A Graph Foundation Model for Unified Clustering

Renda Han[1] **Xiaobao Wang**[1,*] **Longbiao Wang**[1] **Dayu Hu**[2] **Ronghao Fu**[3] **Di Jin**[1]
[1]Tianjin Key Laboratory of Cognitive Computing and Application,
College of Intelligence and Computing, Tianjin University
[2]College of Medicine and Biological Information Engineering, Northeastern University
[3]College of Computer Science and Technology, Jilin University
{rendah,wangxiaobao,longbiao_wang,jindi}@tju.edu.cn
hudy@bmie.neu.edu.cn furh@jlu.edu.cn

Abstract

Graph clustering is a fundamental task for understanding graph-structured data without labels, yet classical end-to-end methods require retraining on each dataset, limiting their generalization ability. Although Graph Foundation Models (GFMs) enable transferable learning across diverse graph tasks, they are not directly applicable to graph clustering. This limitation stems from two key factors: the inability to learn mutually beneficial knowledge on both the node and graph levels, and the lack of a completely unsupervised clustering-oriented design. To bridge this gap, we propose a **G**raph **F**oundation **M**odel for **U**nified **C**lustering (GFM-UC), which pre-trains on the source domain and only requires fine-tuning for fast clustering in the target domain. The entire learning process does not require any labels. Specifically, in the pre-training stage, we select learnable high-confidence unified subgraph prototypes from the source domain and generalize them to train the clustering network collaboratively. In the fine-tuning stage, we align the target domain with the source domain in a distribution-aware manner to further refine the clustering head and obtain more well-separated cluster boundaries. Extensive experiments on six datasets demonstrate that our method significantly outperforms existing state-of-the-art approaches.

1 Introduction

Graph Clustering (GC) plays a central role in molecular property analysis [1], recommendation [2], and community discovery [3]. With the development of Graph Neural Networks (GNNs) [4, 5, 6, 7, 8], Deep Graph Clustering (DGC) has emerged as a powerful paradigm that jointly learns graph representations [9] and clustering assignments in an end-to-end framework [10, 11]. DGC can be divided into two categories: node-level clustering [12, 13, 14, 15, 16, 17, 2] and graph-level clustering [18, 19, 20, 21, 22]. The former aims to assign each node to a latent group by jointly learning node representations on a single large graph. Such methods typically exploit structural dependencies, attribute interactions, or contrastive objectives to enhance node separability. In contrast, the latter focuses on clustering a set of entire graphs, where each instance corresponds to an independent graph sample. This setting requires learning discriminative graph-level representations that capture both local substructure patterns and global topological semantics.

Despite these advances, existing DGC methods still suffer from two key limitations: 1) They focus on a single granularity (node level or graph level). In fact, clustering on the two granularities simultaneously captures complementary knowledge [23]: node clusters reflect local structural regularities, while graph clusters characterize global patterns and higher-order dependencies, generating more stable cluster assignments. 2) Most current approaches are tailored to dataset-specific training pipelines.

*Corresponding author.

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

Their learned representations are not generalizable; once the data distribution/domain changes, the model must be retrained.

Building on the above analysis, an intuitive approach is to leverage Graph Foundation Models (GFMs) [24, 25, 26, 27, 28, 29, 30] to learn generalizable graph structure and node attribute priors from large-scale heterogeneous graphs, enabling clustering on the target domain to benefit from transferable knowledge across domains. Nevertheless, most existing cross-domain GFMs are inherently label-dependent and thus cannot effectively adapt to clustering scenarios without label supervision. Although several label-free methods exist, such as UNPrompt [31], AnomalyGFM [29], and GFM-UAD [32], they are primarily designed for anomaly detection or general graph representation learning. Their optimization and feature extraction mechanisms are not tailored for clustering, limiting their applicability in unsupervised clustering scenarios. Therefore, to achieve a unified cross-domain graph clustering model, two key challenges still need to be addressed: (1) How can node- and graph-level clustering be addressed from a unified perspective so that they mutually enhance each other? (2) How can generalized patterns be effectively extracted and domain knowledge aligned between source and target domains, even under heterogeneous cluster structures? Table 1 summarizes why existing methods cannot simultaneously satisfy these requirements. To solve the first issue, inspired by Uniform [33], we construct agent subgraphs for each node and graph sample, which explicitly encode structural dependencies and attribute interactions around each sample, serving as structural surrogates that preserve sample-specific cluster patterns. To solve the second issue, we dynamically select cluster prototypes to generalize and enrich the domain space, encouraging the model to capture broader domain knowledge and adapt more effectively to unseen graphs. During fine-tuning, we further introduce a multi-distribution weighted alignment to refine cross-domain distribution calibration.

Capability	**DGC**	**GFM**	**OURS**
Label-free	✓	•	✓
Cross-domain	×	•	✓
Clustering-oriented	✓	×	✓

• partially ✓ fully × not **satisfied**

Table 1: Existing advanced DGC and GFM methods are limited in simultaneously satisfying all the above capabilities.

To address these challenges, we propose a Unified Clustering Graph Foundation Model called GFM-UC, designed to jointly improve node-level and graph-level clustering in a fully unsupervised GFM. Its core idea is to leverage high-confidence cluster prototypes to generate transferable samples, which are then used to train a generalized clustering network for cross-domain fine-tuning. Specifically, we first employ agent subgraphs to unify task granularity across clustering settings. We then generate samples from cluster prototypes to enrich the domain space and learn transferable clustering knowledge. Finally, we perform unsupervised fine-tuning with distribution alignment to adapt the clustering head to the target domain. In summary, our contributions are threefold:

- **New Perspective** To the best of our knowledge, this is the first exploration of a unified clustering transfer learning paradigm on GFMs, characterized by being label-free, clustering-oriented, and cross-granularity, making it more challenging than general GFM settings.
- **New Task** We propose a GFM for Unified Clustering, termed GFM-UC, which is pre-trained once on the source domain and simultaneously performs clustering on both node- and graph-level samples across diverse target domains.
- **Superior Performance** Extensive experiments on diverse benchmarks show that GFM-UC consistently surpasses SOTA GFM and DGC methods in both node- and graph-level tasks.

2 Methodology

In this section, we present GFM-UC in detail. The key idea is to unify clustering tasks across multi-level samples through agent subgraphs, and to further leverage high-confidence prototypes from the source domain to train a generalized clustering network for tuning on the target domain. As illustrated in Fig. 1, GFM-UC consists of unified subgraph extraction, domain generalization, and domain adaptation. The overall pipeline is summarized in Algorithm 1.

2.1 Notations

Suppose that we have a source dataset $\mathcal{D}_S$ containing N_G graphs with K clusters. Each graph in $\mathcal{D}_S$ is defined as $G_S = (\mathbf{X}, \mathbf{A})$ with n nodes and m edges, where $\mathbf{X} \in \mathbb{R}^{n \times d}$ represents the node feature

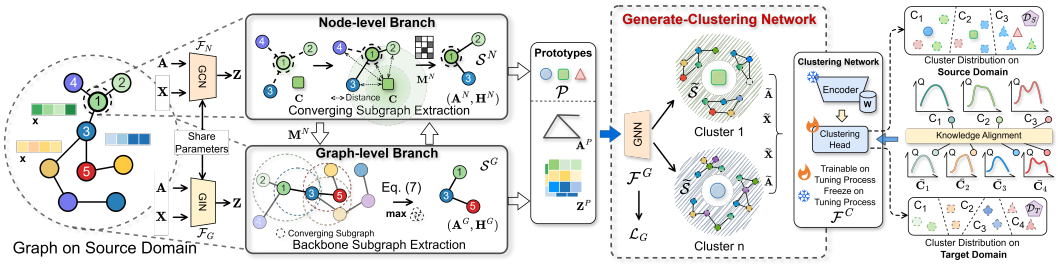

Figure 1: The architecture of GFM-UC. GFM-UC first unifies task granularity through ego-subgraphs and induced subgraphs. It then leverages the cluster prototypes to generate and expand domain space, encouraging the clustering model to learn generalizable knowledge for subsequent fine-tuning.

matrix and $\mathbf{A} \in \{0,1\}^{n \times n}$ denotes the normalized adjacency matrix. During pre-training, the model simultaneously performs node- and graph-level clustering on $\mathcal{D}_S$, while inference is performed on an unseen dataset $\mathcal{D}_T$, which may come from a different domain. Our goal is to develop a GFM that performs unified clustering at both the node and graph levels across multiple graphs in a fully unsupervised setting, with pre-training conducted once and inference conducted on different datasets.

2.2 Unified Subgraph Extraction

To bridge the gap arising from the discrepancy between node-level and graph-level task granularities, we propose to unify these two perspectives via agent subgraphs. Specifically, for each node, we extract a convergence subgraph that captures its local structural context and intrinsic properties. For each graph instance, we further identify a backbone subgraph that preserves the most informative and representative structural patterns of the graph. These convergence subgraphs and backbone subgraphs provide a unified representation space, enabling both node-level and graph-level samples to be consistently projected into a common subgraph-driven domain for clustering.

