Disentangling Homophilic and Heterophilic Patterns for Multi-Domain Graph Foundation Models

Ziyan Wang[1] **Ruiyi Fang**[1,*] **Jingyu Zhao**[1] **Zhimin Mei**[1] **Charles Ling**[1] **Boyu Wang**[1,2]

[1]Western University [2]Vector Institute

{zwan3444, rfang32, jzhao786, zmei22, charles.ling}@uwo.ca
bwang@csd.uwo.ca

Abstract

Graph Foundation Models (GFMs) have attracted growing attention for their ability to model multi-domain graphs universally and generalize to unseen downstream tasks. However, existing GFMs typically rely on a single domain-invariant representation space, which inevitably blurs two fundamental connectivity patterns, *homophily* and *heterophily*, that govern how nodes relate across domains. Additionally, hard-partitioning the graph into disjoint subgraphs discards edges that are essential for self-supervised pre-training. In this work, we introduce **HHGFM**, a **H**omophilic and **H**eterophilic Pattern-Disentangled **G**raph **F**oundation **M**odel that encodes homophily and heterophily separately while keeping the original edge structure intact. Specifically, we propose a node-level soft disentanglement mechanism that estimates each neighbor's homophilic and heterophilic contributions and uses them as edge weights, thereby preserving the original graph structure. Low-pass and high-pass filters then serve as dual backbones to align homophilic and heterophilic patterns separately, supported by our generalization analysis. Extensive experiments on a range of graph datasets validate the effectiveness of our method.

1 Introduction

Foundation models have achieved remarkable success in natural language processing (NLP) [1–3] and computer vision (CV) [4–7], demonstrating strong generalization capabilities across diverse downstream tasks. This success has motivated growing interest in extending the foundation-model paradigm to graph data. Accordingly, Graph Foundation Models (GFMs) have recently attracted increasing attention, particularly for learning from diverse multi-domain graphs and adapting to downstream tasks under few-shot settings [8–13]. A challenge in this paradigm lies in discrepancies among source domains. To address this, a common practice is to learn domain-invariant knowledge to enable cross-domain generalization [12, 8, 14, 10, 11, 9]. SAMGPT [12], MDGFM [9] project graphs from different domains into a unified representation space. In addition, some works [15–21] tokenize graphs and leverage large language models (LLMs) to project multi-domain graphs into a shared text embedding space, implicitly enabling unified modeling of multi-domain knowledge.

However, real-world graphs are far from homogeneous in their connectivity patterns. Two fundamental patterns, *homophily* and *heterophily*, determine how nodes connect [22]: citation networks typically exhibit strong homophily, while social networks and hyperlink graphs frequently show numerous heterophilic patterns [23]. These two connectivity patterns reflect different semantic relations and should not be mixed. Existing GFMs that align all domains into a single invariant space inevitably entangle homophilic and heterophilic signals, making it difficult to distinguish nodes with structurally

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

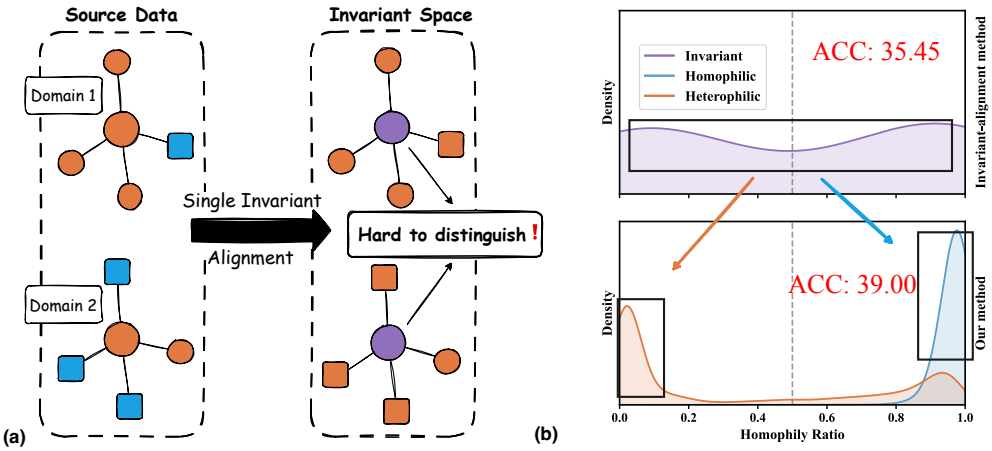

Figure 1: A single invariant space is insufficient under homophily–heterophily discrepancy. (a) Domain 1 is mainly homophilic, whereas Domain 2 is mainly heterophilic. Colors (**Orange**, **Purple** and **Blue**) /shapes (□ and ○) indicate node features/semantic types. After aligning both domains into a single invariant space, the two central nodes obtain similar representations, despite following different relational patterns. (b) Comparison of homophily distributions induced by pre-trained representations: SAMGPT [12] (single invariant) vs. ours (disentangled homophilic/heterophilic). We pre-train on Cora and CiteSeer to obtain node representations, and evaluate transfer on Cornell as the target graph. Invariant representations tend to blur homophily and heterophily, while ours preserves both. See Appendix D.1 for details.

distinct neighborhoods. As a result, the transferable knowledge learned by such alignment is difficult to reflect the true relational rules of each domain, ultimately hindering downstream generalization.

We illustrate this issue in Figure 1. Figure 1(a) shows that Domain 1 is predominantly homophilic while Domain 2 is predominantly heterophilic. Since invariant alignment encourages cross-domain representations to be indistinguishable in order to minimize domain discrepancy, the two central nodes are pulled toward similar representations, even though their neighborhoods follow opposite connectivity patterns. Figure 1(b) further verifies this phenomenon empirically. We compare the homophily ratio distributions induced by the pre-trained representations of an invariant-alignment method and our method. The former yields a nearly flat distribution where homophilic and heterophilic patterns are indistinguishable, whereas ours exhibits two distinct peaks that preserve them, which suggests disentangling homophilic and heterophilic representations is essential for effective multi-domain transfer.

Several recent works [24–28] have also recognized the importance of separating homophilic and heterophilic patterns. For instance, HeterGP [24] and RSGDA [25] partition the graph into two disjoint subgraphs and process them independently. Such partitioning inevitably destroys the original structure. Since edges provide the primary supervision signal for self-supervised GFM pre-training, this loss of connectivity limits what GFMs can learn from unlabeled graphs.

To address this, we propose a node-level soft disentanglement mechanism that separates homophilic and heterophilic patterns by computing the homophilic contribution based on node and structural features as weighted adjacency matrix, which keeping the original edge structure intact. We then employ a low-pass and a high-pass graph filter as dual backbones to obtain homophilic and heterophilic representations respectively. We name our method HHGFM. Our contributions are summarized:

- We empirically and theoretically analyze the limitations of single-invariant alignment in multi-domain GFMs, showing that it blurs homophilic and heterophilic patterns.
- We propose HHGFM, which introduces a soft disentanglement mechanism that separates homophilic and heterophilic patterns while preserving the original edge structure, together with low-pass and high-pass dual backbones that extract homophilic and heterophilic representations respectively.
- Extensive experiments on 8 benchmark graph datasets across 3 domains and comparisons with strong baselines demonstrate the effectiveness of HHGFM.

2 Related Work

Graph Pre-training and Graph Foundation Models. Graph foundation models (GFMs) have progressed rapidly, fueled by self-supervised pre-training on large graphs to learn transferable structural and semantic patterns [29–40]. Standard objectives produce strong initializations such as GCC [41], GPT-GNN [42], GPPT [43], and L2P-GNN [44], alongside many extensions [45–53]. However, most methods assume the pre-training and target graphs are structurally or semantically similar, and can degrade under distribution shift [54–60]. To reduce domain mismatch [61–65], cross-domain graph learning aims to transfer across dissimilar domains, typically via domain-invariant representations (e.g., GCOPE [14], OFA [66], PGPRec [67], UniAug [68], and UniGraph [69]). Yet many are tailored to single-source transfer and struggle to scale to multiple heterogeneous sources. Multi-domain GFMs further unify pre-training across diverse domains: OFA [66] and HiGPT [70] leverage LLM-based conversion but are limited to text-attributed graphs [16, 71]; GCOPE [14] uses virtual bridges but does not explicitly ensure semantic alignment; MDGPT [8] aligns domains with tokens and SVD projections, yet semantic consistency remains challenging. BRIDGE [10] introduces lightweight MoE-based aligners but still relies on a shared invariant space that can obscure domain-specific structure; GRAVER [11] learns transferable graph vocabularies for few-shot adaptation but does not explicitly disentangle homophily and heterophily; and MDGFM [9] balances topology and features, though a unified alignment may still blur distinct structural mechanisms across domains.

Graph Fine-tuning and Prompt Learning Fine-tuning graph foundation models has been extensively studied as a practical way to adapt general-purpose representations to specific downstream problems [72–75, 71, 76–80]. Standard approaches either update the full model or optimize selected components, which require substantial training and memory costs. To improve efficiency, PEFT techniques update a small set of parameters [81–84] or introduce lightweight adaptation modules [85–87]. While these methods preserve most pre-trained knowledge and reduce compute, their cross-domain generalization can still be limited when the target distribution deviates significantly from pre-training. Prompt-based adaptation provides an alternative by conditioning a model on learned prompts rather than directly modifying its weights [88–91]. In graph learning [92–102], prompt tuning typically learns structured prompt embeddings to bias the model toward task-relevant patterns [43, 103–105]. Building on this idea, many graph prompt methods have been proposed [43, 106–108, 103, 109, 110, 8, 111, 112], leveraging learnable prompts to emphasize structural and semantic cues for improved transfer.

Heterophilic Graph Learning. Recent studies have increasingly recognized that heterophily is a critical challenge in graph learning, where connected nodes may have dissimilar labels or features [113, 114]. In graph prompt tuning, HeterGP [24] designs multi-view prompts to unify homophilic and heterophilic scenarios, while ProNoG [115] further considers node-specific non-homophilic patterns for pre-training and prompt learning. In graph foundation models, MDGFM [9] improves transferability by aligning topological information across domains, and GraphAny [116] aims to generalize graph models to broader graph distributions. TFSGFM [117] splits features into homophilic and heterophilic ones, but it overlooks the two underlying connectivity patterns. Graph domain adaptation methods such as RSGDA [25] further explore how to transfer knowledge under structural distribution shifts[118]. These studies show the value of heterophilic patterns, but node-level disentanglement of homophilic and heterophilic patterns in multi-domain pre-training remains underexplored.

3 Preliminary

Notations and Problem Formulation. Formally, let $G_i = (\mathcal{V}_i, \mathcal{E}_i, \boldsymbol{X}_i)$ denote a graph with index i, where $\mathcal{V}_i$ is the node set, $\mathcal{E}_i$ is the edge set, and $\boldsymbol{X}_i \in \mathbb{R}^{n_i \times d_i}$ is the feature matrix of G_i. The adjacency matrix of G_i is denoted by $\boldsymbol{A}_i$. Suppose we have N source graphs $\{G_i^{\mathcal{S}}\}_{i=1}^{N} \subseteq \mathcal{G}^{\mathcal{S}}$ from K source domains, whose distributions form the set $\mathcal{D}^{\mathcal{S}} := \{D_k^{\mathcal{S}}\}_{k=1}^{K}$, where $D_k^{\mathcal{S}}$ denotes the distribution of the k-th source domain. And given a target graph $G^{\mathcal{T}} \in \mathcal{G}^{\mathcal{T}}$ from a target domain $D^{\mathcal{T}} \in \mathcal{D}^{\mathcal{T}}$ (seen or unseen), where $D^{\mathcal{T}}$ denotes the distribution of the target domain, with m labeled nodes randomly selected from each class under the m-shot setting, we need to transfer knowledge from source graphs to make predictions of the unlabeled samples. The major notations used in this paper are summarized in Appendix A.

Homophilic and Heterophilic Patterns. In real-world scenarios, graphs often exhibit a mixture of homophilic and heterophilic characteristics. Each node can exhibit either homophilic or heterophilic patterns, depending on the label relationships with their neighbors. We formally define these two patterns as follows.

Definition 1 (Homophilic and heterophilic patterns [22]). *Given a graph $G = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$ with node labels $\boldsymbol{y}$, let $\mathcal{N}(v) = \{u \mid (u, v) \in \mathcal{E}\}$ be the neighborhood of node v. We say that node v exhibits a* homophilic pattern *if most of its neighbors share the same label as v (i.e., $y_v = y_u$ for most $u \in \mathcal{N}(v)$). Similarly, node v exhibits a* heterophilic pattern *if most of its neighbors have a different label from v (i.e., $y_v \neq y_u$ for most $u \in \mathcal{N}(v)$).*

4 Method

4.1 Homophilic and Heterophilic Patterns Disentanglement

For pre-training, existing GFMs typically learn a shared backbone $\mathbf{\Phi}$ that maps node features from different source domains into a domain-invariant representation space $\mathbb{G}$. Let $\mathcal{X}_k^S$ denote the node feature space of source domain k, and define $\mathcal{X}^S := \bigcup_{k=1}^K \mathcal{X}_k^S$. We write this mapping as:

$$\mathbf{\Phi} : \mathcal{X}^S \to \mathbb{G}. \tag{1}$$

With the help of $\mathbf{\Phi}$, domain-invariant features can be learned, bringing the feature space across source domains closer together. That is, $\mathbf{\Phi}$ is learned to minimize

$$\max_{i,j \in [K]} d_{\mathcal{H}\Delta\mathcal{H}}(D_i^{\mathcal{S}}, D_j^{\mathcal{S}}), \tag{2}$$

where $d_{\mathcal{H}\Delta\mathcal{H}}(\cdot, \cdot)$ denotes the $\mathcal{H}\Delta\mathcal{H}$-divergence induced by a hypothesis class $\mathcal{H}$ (see Appendix B.1 for details). Let $\Lambda^{\mathcal{S}} := \left\{ \sum_{i=1}^K \pi_i D_S^i \ : \ \pi \in \Delta_{K-1} \right\}$ be the convex hull of source-domain distributions. We assume the target distribution is well-approximated by some mixture in this hull, i.e., $D^{\mathcal{T}} \approx \bar{D}_\pi^{\mathcal{S}}$ for some $\bar{D}_\pi^{\mathcal{S}} \in \Lambda^{\mathcal{S}}$. Without assumptions relating the source and target domains, generalization to arbitrary unseen distributions cannot be guaranteed. Under this, solving (2) tightens an upper bound on the target risk, thereby enabling domain adaptation according to [119].

Nevertheless, learning (1) by optimizing (2) has the following limitations:

- Homophilic and heterophilic patterns often coexist in real-world graphs. We assume that each domain can be decomposed as $D_i^{\mathcal{S}} = \alpha_i D_{i,\text{ho}}^{\mathcal{S}} + (1 - \alpha_i) D_{i,\text{he}}^{\mathcal{S}}$. Aligning both components in a single invariant representation space may then map nodes with different local connectivity patterns to similar representations.
- Graph neural networks (GNNs), which are often used as the graph encoder within $\mathbf{\Phi}$, typically rely on the homophily assumption [120, 121]. A homophily-biased encoder entangles homophilic and heterophilic patterns.

Accordingly, to learn more generalizable representations, we propose dual backbones based on homophilic and heterophilic patterns, instead of (1). Formally, we partition nodes into $\mathcal{X}_{\text{ho}}^{\mathcal{S}}$ and $\mathcal{X}_{\text{he}}^{\mathcal{S}}$, and then project them into homophilic and heterophilic representation spaces, i.e.,

$$\mathbf{\Phi}_{\text{ho}} : \mathcal{X}_{\text{ho}}^{\mathcal{S}} \to \mathbb{G}_{\text{ho}}, \quad \mathbf{\Phi}_{\text{he}} : \mathcal{X}_{\text{he}}^{\mathcal{S}} \to \mathbb{G}_{\text{he}},$$

where $\mathbf{\Phi}_{\text{ho}}$ and $\mathbf{\Phi}_{\text{he}}$ denote the homophily- and heterophily-oriented backbones, respectively. To justify the generalization capability of aligning representations separately, we further derive a generalization error upper bound.

