GFMATE: EMPOWERING GRAPH FOUNDATION MODELS WITH PRE-TRAINING-AGNOSTIC TEST-TIME PROMPT TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph prompt tuning has shown great potential in graph learning by introducing trainable prompts to enhance the model performance in conventional single-domain scenarios. Recent research has extended graph prompt methods to Graph Foundation Models (GFMs), aiming to improve their cross-domain generalisability from source domains to an unseen target domain by tuning auxiliary prompts using few-shot samples. Despite their progress, most existing GFM prompt methods embed domain-specific information from source domains into prompts, which serve either as input to GFMs or encoded during the GFM pre-training process. This entanglement of prompts with specific source domains and particular GFM pre-training strategy restricts their generalisability to target domains and different GFMs. Furthermore, existing methods merely rely on few-shot data for prompt tuning, neglecting the rich information in unlabelled target domain test data. Motivated by these insights, this paper aims to empower **GFM**s with a pre-training-**a**gnostic **te**st-time graph prompt tuning framework, named **GFMate**. GFMate introduces a centroid prompt and a layer prompt applied after pre-training on target domains, avoiding entanglement with the source domains and model pre-training. In addition, a test-time complementary learning objective is devised to exploit both labelled and unlabelled target domain data for effective test-time prompt tuning. Extensive experiments on 12 benchmark datasets across diverse domains demonstrate the superior performance and efficiency of GFMate, achieving improvements of up to 30.63%. Code will be released upon acceptance.

1 INTRODUCTION

Graph prompt tuning has demonstrated notable potential in single-domain graph supervised learning (Liu et al., 2023b; Fang et al., 2024; Zi et al., 2024; Sun et al., 2023; Chen et al., 2025), where auxiliary prompts are supervised tuned to improve the performance of backbone models, such as GPPT (Sun et al., 2022) and ProNoG (Yu et al., 2024b). These prompts are typically designed as additive or multiplicative, learnable vectors on input features or encoded embeddings and are supervised trained within a single domain. Recent developments have further extended prompts to Graph Foundation Models (GFMs) (Zhao et al., 2024a), such as MDGPT (Yu et al., 2024c), SAMGPT (Yu et al., 2025) and MDGFM (Wang et al., 2025). Generally, prompts are jointly pre-trained with the GFM on source domains (e.g., financial network and citation network), and subsequently fine-tuned with few-shot samples on an unseen target domain (e.g., social network). This cross-domain scenario greatly enhances the transferability of graph learning models, which is the focus of this paper.

Unlike single-domain supervised graph prompting, applying prompts in cross-domain GFMs, where the target domain is unseen during pre-training, faces two primary challenges: **(i) Existing GFM prompt designs are generally pre-training–entangled on a specific set of source domains and are not easily generalised to unseen target domains**. For example, MDGFM (Wang et al., 2025) injects domain tokens into source domain graphs during pre-training, while SAMGPT (Yu et al., 2025) and MDGPT (Yu et al., 2024c) incorporate domain-related prompt vectors into the encoding process of GFM during pre-training. These GFM prompts are entangled with a set of source-domain graphs, as illustrated in Figure 1 (a). However, domain-specific information learnt on the prompts cannot be straightforwardly transferred from the source domains to the target domains, as the target domain is unseen in GFM cross-

domain scenarios, resulting in substantiate different graph structural and feature distribution (Zhao et al., 2024a; Jin et al., 2023). Additionally, these **pre-training–entangled prompts cannot be easily generalised across different pre-trained models**. For instance, the prompts in MDGFM (Wang et al., 2025) are jointly pre-trained with GFMs by a specifically designed graph contrastive learning objective conditioned on the source domain graph structure and cannot be directly applied to a GFM pre-trained with alternative strategies (e.g., link prediction).

Moreover, **(ii) current graph prompt tuning paradigm relies solely on few shots while neglecting abundant unlabelled samples despite their availability.** In the widely applied few-shot scenarios, GFM prompt vectors are optimised on a target domain graph where few-shot labelled nodes and unlabelled test nodes both exist. As illustrated in Figure 1 (a), the abundant unlabelled nodes from the target domain are available during prompt tuning but only being exploited passively as neighbouring contexts during message passing (Sun et al., 2022; Liu et al., 2023b; Fang et al., 2024; Chen et al., 2025; Yu et al., 2024b; Sun et al., 2023; 2025; Yu et al., 2024c; 2025). This leaves the distribution shift between the limited few-shot data and the abundant unlabelled test data unaddressed, preventing the GFM prompts from adequately capturing target-domain knowledge.