Algorithm 1: Training Procedure of GFM-UC

Input: Source graph set $\mathcal{D}_S$; Target graph set $\mathcal{D}_T$; Training epochs N_{epoch};
Output: Clustering results R.

```
// Pre-training process
```
Extract node embeddings $\mathbf{H}$ using Eq. (1);
Extract converging subgraph $\mathcal{S}^N$ by Eqs. (2)-(5);
Extract backbone subgraph $\mathcal{S}^G$ by Eqs. (6)-(8);
Obtain the prototype for each cluster;
Generate domain samples $\widetilde{\mathbf{X}}$ and $\widetilde{\mathbf{A}}$;
Input $\widetilde{\mathbf{X}}$ and $\widetilde{\mathbf{A}}$ into the clustering network;
Alternately calculate $\mathcal{L}_G$ and $\mathcal{L}_C$ using Eqs. (11)-(12);
Backpropagate and optimize parameters;
```
// Fine-tuning process
```
Extract subgraph shot samples for tuning;
Align source domain knowledge with target domain knowledge using Eq. (13);
Obtain clustering results R;
return R;

Converging Subgraph Extraction For the node-level task, we attempt to extract a convergence subgraph $\mathcal{S}^C$ as a minimal and sufficient structure agent for clustering, which is learned as a minimal and sufficient pattern digest for each node. Specifically, we first learn structure-aware representations via an L-layer dual-branch encoder $\mathcal{F}_N$ and $\mathcal{F}_G$, for subsequent subgraph extraction to obtain node embeddings:

$$\mathbf{H} = \mathcal{F}_{N|G}(\mathbf{A}, \mathbf{X}). \tag{1}$$

Then, we introduce a mask matrix $\mathbf{M} \in \mathbb{R}^{N \times d}$ to parameterize the soft subgraph for each node in the representation space. The induced subgraphs are defined as

$$\mathcal{S}^C = (\mathbf{A}, \mathbf{H}) = \big(\mathbf{A} \odot (\mathbf{M}\mathbf{M}^\top),\ \mathbf{H} \odot \mathbf{M}\big), \tag{2}$$

which simultaneously selects informative nodes and their structural interactions. The subgraph extraction process can be viewed as a stochastic mapping $q_\theta(\mathcal{S}_\phi)$ induced by the mask distribution. The objective is to ensure that $\mathcal{S}_\phi$ preserves cluster-relevant semantics encoded in $\mathbf{H}$ while discarding residual information that is not essential for clustering. To this end, we formulate the following variational information bottleneck objective:

$$\max_{\theta}\ \mathbb{E}_{q_\theta(\mathcal{S}_i)}\big[\mathbb{I}(\mathcal{S}^C; \mathbf{C}) + \sum\nolimits_{i \in \mathcal{V}} \mathbb{I}(\mathcal{S}_i^C; \mathbf{h}_i) - \beta \mathbb{I}(\mathcal{S}^C; \mathbf{H})\big], \tag{3}$$

where $\mathbf{C}$ denotes the latent cluster variable and $\mathbb{I}(\cdot;\cdot)$ denotes mutual information. The first term encourages the extracted subgraph to retain discriminative information for its cluster, while the third term enforces minimality by constraining its dependence on the full representation $\mathbf{H}$. To make the objective tractable, we adopt a variational approximation of the mutual information:

$$\mathbb{I}(\mathcal{S}^C;\mathbf{C}) \geq \mathbb{E}_{q_\theta(\mathcal{S}_\phi)}\left[\log p_\psi(\mathbf{C} \mid \mathcal{S}^C)\right] - \log p(\mathbf{C}). \tag{4}$$

Furthermore, we interpret the optimization of $\mathbf{M}$ as a dynamical process over the representation-induced energy landscape. The mask is then iteratively updated via gradient-driven dynamics:

$$\mathbf{M} = \sigma\big(-\nabla_{\mathbf{M}}(-\mathbb{I}(\mathcal{S}^C;\mathbf{C}) + \beta\mathbb{I}(\mathcal{S}^C;\mathbf{H}) + \gamma\Omega(\mathcal{S}_\phi))\big), \tag{5}$$

where $\Omega(\mathcal{S}^C)$ enforces structural consistency over $\mathbf{A}$, encouraging the selected nodes to form a coherent substructure aligned with the original topology. Eq. (5) can be interpreted as a discrete relaxation process that progressively suppresses residual components while preserving cluster-relevant structures. Under mild conditions, the dynamics converge to a stationary point $\mathbf{m}_i^*$, yielding a deterministic subgraph $\mathcal{S}_i^*$. Importantly, this fixed point satisfies a minimal sufficiency property in the representation space: $p(\mathbf{C} \mid \mathbf{H}) = p(\mathbf{C} \mid \mathcal{S}^*)$, indicating that the convergence subgraph fully preserves clustering semantics encoded by the GNN, while eliminating irrelevant information.

Backbone Subgraph Extraction For the graph-level branch, we elevate the graph to a consensus-induced measure space, where the global structure is not directly extracted but emerges from the collective agreement of node-wise explanations. $\mathbf{M}_i^N \in \mathbf{M}^N$ denotes the soft mask corresponding to the convergence subgraph of node i. We define the graph-level agent as the information-theoretic barycenter on the probability simplex:

$$\mathbf{s}_i = \arg\min_{\mathbf{u}\in\Delta^{n-1}} \sum\nolimits_{i=1}^{n} \mathrm{KL}\left(\mathbf{M}_i^N \middle\| \mathbf{u}\right), \tag{6}$$

which admits the closed-form trick: $\mathbf{s}_j \propto \exp\left(\frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|}\log m_{i,j}\right)$. This geometric averaging endows $\mathbf{s}$ with a multiplicative aggregation property, ensuring that only those structural components consistently supported across node-wise explanations receive high probability mass. Therefore, the graph-level backbone structure is a log-consensus of local evidence, which inherently suppresses isolated or spurious structures. To further promote structural coherence, we define a higher-order consensus objective over the induced measure $\mathbf{p}$, where the graph backbone arises as a stationary point of a self-reinforcing agreement process. Specifically, we consider the functional

$$\mathbf{p}^\star = \arg\max_{\mathbf{p}\in\Delta^{|\mathcal{V}|-1}} \big[\sum\nolimits_{i,j} p_i p_j, \Psi_{ij} - \mathbf{p}^\top\mathbf{L}\mathbf{p} - \sum\nolimits_j p_j \log p_j\big], \tag{7}$$

where the similarity kernel is defined as $\Psi_{ij} = \langle \mathbf{m}_i, \mathbf{m}_j\rangle = \sum_k m_{i,k}m_{j,k}$, where the first term induces a second-order consensus, encouraging nodes with similar explanatory subgraphs to reinforce each other; the second term imposes graph-aware smoothness via the Laplacian, ensuring that the selected backbone respects the intrinsic topology; and the final entropy term naturally enforces sparsity while avoiding degenerate collapse. The resulting optimal measure $\mathbf{p}^\star$ defines the graph-level convergence subgraph via its agent: $\mathcal{B}^\star = \mathrm{supp}(\mathbf{p}^\star)$. This refined node-subgraph consensus yields a coherent backbone that captures higher-order agreement, bridging local explanations and global structure. Then, we select a subset $\mathcal{V}^G = \{i \mid p_i^\star > \tau\}$ of nodes according to $\mathbf{p}^\star$ to construct the backbone subgraph $\mathcal{S}^G = (\mathbf{A}^G, \mathbf{H}^G)$ as:

$$\mathbf{A}^G = \mathbf{A}[\mathcal{V}^G, \mathcal{V}^G], \quad \mathbf{H}^G = \mathbf{H}[\mathcal{V}^G], \tag{8}$$

where τ is a threshold or determined by top-k selection. In this way, the continuous measure $\mathbf{p}^\star$ is converted into a discrete backbone structure, preserving the most consensus-supported nodes and their interactions. Ultimately, GFM-UC extracts surrogate subgraphs at both the node and graph levels, enabling it to learn cluster-oriented knowledge from a unified perspective for generalization.

2.3 Generative Clustering for Domain Generalization

While the unified subgraph extraction establishes a shared representation space across node-level and graph-level samples, the learned embeddings are still confined to the prior empirical source distribution. This limitation restricts the model's ability to generalize to unseen domains, where the underlying cluster distributions may exhibit distributional shifts. To address this issue, we

propose adversarial generative clustering, which aims to (1) enrich and expand the domain of cluster distributions through generation, and (2) refine cluster boundaries by explicitly identifying samples that deviate from cluster manifolds. The key idea is to couple a variational generator with a clustering-oriented discriminator, forming a closed-loop optimization that enhances both representation robustness and clustering consistency.

Subgraph Sample Generation In this stage, the generator produces domain samples that are difficult for the clustering network to recognize, while the clustering network strives to exclude them from existing clusters. The generator $\mathcal{F}^G(\tilde{\mathcal{S}} \mid \mathcal{P})$ aims to reconstruct and generalize subgraph representations in the latent space so that the generated samples are indistinguishable from real high-confidence cluster samples. Specifically, we first use cluster prototypes $\mathcal{P} = (\mathbf{Z}^P, \mathbf{A}^P)$ to guide a conditional variational generation process. The encoder learns a distribution conditioned on prototype representations:

$$q_\phi(\mathbf{Z}^\phi \mid \mathbf{Z}^P) = \mathcal{N}(\mu, \sigma^2\mathbf{I}), \tag{9}$$

where $(\boldsymbol{\mu}, \log \boldsymbol{\sigma}) = \mathrm{MLP}(\mathbf{Z}^P)$. Subsequently, to further impose cluster-aware structure in the latent space, we introduce a mixture prior over cluster prototypes:

$$p(\mathbf{Z}^\phi) = \sum\nolimits_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{Z}^\phi \mid \mu_k, \mathbf{\Sigma}_k), \tag{10}$$

where each $\boldsymbol{\mu}_k, \mathbf{\Sigma}_k$ is derived from the corresponding cluster prototype $\mathbf{z}_k^P$, which encodes aggregated high-confidence semantic information of that cluster. Then, we jointly reconstruct node attributes and the structure of subgraphs. For the node attributes, we decode $\mathbf{Z}^\phi$ into a set of node-wise embeddings to reconstruct as $\widetilde{\mathbf{X}} = \mathrm{MLP}(\mathbf{Z}^\phi)$, For graph structure, we further construct the adjacency through a similarity-induced decoder $\widetilde{\mathbf{A}} = \sigma\big(\mathbf{Z}^\phi \mathbf{W}_g (\mathbf{Z}^\phi)^\top\big)$, where $\mathbf{W}_g$ is a learnable parameter matrix and $\sigma(\cdot)$ denotes a squashing function that produces soft connectivity scores. In this way, the generator produces subgraphs that are aligned with the representation space induced by the preceding GNN and subgraph extraction process. The overall objective of the generator follows a variational lower bound:

$$\mathcal{L}_{\mathrm{G}} = -\mathrm{KL}\big(q_\theta(\widetilde{\mathcal{S}} \mid \mathbf{Z}^\phi) \| p(\mathcal{P})\big) - \mathbb{I}(\mathbf{A}^P; \widetilde{\mathbf{A}}) + \mathbb{E}_{\widetilde{\mathcal{S}} \sim p_\theta}\big[-\mathcal{H}(p_\omega(\mathbf{C} \mid \widetilde{\mathcal{S}}))\big], \tag{11}$$

where $\mathbf{C}$ represents the original cluster assignments. The first term aligns generated samples with the prior distribution, while the second encourages structural diversity. The final term encourages the generator to produce samples with confident cluster assignments, aligning them with cluster manifolds. This objective synthesizes novel subgraphs within or around cluster manifolds, expanding cluster support beyond observed data and improving domain coverage.