For each pattern (homophily and heterophily) $c \in \{\text{ho}, \text{he}\}$, let $D_{i,c}^{\mathcal{S}}$ and $D_c^{\mathcal{T}}$ denote the conditional distributions in the i-th source domain and the target domain, respectively. Let $\mathcal{H}_c$ be the corresponding hypothesis set. We define the source convex hull as $\Lambda_c^{\mathcal{S}} := \{\sum_{i=1}^K \pi_i^c D_{i,c}^{\mathcal{S}} : \pi^c \in \Delta_{K-1}\}$ for each pattern, where Δ_{K-1} is the $K-1$-th dimensional simplex. the distance between distributions of target domain and mixture of source domains as $\gamma_c = \min_{\pi^c \in \Delta_{K-1}} d_{\mathcal{H}_c\Delta\mathcal{H}_c}(D_c^{\mathcal{T}}, \sum_{i=1}^K \pi_i^c D_{i,c}^{\mathcal{S}})$, and the diameter between source domains as $\rho_c = \sup_{\pi^c, \tilde{\pi}^c \in \Delta_{K-1}} d_{\mathcal{H}_c\Delta\mathcal{H}_c}(\sum_{i=1}^K \pi_i^c D_{i,c}^{\mathcal{S}}, \sum_{i=1}^K \tilde{\pi}_i^c D_{i,c}^{\mathcal{S}})$.

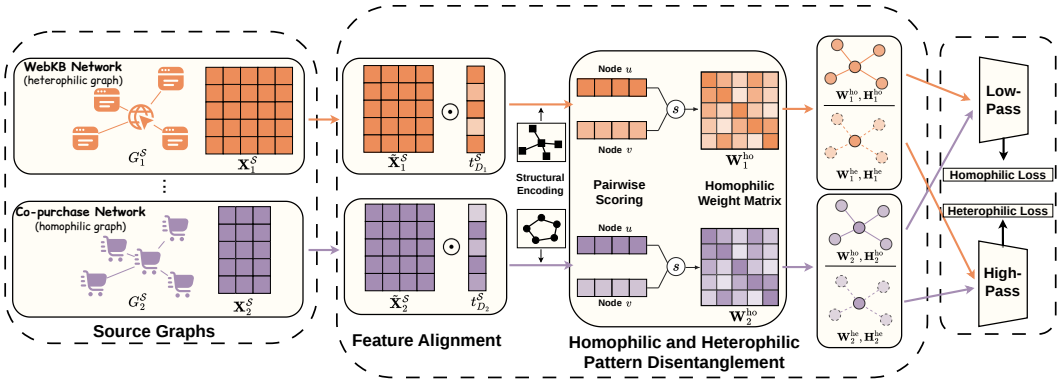

Figure 2: The overall framework of the proposed HHGFM. Features from multiple source graphs are first aligned in dimension and semantics. Homophilic and heterophilic patterns are then softly disentangled through pairwise scoring based on aligned features and structural encodings, while preserving the original graph structure. Low-pass and high-pass encoders subsequently learn homophilic and heterophilic representations, respectively, through separate self-supervised objectives.

Theorem 1 (Upper bound on the target-domain risk [119, 122]). *For hypothesis $h_c \in \mathcal{H}_c$, consider a predictor h satisfying $\epsilon^{\mathcal{T}}(h) \leq \sum_c \alpha_t^c \epsilon_c^{\mathcal{T}}(h_c) + \varepsilon_{\text{dis}}$, where ε_{dis} is the risk caused by imperfect pattern disentanglement. The target risk is bounded by*

$$\epsilon^{\mathcal{T}}(h) \leq \sum_{c\in\{\text{ho},\text{he}\}} \alpha_t^c \left[\sum_{i=1}^{K} \pi_i^{c,*} \epsilon_{i,c}^{\mathcal{S}}(h_c) + \frac{\gamma_c + \rho_c}{2} + \lambda_c \right] + \varepsilon_{\text{dis}} \text{ (Proof in Appendix B.2).} \quad (3)$$

Here, α_t^c is the target proportion of pattern c, with $\alpha_t^{\text{ho}} = \alpha_t$ and $\alpha_t^{\text{he}} = 1 - \alpha_t$; $\pi^{c,*}$ is a mixture weight vector attaining γ_c. The ideal joint risk λ_c between the target component and its closest source mixture.

Theorem 1 states that a single encoder aligns all domains into a single representation space, where the alignment can be affected by the entanglement of homophilic and heterophilic patterns across domains. Additionally, it provides an interpretation of why separate pattern alignment can improve generalization. Consider two domains with similar homophilic and heterophilic conditional distributions, but different pattern proportions: one is predominantly homophilic, whereas the other is predominantly heterophilic. Their overall distributions can differ substantially even though the corresponding patterns are similar across domains. Aligning these overall distributions may therefore match representations from different patterns, potentially obscuring their distinct predictive structures.

Separate alignment compares homophilic representations with homophilic representations, and heterophilic representations with heterophilic representations. Moreover, each target pattern can be approximated by its own source mixture through the weights $\pi^{c,*}$, allowing the two patterns to rely on different combinations of source domains. When within-pattern domain differences are small, this can yield small target-to-hull discrepancies γ_c and source-hull diameters ρ_c, while avoiding discrepancies arising solely from different pattern proportions. The bound makes the associated trade-off explicit: These reductions yield a tighter target-risk upper bound if they outweigh the disentanglement error and any increases in the source-risk and ideal-joint-risk terms.

4.2 Training Framework for HHGFM

Based on section 4.1, we elaborate on the proposed HHGFM with its framework shown in Figure 2.

Multi-Domain Feature Alignment. Graphs in different domains present heterogeneous features with inconsistent dimensions and semantics. To address this, we use aligners introduced in [8, 14, 12] that consequently unify dimension- and semantic-wise features. Given a graph $G_i^{\mathcal{S}} = (\mathcal{V}_i^{\mathcal{S}}, \mathcal{E}_i^{\mathcal{S}}, \boldsymbol{X}_i^{\mathcal{S}})$ from the source domain $D_j^{\mathcal{S}}$, we first align the dimensions of its feature matrix,

$$\tilde{\boldsymbol{X}}_i^{\mathcal{S}} \in \mathbb{R}^{n_i \times d} = \text{SVD}(\boldsymbol{X}_i^{\mathcal{S}}), \quad (4)$$

where SVD$(\cdot)$ is implemented by truncated SVD [123] that aligns feature dimensions as d. Next, we unify their semantic space across various domains as follows,

$$\hat{\boldsymbol{X}}_i^{\mathcal{S}} \in \mathbb{R}^{n_i \times d} = \boldsymbol{t}_{D_j^{\mathcal{S}}} \odot \tilde{\boldsymbol{X}}_i^{\mathcal{S}}, \tag{5}$$

where $\hat{\boldsymbol{X}}_i^{\mathcal{S}}$ is the unified feature matrix after dimension and semantic alignment, $\boldsymbol{t}_{D_j^{\mathcal{S}}}$ is a learnable domain token for $D_j^{\mathcal{S}}$, and $\odot$ denotes the Hadamard product.

Disentangling Homophilic and Heterophilic Patterns. Since pre-training is unsupervised, node labels are unavailable, preventing direct identification of label-based homophilic and heterophilic patterns. We therefore use similarity in a joint representation of node attributes and structural information as an unsupervised proxy, where node attributes capture feature-level similarity and structural encodings characterize nodes' positions and local connectivity patterns. As both are informative about node labels [124, 125], their joint representation can help distinguish homophilic and heterophilic interactions.

Specifically, we construct node structural features by combining degree centrality [126], Laplacian positional encoding [127–131], and random-walk structural encoding [132]. The structural encoding of node v in each source graph is defined as follows, where we omit the source graph index for simplicity.

$$\boldsymbol{x}_v^{\mathrm{SE}} = \mathrm{concat}\left[\boldsymbol{p}_v^{\mathrm{Lap}},\, \boldsymbol{p}_v^{\mathrm{RW}},\, \boldsymbol{D}_{vv}\right]. \tag{6}$$

Here, $\boldsymbol{p}_v^{\mathrm{Lap}}$, $\boldsymbol{p}_v^{\mathrm{RW}}$, and $\boldsymbol{D}_{vv}$ denote the Laplacian positional encoding, the random-walk structural encoding, and the degree centrality of node v, respectively. $\boldsymbol{D}$ is the diagonal degree matrix with $\boldsymbol{D}_{vv} = \sum_{u\in\mathcal{V}} \boldsymbol{A}_{vu}$. For Laplacian encoding, we use the normalized Laplacian $\boldsymbol{L} := \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$ and its eigendecomposition $\boldsymbol{L}\boldsymbol{U} = \boldsymbol{U}\boldsymbol{\Lambda}$, and set $\boldsymbol{p}_v^{\mathrm{Lap}} := [\boldsymbol{U}_{v1}, \boldsymbol{U}_{v2}, \cdots, \boldsymbol{U}_{vr}] \in \mathbb{R}^r$. For random-walk encoding, we define the transition matrix $\boldsymbol{P} := \boldsymbol{D}^{-1}\boldsymbol{A}$ and set $\boldsymbol{p}_v^{\mathrm{RW}} := [\mathbf{RW}_{vv}^0, \mathbf{RW}_{vv}^1, \cdots, \mathbf{RW}_{vv}^r] \in \mathbb{R}^{r+1}$, where $\mathbf{RW}_{vv}^t := (\boldsymbol{P}^t)_{vv}$ denotes the t-step return probability to node v. We then combine the structural encoding with node attributes to obtain a learnable joint embedding for node v:

$$\boldsymbol{z}_v = \mathrm{MLP}\left(\,\mathrm{concat}[\boldsymbol{x}_v^{\mathrm{SE}},\, \hat{\boldsymbol{x}}_v]\,;\, \boldsymbol{\phi}\right).$$

Based on these joint embeddings, we adopt a soft weighting mechanism that preserves the original graph connectivity while assigning each edge complementary contributions to the two branches. This avoids requiring a hard assignment of each edge to a homophilic or heterophilic category based on an imperfect proxy. Specifically, for each node v, we compute a vector of pairwise dissimilarity scores with its neighbors $u \in \mathcal{N}(v)$:

$$\boldsymbol{s}_v \in \mathbb{R}^{|\mathcal{N}(v)|} := \left[\, s_{v,u} \,\right]_{u\in\mathcal{N}(v)} = \left[\|\boldsymbol{z}_v - \boldsymbol{z}_u\|_2^2\right]_{u\in\mathcal{N}(v)}. \tag{7}$$

We convert these dissimilarity scores into homophilic contribution weights using an element-wise exponential kernel, $\boldsymbol{\alpha}_v^{\mathrm{ho}} = \exp(-t\,\boldsymbol{s}_v)$, where $t > 0$ is a temperature hyperparameter. We define the heterophilic contribution weights as $\boldsymbol{\alpha}_v^{\mathrm{he}} = \mathbf{1} - \boldsymbol{\alpha}_v^{\mathrm{ho}}$. Thus, edges connecting nodes with similar joint embeddings receive higher weights in the homophilic branch, whereas those connecting nodes with dissimilar embeddings receive higher weights in the heterophilic branch. Consequently, we obtain the homophilic weight matrix $\boldsymbol{W}^{\mathrm{ho}} \in [0,1]^{n\times n}$ with entries

$$\boldsymbol{W}_{vu}^{\mathrm{ho}} := \begin{cases} \alpha_{v,u}^{\mathrm{ho}}, & u \in \mathcal{N}(v), \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

and $\boldsymbol{W}^{\mathrm{he}} := \boldsymbol{A} - \boldsymbol{W}^{\mathrm{ho}}$. Since $\boldsymbol{W}^{\mathrm{ho}} + \boldsymbol{W}^{\mathrm{he}} = \boldsymbol{A}$, the original edge connectivity is preserved across the two views. This construction allows the two encoders to learn from interactions with different levels of homophilic weights without requiring a perfect partition. Its effectiveness depends on whether the learned similarity provides useful cues for representation learning, rather than whether every edge is correctly identified by labels.

Graph Filtering and Pre-training Loss. To align distributions of multiple source domains based on homophilic and heterophilic patterns and alleviate homophily bias which is induced by GNNs, we

use high-pass and low-pass filter as dual backbones. Specifically, we directly use the GCN as the low-pass filter on homophilic weight adjacent matrix:

$$\boldsymbol{H}_i^{\mathrm{ho},(l)} = \sigma\left((\boldsymbol{H}_i^{\mathrm{ho},(l-1)} + \boldsymbol{W}_i^{\mathrm{ho}}\boldsymbol{H}_i^{\mathrm{ho},(l-1)})\,\boldsymbol{\Theta}^{\mathrm{ho}}\right). \tag{9}$$

For the heterophilic pattern, inspired by [133], we design a high-pass filter to aggregate discrepancies between a node and its neighbors, formulated as

$$\boldsymbol{H}_i^{\mathrm{he},(l)} = \sigma\Big(\Big(\boldsymbol{H}_i^{\mathrm{he},(l-1)} + (\boldsymbol{D}_i^{\mathrm{he}} - \boldsymbol{W}_i^{\mathrm{he}})\boldsymbol{H}_i^{\mathrm{he},(l-1)}\Big)\,\boldsymbol{\Theta}^{\mathrm{he}}\Big), \tag{10}$$

where $\boldsymbol{D}_i^{\mathrm{he}}$ is the degree matrix of $\boldsymbol{W}_i^{\mathrm{he}}$, with $\boldsymbol{D}_{i,vv}^{\mathrm{he}} = \sum_u \boldsymbol{W}_{i,vu}^{\mathrm{he}}$.

Prior studies [107, 11] adopt a self-supervised contrastive link prediction as the pre-training task, which unifies both link prediction with node- or graph-level classification tasks under a shared framework. However, the link prediction loss relies on homophily assumption. To solve this, we compute homophilic and heterophilic losses separately.

Specifically, we construct a set of tuples (v, u^+, u^-) sampled non-redundantly from $\{G^{\mathcal{S}}\}$, where u^+, u^- are the positive and negative neighbor. For the homophilic loss, the objective is to encourage higher semantic similarity between v and its positive neighbor $\boldsymbol{h}_{u^+}^{\mathcal{S}}$, while suppressing that with the negative neighbor $\boldsymbol{h}_{u^-}^{\mathcal{S}}$. For the heterophilic loss, the objective is the opposite. The training objective is formalized as:

$$\begin{aligned}
\mathcal{L}_{\mathrm{ho}}(\boldsymbol{\Theta}^{\mathrm{ho}}, \boldsymbol{t}_{D^{\mathcal{S}}}, \boldsymbol{\phi}; \boldsymbol{H}^{\mathcal{S},\mathrm{ho}}) &= -\sum_{(v,u^+,u^-)} \log \frac{\exp\left(g(\boldsymbol{h}_v^{\mathcal{S},\mathrm{ho}}, \boldsymbol{h}_{u^+}^{\mathcal{S},\mathrm{ho}})/\tau\right)}{\exp\left(g(\boldsymbol{h}_v^{\mathcal{S},\mathrm{ho}}, \boldsymbol{h}_{u^+}^{\mathcal{S},\mathrm{ho}})/\tau\right) + \exp\left(g(\boldsymbol{h}_v^{\mathcal{S},\mathrm{ho}}, \boldsymbol{h}_{u^-}^{\mathcal{S},\mathrm{ho}})/\tau\right)}, \\
\mathcal{L}_{\mathrm{he}}(\boldsymbol{\Theta}^{\mathrm{he}}, \boldsymbol{t}_{D^{\mathcal{S}}}, \boldsymbol{\phi}; \boldsymbol{H}^{\mathcal{S},\mathrm{he}}) &= -\sum_{(v,u^+,u^-)} \log \frac{\exp\left(g(\boldsymbol{h}_v^{\mathcal{S},\mathrm{he}}, \boldsymbol{h}_{u^-}^{\mathcal{S},\mathrm{he}})/\tau\right)}{\exp\left(g(\boldsymbol{h}_v^{\mathcal{S},\mathrm{he}}, \boldsymbol{h}_{u^+}^{\mathcal{S},\mathrm{he}})/\tau\right) + \exp\left(g(\boldsymbol{h}_v^{\mathcal{S},\mathrm{he}}, \boldsymbol{h}_{u^-}^{\mathcal{S},\mathrm{he}})/\tau\right)}, \\
\mathcal{L}_{\mathrm{pre}}(\boldsymbol{\Theta}, \boldsymbol{t}_{D^{\mathcal{S}}}, \boldsymbol{\phi}) &= \mathcal{L}_{\mathrm{ho}}(\boldsymbol{\Theta}^{\mathrm{ho}}, \boldsymbol{t}_{D^{\mathcal{S}}}, \boldsymbol{\phi}) + \lambda\mathcal{L}_{\mathrm{he}}(\boldsymbol{\Theta}^{\mathrm{he}}, \boldsymbol{t}_{D^{\mathcal{S}}}, \boldsymbol{\phi}).
\end{aligned} \tag{11}$$

where λ is the hyperparameter and τ is a temperature parameter. $g = \mathrm{sim}(\cdot,\cdot)$ denotes a similarity function. $\boldsymbol{\Theta}^\star$ and $\boldsymbol{t}_{D^{\mathcal{S}}}^\star = [t_{D_1^{\mathcal{S}}}^\star, \ldots, t_{D_K^{\mathcal{S}}}^\star]^\top$ are frozen after pre-training.