Clustering-oriented Adversarial Discriminator To ensure that the generated subgraphs not only exhibit realism but also adhere to cluster structures, we introduce a clustering network $\mathcal{F}^C(\widetilde{\mathbf{X}}, \widetilde{\mathbf{A}})$ that enforces cluster-level consistency. This network is composed of two key components: an encoder and a discriminator. The encoder is responsible for generating representations of the subgraphs, while the discriminator operates to both cluster the representations and distinguish between in-cluster and outlier samples. The encoder E_ω transforms the input generated samples $\widetilde{\mathcal{S}} = (\widetilde{\mathbf{X}}, \widetilde{\mathbf{A}})$ into a latent representation $\widetilde{\mathbf{Z}} = E_\omega(\widetilde{\mathbf{X}}, \widetilde{\mathbf{A}})$. Then we assign each latent representation to a cluster by clustering head $g_\omega(\cdot)$. Subsequently, we output a cluster assignment distribution: $\widetilde{\mathbf{C}} = g_\omega(\widetilde{\mathbf{Z}})$, where $\widetilde{\mathbf{C}}$ represents the cluster assignments. For real samples, $\mathcal{F}^C$ is trained to produce sharp and consistent cluster assignments, reinforcing the structure of the underlying clusters. For generated samples, the discriminator attempts to detect deviations from the cluster structure by producing uncertain or inconsistent predictions. This leads to the following objective function for the discriminator:

$$\mathcal{L}_{\mathrm{C}} = -\mathbb{E}_{\tilde{\mathcal{S}} \sim p_\theta}\big[\mathcal{H}(p_\omega(\mathbf{C} \mid \mathbf{Z}^P))\big] + \mathbb{E}_{\mathcal{S} \sim \mathcal{D}_{\text{high}}}\big[\log p_\omega(\widetilde{\mathbf{C}} \mid \widetilde{\mathbf{Z}}/\mathbf{Z}^P)\big], \tag{12}$$

where $\mathcal{H}(\cdot)$ denotes entropy. The first term ensures that the discriminator assigns high confidence to real samples, encouraging sharp, consistent cluster assignments. The second term penalizes the consistent predictions for generated samples, helping the discriminator better distinguish between real and generated samples. This formulation encourages the model to generate samples that both exhibit realism and conform to the expected cluster structures.

Optimization Objective The generator and discriminator are optimized in an adversarial manner, where the generator synthesizes subgraphs to enrich the cluster distribution, and the discriminator enforces compact cluster boundaries by rejecting out-of-manifold samples. The generator loss $\mathcal{L}_{\mathrm{G}}$ and

clustering objective $\mathcal{L}_C$ jointly encourage generated samples to conform to cluster structures while penalizing deviations. The overall optimization objective is: $\min_\theta \max_\omega \mathcal{L}_{\mathrm{G}} - \lambda \mathcal{L}_{\mathrm{C}}$. This process enhances cluster consistency in generated samples and improves generalization to unseen domains. The process plays two complementary roles: it extrapolates the data distribution by synthesizing cluster-conditioned subgraphs, mitigating overfitting to source domains, and it sharpens cluster boundaries by identifying out-of-manifold samples. Overall, this design aligns generation with clustering objectives, yielding more robust and transferable knowledge.

2.4 Domain Adaptation via Fine-Tuning

At this stage, our goal is to enable the clustering head g_ω to rapidly adapt to the target domain. To this end, we fine-tune its parameters by aligning the cluster-level distributions between the source and target domains. Specifically, we compute the cluster distributions in the source and target domains as $\mathcal{D}_S = p_S(\widetilde{\mathbf{C}})$ and $\mathcal{D}_T = p_T(\bar{\mathbf{C}})$, respectively. In the target domain, we only select $\mathcal{T}$ high-confidence shot samples to ensure the quality and reliability of the domain data. To address the discrepancy in the number of clusters between the source and target domains, we introduce a learnable weight matrix $\mathbf{\Pi}$, which captures the correspondence between source and target clusters. Each entry in $\mathbf{\Pi}$ reflects the contribution of a source cluster to a target cluster, enabling flexible cross-domain alignment. Based on this, we reweight the source distribution and minimize its divergence from the target distribution:

$$\mathcal{L}_{\text{align}} = \mathrm{KL}\big(\mathbf{\Pi}, \tilde{p}_S(\widetilde{\mathbf{C}}) \mid \mathcal{D}_T)\big), \tag{13}$$

where $\tilde{p}_S(\widetilde{\mathbf{C}})$ denotes the reweighted source cluster distribution. This objective encourages the clustering head g_ω to adapt the source distribution to approximate the target distribution, reducing the domain discrepancy. As a result, the clustering head can effectively transfer cluster knowledge and achieve rapid adaptation in the target domain.

2.5 Efficiency Analysis

We analyze the computational efficiency of GFM-UC in both the pre-training and fine-tuning stages. **For pre-training**, the computational cost is mainly dominated by two components: (i) unified subgraph extraction, (ii) adversarial generative clustering. For unified subgraph extraction, the GNN-based encoder requires $\mathcal{O}(mdL + nd^2L)$. The extraction of agent subgraphs requires $\mathcal{O}(mL)$. The mask-based subgraph construction requires $\mathcal{O}(n^2d)$. For the generative clustering, the variational encoding requires $\mathcal{O}(nd^2L)$ complexity. The KL divergence and entropy regularization are computed in closed form and incur negligible overhead. Overall, the pre-training complexity is approximately: $\mathcal{O}(nd^2L + mdL + n^2d)$,

For fine-tuning, only the clustering head and alignment matrix $\mathbf{\Pi}$ are updated, while the GNN encoder and generator are frozen. The dominant cost lies in computing cluster assignment distributions and domain alignment via KL divergence, which scales as: $\mathcal{O}(\mathcal{T}Kd)$. The learnable alignment matrix $\mathbf{\Pi}$ introduces only $\mathcal{O}(K^2)$ overhead, which is negligible in practice.

3 Experiments

We evaluate GFM-UC across diverse graph clustering scenarios, including benchmarking against state-of-the-art baselines, ablation studies, and additional experiments to comprehensively assess the contribution of each component and the overall generalized clustering design.

3.1 Experimental Setup

Benchmark Datasets We evaluate GFM-UC on benchmark datasets from the TUDataset collection [34], spanning graph-level and node-level clustering across multiple domains. We use COX2, PTC-MM, DHFR, and PROTEINS as standard benchmarks, and PC-3 and SW-620 for large-scale scalability evaluation (see **Appendix** B.1 for more details).

Baseline Methods We compare GFM-UC with end-to-end DGC models and GFMs. The former are further divided into node- and graph-level approaches. For node-level, the models include: SDCN [12], DFCN [13], SCGC [35], ABR [14] and CoCo [36]. For graph-level, the models include: GLCC [37], UDGC [38], DGLC [19], and DCGLC [18]. For the GFM category, we select both GNN and

Method	COX2			PTC-MM			DHFR			PROTEINS		
	ACC	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
Node-level Task												
SDCN	44.3±1.4	24.5±1.5	19.2±1.1	34.1±1.6	10.3±1.4	12.5±2.1	38.6±1.9	7.0±1.9	15.4±2.0	42.9±2.2	18.3±1.6	19.9±3.2
DFCN	47.1±2.3	27.9±2.1	16.7±1.5	43.2±1.8	13.2±1.9	12.4±1.4	42.6±2.5	0.1±0.1	0.1±0.1	37.8±3.1	17.0±1.4	20.2±2.6
SCGC	49.5±2.0	23.3±1.7	20.7±1.4	48.3±1.7	15.8±1.5	23.8±2.0	44.3±0.8	1.3±0.4	0.9±0.5	35.9±0.4	7.7±0.8	9.0±0.5
ABR	48.6±0.4	25.6±0.3	19.8±0.2	43.2±0.7	16.9±0.7	18.4±1.2	23.5±0.1	12.2±0.5	10.7±0.6	43.4±1.9	18.6±1.2	31.6±1.5
CoCo	42.7±1.0	18.5±1.3	15.8±1.2	31.6±2.2	13.9±1.9	14.3±0.9	41.4±1.3	0.5±0.0	0.1±0.1	32.6±2.0	8.7±1.3	16.4±1.3
LangGFM	31.6±2.3	8.0±1.4	3.8±0.8	32.4±1.1	5.7±2.9	2.6±1.3	24.9±5.2	5.0±4.7	4.8±3.3	34.3±4.7	6.0±4.6	13.6±3.7
TS-GFM	27.8±1.3	6.4±3.5	7.7±4.2	33.6±0.5	2.4±0.8	3.1±0.1	26.6±1.7	5.9±1.6	5.2±2.0	41.9±2.0	13.9±1.8	21.6±2.3
GPC	61.5±1.5	31.0±1.6	32.6±1.2	60.6±1.9	15.6±1.8	17.2±1.3	46.3±1.2	13.2±1.4	14.2±1.5	47.0±1.7	14.7±1.2	23.9±1.6
AnyGraph	56.9±2.0	26.8±2.4	24.7±1.8	54.2±1.3	7.9±1.1	7.0±1.4	38.5±2.4	4.0±0.5	3.1±0.4	45.5±1.6	11.5±2.8	16.3±2.2
GraphAny	51.2±1.7	24.2±2.3	23.5±2.0	52.2±1.5	7.9±1.6	3.8±1.2	45.8±2.0	1.1±1.3	6.9±0.6	52.0±1.8	16.9±0.5	10.4±1.7
GFT	55.5±2.3	21.1±1.7	22.0±1.5	56.1±1.4	19.6±1.9	14.8±1.3	44.7±1.7	8.9±1.6	6.2±1.6	48.0±1.5	15.2±1.2	17.0±1.6
UniGraph	47.0±1.6	26.3±2.1	25.6±3.4	41.0±1.6	16.4±2.1	13.9±1.8	47.6±2.1	15.1±1.8	16.2±1.4	50.3±2.6	13.8±1.7	15.5±1.1
SAMGPT	54.9±2.0	29.5±2.7	28.8±3.1	45.2±1.9	11.3±1.8	12.5±1.0	49.3±2.3	13.6±1.3	13.1±2.6	51.3±0.9	7.8±0.4	8.6±0.9
GraphCLIP	48.6±0.2	20.1±1.4	22.5±0.5	55.4±0.8	14.8±1.3	15.3±0.7	39.0±0.8	3.2±0.6	3.5±0.7	43.4±1.0	14.6±1.1	24.1±0.8
MDGFM	55.4±1.3	16.0±1.2	20.3±1.7	40.5±2.9	5.0±0.3	2.0±0.6	50.1±2.3	3.1±0.8	1.5±0.4	53.9±1.8	13.0±0.6	15.4±1.0
OURS	**64.6±1.2**	**33.3±1.5**	**35.6±1.7**	**62.7±2.1**	**26.7±1.6**	**27.4±1.8**	**52.3±2.6**	**18.7±1.1**	**20.3±1.6**	**55.9±2.1**	**20.4±2.0**	**26.0±2.4**
Graph-level Task												
	ACC	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
GLCC	54.0±1.7	1.7±0.2	2.0±0.6	50.1±1.2	2.5±0.1	1.3±0.1	52.8±1.0	3.8±0.3	2.7±0.4	51.0±1.6	0.5±0.0	0.3±0.1
UDGC	50.5±0.8	0.7±0.4	1.3±1.1	55.1±0.4	0.3±0.1	0.5±1.2	57.6±0.9	4.8±0.0	1.3±0.0	52.6±0.9	0.2±0.8	0.3±1.4
DGLC	52.7±1.1	3.9±0.6	2.5±0.7	46.3±1.1	2.5±1.0	1.2±0.6	55.7±0.9	0.8±0.2	0.1±0.3	57.9±0.4	7.6±1.5	5.9±1.0
DCGLC	54.8±1.9	6.4±2.1	5.3±1.4	47.7±1.4	3.8±1.7	3.6±1.4	58.3±1.5	5.1±1.4	4.6±1.7	59.8±1.5	6.3±0.4	4.8±1.2
LangGFM	56.5±3.4	0.1±0.1	0.3±0.2	51.1±0.3	0.2±0.1	0.4±0.2	53.4±0.7	0.4±0.2	0.3±0.0	56.9±0.2	0.1±0.0	0.2±0.1
TS-GFM	54.2±1.1	2.5±0.2	0.2±0.2	52.8±1.0	0.0±0.1	0.1±0.0	56.7±0.5	5.9±0.2	6.1±0.3	53.5±0.9	5.7±0.5	4.2±1.2
GPC	69.5±1.7	4.5±0.7	3.4±0.8	70.1±1.9	5.3±0.9	4.1±0.2	63.4±1.5	8.9±1.7	7.6±2.0	62.4±1.2	8.9±0.7	12.6±0.8
AnyGraph	53.2±1.0	4.0±0.5	2.9±0.7	62.6±1.4	4.7±1.3	0.8±0.1	60.9±1.1	3.8±0.2	3.7±1.8	57.1±2.3	2.6±0.4	3.0±0.2
GraphAny	57.3±1.1	6.2±0.8	4.7±0.4	43.1±0.0	3.1±0.2	5.8±0.5	OOM	OOM	OOM	OOM	OOM	OOM
GFT	58.3±2.2	6.6±0.9	5.0±0.7	52.5±0.4	3.9±0.3	0.0±0.0	59.3±0.2	4.1±0.1	0.7±0.2	55.0±3.6	3.9±2.8	4.1±2.1
UniGraph	61.5±0.7	2.4±1.3	3.6±1.9	56.8±1.7	2.3±1.1	0.9±0.8	52.3±1.9	0.4±0.1	0.1±0.2	50.1±3.4	8.2±1.6	11.0±1.4
SAMGPT	63.4±0.6	4.0±1.6	4.4±1.1	62.6±1.5	0.9±1.2	3.5±1.1	55.6±1.2	0.8±0.0	0.9±0.0	61.7±1.6	6.8±1.1	10.5±1.3
GraphCLIP	56.4±1.2	3.7±0.9	4.0±1.0	54.9±3.6	1.1±1.4	2.5±0.8	50.6±1.9	0.5±0.8	6.3±0.7	52.9±3.2	3.9±0.2	3.0±0.6
MDGFM	55.1±2.3	3.1±1.0	3.0±1.5	65.8±2.2	4.6±1.7	3.9±1.7	58.3±3.0	10.4±1.0	8.1±0.8	56.6±2.4	5.9±1.8	7.3±1.5
OURS	**73.2±1.4**	**8.5±1.6**	**6.8±0.8**	**73.8±3.1**	**6.2±1.4**	**7.7±1.5**	**66.9±1.4**	**12.1±1.2**	**13.4±0.8**	**64.6±1.7**	**13.7±1.5**	**16.2±1.8**

Table 2: Performance comparison results (expressed as mean ± standard deviation) of the node-level and graph-level clustering tasks on four benchmark datasets. **Bold** and underlined values indicate the best and second-best results in each column, respectively. "OOM" denotes out-of-memory.

LLM-based models. The GNN-based models include: GPC [7], AnyGraph [26], GraphAny [27], GFT [39], UniGraph [28], SAMGPT [40], GraphCLIP [41], and MDGFM [42]. The LLM-based GFMs include: LangGFM [43], TS-GFM [44] and GraphICL [45]. Details are shown in **Appendix** B.2.

Evaluation Metrics We employ Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI) [46, 47, 48, 49, 50]; details are in **Appendix** B.3.

Implementation Details We implement the model in PyTorch, adopt a pre-train-then-fine-tune training strategy, and use a learning rate of 1e-3 with the Adam optimizer [51, 52, 48, 53]. The batch size is set to 128. More details of implementation are shown in **Appendix** B.4.

3.2 Comparison Experiments

The results of the comparison experiment are shown in Table 2 and **Appendix** D for large-scale datasets. Based on the results, we can observe: i) Compared with DGC methods, our approach achieves significantly better performance. This is mainly because these methods are trained specifically for particular datasets and lack the ability to generalize to other domains. As a result, their performance drops sharply when applied to inference tasks in unseen domains. ii) Compared with representation-learning-based GFM models, our method consistently achieves superior performance. Existing GFM models often rely heavily on label supervision during training and are not explicitly designed for clustering objectives. As a result, the learned representations lack clear cluster separability in the latent space, making it difficult to form well-defined and compact clusters. iii) For

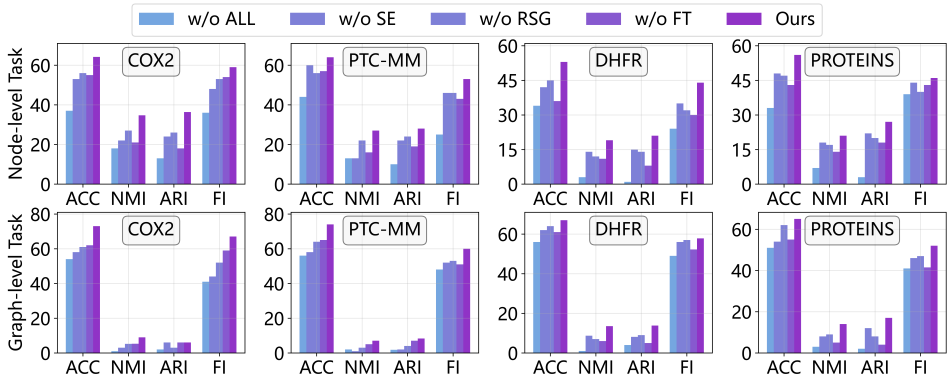

Figure 2: Ablation study results across node- and graph-level clustering tasks on four benchmark datasets.

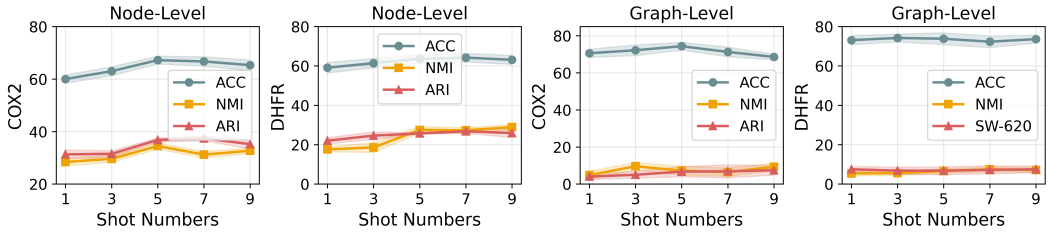

Figure 3: Clustering performance under various fine-tuning shots $S \in [1, 9]$.

large-scale datasets, GFM-UC still maintains a performance advantage, mainly driven by effective generalization to unseen samples, stemming from high-quality prototypes and diverse generalization under adversarial conditions. The prototypes provide representative clustering patterns, while sample generation expands the domain space beyond the observed source samples. Together, these components help the clustering network transfer the learned patterns to larger target datasets.