Target Domain Adaptation with Prompt Tuning. we apply learnable graph prompts [12, 10, 11] to transfer relevant knowledge:

$$\mathcal{P}(\widetilde{\mathbf{X}}^{\mathcal{T}}) = \widetilde{\mathbf{X}}^{\mathcal{T}} \odot \left(\sum_{k=1}^{K} \beta_k^{\mathcal{T}} \mathbf{t}_{D_k^{\mathcal{S}}}^\star\right), \tag{12}$$

where $\boldsymbol{\beta}^{\mathcal{T}} = [\beta_1, \ldots, \beta_K]^\top$ are learnable parameters, and $\tilde{\boldsymbol{X}}_i^{\mathcal{T}}$ is dimension-aligned via (4). Similar to the source graphs, we obtain homophilic and heterophilic representations $\boldsymbol{H}^{\mathcal{T},\mathrm{ho}}$ and $\boldsymbol{H}^{\mathcal{T},\mathrm{he}}$ with frozen parameters $\boldsymbol{\Theta}^\star$ following (6)–(10). Next, to better select relevant homophilic and heterophilic knowledge, we introduce $\boldsymbol{\sigma}$, a lightweight vector.

$$\boldsymbol{H}^{\mathcal{T}} = \sigma_{\mathrm{ho}}\boldsymbol{H}^{\mathcal{T},\mathrm{ho}} + \sigma_{\mathrm{he}}\boldsymbol{H}^{\mathcal{T},\mathrm{he}}. \tag{13}$$

To handle both node- and graph-level classification, we use a unified subgraph similarity-based task template. Specifically, training nodes $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\}$, the loss function is :

$$\mathcal{L}_{\mathrm{down}}(\boldsymbol{\beta}^{\mathcal{T}}, \boldsymbol{\sigma}; \boldsymbol{H}^{\mathcal{T}}) = -\sum_{(\boldsymbol{x}_i, y_i)} \log \frac{\exp\left(g(\boldsymbol{h}_i^{\mathcal{T}}, \overline{\boldsymbol{h}}_{y_i}^{\mathcal{T}})/\tau\right)}{\sum_{y\in\{Y^{\mathcal{T}}\}} \exp\left(g(\boldsymbol{h}_i^{\mathcal{T}}, \overline{\boldsymbol{h}}_{y}^{\mathcal{T}})/\tau\right)}, \tag{14}$$

where $\overline{\boldsymbol{h}}_{y_i}^{\mathcal{T}}$ denotes average embedding of all training nodes of class y_i.

The whole steps of HHGFM are summarized in Algorithms 1 and 2, Appendix C.1. The time complexity analysis is provided in Appendix C.2.

5 Experiment

5.1 Experimental Setup

Dataset. We evaluate our method on graph benchmarks from three domains: **citation networks** (Cora [134], CiteSeer [135], and PubMed [136]), **co-purchase graphs** (Photo and Computers [137]), and **web-page graphs** (Cornell, Texas, and Wisconsin [138]). Citation networks and co-purchase graphs are typically *homophilic*, whereas Web page graphs are known to be more *heterophilic*. These domains cover diverse graph semantics and feature spaces, enabling a comprehensive assessment of multi-domain transfer performance. Additional details are provided in Appendix D.2.

Baselines. We compare our HHGFM with state-of-the-art baselines grouped into three categories, including: **1) Vanilla Graph Neural Networks:** GCN [139], GAT [140]; **2) Self-supervised Graph Representation Learning:** GCC [41], DGI [141], GraphCL [142], DSSL [143], GraphACL [144]; **3) Graph Prompt tuning:** GPPT [43], GraphPrompt [107], GPF [109], ProNoG [115]; **4) Multi-domain Pre-training:** GCOPE [14], MDGPT [8], SAMGPT [12], BRIDGE [10]. Additional details are provided in Appendix D.3

Experiment Details. We evaluate HHGFM under multi-domain pre-training and few-shot fine-tuning settings. For multi-domain pre-training, we consider two transfer scenarios: cross-dataset, where source and target datasets come from the same domain but are different datasets, and cross-domain, where the target dataset belongs to an unseen domain. For few-shot fine-tuning, we adopt the m-shot setting, where m labeled samples per class are randomly selected. For graph classification, we follow prior work [43, 107, 109, 145] to construct ego-graphs centered on target nodes and assign labels according to the center nodes. Accuracy is used as the evaluation metric.

Implementation Details. All experiments are conducted on NVIDIA H100 and RTX A5000 GPUs. Following [12], in all experiments for the proposed HHGFM, we adopt a 3-layer high-pass and low-pass filters, with a hidden dimension of 256. For multi-domain feature alignment, we apply truncated SVD to unify input feature dimensions to $d = 50$. The structural encoding $\boldsymbol{x}_v^{\mathrm{SE}}$ is a 12-dimensional vector concatenating 5-dimensional Laplacian positional encoding, 5-step random-walk structural encoding, and normalized degree centrality. The contribution score temperature is fixed at $t = 1.0$. For pre-training, we use the Adam optimizer with a learning rate of 0.01. The hyperparameter λ in $\mathcal{L}_{\text{pre}}$ is searched over $\{0.0001, 0.001, 0, 01, 0.1\}$. For downstream prompt tuning, we use Adam with a learning rate of 0.01. We evaluate under m-shot and report the mean accuracy over 20 independent runs.

5.2 Experimental Results

m-shot Node and Graph Classification Table 1 presents the results in the one-shot setting. We have the following observations: (1) HHGFM achieves state-of-the-art performance, outperforming baselines, with consistent gains across both cross-dataset and cross-domain transfer scenarios. This demonstrates that explicitly disentangling homophilic and heterophilic patterns yields more transferable representations under extreme one-shot supervision. Such transferability is particularly valuable when target labels are insufficient to learn effective representations from scratch. (2) In the homophilic setting where both source and target domains consist of homophilic graphs, HHGFM remains highly competitive. This suggests that modeling heterophilic patterns does not compromise its ability to exploit the shared homophilic patterns across graphs. (3) In the more realistic scenario where homophilic and heterophilic graphs coexist as either source or target domains, HHGFM consistently achieves state-of-the-art performance. These results support our motivation: a single homophily-oriented encoder can entangle the two patterns, whereas our dual-backbone design with high-pass filtering preserves discriminative signals from heterophilic neighborhoods, enabling better transfer. By aligning the two patterns separately, HHGFM facilitates knowledge transfer across graphs with different mixtures of homophilic and heterophilic patterns. Additional results for **five-shot** and **ten-shot** settings are in Tables 5 and 6, Appendix E.1.

We also conduct experiments to assess performance under varying m, comparing with three baselines GCOPE, MDGPT and SAMGPT. As shown in Figure 3(a), the performance of all methods generally improves as the number of shots increases on both Cora and Wisconsin. HHGFM consistently

Table 1: Accuracy (% ± std over 20 runs) of **one-shot classification**. Best results are shown in **bold**, and runner-up results are underlined. CR† = Cora, CS† = CiteSeer, PM† = PubMed; PT‡ = Photo, CP‡ = Computers; CN§ = Cornell, TX§ = Texas, WCS§ = Wisconsin. Datasets with the same superscript symbol are from the same domain.

Source	**Cross-Dataset**			**Cross-Domain**			
	CS† PM† PT‡ CP‡	CR† PM† PT‡ CP‡	CR† CS† PT‡ CN§	PT‡ CP‡ CN§ TX§	PT‡ CP‡ CN§ TX§	CR† CS† PT‡ CP‡	CR† CS† PT‡ CP‡
Model / Target	CR†	CS†	PM†	CR†	CS†	WCS§	CN§
Node Classification							
GCN [139]	29.84 ± 4.72	26.91 ± 4.35	38.42 ± 5.18	27.96 ± 5.14	23.42 ± 5.36	24.86 ± 6.72	25.14 ± 6.48
GAT [140]	30.52 ± 4.81	27.46 ± 4.49	39.15 ± 5.26	28.54 ± 5.22	24.18 ± 5.61	25.37 ± 6.84	25.92 ± 6.35
GCC [41]	34.16 ± 5.36	29.73 ± 5.14	43.62 ± 5.91	32.48 ± 5.76	25.86 ± 6.48	26.74 ± 7.52	27.18 ± 7.36
DGI [141]	35.27 ± 5.48	30.46 ± 5.21	44.31 ± 6.02	33.62 ± 5.93	26.42 ± 6.72	27.36 ± 7.84	27.65 ± 7.61
GraphCL [142]	36.84 ± 5.89	31.52 ± 5.63	46.18 ± 6.21	35.21 ± 6.08	27.38 ± 7.16	28.94 ± 8.23	28.41 ± 7.92
DSSL [143]	35.91 ± 5.76	30.84 ± 5.42	45.36 ± 6.14	34.57 ± 5.97	26.91 ± 6.83	28.12 ± 8.04	27.86 ± 7.74
GraphACL [144]	38.46 ± 5.64	32.71 ± 5.52	47.84 ± 6.03	36.82 ± 5.92	28.74 ± 7.28	29.53 ± 8.36	29.21 ± 8.08
GPPT [43]	37.82 ± 6.04	32.18 ± 5.91	47.15 ± 6.38	36.24 ± 6.21	28.16 ± 7.34	29.42 ± 8.18	28.94 ± 8.07
GraphPrompt [107]	40.36 ± 6.18	34.72 ± 6.07	49.68 ± 6.52	38.92 ± 6.31	29.64 ± 7.82	30.46 ± 8.64	30.18 ± 8.36
GPF [109]	41.27 ± 6.31	35.46 ± 6.19	50.43 ± 6.64	39.78 ± 6.44	30.18 ± 8.04	30.83 ± 8.72	30.67 ± 8.45
ProNoG [115]	43.18 ± 6.12	37.64 ± 6.28	52.75 ± 6.49	41.36 ± 6.37	31.06 ± 8.31	31.24 ± 8.87	31.58 ± 8.62
GCOPE [14]	36.26 ± 7.37	38.14 ± 4.26	44.41 ± 8.12	40.62 ± 8.84	31.65 ± 9.37	28.72 ± 6.91	31.28 ± 10.24
MDGPT [8]	42.71 ± 5.18	39.26 ± 8.74	49.62 ± 4.91	47.41 ± 9.18	32.18 ± 9.84	30.56 ± 7.42	32.05 ± 10.73
SAMGPT [12]	**47.36 ± 6.84**	38.33 ± 8.85	45.84 ± 10.15	**53.20 ± 9.63**	33.20 ± 10.26	31.80 ± 6.38	36.40 ± 11.46
BRIDGE [10]	46.58 ± 8.93	39.84 ± 5.62	55.12 ± 6.37	52.84 ± 8.76	32.74 ± 9.56	31.12 ± 7.18	34.76 ± 10.38
HHGFM (ours)	47.12 ± 7.86	**42.18 ± 9.42**	**57.24 ± 7.51**	52.46 ± 8.73	**34.07 ± 10.16**	**33.80 ± 9.24**	**39.18 ± 8.71**
Graph Classification							
GCN [139]	31.76 ± 5.28	27.14 ± 4.91	40.62 ± 5.84	30.48 ± 5.73	24.36 ± 5.86	25.62 ± 6.74	26.18 ± 6.45
GAT [140]	32.41 ± 5.47	27.86 ± 5.12	41.36 ± 6.03	31.16 ± 5.94	25.12 ± 6.04	26.28 ± 6.93	26.74 ± 6.71
GraphCL [142]	39.84 ± 6.58	33.76 ± 6.24	48.72 ± 7.16	38.42 ± 6.86	29.36 ± 7.28	30.42 ± 8.36	30.16 ± 8.08
DSSL [143]	38.91 ± 6.42	33.08 ± 6.12	47.93 ± 7.04	37.81 ± 6.74	28.74 ± 7.14	29.86 ± 8.12	29.42 ± 7.86
GraphACL [144]	41.62 ± 6.23	35.41 ± 5.98	50.37 ± 6.84	40.18 ± 6.52	30.16 ± 7.46	31.04 ± 8.54	30.72 ± 8.31
GraphPrompt [107]	43.76 ± 6.74	37.18 ± 6.48	52.46 ± 7.62	42.27 ± 7.21	31.24 ± 7.86	32.36 ± 8.71	32.08 ± 8.44
GPF [109]	44.58 ± 6.91	37.92 ± 6.63	53.18 ± 7.84	43.08 ± 7.38	31.84 ± 8.04	32.92 ± 8.92	32.64 ± 8.68
ProNoG [115]	46.84 ± 6.52	39.81 ± 6.28	55.64 ± 7.46	44.96 ± 7.08	32.78 ± 8.26	33.46 ± 9.04	33.12 ± 8.81
GCOPE [14]	47.92 ± 8.12	40.26 ± 4.73	56.38 ± 8.74	55.62 ± 8.84	33.46 ± 9.37	32.72 ± 6.91	34.28 ± 10.24
MDGPT [8]	49.31 ± 5.84	41.58 ± 9.18	57.62 ± 5.36	56.41 ± 9.18	34.18 ± 9.84	33.56 ± 7.42	34.05 ± 10.73
SAMGPT [12]	50.86 ± 7.31	42.84 ± 4.28	59.41 ± 8.02	62.67 ± 5.84	34.28 ± 8.93	34.76 ± 4.93	38.96 ± 7.54
BRIDGE [10]	51.24 ± 9.04	**45.96 ± 6.08**	58.73 ± 6.84	58.91 ± 4.62	35.52 ± 8.26	**37.18 ± 6.41**	36.13 ± 4.07
HHGFM (ours)	**52.18 ± 8.46**	44.36 ± 9.12	**61.24 ± 8.03**	**62.82 ± 8.74**	**36.92 ± 10.26**	36.63 ± 9.41	**40.72 ± 8.88**

achieves the best or highly competitive results across different shot settings, demonstrating its effectiveness. We further evaluate HHGFM in the zero-shot setting following [146], where it consistently outperforms the compared baselines across all three transfer scenarios, as shown in Figure 3(d). These results suggest that pattern disentanglement facilitates knowledge transfer even without labeled target examples.

Generalization across Homophily and Heterophily. To further evaluate the performance of HHGFM across homophilic and heterophilic graphs, we conduct experiments on homophilic datasets (Cora [134], CiteSeer [135] and Photo [137]) and heterophilic datasets(Cornell [138], Texas [138]). As shown in Table 2, HHGFM achieves superior performance in both node and graph classification. When transferring from Cora and CiteSeer to Cornell, HHGFM improves over the strongest baseline in node and graph classification. These consistent improvements across both transfer directions suggest that HHGFM effectively leverages transferable knowledge from multi-domain homophilic and heterophilic patterns, even when the source and target graphs exhibit different homophily characteristics.

Hyperparameter Analysis and Ablation Study. In our HHGFM, there is only one crucial hyperparameter, λ, which controls the trade-off between the losses associated with the high-pass and low-pass filters. Figure 3 (c) shows the influence of λ on Cornell [138], Photo [137], and Cora [134]. When λ is too small, the model may underemphasize the high-pass filtering objective, leading to insufficient preservation of heterophilic signals. When $\lambda = 0$, we disable the high-pass filter. When λ is too large, the high-pass filtering loss may be overemphasized. We see that when λ is zero or is too large, the performance is not satisfactory.