3.3 Ablation Study

To evaluate the contribution of each component, we design four variants: (1) removing the subgraph extraction **(w/o SE)**; (2) removing the sample generation **(w/o RSG)**; (3) removing the fine-tuning (w/o **FT**); and (4) removing all components, denoted as **(w/o All)**. As shown in Fig. 2, we can observe: 1) Removing the subgraph extraction degrades clustering performance because the model fails to leverage unified information across different granularities. 2) Removing sample generation significantly degrades performance, primarily because the model cannot effectively use expanded domain samples to acquire transferable knowledge. 3) Removing fine-tuning leads to a noticeable performance drop because cluster centers across domains cannot be effectively aligned. Nevertheless, it still outperforms the **w/o All** variant, showing the benefit of the generalization process.

3.4 Hyperparameter Sensitivity Analysis

Number of Shot Samples S We analyze the sensitivity of GFM-UC to the number of fine-tuning shots by varying S from 1 to 9. As shown in Fig. 3, performance is stable only when S falls within a moderate range. Too few shots leave the core domain space incomplete, while too many unrepresentative samples distort the target domain space and degrade alignment. This suggests that the representativeness of the selected samples, rather than their number alone, is important for calibrating source and target distributions. Graph-level tasks are more sensitive to S because their larger granularity involves subtler patterns.

Depth of Subgraph N_s We study the effect of the subgraph depth $N_s \in [1, 5]$. Based on the results shown in Fig. 4, we can observe that $N_s = 3$ achieves the best performance. When the value is too

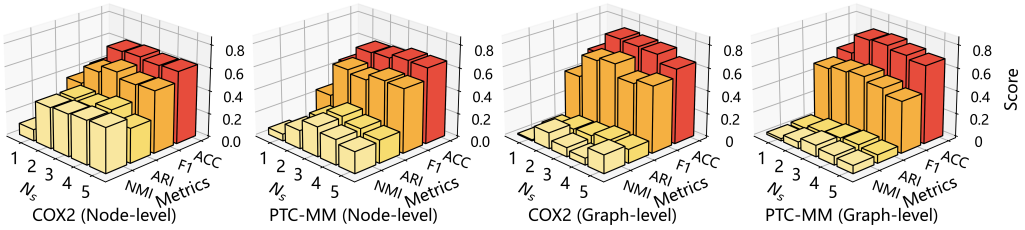

Figure 4: Clustering performance under varying $N_s \in [1, 5]$ on COX2 and PTC-MM datasets.

small, it fails to capture consistent patterns aligned with the unified subgraph signal, whereas when it is too large, it introduces additional noise. Choosing an appropriate value yields the best performance.

3.5 Generalization Test

We evaluate generalizability by pre-training on different datasets and inferring on four target datasets. This setting examines whether the learned clustering knowledge can transfer when the source dataset changes, rather than relying on a single source-target combination. As shown in Fig. 5, performance varies with source-target distribution gaps but remains stable due to the generalized generative domain space. By expanding the source domain around representative prototypes, GFM-UC encourages the clustering network to learn patterns that are useful beyond the original pre-training samples. The results support the applicability of the framework across different pre-training datasets.

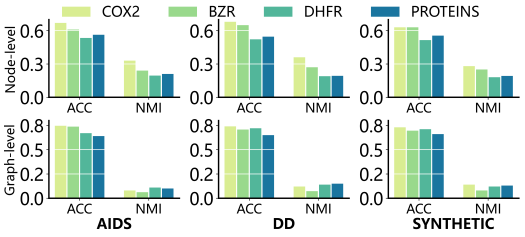

Figure 5: Clustering performance under different pre-training datasets.

3.6 Ablation Study on Task Granularity

To verify the superiority of unified clustering, we compare it with single-granularity clustering. The experimental results, as shown in Table 3, show that single-granularity clustering significantly underperforms unified clustering. This highlights the complementary nature of node structures and global graph-level patterns, where multi-granularity clustering enables more precise and robust prototypes and produces a smoother domain space for unseen samples. Node-level clustering captures local structural regularities, while graph-level clustering supplies information about the organization of the entire graph. Their joint use allows the model to incorporate both types of information when learning transferable clustering patterns.

Dataset	Type	Node-level Task		Graph-level Task		Δ. Avg
		ACC	NMI	ACC	NMI	
COX2	Unified	64.6±1.2	33.3±1.5	73.2±1.4	8.5±1.6	8.9↑
	Single	58.7±1.9	20.4±2.1	61.5±0.8	3.3±0.2	
PTC-MM	Unified	62.8±2.1	26.7±1.6	73.8±3.1	6.2±1.4	12.0↑
	Single	49.3±1.0	18.4±1.3	60.3±1.5	2.1±1.9	
DHFR	Unified	52.3±2.3	18.7±1.1	66.9±1.4	12.1±1.2	8.3↑
	Single	43.5±2.4	8.4±0.7	54.0±1.7	6.3±1.5	

Table 3: Clustering results at different granularities.

4 Conclusion

In this paper, we propose GFM-UC, a graph foundation model for unified clustering that is trained once and then applied to joint node- and graph-level clustering across domains. Through agent subgraphs, the framework brings both task granularities into a common representation space. High-confidence prototypes guide sample generation during pre-training, and distribution-aware alignment adapts the clustering head to the target domain without labels. GFM-UC addresses the limited transferability and single-granularity design of existing methods, and extensive experiments validate its effectiveness. Future work will extend the framework to dynamic graphs and larger-scale pre-training across more domains.

Acknowledgments and Disclosure of Funding

This work was supported by the Tianjin Key Laboratory of Cognitive Computing and Application, the Science and Technology Major Project of the Science and Technology Program of the Xizang Autonomous Region (XZ202601ZD0010), and the National Natural Science Foundation of China (Nos. 62302333, U23B2053, 92370111 and 62272340).

References

[1] Ke Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, Xinwang Liu, Fuchun Sun, and Kunlun He. A survey of Knowledge Graph Reasoning on Graph Types: Static, Dynamic, and Multi-Modal. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9456–9478, 2024.

[2] Xiaobao Wang, Yujing Wang, Dongxiao He, Zhe Yu, Yawen Li, Longbiao Wang, Jianwu Dang, and Di Jin. Elevating Knowledge-Enhanced Entity and Relationship Understanding for Sarcasm Detection. *IEEE Transactions on Knowledge and Data Engineering*, 37:3365–3371, 2025.

[3] Ke Liang, Lingyuan Meng, Hao Li, Jun Wang, Long Lan, Miaomiao Li, Xinwang Liu, and Huaimin Wang. From Concrete to Abstract: Multi-View Clustering on Relational Knowledge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(10):9043–9060, 2025.

[4] Meng Liu, Ke Liang, Hao Yu, Lingyuan Meng, Siwei Wang, Sihang Zhou, and Xinwang Liu. Multiview Temporal Graph Clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 36(10):18383–18396, 2025.

[5] Meng Liu, Yue Liu, KE LIANG, Wenxuan Tu, Siwei Wang, Xinwang Liu, et al. Deep Temporal Graph Clustering. In *Proceedings of the International Conference on Learning Representations*, 2023.

[6] Tianchi Liao, Lele Fu, Jialong Chen, Zhen Wang, Zibin Zheng, and Chuan Chen. A swiss Army Knife for Heterogeneous Federated Learning: Flexible Coupling via Trace Norm. *Advances in Neural Information Processing Systems*, 37:139886–139911, 2024.

[7] Man-Sheng Chen, Pei-Yuan Lai, De-Zhang Liao, Chang-Dong Wang, and Jian-Huang Lai. Graph Prompt Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7):5794–5805, 2025.

[8] Ke Liang, Sihang Zhou, Meng Liu, Yue Liu, Wenxuan Tu, Yi Zhang, Liming Fang, Zhe Liu, and Xinwang Liu. Hawkes-Enhanced Spatial-Temporal Hypergraph Contrastive Learning Based on Criminal Correlations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8733–8741, 2024.

[9] Yongheng Xing, Liang Hu, Xinqi Du, Zhiqi Shen, Juncheng Hu, and Feng Wang. Ccdf-tap: A context-aware conflict detection framework for iot trigger-action programming with graph neural network. *IEEE Internet of Things Journal*, 11(19):31534–31544, 2024.

[10] Lin Li, Ziqi Jiang, Gefan Ye, Zhenqi He, Jiahui Li, Jun Xiao, Kwang-Ting Cheng, and Long Chen. Path-decoupled hyperbolic flow matching for few-shot adaptation. In *ICML*, 2026.

[11] Lin Li, Guikun Chen, Zhen Wang, Jun Xiao, and Long Chen. Compositional zero-shot learning via progressive language-based observations. In *ACM MM*, pages 3827–3836, 2025.

[12] Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. Structural Deep Clustering Network. In *Proceedings of the Web Conference 2020*, pages 1400–1410, 2020.

[13] Wenxuan Tu, Sihang Zhou, Xinwang Liu, Xifeng Guo, Zhiping Cai, En Zhu, and Jieren Cheng. Deep Fusion Clustering Network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9978–9987, 2021.

[14] Jingxin Liu, Xiangyan Tang, Renda Han, Wenxuan Tu, and Ruili Wang. Adaptive Feature Boosting and Distribution Refinement for Graph Clustering. *Pattern Recognition*, page 112309, 2025.

[15] Wenxuan Tu, Sihang Zhou, Xinwang Liu, Yue Liu, Zhiping Cai, En Zhu, Changwang Zhang, and Jieren Cheng. Initializing Then Refining: A simple Graph Attribute Imputation Network. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 3494–3500, 2022.

[16] Jingxin Liu, Renda Han, Wenxuan Tu, Haotian Wang, Junlong Wu, and Jieren Cheng. Federated Node-Level Clustering Network with Cross-Subgraph Link Mending. In *International Conference on Machine Learning*, pages 38540–38556, 2025.

[17] Jingxin Liu, Wenxuan Tu, Haotian Wang, Renda Han, Haoyi Li, Junlong Wu, and Xiangyan Tang. Causally-Aware Attribute Completion for Incomplete Federated Graph Clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 23730–23738, 2026.

[18] Jinyu Cai, Yi Han, Wenzhong Guo, and Jicong Fan. Deep Graph-level Clustering Using Pseudo-label-guided Mutual Information Maximization Network. *Neural Computing and Applications*, 36(16):9551–9566, 2024.

[19] Jinyu Cai, Yunhe Zhang, Jicong Fan, Yali Du, and Wenzhong Guo. Dual Contrastive Graph-Level Clustering with Multiple Cluster Perspectives Alignment. In *International Joint Conference on Artificial Intelligence*, pages 3770–3779, 2024.

[20] Jinyu Cai, Yunhe Zhang, Fusheng Liu, and See-Kiong Ng. Leveraging Diffusion Model as Pseudo-Anomalous Graph Generator for Graph-Level Anomaly Detection. In *Proceedings of the International Conference on Machine Learning*, pages 3782–3794, 2025.

[21] Renda Han, Junlong Wu, Wenxuan Tu, Jingxin Liu, Haotian Wang, and Jieren Cheng. Federated Graph-level Clustering Network with Attribute Inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 21585–21593, 2026.

[22] Xiaobao Wang, Renda Han, Ronghao Fu, and Di Jin. Federated graph-level clustering network with dual knowledge separation. In *The Fourteenth International Conference on Learning Representations*, 2026.

[23] Renda Han, Xiaobao Wang, Longbiao Wang, Wenxin Zhang, Ronghao Fu, Kaiming Wang, Zeyu Zhang, and Kuntharrgyal Khysru. A unified graph clustering network. In *Proceedings of the ACM Web Conference 2026*, pages 845–856, 2026.

[24] 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. *ArXiv Preprint ArXiv:2310.00149*, 2023.

[25] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. All in One: Multi-task Prompting for Graph Neural Networks. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2120–2131, 2023.

[26] Lianghao Xia and Chao Huang. Anygraph: Graph Foundation Model in the Wild. *ArXiv Preprint ArXiv:2408.10700*, 2024.

[27] Jianan Zhao, Hesham Mostafa, Michael Galkin, Michael Bronstein, Zhaocheng Zhu, and Jian Tang. Graphany: A foundation Model for Node Classification on Any Graph. *ArXiv Preprint ArXiv:2405.20445*, 29, 2024.

[28] Yufei He, Yuan Sui, Xiaoxin He, and Bryan Hooi. Unigraph: Learning a Unified Cross-domain Foundation Model for Text-attributed Graphs. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 448–459, 2025.

[29] Hezhe Qiao, Chaoxi Niu, Ling Chen, and Guansong Pang. Anomalygfm: Graph Foundation Model for Zero/few-shot Anomaly Detection. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 2326–2337, 2025.

[30] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. Gpt-gnn: Generative Pre-training of Graph Neural Networks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1857–1867, 2020.

[31] Aodong Li, Chen Qiu, Marius Kloft, Padhraic Smyth, Maja Rudolph, and Stephan Mandt. Zero-shot Anomaly Detection via Batch Normalization. In *Proceedings of the Conference on Neural Information Processing Systems*, volume 36, 2024.

[32] Renda Han, Xiaobao Wang, Luzhi Wang, Wenxin Zhang, Guangzhen Yao, and Hongxiang Liang. A graph foundation model for unified anomaly detection. In *Proceedings of the ACM Web Conference 2026*, pages 523–534, 2026.

[33] Chuancheng Song, Xixun Lin, Hanyang Shen, Yanmin Shang, and Yanan Cao. Uniform: Towards unified framework for anomaly detection on graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12559–12567, 2025.

[34] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of Benchmark Datasets for Learning with Graphs. *ArXiv Preprint ArXiv:2007.08663*, 2020.

[35] Gayan K Kulatilleke, Marius Portmann, and Shekhar S Chandra. SCGC: Self-supervised Contrastive Graph Clustering. *Neurocomputing*, 611:128629, 2025.

[36] Wei Ju, Siyu Yi, Kangjie Zheng, Yifan Wang, Ziyue Qiao, Li Shen, Yongdao Zhou, Xiaochun Cao, and Jiancheng Lv. Compactness and consistency: A conjoint framework for deep graph clustering. In *The Fourteenth International Conference on Learning Representations*, 2026.

[37] Wei Ju, Yiyang Gu, Binqi Chen, Gongbo Sun, Yifang Qin, Xingyuming Liu, Xiao Luo, and Ming Zhang. Glcc: A general Framework for Graph-level Clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4391–4399, 2023.

[38] Mengling Hu, Chaochao Chen, Weiming Liu, Xinyi Zhang, Xinting Liao, and Xiaolin Zheng. Learning Uniform Clusters on Hypersphere for Deep Graph-level Clustering. *ArXiv E-prints*, pages arXiv–2311, 2023.

[39] Zehong Wang, Zheyuan Zhang, Nitesh V Chawla, Chuxu Zhang, and Yanfang Ye. Gft: Graph Foundation Model with Transferable Tree Vocabulary. *Advances in Neural Information Processing Systems*, 37:107403–107443, 2024.

[40] 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 *Proceedings of the ACM on Web Conference 2025*, pages 1142–1153, 2025.

[41] Yun Zhu, Haizhou Shi, Xiaotang Wang, Yongchao Liu, Yaoke Wang, Boci Peng, Chuntao Hong, and Siliang Tang. Graphclip: Enhancing Transferability in Graph Foundation Models for Text-attributed Graphs. In *Proceedings of the ACM on Web Conference*, pages 2183–2197, 2025.

[42] Shuo Wang, Bokui Wang, Zhixiang Shen, Boyan Deng, and Zhao Kang. Multi-domain graph foundation models: Robust knowledge transfer via topology alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, pages 64806–64821, 2025.

[43] Tianqianjin Lin, Pengwei Yan, Kaisong Song, Zhuoren Jiang, Yangyang Kang, Jun Lin, Weikang Yuan, Junjie Cao, Changlong Sun, and Xiaozhong Liu. Langgfm: A large Language Model Alone Can Be a Powerful Graph Foundation Model. *ArXiv Preprint ArXiv:2410.14961*, 2024.

[44] Zhikai Chen, Haitao Mao, Jingzhe Liu, Yu Song, Bingheng Li, Wei Jin, Bahare Fatemi, Anton Tsitsulin, Bryan Perozzi, Hui Liu, et al. Text-space Graph Foundation Models: Comprehensive Benchmarks and New Insights. *Advances in Neural Information Processing Systems*, 37:7464–7492, 2024.

[45] Yuanfu Sun, Zhengnan Ma, Yi Fang, Jing Ma, and Qiaoyu Tan. Graphicl: Unlocking Graph Learning Potential in Llms Through Structured Prompt Design. In *Findings of the Association for Computational Linguistics*, pages 2440–2459, 2025.

[46] Tianchi Liao, Lei Zhang, Jinghua Yang, Chuan Chen, and Zibin Zheng. CoaTr: A convolutional Autoregressive Tensor-Ring Decomposition Method for Sparse Spatio-temporal Traffic Data. *Neurocomputing*, 617:129006, 2025.

[47] Tianchi Liao, Lele Fu, Lei Zhang, Lei Yang, Chuan Chen, Michael K. Ng, Huawei Huang, and Zibin Zheng. Privacy-Preserving Vertical Federated Learning With Tensor Decomposition for Data Missing Features. *IEEE Transactions on Information Forensics and Security*, 20:3445–3460, 2025.

[48] Tianchi Liao, Ziyue Xu, Qing Hu, Hong-Ning Dai, Huaiwei Huang, Zibin Zheng, and Chuan Chen. FedBrb: A solution to the Small-to-Large Scenario in Device-Heterogeneity Federated Learning. *IEEE Transactions on Mobile Computing*, pages 1–14, 2025.

[49] Dayu Hu, Tagrid Abdullah N. Alshalali, Xu Xu, Por Lip Yee, Zaffar Ahmed Shaikh, Jing Yang, and Thippa Reddy Gadekallu. Pagm: Partially-aligned global and marginal multi-view contrastive clustering for facial recognition in consumer electronics. *IEEE Transactions on Consumer Electronics*, pages 1–1, 2025.

[50] Dayu Hu, Zhibin Dong, Ke Liang, Hao Yu, Siwei Wang, and Xinwang Liu. High-order topology for deep single-cell multi-view fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, 2024.

[51] Zhaowei Liu, Dong Yang, Shenqiang Wang, and Hang Su. Adaptive Multi-Channel Bayesian Graph Attention Network for Iot Transaction Security. *Digital Communications and Networks*, 10(3):631–644, 2024.

[52] Zhaowei Liu, Dong Yang, Yingjie Wang, Mingjie Lu, and Ranran Li. EGnn: Graph Structure Learning Based on Evolutionary Computation Helps More in Graph Neural Networks. *Applied Soft Computing*, 135:110040, 2023.

[53] Chengyang Zhou, Zijian Zhang, Chunxu Zhang, Hao Miao, Yulin Zhang, Kedi Lyu, and Juncheng Hu. Feddis: A causal disentanglement framework for federated traffic prediction. In *Proceedings of the ACM Web Conference 2026*, pages 7541–7551, 2026.

[54] Xunqiang Jiang, Tianrui Jia, Yuan Fang, Chuan Shi, Zhe Lin, and Hui Wang. Pre-training on large-scale heterogeneous graph. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 756–766, 2021.

[55] Qinglong Cao, Zhengqin Xu, Yuntian Chen, Chao Ma, and Xiaokang Yang. Domain-controlled prompt learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 936–944, 2024.

[56] Shengrui Li, Xueting Han, and Jing Bai. Adaptergnn: Parameter-efficient Fine-tuning Improves Generalization in Gnns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13600–13608, 2024.

[57] Jingyan Chen, Guanghui Zhu, Guansong Pang, Chunfeng Yuan, and Yihua Huang. Affinitytune: A prompt-tuning framework for few-shot anomaly detection on graphs. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 179–190, 2025.