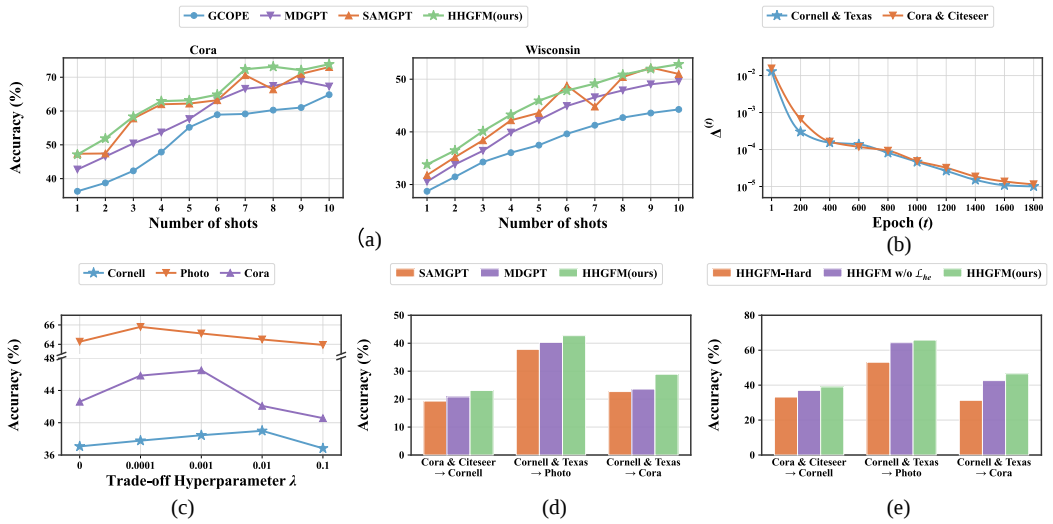

Figure 3: (a) m-shot node classification. (b) Stability of homophilic and heterophilic pattern disentanglement. (c) Hyperparameter sensitivity λ in the one-shot setting. (d) Zero-shot node classification. (e) Ablation study on pattern disentanglement.

We further conduct an ablation study in Figure 3(e) to evaluate soft disentanglement and the heterophilic learning objective. We construct HHGFM-Hard by binarizing the homophilic contribution with a threshold of 0.5, assigning each edge exclusively to either the homophilic or heterophilic channel. The full model consistently outperforms both variants across all three transfer settings. The larger performance drop caused by hard partitioning highlights the benefit of retaining soft edge contributions to both patterns, while the degradation without $\mathcal{L}_{he}$ supports the importance of explicitly learning heterophilic patterns.

Table 2: Analysis of **homophilic** and **heterophilic** graph. Accuracy (% ± std over 20 runs) of one-shot classification. Best results are shown in **bold**. Datasets with the same superscript symbol are from the same domain.

Source	Cora† CiteSeer†	Cornell§ Texas§	Cornell§ Texas§
Model / Target	Cornell§	Photo‡	Cora†
Node Classification			
GCOPE [14]	28.80±7.96	54.42±9.18	38.73±7.54
MDGPT [8]	30.26±8.14	56.37±9.26	43.15±8.02
SAMGPT [12]	35.45±8.24	61.05±9.45	44.90±7.88
HHGFM(ours)	**39.00±8.66**	**65.80±9.54**	**46.50±9.69**
Graph Classification			
GCOPE [14]	29.87±8.32	60.13±8.40	52.83±7.36
MDGPT [8]	30.20±7.37	63.29±6.86	53.36±7.49
SAMGPT [12]	33.20±10.38	65.80±9.86	61.60±8.02
HHGFM(ours)	**39.65±7.46**	**69.80±7.03**	**61.73±10.73**

Figure 3(b) shows the stability of pattern disentanglement. $\Delta^{(t)}$ measures how much edge contributions change between consecutive epochs. Its decrease indicates that the disentanglement becomes more stable during training.

6 Conclusion

This paper proposed HHGFM, a Homophilic and Heterophilic Pattern-Disentangled Graph Foundation Model for multi-domain graph pre-training. Unlike prior GFMs that map all source domains into a single invariant representation space and inevitably blur homophilic and heterophilic patterns, our HHGFM aligns the two patterns separately through low-pass and high-pass dual encoders, while preserving the original edge structure. Extensive experiments on 8 benchmark datasets demonstrate that HHGFM consistently outperforms state-of-the-art graph foundation models.

Acknowledgments and Disclosure of Funding

This work is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), Discovery Grants program.

References

[1] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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*, 2019.

[3] 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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[4] 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.

[5] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[6] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM Computing Surveys*, 2022.

[7] Jincheng Huang, Jie Xu, Xiaoshuang Shi, Ping Hu, Lei Feng, and Xiaofeng Zhu. Revisiting confidence calibration for misclassification detection in vlms. In *International Conference on Learning Representations (ICLR)*, 2026.

[8] Xingtong Yu, Chang Zhou, Yuan Fang, and Xinming Zhang. Text-free multi-domain graph pre-training: Toward graph foundation models. *arXiv preprint arXiv:2405.13934*, 2024.

[9] Shuo Wang, Bokui Wang, Zhixiang Shen, Boyan Deng, and Zhao Kang. Multi-domain graph foundation models: Robust knowledge transfer via topology alignment. In *International Conference on Machine Learning (ICML)*, 2025.

[10] Haonan Yuan, Qingyun Sun, Junhua Shi, Xingcheng Fu, Bryan Hooi, Jianxin Li, and Philip S Yu. How much can transfer? BRIDGE: Bounded multi-domain graph foundation model with generalization guarantees. In *International Conference on Machine Learning (ICML)*, 2025.

[11] Haonan Yuan, Qingyun Sun, Junhua Shi, Xingcheng Fu, Bryan Hooi, Jianxin Li, and Philip S Yu. Graver: Generative graph vocabularies for robust graph foundation models fine-tuning. *arXiv preprint arXiv:2511.05592*, 2025.

[12] Xingtong Yu, Zechuan Gong, Chang Zhou, Yuan Fang, and Hui Zhang. SAMGPT: Text-free graph foundation model for multi-domain pre-training and cross-domain adaptation. In *The ACM Web Conference (WWW)*, 2025.

[13] Xingtong Yu, Shenghua Ye, Ruijuan Liang, Chang Zhou, Hong Cheng, Xinming Zhang, and Yuan Fang. Evaluating progress in graph foundation models: A comprehensive benchmark and new insights. *arXiv preprint arXiv:2603.10033*, 2026.

[14] Haihong Zhao, Aochuan Chen, Xiangguo Sun, Hong Cheng, and Jia Li. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2024.

[15] Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Language is all a graph needs. In *Findings of the Association for Computational Linguistics: EACL*, 2024.

[16] Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. Can language models solve graph problems in natural language? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[17] Jiayan Guo, Lun Du, Hengyu Liu, Mengyu Zhou, Xinyi He, and Shi Han. Gpt4graph: Can large language models understand graph structured data? an empirical evaluation and benchmarking. *arXiv preprint arXiv:2305.15066*, 2023.

[18] Haiteng Zhao, Shengchao Liu, Ma Chang, Hannan Xu, Jie Fu, Zhihong Deng, Lingpeng Kong, and Qi Liu. Gimlet: A unified graph-text model for instruction-based molecule zero-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[19] Limei Wang, Kaveh Hassani, Si Zhang, Dongqi Fu, Baichuan Yuan, Weilin Cong, Zhigang Hua, Hao Wu, Ning Yao, and Bo Long. Learning graph quantized tokenizers. *arXiv preprint arXiv:2410.13798*, 2024.

[20] Zihao Jing, Qiuhao Zeng, Ruiyi Fang, Yan Sun, Boyu Wang, and Pingzhao Hu. Entropy-guided dynamic tokens for graph-llm alignment in molecular understanding. In *International Conference on Learning Representations (ICLR)*, 2026.

[21] Zihao Jing, Qiuhao Zeng, Ruiyi Fang, Yan Yi Li, Yan Sun, Boyu Wang, and Pingzhao Hu. Scaling-aware adapter for structure-grounded llm reasoning. In *Proceedings of the 43nd International Conference on Machine Learning*, 2026.

[22] Haitao Mao, Zhikai Chen, Wei Jin, Haoyu Han, Yao Ma, Tong Zhao, Neil Shah, and Jiliang Tang. Demystifying structural disparity in graph neural networks: Can one size fit all? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[23] Xin Zheng, Yi Wang, Yixin Liu, Ming Li, Miao Zhang, Di Jin, Philip S Yu, and Shirui Pan. Graph neural networks for graphs with heterophily: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2026.

[24] Fengyu Yan, Xiaobao Wang, Dongxiao He, Longbiao Wang, Jianwu Dang, and Di Jin. Hetergp: Bridging heterogeneity in graph neural networks with multi-view prompting. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

[25] Ruiyi Fang, Shuo Wang, Ruizhi Pu, Qiuhao Zeng, Hao Zheng, Ziyan Wang, Jiale Cai, Zhimin Mei, Song Tang, Charles Ling, et al. Graph domain adaptation via homophily-agnostic reconstructing structure. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2026.

[26] Ruiyi Fang, Bingheng Li, Jingyu Zhao, Ruizhi Pu, Qiuhao Zeng, Gezheng Xu, Charles Ling, and Boyu Wang. Homophily enhanced graph domain adaptation. *arXiv preprint arXiv:2505.20089*, 2025.

[27] Jincheng Huang, Lun Du, Xu Chen, Qiang Fu, Shi Han, and Dongmei Zhang. Robust mid-pass filtering graph convolutional networks. In *The ACM Web Conference (WWW)*, 2023.

[28] Xuanting Xie, Wenyu Chen, and Zhao Kang. Robust graph structure learning under heterophily. *Neural Networks*, 185:107206, 2025.

[29] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and S Yu Philip. Graph self-supervised learning: A survey. *IEEE transactions on knowledge and data engineering*, 2022.

[30] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2020.

[31] Yaochen Xie, Zhao Xu, Jingtun Zhang, Zhengyang Wang, and Shuiwang Ji. Self-supervised learning of graph neural networks: A unified review. *IEEE transactions on pattern analysis and machine intelligence*, 2022.

[32] Ruiyi Fang, Liangjian Wen, Zhao Kang, and Jianzhuang Liu. Structure-preserving graph representation learning. In *IEEE International Conference on Data Mining (ICDM)*, 2022.

[33] Jiaming Zhuo, Can Cui, Kun Fu, Bingxin Niu, Dongxiao He, Chuan Wang, Yuanfang Guo, Zhen Wang, Xiaochun Cao, and Liang Yang. Graph contrastive learning reimagined: Exploring universality. In *The ACM Web Conference (WWW)*, 2024.

[34] Jiaming Zhuo, Feiyang Qin, Can Cui, Kun Fu, Bingxin Niu, Mengzhu Wang, Yuanfang Guo, Chuan Wang, Zhen Wang, Xiaochun Cao, and Liang Yang. Improving graph contrastive learning via adaptive positive sampling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[35] Jiaming Zhuo, Yintong Lu, Hui Ning, Kun Fu, Bingxin Niu, Dongxiao He, Chuan Wang, Yuanfang Guo, Zhen Wang, Xiaochun Cao, and Liang Yang. Unified graph augmentations for generalized contrastive learning on graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[36] Liang Yang, Zhenna Li, Jiaming Zhuo, Jing Liu, Ziyi Ma, Chuan Wang, Zhen Wang, and Xiaochun Cao. Graph contrastive learning with joint spectral augmentation of attribute and topology. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

[37] Liang Yang, Yukun Cai, Hui Ning, Jiaming Zhuo, Di Jin, Ziyi Ma, Yuanfang Guo, Chuan Wang, and Zhen Wang. Universal graph self-contrastive learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2025.

[38] Wei Chen, Xingyu Guo, Shuang Li, Zhao Zhang, Yan Zhong, Fuzhen Zhuang, and Deqing Wang. Learning adaptive distribution alignment with neural characteristic function for graph domain adaptation. In *International Conference on Learning Representations (ICLR)*, 2026.

[39] Wei Chen, Xingyu Guo, Shuang Li, Yan Zhong, Zhao Zhang, Fuzhen Zhuang, Hongrui Liu, Libang Zhang, Guo Ye, and Huimei He. Learning structure-semantic evolution trajectories for graph domain adaptation. In *International Conference on Learning Representations (ICLR)*, 2026.

[40] Xuanting Xie, Bingheng Li, Erlin Pan, Xinyi Wang, Keren He, Wenyu Chen, and Zhao Kang. Aggregation-aware mlp: An unsupervised approach for graph message-passing. *Pattern Recognition*, page 113877, 2026.

[41] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. GCC: Graph contrastive coding for graph neural network pre-training. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2020.

[42] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. GPT-GNN: Generative pre-training of graph neural networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2020.

[43] Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. GPPT: Graph pre-training and prompt tuning to generalize graph neural networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2022.

[44] Yuanfu Lu, Xunqiang Jiang, Yuan Fang, and Chuan Shi. Learning to pre-train graph neural networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[45] Jiarong Xu, Renhong Huang, Xin Jiang, Yuxuan Cao, Carl Yang, Chunping Wang, and Yang Yang. Better with less: A data-active perspective on pre-training graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[46] Xunqiang Jiang, Yuanfu Lu, Yuan Fang, and Chuan Shi. Contrastive pre-training of gnns on heterogeneous graphs. In *ACM International Conference on Information and Knowledge Management (CIKM)*, 2021.

[47] Yaming Yang, Ziyu Guan, Zhe Wang, Wei Zhao, Cai Xu, Weigang Lu, and Jianbin Huang. Self-supervised heterogeneous graph pre-training based on structural clustering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[48] Ke-Jia Chen, Jiajun Zhang, Linpu Jiang, Yunyun Wang, and Yuxuan Dai. Pre-training on dynamic graph neural networks. *Neurocomputing*, 2022.

[49] Yuxuan Cao, Jiarong Xu, Carl Yang, Jiaan Wang, Yunchao Zhang, Chunping Wang, Lei Chen, and Yang Yang. When to pre-train graph neural networks? from data generation perspective! In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2023.

[50] Jun Yin, Chaozhuo Li, Hao Yan, Jianxun Lian, and Senzhang Wang. Train once and explain everywhere: Pre-training interpretable graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[51] Jiaming Zhuo, Yuwei Liu, Yintong Lu, Ziyi Ma, Kun Fu, Chuan Wang, Yuanfang Guo, Zhen Wang, Xiaochun Cao, and Liang Yang. Dualformer: Dual graph transformer. In *International Conference on Learning Representations (ICLR)*, 2025.

[52] Jiaming Zhuo, Ziyi Ma, Kun Fu, Di Jin, Chuan Wang, Zhen Wang, Xiaochun Cao, Huazhu Fu, and Liang Yang. Rethinking graph transformers as graph signal denoisers: The role of block-diagonal priors. In *International Conference on Machine Learning (ICML)*, 2026.

[53] Jiaming Zhuo, Ziyi Ma, Yintong Lu, Yuwei Liu, Kun Fu, Di Jin, Chuan Wang, Wenning Wu, Zhen Wang, Xiaochun Cao, and Liang Yang. A closer look at graph transformers: Cross-aggregation and beyond. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[54] Qian Huang, Hongyu Ren, and Jure Leskovec. Few-shot relational reasoning via connection subgraph pretraining. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[55] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Chee-Kong Lee. Motif-based graph self-supervised learning for molecular property prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[56] Junwei Gong, Xiao Shen, Zhihao Chen, Shirui Pan, Xiao Wang, and Xi Zhou. negmix: Negative mixup for ood generalization in open-set node classification. In *The ACM Web Conference (WWW)*, 2026.

[57] Jiaming Zhuo, Can Cui, Kun Fu, Bingxin Niu, Dongxiao He, Yuanfang Guo, Zhen Wang, Chuan Wang, Xiaochun Cao, and Liang Yang. Propagation is all you need: A new framework for representation learning and classifier training on graphs. In *ACM International Conference on Multimedia (ACM MM)*, 2023.

[58] Ruiyi Fang, Changhong Chen, Qiang Wang, Charles Ling, Boyu Wang, and Hong Zhang. Transfer learning with graph neural networks to predict polymer solubility parameters. *npj Computational Materials*, 2026.

[59] Xiao Shen, Chuanyun Lin, Zhihao Chen, Junwei Gong, and Xi Zhou. Prototypical multi-source graph contrastive domain adaptation. *Pattern Recognition*, page 114312, 2026.

[60] Xiao Shen, Zhihao Chen, Shirui Pan, Shuang Zhou, Laurence T Yang, and Xi Zhou. Open-set cross-network node classification via unknown-excluded adversarial graph domain alignment. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

[61] Ruiyi Fang, Bingheng Li, Qiuhao Zeng, Nima Hosseini Dashtbayaz, Ruizhi Pu, Charles Ling, Boyu Wang, et al. On the benefits of attribute-driven graph domain adaptation. In *International Conference on Learning Representations (ICLR)*, 2025.

[62] Ruiyi Fang, Jingyu Zhao, Shuo Wang, Ruizhi Pu, Bingheng Li, Jiale Cai, Zhihao Li, Zihao Jing, Jian Zhu, Song Tang, et al. Saga: Structural aggregation guided alignment with dynamic view and neighborhood order selection for multiview graph domain adaptation. In *International Conference on Learning Representations (ICLR)*, 2026.