[58] Yixin Liu, Shiyuan Li, Yu Zheng, Qingfeng Chen, Chengqi Zhang, and Shirui Pan. ARc: A generalist Graph Anomaly Detector with In-Context Learning. *ArXiv:2405.16771*, 2024.

[59] Chaoxi Niu, Hezhe Qiao, Changlu Chen, Ling Chen, and Guansong Pang. Zero-shot Generalist Graph Anomaly Detection with Unified Neighborhood Prompts. *ArXiv Preprint ArXiv:2410.14886*, 2024.

[60] Thomas N Kipf and Max Welling. Semi-supervised Classification with Graph Convolutional Networks. *ArXiv:1609.02907*, 2016.

[61] D.P. Kingma and J.B. Ba. Adam: A method for Stochastic Optimization. *ArXiv Preprint ArXiv:1412.6980*, 2014.

A Related Work

A.1 Deep Graph Clustering

Recent advances in deep graph clustering have focused on jointly modeling node attributes and graph topology with effective self-supervised learning strategies, spanning both node-level and graph-level settings. For node-level clustering, SDCN [12] bridges autoencoder representations with GCN embeddings via a delivery operator and enables interaction between feature and structure spaces. It further adopts a dual self-supervised mechanism to jointly optimize representation learning and clustering assignments, improving structural consistency. DFCN [13] extends this paradigm by introducing a dynamic fusion mechanism to adaptively integrate attribute and structure representations. It also refines clustering assignments through a self-supervised target distribution, leading to more robust and discriminative embeddings. ABR [14] enhances multi-view representation learning by adaptively reweighting local and global features. Meanwhile, it constructs a reliable target distribution via dynamic refinement of multiple soft assignments, improving clustering stability. CoCo [36] addresses redundancy by learning low-rank compact embeddings that remove noise while preserving intrinsic structure. In addition, it introduces cross-view consistency learning to align similarity distributions and enhance semantic representation. Furthermore, contrastive learning has been widely adopted, where SCGC [35] improves discriminability by maximizing agreement across augmented views. In contrast, graph-level clustering targets the grouping of multiple graphs based on their global characteristics, typically relying on hierarchical pooling or readout operations to aggregate node features into graph-level representations before performing clustering in the embedding space. For graph-level clustering, existing methods focus on learning expressive graph representations and designing clustering-friendly objectives. GLCC [37] proposes a general framework that unifies graph-level clustering by decoupling representation learning and clustering optimization, enabling flexible adaptation to different graph encoders. It further incorporates clustering-oriented objectives to improve the quality and separability of graph embeddings. UDGC [38] constrains graph representations onto a hypersphere manifold and encourages uniform distribution of clusters, which enhances inter-cluster separability and alleviates representation collapse. It also introduces normalization strategies to stabilize training and improve embedding geometry. DGLC [19] adopts a pseudolabel-guided mutual information maximization scheme, where clustering assignments are iteratively refined through self-supervision. This design enables more reliable global representation learning and improves clustering consistency across graphs. DCGLC [18] further explores dual contrastive learning by aligning multiple cluster perspectives, capturing complementary structural semantics across different views. By enforcing consistency across these perspectives, it enhances robustness and leads to more discriminative graph-level clustering results. However, such aggregation schemes often oversimplify complex structural patterns, potentially discarding fine-grained information and limiting their ability to capture diverse and hierarchical relationships across graphs.

A.2 GNN-based Graph Foundation Model

Recent advances in graph machine learning have highlighted GFM as a promising paradigm. It is pre-trained on graphs from one domain and supports broad transferability across downstream tasks with zero- or few-shot prompting. GFMs can generally be categorized into two types of transfer scenarios, which include cross-domain transfer and cross-task transfer. For cross-task methods, classic pre-training methods [30] typically learn general representations and transfer them to various downstream tasks. An improved version, PT-HGNN [54], is further pre-trained on large-scale heterogeneous graphs by jointly designing node- and schema-level contrastive tasks to capture transferable semantics. Subsequently, prompt learning [55] methods have been introduced to reformulate tasks via task-specific prompts. For cross-domain transfer methods, existing approaches focus on transferring knowledge across different graph distributions, where models trained on a source domain are adapted to target domains with distribution shifts, improving generalization under domain discrepancy. Early methods, such as All-in-One [25], address the negative transfer problem in cross-domain pre-training of a single model by extracting shared transferable knowledge from a unified heterogeneous graph dataset. One-for-All [24] unifies heterogeneous graph data as text-attributed graphs, maps them into a shared embedding space via language models, and standardizes node, link, and graph tasks through nodes-of-interest. Subsequently, AnyGraph [26] utilizes a Mixture-of-Experts architecture to address structural and feature heterogeneity, enabling fast cross-domain adaptation, Similarly, AdapterGNN [56] introduces highly expressive adapters for GNNs that preserve the knowledge of large pretrained

models, enabling effective adaptation to downstream tasks with only a small number of parameters. GraphAny [27] performs inference on unseen graphs as an analytical solution to a LinearGNN and further employs an attention mechanism over multiple LinearGNNs to enhance generalization. UniGraph [28] leverages text as a unifying medium to train a graph foundation model capable of generalizing to unseen graphs and tasks across diverse domains. GFT [39] treats computation trees from message passing as a transferable vocabulary to enable cross-task and cross-domain generalization. GraphCLIP [41] employs a self-supervised contrastive graph summary powered by LLM-generated data and invariant learning for pre-training, enabling strong cross-domain zero-shot and few-shot transferability for graph foundation models. MDGFM [42] adaptively balances structural and feature information while refining graphs for denoising and topology alignment to address topological disparities across graph domains and real-world issues. SAMGPT [40] uses structure tokens and dual prompts to enable cross-domain adaptation for text-free multi-domain graph pre-training. For GFMs designed for anomaly detection, AffinityTune [57] and AnomalyGFM [29] align learnable, data-independent normal and abnormal class prototypes with node representation residuals to capture dataset-specific patterns. Moreover, ARC [58] directly extracts data-specific knowledge from the target dataset using few-shot normal samples during inference. Similarly, UNPrompt [59] leverages latent node attribute predictability as a generalized anomaly measure and learns universal graph patterns through a normalized node attribute space.

B Details of Experiment Setup

B.1 Benchmark Datasets

We evaluate GFM-UC on a diverse collection of benchmark datasets widely used for both graph-level and node-level clustering tasks. All datasets are drawn from the publicly available TUDataset collection [34], which spans multiple application domains and exhibits substantial heterogeneity in structural patterns and semantic distributions. In particular, these datasets cover a wide range of real-world scenarios, including small-molecule chemistry, bioinformatics, computer vision, and social networks, thereby providing a comprehensive evaluation of clustering performance under diverse data distributions.

Specifically, we select COX2, PTC-MM, DHFR, and PROTEINS as representative medium-scale benchmarks, which differ in terms of graph topology complexity and attribute richness. To further evaluate the scalability and robustness of the proposed method under large-scale settings, we additionally include two larger datasets, PC-3 and SW-620, which contain significantly more samples and more complex structural variations.

As summarized in Table 4, these datasets exhibit diverse characteristics in terms of graph scale, structural complexity, and attribute completeness. This diversity makes them well-suited for evaluating clustering performance under incomplete and heterogeneous settings, providing a comprehensive and challenging testbed for the proposed framework.

- **COX2, DHFR**: A molecular graph datasets representing chemical compounds, where nodes correspond to atoms and edges denote chemical bonds. Both datasets contain continuous node attributes describing atomic properties.
- **PTC-MM**: A molecular dataset for carcinogenicity prediction. Unlike COX2 and DHFR, it does not provide explicit node attributes.
- **PROTEINS**: A bioinformatics dataset where each graph represents a protein structure, and nodes correspond to secondary structure elements.

To further assess scalability, we additionally include two large-scale datasets:

- **PC-3:** A large-scale molecular graph dataset derived from the PC-3 human prostate cancer cell line. Each sample corresponds to a chemical compound represented as a graph, where atoms are modeled as nodes and chemical bonds as edges. The dataset contains a substantial number of molecular graphs.
- **SW-620:** A large-scale molecular graph dataset collected from the SW-620 human colorectal cancer cell line. Similar to PC-3, each sample is represented as a molecular graph with atoms as nodes and bonds as edges. The dataset is characterized by its large sample size, resulting in purely structure-based graph representations.

Table 4: Statistics of benchmark datasets used in our experiments. "SM" and "BIO" denote small molecule and bioinformatics, respectively. "G.Clu" and "N.Clu" denote the number of clusters of graph-level and node-level, respectively.

Dataset	Domain	#Graphs	#G. Clu	#N. Clu	Avg. Nodes	A. Edges	N. Attr
COX2	SM	467	2	8	41.22	43.45	+3
PTC-MM	SM	336	2	20	13.97	14.32	-
DHFR	BIO	756	2	9	42.43	44.54	+3
PROTEINS	BIO	1113	2	3	39.06	72.82	+1
PC-3	SM	27,509	2	45	26.36	28.49	-
SW-620	SM	40,532	2	65	26.06	28.09	-

B.2 Baseline Methods

We adapt all end-to-end deep clustering models to the graph foundation model setting through appropriate modifications. For label-dependent GFMs, we replace supervision with reconstruction-based loss to eliminate reliance on labels. In addition, we tailor other baseline models toward clustering-oriented objectives to ensure fair comparisons. The following provides detailed descriptions of the baselines not discussed in the related work section.

- **Lang-GFM** reformulates graph learning as a pure language modeling problem by converting graphs into textual representations and directly applying large language models for downstream tasks. It unifies multiple graph tasks in a text-to-text paradigm without using graph neural networks. However, it heavily depends on graph textualization and may lose fine-grained structural information.
- **TS-GFM** maps graph data into a shared text space via language models and then leverages graph neural networks for prediction. It enables heterogeneous graphs to be aligned in a unified semantic space while preserving structural inductive bias. However, it still relies on GNN-based architectures and is mainly effective on text-attributed graphs.
- **GPC** uses structured prompts and in-context learning to perform graph reasoning with large language models without training. It encodes graph structure into prompts and relies on few-shot examples for task adaptation. However, its performance is highly sensitive to prompt design and does not learn explicit graph representations.