[63] Liang Yang, Xin Chen, Jiaming Zhuo, Di Jin, Chuan Wang, Xiaochun Cao, Zhen Wang, and Yuanfang Guo. Disentangled graph spectral domain adaptation. In *International Conference on Machine Learning (ICML)*, 2025.

[64] Luzhi Wang, Xuanshuo Fu, He Zhang, Chuang Liu, Xiaobao Wang, and Hongbo Liu. From subtle to significant: Prompt-driven self-improving optimization in test-time graph ood detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 15851–15859, 2026.

[65] Luzhi Wang, Dongxiao He, He Zhang, Yixin Liu, Wenjie Wang, Shirui Pan, Di Jin, and Tat-Seng Chua. Goodat: Towards test-time graph out-of-distribution detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[66] Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. One for all: Towards training one graph model for all classification tasks. In *International Conference on Learning Representations (ICLR)*, 2024.

[67] Zixuan Yi, Iadh Ounis, and Craig Macdonald. Contrastive graph prompt-tuning for cross-domain recommendation. *ACM Transactions on Information Systems*, 2023.

[68] Wenzhuo Tang, Haitao Mao, Danial Dervovic, Ivan Brugere, Saumitra Mishra, Yuying Xie, and Jiliang Tang. Cross-domain graph data scaling: A showcase with diffusion models. *arXiv preprint arXiv:2406.01899*, 2024.

[69] Yufei He, Yuan Sui, Xiaoxin He, and Bryan Hooi. UniGraph: Learning a unified cross-domain foundation model for text-attributed graphs. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2025.

[70] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. HiGPT: Heterogeneous graph language model. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2024.

[71] Yanchao Tan, Zihao Zhou, Hang Lv, Weiming Liu, and Carl Yang. Walklm: A uniform language model fine-tuning framework for attributed graph embedding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[72] Jiying Zhang, Xi Xiao, Long-Kai Huang, Yu Rong, and Yatao Bian. Fine-tuning graph neural networks via graph topology induced optimal transport. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.

[73] Yifei Sun, Qi Zhu, Yang Yang, Chunping Wang, Tianyu Fan, Jiajun Zhu, and Lei Chen. Fine-tuning graph neural networks by preserving graph generative patterns. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

[74] Wang Zhili, Di Shimin, Chen Lei, and Zhou Xiaofang. Search to fine-tune pre-trained graph neural networks for graph-level tasks. In *IEEE International Conference on Data Engineering (ICDE)*, 2024.

[75] Yuchen Yan, Peiyan Zhang, Zheng Fang, and Qingqing Long. Inductive graph alignment prompt: Bridging the gap between graph pre-training and inductive fine-tuning from spectral perspective. In *The ACM Web Conference (WWW)*, 2024.

[76] Wangyang Ying, Dongjie Wang, Xuanming Hu, Yuanchun Zhou, Charu C Aggarwal, and Yanjie Fu. Unsupervised generative feature transformation via graph contrastive pre-training and multi-objective fine-tuning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2024.

[77] Renhong Huang, Jiarong Xu, Xin Jiang, Chenglu Pan, Zhiming Yang, Chunping Wang, and Yang Yang. Measuring task similarity and its implication in fine-tuning graph neural networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

[78] Bo Jiang, Hao Wu, Beibei Wang, Jin Tang, and Bin Luo. Reliable and compact graph fine-tuning via graphsparse prompting. *arXiv preprint arXiv:2410.21749*, 2024.

[79] 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.

[80] Jincheng Huang, Jie Xu, Xiaoshuang Shi, Ping Hu, Lei Feng, and Xiaofeng Zhu. The final layer holds the key: A unified and efficient gnn calibration framework. *arXiv preprint arXiv:2505.11335*, 2025.

[81] Qi Zhu, Da Zheng, Xiang Song, Shichang Zhang, Bowen Jin, Yizhou Sun, and George Karypis. Parameter-efficient tuning large language models for graph representation learning. *arXiv preprint arXiv:2404.18271*, 2024.

[82] Shiyu Tian, Yangyang Luo, Tianze Xu, Caixia Yuan, Huixing Jiang, Chen Wei, and Xiaojie Wang. KG-Adapter: Enabling knowledge graph integration in large language models through parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics (ACL Findings)*, 2024.

[83] Anchun Gui, Jinqiang Ye, and Han Xiao. G-Adapter: Towards structure-aware parameter-efficient transfer learning for graph transformer networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

[84] Rui Xue, Xipeng Shen, Ruozhou Yu, and Xiaorui Liu. Efficient end-to-end language model fine-tuning on graphs. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2025.

[85] Mingkai Lin, Wenzhong Li, Xiaobin Hong, and Sanglu Lu. Scalable multi-source pre-training for graph neural networks. In *ACM International Conference on Multimedia (ACM MM)*, 2024.

[86] Yun Zhu, Yaoke Wang, Haizhou Shi, and Siliang Tang. Efficient tuning and inference for large language models on textual graphs. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2024.

[87] Yuhao Yang, Lianghao Xia, Da Luo, Kangyi Lin, and Chao Huang. GraphPro: Graph pre-training and prompt learning for recommendation. In *The ACM Web Conference (WWW)*, 2024.

[88] 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. *ACM Computing Surveys*, 2023.

[89] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International journal of computer vision*, 2022.

[90] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[91] Yu Du, Fangyun Wei, Zihe Zhang, Miaojing Shi, Yue Gao, and Guoqi Li. Learning to prompt for open-vocabulary object detection with vision-language model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[92] Di Jin, Luzhi Wang, Yizhen Zheng, Guojie Song, Fei Jiang, Xiang Li, Wei Lin, and Shirui Pan. Dual intent enhanced graph neural network for session-based new item recommendation. In *The ACM Web Conference (WWW)*, 2023.

[93] Xuanting Xie, Bingheng Li, Erlin Pan, Zhaochen Guo, Zhao Kang, and Wenyu Chen. One node one model: Featuring the missing-half for graph clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[94] Xuanting Xie, Bingheng Li, Erlin Pan, Rui Hou, Wenyu Chen, and Zhao Kang. Attention beyond neighborhoods: Reviving transformer for graph clustering. *IEEE Transactions on Circuits and Systems for Video Technology*, 2026.

[95] Xuanting Xie, Zhaochen Guo, Bingheng Li, Xingtong Yu, Zhifei Liao, Zhao Kang, and Yuan Fang. Clustering as reasoning: A k-means interpretation of chain-of-thought graph learning. In *International Conference on Machine Learning (ICML)*, 2026.

[96] Xuanting Xie, Erlin Pan, Zhao Kang, Wenyu Chen, and Bingheng Li. Provable filter for real-world graph clustering. *IEEE Transactions on Cybernetics*, 2026.

[97] Liang Yang, Yuwei Liu, Jiaming Zhuo, Di Jin, Chuan Wang, Zhen Wang, and Xiaochun Cao. Do we really need message passing in brain network modeling? In *International Conference on Machine Learning (ICML)*, 2025.

[98] Jiaming Zhuo, Shuai Zhai, Ziyi Ma, Kun Fu, Chuan Wang, Di Jin, Zhen Wang, Xiaochun Cao, Huazhu Fu, and Liang Yang. On the spectral unreachability of brain graph learning. In *International Conference on Machine Learning (ICML)*, 2026.

[99] Jie Xu, Na Zhao, Gang Niu, Masashi Sugiyama, and Xiaofeng Zhu. Robust multi-view learning via representation fusion of sample-level attention and alignment of simulated perturbation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

[100] Jie Xu, Shuo Chen, Yazhou Ren, Xiaoshuang Shi, Hengtao Shen, Gang Niu, and Xiaofeng Zhu. Self-weighted contrastive learning among multiple views for mitigating representation degeneration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[101] Jie Xu, Huayi Tang, Yazhou Ren, Liang Peng, Xiaofeng Zhu, and Lifang He. Multi-level feature learning for contrastive multi-view clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[102] Hourun Li, Yifan Wang, Zhiping Xiao, Jia Yang, Changling Zhou, Ming Zhang, and Wei Ju. Disco: graph-based disentangled contrastive learning for cold-start cross-domain recommendation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

[103] Yun Zhu, Jianhao Guo, and Siliang Tang. SGL-PT: A strong graph learner with graph prompt tuning. *arXiv preprint arXiv:2302.12449*, 2023.

[104] Reza Shirkavand and Heng Huang. Deep prompt tuning for graph transformers. *arXiv preprint arXiv:2309.10131*, 2023.

[105] Peiyan Zhang, Yuchen Yan, Xi Zhang, Liying Kang, Chaozhuo Li, Feiran Huang, Senzhang Wang, and Sunghun Kim. GPT4Rec: Graph prompt tuning for streaming recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2024.

[106] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. All in one: Multi-task prompting for graph neural networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2023.

[107] Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. GraphPrompt: Unifying pre-training and downstream tasks for graph neural networks. In *The ACM Web Conference (WWW)*, 2023.

[108] Chenghua Gong, Xiang Li, Jianxiang Yu, Yao Cheng, Jiaqi Tan, and Chengcheng Yu. Self-pro: A self-prompt and tuning framework for graph neural networks. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2024.

[109] Taoran Fang, Yunchao Zhang, Yang Yang, Chunping Wang, and Lei Chen. Universal prompt tuning for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[110] Qian Huang, Hongyu Ren, Peng Chen, Gregor Kržmanc, Daniel Zeng, Percy S Liang, and Jure Leskovec. PRODIGY: Enabling in-context learning over graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[111] Xingtong Yu, Yuan Fang, Zemin Liu, and Xinming Zhang. HGPrompt: Bridging homogeneous and heterogeneous graphs for few-shot prompt learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

[112] Chenyi Zi, Haihong Zhao, Xiangguo Sun, Yiqing Lin, Hong Cheng, and Jia Li. ProG: A graph prompt learning benchmark. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[113] Shuo Wang, Xiangyu Wang, Quanxin Wang, Bailin Wu, Bokui Wang, Shunyang Huang, Boyan Deng, Haonan Liu, Ruiyi Fang, Zhenxiang Xu, et al. The post-gcn decade revisited: Curvature-stratified evaluation of relational learning. *arXiv preprint arXiv:2606.06397*, 2026.

[114] Jincheng Huang, Yujie Mo, Ping Hu, Xiaoshuang Shi, Shangbo Yuan, Zeyu Zhang, and Xiaofeng Zhu. Exploring the role of node diversity in directed graph representation learning. In *IJCAI*, 2024.

[115] Xingtong Yu, Jie Zhang, Yuan Fang, and Renhe Jiang. Non-homophilic graph pre-training and prompt learning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2025.

[116] Jianan Zhao, Zhaocheng Zhu, Mikhail Galkin, Hesham Mostafa, Michael Bronstein, and Jian Tang. Fully-inductive node classification on arbitrary graphs. *arXiv preprint arXiv:2405.20445*, 2024.

[117] Yi Wang, Jitao Zhao, Dongxiao He, Jia Li, Yuxiao Huang, and Zhiyong Feng. Topology-aware feature sorting enables universal modeling on homophilic and heterophilic graphs. In *The ACM Web Conference (WWW)*, 2026.

[118] Xiao Shen, Mengqiu Shao, Shirui Pan, Laurence T Yang, and Xi Zhou. Domain-adaptive graph attention-supervised network for cross-network edge classification. *IEEE Transactions on Neural Networks and Learning Systems*, 35(12):17842–17855, 2023.

[119] Isabela Albuquerque, João Monteiro, Mohammad Darvishi, Tiago H Falk, and Ioannis Mitliagkas. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804*, 2019.

[120] Jincheng Huang, Yujie Mo, Xiaoshuang Shi, Lei Feng, and Xiaofeng Zhu. Enhancing the influence of labels on unlabeled nodes in graph convolutional networks. In *International Conference on Machine Learning (ICML)*, 2025.

[121] Jincheng Huang, Jialie Shen, Xiaoshuang Shi, and Xiaofeng Zhu. On which nodes does gcn fail? enhancing gcn from the node perspective. In *International Conference on Machine Learning (ICML)*, 2024.

[122] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 2010.

[123] Gilbert W Stewart. On the early history of the singular value decomposition. *SIAM Review*, 1993.

[124] Yilun Zheng, Sitao Luan, and Lihui Chen. What is missing for graph homophily? disentangling graph homophily for graph neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[125] Liang Yang, Mengzhe Li, Liyang Liu, Chuan Wang, Xiaochun Cao, Yuanfang Guo, et al. Diverse message passing for attribute with heterophily. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[126] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[127] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

[128] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[129] Dominique Beaini, Saro Passaro, Vincent Létourneau, Will Hamilton, Gabriele Corso, and Pietro Liò. Directional graph networks. In *International Conference on Machine Learning (ICML)*, 2021.

[130] Haorui Wang, Haoteng Yin, Muhan Zhang, and Pan Li. Equivariant and stable positional encoding for more powerful graph neural networks. *arXiv preprint arXiv:2203.00199*, 2022.

[131] Derek Lim, Joshua Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. *arXiv preprint arXiv:2202.13013*, 2022.

[132] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*, 2021.

[133] Erlin Pan and Zhao Kang. Beyond homophily: Reconstructing structure for graph-agnostic clustering. In *International Conference on Machine Learning (ICML)*, 2023.

[134] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 2000.

[135] C Lee Giles, Kurt D Bollacker, and Steve Lawrence. CiteSeer: An automatic citation indexing system. In *ACM Conference on Digital libraries*, 1998.

[136] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 2008.

[137] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

[138] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

[139] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[140] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengi. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[141] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations (ICLR)*, 2019.

[142] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[143] Teng Xiao, Zhengyu Chen, Zhimeng Guo, Zeyang Zhuang, and Suhang Wang. Decoupled self-supervised learning for graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[144] Teng Xiao, Huaisheng Zhu, Zhengyu Chen, and Suhang Wang. Simple and asymmetric graph contrastive learning without augmentations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[145] Luzhi Wang, Yizhen Zheng, Di Jin, Fuyi Li, Yongliang Qiao, and Shirui Pan. Contrastive graph similarity networks. *The ACM Web Conference (WWW)*, 2024.

[146] Yuhan Li, Peisong Wang, Zhixun Li, Jeffrey Xu Yu, and Jia Li. ZeroG: Investigating cross-dataset zero-shot transferability in graphs. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2024.

A Notations

Table 3: Summary of major notations used in this paper.

Notations	Descriptions
$G_i^{\mathcal{S}} = (\mathcal{V}_i^{\mathcal{S}}, \mathcal{E}_i^{\mathcal{S}}, \boldsymbol{X}_i^{\mathcal{S}})$	The i-th graph in source domains
$\boldsymbol{A}_i$	Adjacency matrix of the i-th graph
N	Number of source graphs
K	Number of source domains
n_i	Number of nodes of $G_i^{\mathcal{S}}$
d_i	raw feature dimension of $G_i^{\mathcal{S}}$
$G^{\mathcal{T}}$	target graphs
$D_k^{\mathcal{S}}$, $D^{\mathcal{T}}$	Distributions of the k-th source domain and the target domain
$\boldsymbol{y}^{\mathcal{T}}$, m	Label vector and number of labeled nodes in m-shot setting
$\mathcal{N}(u)$	The neighborhood of node u
$\boldsymbol{\Phi}$	The universal backbone
$\boldsymbol{\Phi}_{\text{ho}}$, $\boldsymbol{\Phi}_{\text{he}}$	Homophilic / heterophilic backbones
$\mathbb{G}$, $\mathbb{G}_{\text{ho}}$, $\mathbb{G}_{\text{he}}$	Unified space , homophilic unified space and heterophilic unified space
$\mathcal{H}$	Hypothesis class
$d_{\mathcal{H}}(\cdot,\cdot)$, γ_c, ρ_c, λ_c	$\mathcal{H}$-divergence; source-target distance, source diameter, ideal joint risk
α_i, α_t	Proportion of homophilic patterns in source / target domain
$\epsilon^{\mathcal{T}}(\cdot)$, $\epsilon^{\mathcal{S}}(\cdot)$, ε_{dis}	Target risk, source risk, and pattern-disentanglement error
$\tilde{\boldsymbol{X}}_i^{\mathcal{S}}$, $\hat{\boldsymbol{X}}_i^{\mathcal{S}}$	Dimension-aligned and semantic-aligned feature matrices
$\boldsymbol{t}_{D_j^{\mathcal{S}}}$	Domain token
d	unified feature dimension
$\boldsymbol{p}_v^{\text{Lap}}$, $\boldsymbol{p}_v^{\text{RW}}$, $\boldsymbol{x}_v^{\text{SE}}$	Laplacian PE, random-walk SE, and combined structural encoding
r	Order of structural encoding (top-r LapPE / r-step RW)
$\boldsymbol{z}_v$, $\boldsymbol{\phi}$	Disentanglement embedding and MLP parameters
$\boldsymbol{s}_v$, $\boldsymbol{\alpha}_v^{\text{ho}}$, $\boldsymbol{\alpha}_v^{\text{he}}$, t	Dissimilarity scores, contribution scores, and their temperature hyperparameter
$\boldsymbol{W}_i^{\text{ho}}$, $\boldsymbol{W}_i^{\text{he}}$	Disentangled homophilic / heterophilic weighted adjacency matrices
$\boldsymbol{\Theta}^{\text{ho}}$, $\boldsymbol{\Theta}^{\text{he}}$,	Low-pass / high-pass filter parameters
$\boldsymbol{\Theta}^{\star} = \{\boldsymbol{\Theta}_{\star}^{\text{Ho}}, \boldsymbol{\Theta}_{\star}^{\text{He}}\}$	Frozen parameters in backbones after pre-training
$\boldsymbol{H}_i^{\text{ho},(l)}$, $\boldsymbol{H}_i^{\text{he},(l)}$, L	Layer-l homophilic / heterophilic embeddings; number of filter layers
(u, v^+, v^-), $g(\cdot,\cdot)$, τ	Contrastive triple, pairwise similarity, temperature hyperparameter
λ; $\mathcal{L}_{\text{ho}}$, $\mathcal{L}_{\text{he}}$, $\mathcal{L}_{\text{pre}}$	Trade-off hyperparameter and homophilic / heterophilic / total pre-training losses
$\boldsymbol{\beta}^{\mathcal{T}}$, $\boldsymbol{\mathcal{P}}(\cdot)$	Learnable prompt weights and prompted feature transformation function
$\boldsymbol{\sigma} = [\sigma_{\text{Ho}}, \sigma_{\text{He}}]$	Pattern-mixing weights for target embeddings
$\boldsymbol{H}^{\mathcal{T},\text{ho}}$, $\boldsymbol{H}^{\mathcal{T},\text{he}}$, $\boldsymbol{H}^{\mathcal{T}}$	Target homophilic, heterophilic, and mixed representations
$\overline{\boldsymbol{h}}_y^{\mathcal{T}}$	Class-y mean representation
$\mathcal{L}_{\text{down}}$	downstream classification loss
E_{pre}, E_{down}	Number of pre-training and adaptation epochs

B Proofs

B.1 Preliminaries

Let $\mathcal{X}$ denote the node representation space and $\mathcal{Y} = \{1, \ldots, C\}$ the label space, where C is the number of classes. A domain is defined as a tuple $\langle D, f_D \rangle$, where D is a probability distribution over $\mathcal{X}$ and $f_D : \mathcal{X} \to \mathcal{Y}$ is a deterministic labeling function. Let $\mathcal{H}$ a set of candidate hypothesis. The risk of $h \in \mathcal{H}$ on this domain is defined as

$$\epsilon(h) := \mathbb{E}_{x \sim D} \left[\ell\big(h(x), f_D(x)\big) \right],$$

where the loss $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ quantifies the discrepancy between the prediction $h(x)$ and the true label $y = f_D(x)$ for a given instance (x, y).

For our multiclass classification task, we use the $\mathcal{H}\Delta\mathcal{H}$-divergence to measure distribution discrepancies through disagreements between classifiers. Given a hypothesis set $\mathcal{H}$ of mappings h, we define the disagreement as

$$\mathcal{H}\Delta\mathcal{H} := \{x \mapsto \mathbb{I}\{h(x) \neq h'(x)\} \mid h, h' \in \mathcal{H}\},$$

where $\mathbb{I}\{\cdot\}$ denotes the indicator function. For two distributions D and D' over $\mathcal{X}$, the $\mathcal{H}\Delta\mathcal{H}$-divergence is defined as

$$d_{\mathcal{H}\Delta\mathcal{H}}(D, D') := 2 \sup_{h,h' \in \mathcal{H}} \left| \Pr_{x \sim D}[h(x) \neq h'(x)] - \Pr_{x \sim D'}[h(x) \neq h'(x)] \right|. \tag{15}$$

Although h and h' produce multiclass predictions, their disagreement is binary, making this definition applicable to our setting.

B.2 Proof of Theorem 1

Our proof builds on the convex-hull domain-generalization bound of Albuquerque et al. [119] and the classical $\mathcal{H}$-divergence bound of Ben-David et al. [122], applied separately in the homophilic and heterophilic representation spaces. We first restate lemma 1, then give the proof.

Lemma 1 (Upper bound on the target-domain risk for a single encoder [119]). *Let $\mathcal{H}$ be a hypothesis set. Given source distributions $\{D_i^{\mathcal{S}}\}_{i=1}^K$ and a target distribution $D^{\mathcal{T}}$ on a shared input space, define*

$$\gamma := \min_{\pi \in \Delta_{K-1}} d_{\mathcal{H}\Delta\mathcal{H}}\left(D^{\mathcal{T}}, \sum_{i=1}^K \pi_i D_i^{\mathcal{S}}\right), \quad \rho := \sup_{\pi, \tilde{\pi} \in \Delta_{K-1}} d_{\mathcal{H}\Delta\mathcal{H}}\left(\sum_{i=1}^K \pi_i D_i^{\mathcal{S}}, \sum_{i=1}^K \tilde{\pi}_i D_i^{\mathcal{S}}\right).$$

Let π^* be a minimizer defining γ, and let $\lambda := \inf_{g \in \mathcal{H}} \left[\epsilon^{\mathcal{T}}(g) + \sum_{i=1}^K \pi_i^* \epsilon_i^{\mathcal{S}}(g)\right]$ which denotes the ideal joint risk between the target and its closets source mixture.

Then, for any $h \in \mathcal{H}$,

$$\epsilon^{\mathcal{T}}(h) \leq \sum_{i=1}^K \pi_i^* \epsilon_i^{\mathcal{S}}(h) + \frac{\gamma + \rho}{2} + \lambda. \tag{16}$$

Proof of Lemma 1 Let $\bar{D}^{\mathcal{S}} = \sum_{i=1}^K \pi_i^* D_i^{\mathcal{S}}$. Its risk satisfies $\bar{\epsilon}^{\mathcal{S}}(h) = \sum_{i=1}^K \pi_i^* \epsilon_i^{\mathcal{S}}(h)$. For a Domain distribution D, define the disagreement probability $\mathrm{dis}_D(h, g) := \Pr_{x \sim D}[h(x) \neq g(x)]$. By the definition of the $\mathcal{H}\Delta\mathcal{H}$-divergence, for any $h, g \in \mathcal{H}$ and each source domain i,

$$\left|\mathrm{dis}_{D^{\mathcal{T}}}(h, g) - \mathrm{dis}_{D_i^{\mathcal{S}}}(h, g)\right| \leq \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(D^{\mathcal{T}}, D_i^{\mathcal{S}}).$$

Then, we have

$$\begin{aligned}
\mathrm{dis}_{D^{\mathcal{T}}}(h, g) &\leq \mathrm{dis}_{D_i^{\mathcal{S}}}(h, g) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(D^{\mathcal{T}}, D_i^{\mathcal{S}}) \\
&\leq \mathrm{dis}_{D_i^{\mathcal{S}}}(h, g) + \frac{1}{2}(d_{\mathcal{H}\Delta\mathcal{H}}(D^{\mathcal{T}}, \bar{D}^{\mathcal{S}})) + d_{\mathcal{H}\Delta\mathcal{H}}(\bar{D}^{\mathcal{S}}, D_i^{\mathcal{S}})) \\
&\leq \mathrm{dis}_{D_i^{\mathcal{S}}}(h, g) + \frac{1}{2}(\gamma + \rho).
\end{aligned}$$

Using the pointwise inequalities $\mathbb{I}\{h(x) \neq y\} \leq \mathbb{I}\{h(x) \neq g(x)\} + \mathbb{I}\{g(x) \neq y\}$ and $\mathbb{I}\{h(x) \neq g(x)\} \leq \mathbb{I}\{h(x) \neq y\} + \mathbb{I}\{g(x) \neq y\}$, which hold for multiclass labels, we obtain

$$\begin{aligned}
\epsilon^{\mathcal{T}}(h) &= \mathbb{E}_{(x,y) \sim P^{\mathcal{T}}}\left[\mathbb{I}\{h(x) \neq y\}\right] \\
&\leq \mathbb{E}_{(x,y) \sim P^{\mathcal{T}}}\left[\mathbb{I}\{h(x) \neq g(x)\}\right] + \mathbb{E}_{(x,y) \sim P^{\mathcal{T}}}\left[\mathbb{I}\{g(x) \neq y\}\right] \\
&= \mathrm{dis}_{D^{\mathcal{T}}}(h, g) + \epsilon^{\mathcal{T}}(g) \\
&\leq \epsilon^{\mathcal{T}}(g) + \mathrm{dis}_{D_i^{\mathcal{S}}}(h, g) + \frac{1}{2}(\gamma + \rho) \\
&\leq \epsilon^{\mathcal{T}}(g) + \bar{\epsilon}^{\mathcal{S}}(h) + \bar{\epsilon}^{\mathcal{S}}(g) + \frac{1}{2}(\gamma + \rho) \\
&\leq \epsilon^{\mathcal{T}}(g) + \sum_{i=1}^K \pi_i^* \epsilon_i^{\mathcal{S}}(h) + \sum_{i=1}^K \pi_i^* \epsilon_i^{\mathcal{S}}(g) + \frac{1}{2}(\gamma + \rho).
\end{aligned}$$

Taking the infimum over $g \in \mathcal{H}$ yields

$$\epsilon^{\mathcal{T}}(h) \leq \sum_{i=1}^{K} \pi_i^* \epsilon_i^{\mathcal{S}}(h) + \frac{\gamma + \rho}{2} + \lambda.$$

. □

Proof of Theorem 1. We first apply Lemma 1 within each pattern-specific space and then combine the resulting bounds using the assumed target-risk decomposition.

By the assumption in Theorem 1, the predictor h satisfies

$$\epsilon^{\mathcal{T}}(h) \leq \sum_{c \in \{\mathrm{ho}, \mathrm{he}\}} \alpha_t^c \, \epsilon_c^{\mathcal{T}}(h_c) + \varepsilon_{\mathrm{dis}}, \tag{17}$$

where $\alpha_t^{\mathrm{ho}} = \alpha_t$ and $\alpha_t^{\mathrm{he}} = 1 - \alpha_t$ are nonnegative and sum to one.

Fix a pattern $c \in \{\mathrm{ho}, \mathrm{he}\}$. For the fixed encoder $\mathbf{\Phi}_c$, let $\{D_{i,c}^{\mathcal{S}}\}_{i=1}^{K}$ and $D_c^{\mathcal{T}}$ denote the pattern-conditional distributions on the shared space $\mathcal{X}_c$, and let $\mathcal{H}_c$ be the corresponding hypothesis set. Define the closest source mixture as

$$\bar{D}_c^{\mathcal{S}} := \sum_{i=1}^{K} \pi_i^{c,*} D_{i,c}^{\mathcal{S}}, \qquad \pi^{c,*} \in \operatorname*{arg\,min}_{\pi^c \in \Delta_{K-1}} d_{\mathcal{H}_c \Delta \mathcal{H}_c} \left(D_c^{\mathcal{T}}, \sum_{i=1}^{K} \pi_i^c D_{i,c}^{\mathcal{S}} \right).$$

The corresponding ideal joint risk is

$$\lambda_c := \inf_{g_c \in \mathcal{H}_c} \left[\epsilon_c^{\mathcal{T}}(g_c) + \sum_{i=1}^{K} \pi_i^{c,*} \epsilon_{i,c}^{\mathcal{S}}(g_c) \right].$$

Applying Lemma 1 to these distributions with hypothesis set $\mathcal{H}_c$, the quantities γ and ρ in the lemma become γ_c and ρ_c, respectively. Thus, for every $h_c \in \mathcal{H}_c$,

$$\epsilon_c^{\mathcal{T}}(h_c) \leq \sum_{i=1}^{K} \pi_i^{c,*} \epsilon_{i,c}^{\mathcal{S}}(h_c) + \frac{\gamma_c + \rho_c}{2} + \lambda_c. \tag{18}$$

Multiplying (18) by $\alpha_t^c \geq 0$, summing over the two patterns, and substituting into (17), we obtain

$$\begin{aligned} \epsilon^{\mathcal{T}}(h) &\leq \sum_{c \in \{\mathrm{ho}, \mathrm{he}\}} \alpha_t^c \, \epsilon_c^{\mathcal{T}}(h_c) + \varepsilon_{\mathrm{dis}} \\ &\leq \sum_{c \in \{\mathrm{ho}, \mathrm{he}\}} \alpha_t^c \left[\sum_{i=1}^{K} \pi_i^{c,*} \epsilon_{i,c}^{\mathcal{S}}(h_c) + \frac{\gamma_c + \rho_c}{2} + \lambda_c \right] + \varepsilon_{\mathrm{dis}}. \end{aligned}$$

This proves the stated bound. □

C Algorithm and Complexity

C.1 Algorithm

We summarize the overall procedures of pre-training and downstream adaptation in Algorithms 1 and 2.

C.2 Complexity Analysis

In the **pre-training phase**, each of the N source graphs $\{G_i^{\mathcal{S}}\}_{i=1}^{N}$ with $G_i^{\mathcal{S}} = (\mathcal{V}_i^{\mathcal{S}}, \mathcal{E}_i^{\mathcal{S}}, \boldsymbol{X}_i^{\mathcal{S}})$ with $n_i = |\mathcal{V}_i^{\mathcal{S}}|$ and raw feature dimension d_i is processed independently. The model first aligns node features via truncated SVD in (4), which reduces the input dimension from d_i to d at a cost of $\mathcal{O}(|n_i| \cdot d \cdot \log d_i)$. The semantic alignment with the domain token $\boldsymbol{t}_{D_j^{\mathcal{S}}}$ in (5) then takes $\mathcal{O}(n_i \cdot d)$. Next, the structural encoding $\boldsymbol{x}_v^{\mathrm{SE}}$ in (6), which combines top-r Laplacian positional encoding

Algorithm 1: Pre-training of HHGFM.

Input: N source graphs $\{G_i^{\mathcal{S}}\} \in \mathcal{G}^{\mathcal{S}}$ from multi-domain $\{D_k^{\mathcal{S}}\}_{k=1}^{K} \in \mathcal{D}^{\mathcal{S}}$, each of the graph has its feature matrix $\boldsymbol{X}_i \in \mathbb{R}^{n_i \times d_i}$ and adjacency matrix $\boldsymbol{A}_i \in \mathbb{R}^{n_i \times n_i}$; Number of the pre-training epochs E_{pre}; Temperature parameter γ for computing homophily and heterophily score; Number of layers of high-pass and low-pass filters L; Hyperparameter λ for trading- off homophilic and heterophilic losses.

Output: Pre-trained dual encoders $\boldsymbol{\Phi}^{\text{ho}}_{\boldsymbol{\Theta}^{\text{ho}}_{\star}}(\cdot)$ and $\boldsymbol{\Phi}^{\text{he}}_{\boldsymbol{\Theta}^{\text{he}}_{\star}}(\cdot)$; Domain alignment tokens vector $\boldsymbol{t}^{\star}_{D^{\mathcal{S}}}$.