B.3 Evaluation Metrics

To comprehensively evaluate clustering performance, we adopt three widely used metrics, including Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI). These metrics assess clustering quality from complementary perspectives, including label alignment, information-theoretic consistency, and pairwise agreement.

Accuracy (ACC). ACC measures clustering accuracy by finding the optimal one-to-one mapping between predicted cluster assignments and ground-truth labels via the Hungarian algorithm:

$$\text{ACC} = \max_{\mathcal{M}} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{y_i = \mathcal{M}(c_i)\}, \tag{14}$$

where y_i and c_i denote the ground-truth and predicted labels of sample i, respectively, and $\mathcal{M}$ is the optimal permutation mapping.

Normalized Mutual Information (NMI). NMI evaluates the mutual dependence between clustering assignments and ground-truth labels:

$$\text{NMI} = \frac{2 \cdot I(Y; C)}{H(Y) + H(C)}, \tag{15}$$

where $I(Y; C)$ denotes the mutual information between Y and C, and $H(\cdot)$ represents entropy. NMI ranges from 0 to 1, with higher values indicating better agreement.

Adjusted Rand Index (ARI). ARI measures the similarity between two assignments by considering all sample pairs and correcting for random chance:

$$\text{ARI} = \frac{\text{RI} - \mathbb{E}[\text{RI}]}{\max(\text{RI}) - \mathbb{E}[\text{RI}]}, \tag{16}$$

where RI denotes the Rand Index. ARI ranges from -1 to 1, where higher values indicate better clustering performance.

Discussion. These metrics jointly provide a comprehensive evaluation of clustering performance. Specifically, ACC reflects the best label alignment under permutation, NMI captures global information consistency, and ARI corrects for random agreement.

B.4 Implementation Details

Software Environment All experiments are implemented in Python 3.10, with the core code developed using PyTorch 2.4 for model training and inference, ensuring reproducibility and transparency.

Hardware Environment All experiments are performed on a workstation equipped with an Intel Core i9-13900K CPU, an NVIDIA GeForce RTX 4090 GPU (24GB memory), and 64GB RAM, with data stored on a 2TB NVMe SSD. This setup enables efficient training on large-scale graph datasets.

Model Architecture GFM-UC adopts a graph neural network (GNN) backbone integrated with a unified clustering architecture. Specifically, we employ a standard Graph Convolutional Network (GCN) [60] as the feature extraction module. The learned node representations are then fed into the unified clustering module, which jointly performs node-level and graph-level clustering.

Optimization The model is optimized using the Adam optimizer [61] with an initial learning rate of 1e-3. A learning rate scheduler is applied to decay the learning rate by a factor of 0.1 every 20 epochs. The batch size is set to 128, and training is conducted for a maximum of 100 epochs.

C Theorem Proof

Theorem 1 (KL Barycenter of Node-wise Masks)**.** *Let $\{\mathbf{m}_i\}_{i=1}^N$ be a set of probability vectors defined on the simplex Δ^{d-1}, where each $\mathbf{m}_i \in \mathbb{R}^d$ satisfies $m_{i,j} \geq 0$ and $\sum_{j=1}^d m_{i,j} = 1$. The KL barycenter is defined as*

$$\mathbf{p}^\star = \arg\min_{\mathbf{p}\in\Delta^{d-1}} \sum_{i=1}^N \mathrm{KL}(\mathbf{p} \,\|\, \mathbf{m}_i) \tag{17}$$

admits the closed-form solution:

$$p_j^\star = \frac{\exp\left(\frac{1}{N}\sum_{i=1}^N \log m_{i,j}\right)}{\sum_{k=1}^d \exp\left(\frac{1}{N}\sum_{i=1}^N \log m_{i,k}\right)}. \tag{18}$$

Proof. We begin by expanding the KL divergence term:

$$\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{m}_i) = \sum_{j=1}^d p_j \log \frac{p_j}{m_{i,j}}. \tag{19}$$

Substituting into the objective, we obtain:

$$\begin{aligned}\mathcal{L}(\mathbf{p}) &= \sum_{i=1}^N \sum_{j=1}^d p_j \log \frac{p_j}{m_{i,j}} \\ &= \sum_{j=1}^d p_j \sum_{i=1}^N \log p_j - \sum_{j=1}^d p_j \sum_{i=1}^N \log m_{i,j}.\end{aligned} \tag{20}$$

Since $\sum_{i=1}^{N} \log p_j = N \log p_j$, the objective becomes:

$$\mathcal{L}(\mathbf{p}) = N \sum_{j=1}^{d} p_j \log p_j - \sum_{j=1}^{d} p_j \sum_{i=1}^{N} \log m_{i,j}. \tag{21}$$

We introduce the Lagrangian to enforce the simplex constraint:

$$\mathcal{J}(\mathbf{p}, \lambda) = N \sum_{j=1}^{d} p_j \log p_j - \sum_{j=1}^{d} p_j \sum_{i=1}^{N} \log m_{i,j} + \lambda \left(\sum_{j=1}^{d} p_j - 1 \right). \tag{22}$$

Taking derivative with respect to p_j, we obtain:

$$\frac{\partial \mathcal{J}}{\partial p_j} = N(1 + \log p_j) - \sum_{i=1}^{N} \log m_{i,j} + \lambda = 0. \tag{23}$$

Rearranging terms gives:

$$\log p_j = \frac{1}{N} \sum_{i=1}^{N} \log m_{i,j} - \frac{1+\lambda}{N}. \tag{24}$$

Let $C = \exp\left(-\frac{1+\lambda}{N}\right)$, we obtain:

$$p_j = C \cdot \exp\left(\frac{1}{N} \sum_{i=1}^{N} \log m_{i,j} \right). \tag{25}$$

Finally, enforcing $\sum_j p_j = 1$ yields the normalization constant:

$$C^{-1} = \sum_{k=1}^{d} \exp\left(\frac{1}{N} \sum_{i=1}^{N} \log m_{i,k} \right). \tag{26}$$

Thus, the optimal solution is:

$$p_j^\star = \frac{\exp\left(\frac{1}{N} \sum_{i=1}^{N} \log m_{i,j} \right)}{\sum_{k=1}^{d} \exp\left(\frac{1}{N} \sum_{i=1}^{N} \log m_{i,k} \right)}. \tag{27}$$

This completes the proof. □

D Performance Comparison on Large-Scale Datasets

Performance comparison experimental results on large-scale datasets are shown in Table 5.

E Limitations

GFM-UC focuses on unified clustering with cross-domain generalization and is mainly designed for clustering scenarios. Other downstream tasks, such as anomaly detection, are not explicitly studied in this work.

F Broader Impacts

This work studies general graph representation learning, federated graph clustering, and clustering methods, without targeting specific applications or deployment scenarios. Its impact is mainly positive in broad data analysis settings, and no societal risks are expected.

Table 5: Node-level and graph-level clustering performance comparison on the PC-3 and SW-620 datasets.

Dataset	Method	Node-level			Graph-level		
		ACC	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
PC-3	ABR	36.5±1.6	5.3±1.2	4.1±1.2	N/A	N/A	N/A
	SAMGPT	42.3±2.4	8.5±2.3	6.7±2.0	52.6±1.9	0.1±0.0	1.0±0.1
	MDGFM	39.6±2.2	0.4±0.1	0.6±0.2	55.4±3.0	1.2±0.5	1.3±0.1
	Ours	**44.1±1.3**	**12.9±1.6**	**15.3±1.2**	**64.3±2.4**	**5.7±2.1**	**3.6±1.8**
SW-620	DCGLC	N/A	N/A	N/A	53.3±1.7	0.7±0.2	0.1±0.1
	GPC	N/A	N/A	N/A	54.0±2.0	7.9±1.1	8.7±1.4
	SAMGPT	26.4±3.3	1.2±0.1	3.5±0.3	53.5±2.4	-0.0±0.1	0.0±0.2
	MDGFM	18.7±4.8	6.9±4.1	0.5±0.1	OOM	OOM	OOM
	Ours	**40.4±1.6**	**8.3±2.5**	**6.4±2.0**	**56.7±3.2**	**8.0±1.8**	**9.3±2.4**

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the contributions and scope of this paper.

   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: We discussed the limitations in the Appendix.

   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: [N/A]

   Justification: This paper does not claim theoretical contributions. However, we provide theoretical analyses for several key results in the Appendix.

   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: The paper fully disclose all the information needed to reproduce the main experimental results.

   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: The paper provides open access to the data, with sufficient instructions to faithfully reproduce the main experimental results.

   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 the Experiment Setup section and the Appendix for more details.

   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: We report the average performance over multiple runs with different random seeds, together with ±1 standard deviation as error bars to quantify the stochastic variability of the training process.

   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: We provide detailed descriptions of the experimental environment, including hardware and software configurations. Specifically, experiments are conducted on GPU-based platforms, and all implementation details are reported to facilitate reproducibility. Although we do not explicitly report memory consumption or exact wall-clock training time, we provide a thorough computational complexity analysis of the proposed method to characterize its efficiency.

   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: This work complies with the NeurIPS Code of Ethics. The research does not involve human subjects, personal data, or sensitive information, and no ethical concerns are raised by the datasets or experimental procedures used in this paper.

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: [Yes]

Justification: We provide the discussion of broader impacts in the Appendix.

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: The paper poses no such risks.

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: [N/A]

    Justification: The creators or original owners of the assets used in this paper are properly credited, and the corresponding licenses and terms of use are explicitly stated and fully respected.

    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: The paper does not release new assets.

    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: The paper does not involve crowdsourcing nor research with human subjects.

    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: The paper does not involve crowdsourcing nor research with human subjects.

    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: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    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.