Initialize all parameters randomly, including:

Domain alignment tokens vector $\boldsymbol{t}_{D^{\mathcal{S}}}$; Learnable parameters $\boldsymbol{\phi}$ in $\text{MLP}(\cdot)$ in the disentangle step; parameters $\boldsymbol{\Theta}^{\text{Ho}}$ and $\boldsymbol{\Theta}^{\text{He}}$ ($\boldsymbol{\Theta} = \{\boldsymbol{\Theta}^{\text{Ho}}, \boldsymbol{\Theta}^{\text{He}}\}$in dual backbones.

for $e = 1, 2, \cdots, E_{\text{pre}}$ **do**

```
// domain feature alignment
```

Align feature dimensions: $\boldsymbol{X}_i^{\mathcal{S}} \leftarrow$ (4) for each $G_i^{\mathcal{S}}$;

Align semantical features: $\hat{\boldsymbol{X}}_i^{\mathcal{S}} \leftarrow$ (5) for each $G_i^{\mathcal{S}}$.

```
// disentangle homophilic and heterophilic patterns
```

Compute homophilic and heterophilic contribution: $\boldsymbol{\alpha}_v^{\text{ho}} \leftarrow$ (6)-(7) and $\boldsymbol{\alpha}_v^{\text{he}} \leftarrow 1 - \boldsymbol{\alpha}_v^{\text{ho}}$ for each node in $G_i^{\mathcal{S}}$;

disentangled homophilic and heterophilic subgraphs: $\boldsymbol{W}_i^{\text{ho}} \leftarrow (8)$ and $\boldsymbol{W}_i^{\text{he}} = \boldsymbol{A} - \boldsymbol{W}_i^{\text{ho}}$ for each $G_i^{\mathcal{S}}$.

```
// high-pass and low-pass graph filters
```

for $l = 1, 2, \cdots, L$ **do**

Updated homophilic embeddings: $\boldsymbol{H}_i^{\mathcal{S},(l)} \leftarrow$ Eq. (9) for $G_i^{\mathcal{S}}$;

Updated heterophilic embeddings: $\boldsymbol{H}_i^{\mathcal{S},(l)} \leftarrow$ Eq. (10) for $G_i^{\mathcal{S}}$;

```
// self-supervised pre-training
```

Calculate the pre-training loss: $\mathcal{L}_{\text{pre}} \leftarrow$ Eq. (11);

Update $\boldsymbol{\Theta}$ and $\boldsymbol{t}_{D^{\mathcal{S}}}$ by minimizing $\mathcal{L}_{\text{pre}}$ and back-propagation;

$\boldsymbol{p}_v^{\text{Lap}}$, r-step random-walk encoding $\boldsymbol{p}_v^{\text{RW}}$, and the degree centrality $\boldsymbol{D}_{vv}$, costs $\mathcal{O}(r \cdot |\mathcal{E}_i^{\mathcal{S}}|)$ when implemented via Lanczos iteration and sparse matrix multiplication. The shared $\text{MLP}(\cdot; \boldsymbol{\phi})$ then maps the concatenated features to embeddings $\boldsymbol{z}_v$ at $\mathcal{O}(|n_i| \cdot d^2)$, after which the per-edge homophilic and heterophilic contribution scores $\boldsymbol{\alpha}_v^{\text{ho}}, \boldsymbol{\alpha}_v^{\text{he}}$ in (7) are obtained in $\mathcal{O}(|\mathcal{E}_i^{\mathcal{S}}| \cdot d)$, yielding the disentangled weighted subgraphs $\boldsymbol{W}_i^{\text{ho}}, \boldsymbol{W}_i^{\text{he}}$ in (8) at $\mathcal{O}(|\mathcal{E}_i^{\mathcal{S}}|)$. The dual L-layer low-pass and high-pass filters $\boldsymbol{\Phi}^{\text{ho}}_{\boldsymbol{\Theta}^{\text{ho}}}$ and $\boldsymbol{\Phi}^{\text{he}}_{\boldsymbol{\Theta}^{\text{he}}}$ in (9)–(10) contribute an additional $\mathcal{O}(L \cdot n_i \cdot d^2 + L \cdot |\mathcal{E}_i^{\mathcal{S}}| \cdot d)$. Finally, the contrastive link-prediction loss $\mathcal{L}_{\text{pre}}$ in (11) over the sampled triples (u, v^+, v^-) adds $\mathcal{O}(|\mathcal{E}_i^{\mathcal{S}}| \cdot d)$. Since r and $\log d_i$ are typically much smaller than $L \cdot d$, the lower-order terms are absorbed by the L-layer message-passing cost. Therefore, the total pre-training complexity across all source graphs for each epoch simplifies to $\mathcal{O}\Big(L \cdot \sum_{i=1}^{N} \big(|\mathcal{V}_i^{\mathcal{S}}| \cdot d^2 + |\mathcal{E}_i^{\mathcal{S}}| \cdot d\big)\Big)$.

In the **downstream phase**, given the target graph $G^{\mathcal{T}} = (\mathcal{V}^{\mathcal{T}}, \mathcal{E}^{\mathcal{T}}, \boldsymbol{X}^{\mathcal{T}})$ with $n_{\mathcal{T}} = |\mathcal{V}^{\mathcal{T}}|$ and raw feature dimension $d_{\mathcal{T}}$, the model first aligns node features via truncated SVD in (4) at a cost of $\mathcal{O}(n_{\mathcal{T}} \cdot d \cdot \log d_{\mathcal{T}})$. The prompt-tuning step that fuses the learnable prompt $\boldsymbol{\beta}^{\mathcal{T}}$ with the frozen source tokens $\boldsymbol{t}^{\star}_{D^{\mathcal{S}}}$ and modulates $\tilde{\boldsymbol{X}}^{\mathcal{T}}$ then takes $\mathcal{O}(n_{\mathcal{T}} \cdot d)$. Next, the structural encoding $\boldsymbol{x}_v^{\text{SE}}$ in (6), which combines top-r Laplacian positional encoding $\boldsymbol{p}_v^{\text{Lap}}$, r-step random-walk encoding $\boldsymbol{p}_v^{\text{RW}}$, and the degree centrality $\boldsymbol{D}_{vv}$, costs $\mathcal{O}(r \cdot |\mathcal{E}^{\mathcal{T}}|)$. The shared $\text{MLP}(\cdot; \boldsymbol{\phi})$ maps the concatenated features to embeddings $\boldsymbol{z}_v$ at $\mathcal{O}(n_{\mathcal{T}} \cdot d^2)$, after which the per-edge homophilic and heterophilic contribution scores $\boldsymbol{\alpha}_v^{\text{ho}}, \boldsymbol{\alpha}_v^{\text{he}}$ in (7) are obtained in $\mathcal{O}(|\mathcal{E}^{\mathcal{T}}| \cdot d)$, yielding the disentangled weighted subgraphs $\boldsymbol{W}^{\text{ho}}, \boldsymbol{W}^{\text{he}}$ in (8) at $\mathcal{O}(|\mathcal{E}^{\mathcal{T}}|)$. The frozen dual L-layer low-pass and high-pass filters $\boldsymbol{\Phi}^{\text{ho}}_{\boldsymbol{\Theta}^{\text{ho}}_{\star}}$ and $\boldsymbol{\Phi}^{\text{he}}_{\boldsymbol{\Theta}^{\text{he}}_{\star}}$ in (9)–(10) contribute an additional $\mathcal{O}(L \cdot n_{\mathcal{T}} \cdot d^2 + L \cdot |\mathcal{E}^{\mathcal{T}}| \cdot d)$, followed by the lightweight pattern mixing $\boldsymbol{H}^{\mathcal{T}} = \sigma_{\text{Ho}} \boldsymbol{H}^{\mathcal{T},\text{ho}} + \sigma_{\text{He}} \boldsymbol{H}^{\mathcal{T},\text{he}}$ in (13) at $\mathcal{O}(n_{\mathcal{T}} \cdot d)$. Finally, the prototype-based downstream loss $\mathcal{L}_{\text{down}}$ in (14), which compares each training node with the $|Y^{\mathcal{T}}|$ class centroids, adds $\mathcal{O}(n_{\mathcal{T}} \cdot |Y^{\mathcal{T}}| \cdot d)$. Since r, $\log d_{\mathcal{T}}$, and $|Y^{\mathcal{T}}|$ are typically much smaller than

Algorithm 2: Downstream adaptation.

Input: Target graph $G^{\mathcal{T}} = (\mathcal{V}^{\mathcal{T}}, \mathcal{E}^{\mathcal{T}}, \boldsymbol{X}^{\mathcal{T}})$ with training nodes $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$ (node classification) or target graphs $\{G_i^{\mathcal{T}}\}$ with labels (graph classification); Pre-trained dual encoders $\boldsymbol{\Phi}^{\text{ho}}_{\boldsymbol{\Theta}^{\text{ho}}_{\star}}(\cdot)$ and $\boldsymbol{\Phi}^{\text{he}}_{\boldsymbol{\Theta}^{\text{he}}_{\star}}(\cdot)$; Frozen source domain tokens $\boldsymbol{t}^{\star}_{D^{\mathcal{S}}}$; Number of adaptation epochs E_{down}; Temperature parameter τ.

Output: Predicted labels for unlabeled nodes/graphs $\hat{\boldsymbol{y}}^{\mathcal{T}}$.

Initialize randomly, including:
Prompt parameter $\boldsymbol{\beta}^{\mathcal{T}}$; Pattern-mixing weights $\boldsymbol{\sigma} = [\sigma_{\text{Ho}}, \sigma_{\text{He}}]$.
for $e = 1, 2, \cdots, E_{\text{down}}$ **do**

```
// dimension alignment and prompt tuning
```
Align feature dimensions: $\tilde{\boldsymbol{X}}^{\mathcal{T}} \leftarrow (4)$;
Align semantical features: $\hat{\boldsymbol{X}}^{\mathcal{T}} \leftarrow (12)$;

```
// pattern disentanglement
```
Compute homophilic and heterophilic contribution: $\boldsymbol{\alpha}_v^{\text{ho}} \leftarrow (6)$-$(7)$ and $\boldsymbol{\alpha}_v^{\text{he}} \leftarrow 1 - \boldsymbol{\alpha}_v^{\text{ho}}$ for each node in $G^{\mathcal{T}}$;
disentangled homophilic and heterophilic subgraphs: $\boldsymbol{W}^{\text{ho}} \leftarrow (8)$ and $\boldsymbol{W}^{\text{he}} = \boldsymbol{A} - \boldsymbol{W}^{\text{ho}}$ for $G^{\mathcal{T}}$.

```
// Embedding encoding with frozen backbones
```
Encode homophilic embeddings: $\boldsymbol{H}^{\mathcal{T},\text{ho}} \leftarrow (9)$;
Encode heterophilic embeddings: $\boldsymbol{H}^{\mathcal{T},\text{he}} \leftarrow (10)$;
Mix representations: $\boldsymbol{H}^{\mathcal{T}} \leftarrow \sigma_{\text{Ho}}\boldsymbol{H}^{\mathcal{T},\text{ho}} + \sigma_{\text{He}}\boldsymbol{H}^{\mathcal{T},\text{he}}$, i.e., (13);

```
// downstream optimization
```
Compute downstream loss: $\mathcal{L}_{\text{down}}(\boldsymbol{\beta}^{\mathcal{T}}, \boldsymbol{\sigma}; \boldsymbol{H}^{\mathcal{T}}) \leftarrow (14)$;
Update $\boldsymbol{\beta}^{\mathcal{T}}$ and $\boldsymbol{\sigma}$ by minimizing $\mathcal{L}_{\text{down}}$ and back-propagation;

$L \cdot d$, the lower-order terms are absorbed by the L-layer message-passing cost. Therefore, the total downstream complexity for each epoch simplifies to $\mathcal{O}\big(L \cdot \big(|\mathcal{V}^{\mathcal{T}}| \cdot d^2 + |\mathcal{E}^{\mathcal{T}}| \cdot d\big)\big)$.

D Experiment Details

D.1 Details of Homophily Measurement

To probe whether a learned representation preserves homophilic and heterophilic information, we measure how much more similar each node is to its neighbors than to random non-neighbors in the embedding space. For a graph $G = (\mathcal{V}, \mathcal{E})$ with learned node embeddings $\{\boldsymbol{h}_v\}_{v\in\mathcal{V}}$, we first ℓ_2-normalize the embeddings and define the cosine similarity $\cos(\boldsymbol{h}_u, \boldsymbol{h}_v) = \boldsymbol{h}_u^{\top}\boldsymbol{h}_v/(\|\boldsymbol{h}_u\|_2\|\boldsymbol{h}_v\|_2)$. For each node $v \in \mathcal{V}$, we compute its average similarity to neighbors and to a random non-neighbor sample $\mathcal{R}(v) \subset \mathcal{V} \setminus (\mathcal{N}(v) \cup \{v\})$ with $|\mathcal{R}(v)| = R$:

$$\bar{s}_v^{\text{in}} = \frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} \cos(\boldsymbol{h}_v, \boldsymbol{h}_u), \qquad \bar{s}_v^{\text{out}} = \frac{1}{R}\sum_{u\in\mathcal{R}(v)} \cos(\boldsymbol{h}_v, \boldsymbol{h}_u). \tag{19}$$

The representation homophily ratio of node v is then

$$\eta_v \;=\; \text{sigmoid}\big(\kappa \cdot (\bar{s}_v^{\text{in}} - \bar{s}_v^{\text{out}})\big) \in (0, 1), \tag{20}$$

where $\kappa > 0$ is a scaling factor (we set $\kappa = 10$) used to spread the otherwise narrow gap $\bar{s}_v^{\text{in}} - \bar{s}_v^{\text{out}}$. We use $R = 50$ random non-neighbors per node.

D.2 Details of Datasets

The dataset statistics are summarized in Table 4.

To study multi-domain pre-training, we benchmark on **eight** graphs drawn from **three** different domains.

Table 4: Data statistics.

Dataset	Domain	# Nodes	# Edges	# Feature Dimensions	# Classes	Avg. # Deg.	Homophily ratio
Cora [134]	Citation	2,708	5,429	1,433	7	4.00	0.81
CiteSeer [135]	Citation	3,186	4,277	3,703	6	2.57	0.74
PubMed [136]	Citation	19,717	44,338	500	3	4.50	0.80
Photo [137]	Co-purchase	7,650	119,081	745	8	31.13	0.83
Computers [137]	Co-purchase	13,752	245,861	767	10	35.76	0.78
Cornell [138]	Web page	183	295	1,703	5	3.22	0.31
Texas [138]	Web page	183	309	1,703	5	3.38	0.11
Wisconsin [138]	Web page	251	499	1,703	5	3.98	0.21

Citation Domain. We use several standard citation-network benchmarks, where one graph shows citations between papers. Nodes correspond to papers, edges indicate citation links, and node labels denote the paper's research area.

- Cora [134]: A citation graph of 2,708 papers in machine learning, grouped into 7 topics. Node attributes are 1,433-dimensional sparse bag-of-words vectors, and the graph contains 5,429 citation edges.
- CiteSeer [135]: A network with 3,186 papers from 6 research categories. Each node is represented by a 3,703-dimensional word-occurrence feature vector, with 4,277 citation edges in total.
- PubMed [136]: A biomedical citation network of 19,717 diabetes-related articles labeled into 3 classes. Node features are 500-dimensional TF-IDF vectors, and the graph includes 44,338 citation edges.

Co-purchase Domain. We consider two Amazon co-purchase graphs, Amazon-Photo and Amazon-Computers [137]. In each graph, nodes represent products, edges indicate that two products are frequently bought together, node features are bag-of-words representations of product reviews, and labels correspond to product categories.

- Photo: A co-purchase graph consisting of products from the photo category, used for product-category prediction.
- Computers: A co-purchase graph consisting of computer-related products, following the same node, edge, feature, and label semantics.

Web Page Domain. We consider three web-page graphs, Cornell, Texas, and Wisconsin [138]. In these datasets, nodes represent web pages, edges correspond to hyperlinks between pages, node features are extracted from textual content, and labels denote the category of each page.

- Cornell [138]: A webpage graph collected from the Cornell University computer science department, where hyperlinks define the connectivity between pages.
- Texas [138]: A webpage graph collected from the Texas University computer science department, following the same hyperlink-based graph construction and node classification setting.
- Wisconsin [138]: A webpage graph collected from the Wisconsin University computer science department, where nodes are webpages and edges indicate hyperlinks between them.

D.3 Baseline Details

We compare Ours with **15** representative baselines from four groups: vanilla GNNs, self-supervised graph pre-training methods, prompt-based tuning methods, and multi-domain pre-training frameworks. These baselines cover both conventional graph learning and recent transfer-oriented paradigms.

Vanilla Graph Neural Networks. These methods are trained directly on downstream tasks without pre-training.

- GCN [139]: A classic neighborhood aggregation model based on spectral graph convolution. We use it as a backbone for node and graph classification.
- GAT [140]: An attention-based GNN that learns different weights for neighboring nodes during message passing. We evaluate it on both node-level and graph-level tasks.

Table 5: Accuracy (% ± std over 20 runs) of **five-shot classification**. Best results are shown in **bold**, and runner-up results are underlined. CR† = Cora, CS† = CiteSeer, PM† = PubMed; PT‡ = Photo, CP‡ = Computers; CN§ = Cornell, TX§ = Texas, WCS§ = Wisconsin. Datasets with the same superscript symbol are from the same domain.

Source	**Cross-Dataset**			**Cross-Domain**			
	CS† PM† PT‡ CP‡	CR† PM† PT‡ CP‡	CR† CS† PT‡ CN§	PT‡ CP‡ CN§ TX§	PT‡ CP‡ CN§ TX§	CR† CS† PT‡ CP‡	CR† CS† PT‡ CP‡
Model / Target	CR†	CS†	PM†	CR†	CS†	WCS§	CN§
Node Classification							
GCN [139]	31.84 ± 4.83	28.16 ± 4.52	40.28 ± 5.41	29.47 ± 5.26	25.72 ± 5.01	31.64 ± 5.54	30.28 ± 5.32
GAT [140]	32.47 ± 4.92	28.81 ± 4.61	41.12 ± 5.58	30.16 ± 5.41	26.31 ± 5.14	32.28 ± 5.76	31.07 ± 5.48
GCC [41]	37.26 ± 5.42	33.84 ± 5.28	44.63 ± 6.03	38.52 ± 5.84	32.81 ± 5.61	35.84 ± 6.08	34.41 ± 5.87
DGI [141]	38.14 ± 5.63	34.47 ± 5.37	45.26 ± 6.21	39.18 ± 6.04	33.84 ± 5.82	36.73 ± 6.16	35.26 ± 5.94
GraphCL [142]	42.62 ± 5.98	38.81 ± 5.74	47.84 ± 6.48	43.82 ± 6.26	36.68 ± 6.02	38.41 ± 6.38	36.94 ± 6.12
DSSL [143]	41.96 ± 5.87	37.07 ± 5.63	46.93 ± 6.37	42.96 ± 6.12	35.07 ± 5.91	37.68 ± 6.24	36.42 ± 6.04
GraphACL [144]	44.26 ± 5.74	40.12 ± 5.61	49.18 ± 6.21	45.74 ± 6.04	38.42 ± 5.86	39.86 ± 6.03	38.61 ± 5.88
GPPT [43]	43.62 ± 6.17	39.73 ± 5.96	48.64 ± 6.71	44.52 ± 6.41	37.48 ± 6.18	38.84 ± 6.31	37.46 ± 6.08
GraphPrompt [107]	48.84 ± 6.42	45.96 ± 6.21	52.13 ± 7.04	51.71 ± 6.78	43.36 ± 6.51	41.92 ± 6.74	40.57 ± 6.48
GPF [109]	50.06 ± 6.58	47.84 ± 6.34	53.48 ± 7.18	53.83 ± 6.94	45.18 ± 6.72	42.84 ± 6.91	41.52 ± 6.68
ProNoG [115]	53.12 ± 6.31	50.46 ± 6.08	55.76 ± 6.94	56.72 ± 6.74	48.84 ± 6.52	44.13 ± 6.72	42.81 ± 6.51
GCOPE [14]	55.18 ± 7.84	58.36 ± 4.91	56.12 ± 8.43	58.58 ± 4.37	52.62 ± 7.61	37.48 ± 5.93	47.82 ± 8.84
MDGPT [8]	57.62 ± 5.97	59.31 ± 8.92	58.36 ± 5.62	60.21 ± 8.24	55.48 ± 4.36	42.26 ± 8.42	48.68 ± 5.97
SAMGPT [12]	62.20 ± 10.90	**65.20 ± 3.83**	53.20 ± 9.99	65.80 ± 9.99	60.60 ± 7.77	43.60 ± 7.92	51.60 ± 7.47
BRIDGE [10]	59.84 ± 9.04	61.96 ± 6.08	57.73 ± 6.84	62.91 ± 4.62	57.52 ± 8.26	42.98 ± 6.41	49.13 ± 4.07
HHGFM (ours)	**63.18 ± 8.18**	64.36 ± 8.84	**61.82 ± 7.74**	**67.31 ± 8.46**	**62.14 ± 9.86**	**45.92 ± 8.91**	**52.38 ± 8.42**
Graph Classification							
GCN [139]	33.42 ± 5.12	29.86 ± 4.84	42.18 ± 5.73	31.84 ± 5.58	27.41 ± 5.26	33.84 ± 5.92	32.16 ± 5.71
GAT [140]	34.16 ± 5.28	30.52 ± 5.01	42.94 ± 5.92	32.48 ± 5.76	28.07 ± 5.43	34.52 ± 6.03	32.84 ± 5.86
GraphCL [142]	44.84 ± 6.12	40.73 ± 5.88	50.16 ± 6.74	45.72 ± 6.42	38.41 ± 6.18	40.62 ± 6.48	39.36 ± 6.24
DSSL [143]	43.92 ± 6.03	39.84 ± 5.74	49.26 ± 6.58	44.83 ± 6.28	37.56 ± 6.02	39.84 ± 6.31	38.72 ± 6.08
GraphACL [144]	46.31 ± 5.91	42.48 ± 5.67	51.72 ± 6.41	47.16 ± 6.14	39.82 ± 5.94	41.26 ± 6.08	40.18 ± 5.87
GraphPrompt [107]	52.41 ± 6.54	48.72 ± 6.28	55.84 ± 7.18	55.16 ± 6.92	46.73 ± 6.74	43.74 ± 6.92	42.31 ± 6.66
GPF [109]	54.12 ± 6.72	50.16 ± 6.43	57.21 ± 7.36	57.34 ± 7.08	48.41 ± 6.91	44.68 ± 7.03	43.18 ± 6.82
ProNoG [115]	57.46 ± 6.46	54.38 ± 6.21	59.82 ± 7.04	60.84 ± 6.83	51.62 ± 6.64	46.12 ± 6.81	44.57 ± 6.57
GCOPE [14]	59.16 ± 7.92	61.38 ± 5.08	60.42 ± 8.31	62.18 ± 4.62	55.74 ± 7.74	40.62 ± 6.08	49.84 ± 8.91
MDGPT [8]	61.74 ± 6.08	63.52 ± 9.04	62.84 ± 5.84	64.31 ± 8.42	58.62 ± 4.58	44.18 ± 8.53	50.76 ± 6.08
SAMGPT [12]	66.20 ± 8.84	**68.60 ± 3.74**	58.20 ± 9.76	69.40 ± 9.52	63.40 ± 7.62	46.80 ± 7.74	54.60 ± 7.31
BRIDGE [10]	63.84 ± 8.92	65.84 ± 6.21	61.42 ± 6.91	66.72 ± 4.84	60.18 ± 8.31	45.92 ± 6.52	52.84 ± 4.24
HHGFM (ours)	**67.41 ± 6.06**	67.32 ± 8.74	**64.26 ± 9.62**	**71.16 ± .34**	**64.12 ± 9.74**	**47.92 ± 8.64**	**55.36 ± 8.21**

Self-Supervised Graph Pre-training. These methods learn graph representations from unlabeled data through auxiliary objectives.

- GCC [41]: A contrastive pre-training method that learns transferable structural representations from sampled subgraphs. It is used for node classification.
- DGI [141]: A mutual-information-based method that contrasts node representations with graph-level summaries. It is used for node classification.
- GraphCL [142]: A contrastive learning framework based on graph augmentations such as node dropping and edge perturbation. It is evaluated on both node and graph classification.
- DSSL [143]: A self-supervised method that separates semantic and structural factors to better handle different neighborhood patterns. It is evaluated on both node and graph classification.
- GraphACL [144]: An asymmetric contrastive method designed for heterophilic graphs, without relying on manually designed augmentations. It is evaluated on both node and graph classification.

Prompt-based Graph Fine-tuning. These methods adapt pre-trained graph models by adding learnable prompts while keeping the tuning process lightweight.

- GPPT [43]: A prompt-based method that converts node classification into an edge-prediction-style task to reduce the gap between pre-training and fine-tuning. It is used for node classification.
- GraphPrompt [107]: A unified prompting framework that reformulates different graph tasks with learnable prompts. It is evaluated on both node and graph classification.
- GPF [109]: A general prompt tuning method that adds prompts to the input feature space and can be used with different pre-trained GNNs. It is evaluated on both node and graph classification.

Table 6: Accuracy (% ± std over 20 runs) of **ten-shot classification**. Best results are shown in **bold**, and runner-up results are underlined. CR† = Cora, CS† = CiteSeer, PM† = PubMed; PT‡ = Photo, CP‡ = Computers; CN§ = Cornell, TX§ = Texas, WCS§ = Wisconsin. Datasets with the same superscript symbol are from the same domain.

Source	**Cross-Dataset**			**Cross-Domain**			
	CS† PM† PT‡ CP‡	CR† PM† PT‡ CP‡	CR† CS† PT‡ CN§	PT‡ CP‡ CN§ TX§	PT‡ CP‡ CN§ TX§	CR† CS† PT‡ CP‡	CR† CS† PT‡ CP‡
Model / Target	CR†	CS†	PM†	CR†	CS†	WCS§	CN§
Node Classification							
GCN [139]	36.42 ± 4.26	31.84 ± 3.98	44.28 ± 5.08	33.62 ± 4.91	29.41 ± 4.63	35.18 ± 5.12	33.84 ± 4.96
GAT [140]	37.18 ± 4.41	32.56 ± 4.16	45.06 ± 5.27	34.36 ± 5.04	30.14 ± 4.82	35.96 ± 5.31	34.52 ± 5.13
GraphCL [142]	48.84 ± 5.48	43.92 ± 5.16	52.38 ± 6.12	48.72 ± 5.86	41.36 ± 5.63	42.74 ± 5.92	41.18 ± 5.76
DSSL [143]	47.92 ± 5.36	42.86 ± 5.03	51.64 ± 5.94	47.84 ± 5.71	40.52 ± 5.48	41.86 ± 5.78	40.31 ± 5.62
GraphACL [144]	50.42 ± 5.18	45.38 ± 4.94	53.91 ± 5.82	50.36 ± 5.54	42.81 ± 5.31	43.92 ± 5.61	42.28 ± 5.46
GraphPrompt [107]	57.36 ± 5.86	52.18 ± 5.52	58.84 ± 6.41	59.26 ± 6.12	49.72 ± 5.92	46.84 ± 6.04	44.92 ± 5.87
GPF [109]	59.18 ± 6.04	54.42 ± 5.74	60.36 ± 6.58	61.48 ± 6.31	51.84 ± 6.08	47.92 ± 6.22	46.38 ± 6.03
ProNoG [115]	62.41 ± 5.92	58.16 ± 5.61	63.72 ± 6.41	65.84 ± 6.12	55.18 ± 5.96	49.38 ± 6.07	47.62 ± 5.88
GCOPE [14]	64.84 ± 7.24	61.72 ± 4.48	63.48 ± 7.92	67.84 ± 4.02	58.16 ± 7.18	44.26 ± 5.72	56.48 ± 8.12
MDGPT [8]	67.26 ± 5.61	64.84 ± 8.32	65.72 ± 5.16	70.18 ± 7.84	61.74 ± 4.16	49.62 ± 8.04	58.92 ± 5.74
SAMGPT [12]	73.00 ± 3.32	68.60 ± 2.07	61.60 ± 6.88	73.00 ± 3.32	66.60 ± 3.05	51.00 ± 4.85	62.20 ± 4.49
BRIDGE [10]	70.42 ± 8.18	66.84 ± 5.72	64.38 ± 6.41	71.62 ± 4.38	64.12 ± 7.84	50.18 ± 6.04	60.84 ± 4.18
HHGFM (ours)	**73.84 ± 7.62**	**70.42 ± 8.18**	**67.18 ± 7.14**	**73.36 ± 7.84**	**67.84 ± 9.12**	**52.84 ± 8.31**	**63.72 ± 7.96**
Graph Classification							
GCN [139]	38.16 ± 4.42	33.84 ± 4.18	46.72 ± 5.14	35.64 ± 4.93	31.26 ± 4.72	37.84 ± 5.21	36.28 ± 5.03
GAT [140]	38.94 ± 4.58	34.62 ± 4.31	47.54 ± 5.31	36.42 ± 5.08	32.04 ± 4.87	38.63 ± 5.36	37.04 ± 5.17
GraphCL [142]	51.72 ± 5.34	46.84 ± 5.08	55.48 ± 5.92	53.18 ± 5.67	45.72 ± 5.43	45.38 ± 5.71	43.84 ± 5.54
DSSL [143]	50.84 ± 5.22	45.96 ± 4.96	54.62 ± 5.76	52.31 ± 5.52	44.84 ± 5.31	44.52 ± 5.58	43.08 ± 5.41
GraphACL [144]	53.28 ± 5.04	48.42 ± 4.81	56.84 ± 5.63	54.72 ± 5.41	47.16 ± 5.18	46.26 ± 5.42	44.84 ± 5.28
GraphPrompt [107]	60.18 ± 5.68	55.42 ± 5.36	62.18 ± 6.14	63.84 ± 5.92	53.72 ± 5.74	49.82 ± 5.91	47.92 ± 5.73
GPF [109]	61.84 ± 5.86	57.36 ± 5.54	63.84 ± 6.31	65.72 ± 6.08	55.64 ± 5.91	50.84 ± 6.04	48.84 ± 5.87
ProNoG [115]	65.18 ± 5.74	61.24 ± 5.42	67.12 ± 6.18	69.84 ± 5.94	59.16 ± 5.78	52.48 ± 5.92	50.36 ± 5.74
GCOPE [14]	67.84 ± 7.02	64.62 ± 4.31	66.28 ± 7.61	72.18 ± 3.91	62.48 ± 6.94	47.16 ± 5.56	58.74 ± 7.86
MDGPT [8]	70.42 ± 5.43	67.18 ± 8.04	68.54 ± 4.98	74.84 ± 7.52	65.72 ± 4.04	52.84 ± 7.82	61.48 ± 5.56
SAMGPT [12]	75.20 ± 3.08	71.40 ± 1.96	64.80 ± 6.42	76.60 ± 3.14	70.20 ± 2.88	54.80 ± 4.62	66.40 ± 4.28
BRIDGE [10]	72.84 ± 7.86	69.16 ± 5.48	67.18 ± 6.14	75.18 ± 4.22	67.82 ± 7.52	53.42 ± 5.84	64.18 ± 4.06
HHGFM (ours)	**77.84 ± 7.31**	**73.62 ± 7.86**	**70.16 ± 6.92**	**77.24 ± 7.54**	**72.84 ± 8.76**	**57.36 ± 7.98**	**69.82 ± 7.62**

- ProNoG [115]: A prompt-based method for non-homophilic graphs that adjusts prompts according to node-level structural information. It is evaluated on both node and graph classification.

Multi-Domain Graph Pre-training. These methods are most related to our setting, as they aim to transfer knowledge across graphs from different domains.

- GCOPE [14]: A multi-dataset pre-training framework that learns common graph knowledge while reducing negative transfer. It is evaluated on both node and graph classification.
- MDGPT [8]: A multi-domain pre-training method that uses domain tokens and prompts to align heterogeneous graph domains. It is evaluated on both node and graph classification.
- SAMGPT [12]: A multi-domain framework that uses structure tokens and prompts to transfer both shared and domain-specific structural knowledge. It is evaluated on both node and graph classification.

Cross-Dataset and Cross-Domain Settings. We consider two types of domain adaptation scenarios. For *cross-dataset* transfer, the target dataset is unseen during pre-training but comes from the same domain as the source datasets. For *cross-domain* transfer, the target dataset originates from a different domain altogether. It is worth noting that, while some prior works on cross-domain graph pre-training and fine-tuning [14, 8, 12] treat each dataset as a separate domain, we argue that such treatment is overly coarse. Instead, we define domains based on broader semantic or application categories. As a result, our *cross-dataset* setting corresponds to what previous literature often calls *cross-domain*, whereas our *cross-domain* setting reflects a higher-level shift across distinct fields.

E Additional Results

E.1 m-shot classification

We further report results under larger few-shot budgets. Tables 5 and 6 summarize the performance in the **5-shot** and **10-shot** settings, respectively, covering both node and graph classification.

F Limitations and Future Work

In this work, we primarily focus on three domains of datasets, citation, co-purchase, and Web Page. In the real world, there also exist other types of datasets, such as social network. Therefore, we will conduct further studies on more data types in future work based on HHGFM.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please see Appendix F.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Please see Section 4.1 and Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please see Section 5.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our code will be public after the paper is accepted.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Please see Section 5.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Please see Section 5.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please see Section 5.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification:

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.