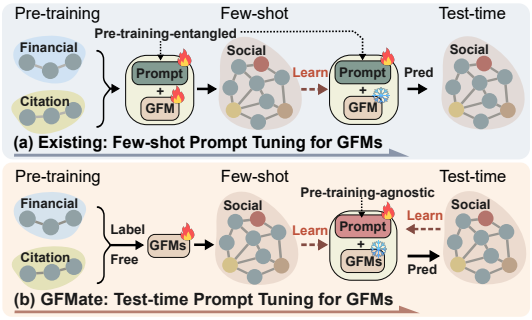


Figure 1: Comparison between GFMate and existing GFM prompts. (a) Existing methods entangle prompts with a specific set of source domains and GFM pre-training strategies, limiting generalisability across domains and models. (b) GFMate instead proposes pre-training-agnostic test-time prompts to learn from all target-domain data.

In light of the above observations, designing GFM prompts for cross-domain scenarios should satisfy two key properties: (i) the prompt should be **pre-training-agnostic**, placing emphasis on the GFM downstream stage without relying on prior assumptions about source domains or specific pre-training strategies; (ii) the **rich information contained in target-domain test data** should be effectively exploited to learn the prompt, thereby enabling improved GFM adaptation to the unseen target domains. To this end, a novel GFMate framework is proposed in this paper. GFMate introduces a centroid prompt and a layer prompt only after pre-training to achieve generalisability across domains and different GFMs. To further exploit the rich information in both few-shot labelled training nodes and unlabelled testing nodes in target domains for GFM prompt tuning, a test-time graph complementary learning objective is proposed. Extensive experiments on node and graph classifications across 12 datasets from diverse domains demonstrate the superior performance and efficiency of GFMate, achieving performance improvements of up to **30.63%**. Our contributions are:

- A novel **pre-training-agnostic prompt paradigm** is proposed to enhance GFMs with improved generalisability, allowing the prompts to be generalised across target domains and GFMs.
- A **test-time graph complementary learning** objective is proposed to exploit both labelled and unlabelled target domain data for effective test-time GFM prompt tuning.
- Extensive experiments across 12 graph datasets from diverse domains verify that GFMate **achieves state-of-the-art performance with significant efficiency and generalisability gains.**

2 Preliminary

A graph is denoted as $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are sets of nodes and edges, which can also be represented as the feature $\boldsymbol{X} \in \mathbb{R}^{N \times d}$ and adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$, where N is the number of nodes and d is the feature dimension. A graph foundation model is pre-trained on source domain graphs and adapted to target domains. Generally, target domains are unseen during pre-training.

Few-shot Classification. Most GFMs focus on few-shot node and graph classification tasks (Zhao et al., 2024a; Yu et al., 2024c; 2025; Wang et al., 2025; Yuan et al., 2025). In few-shot node classification, each node $v \in \mathcal{V}$ is assigned a class $y \in \mathcal{Y}$, where $\mathcal{Y}$ is the set of node classes. For graph classification over a set of graphs $\mathcal{G}$, each graph $G \in \mathcal{G}$ receives a label $y \in \mathcal{Y}$, where $\mathcal{Y}$ denotes the graph-level classes. An m-shot task provides m labelled instances per class.

Definition 1: Prompt Tuning for GFMs in Few-shot Classification (Figure 1 (a)). Given a pre-trained $\text{GFM}_{\boldsymbol{\theta}^*}$ and associated prompts $\mathcal{B}^*_{\text{Pre}}$ optimised with a set of source training graphs $\mathcal{G}_{\text{Pre}}$ on

task $\mathcal{L}_{\text{Pre}}$, current GFM prompt tuning aims to minimise the downstream task loss $\mathcal{L}_{\text{Fs}}$ by tuning the prompts $\mathcal{B}_{\text{Fs}}$ only using few shots $(\mathcal{V}_{\text{Fs}}, \mathcal{Y}_{\text{Fs}})$ from a target domain graph $\mathcal{G}_{\text{Tar}}$. The objective is:

$$\arg\min_{\mathcal{B}_{\text{Fs}}} \mathcal{L}_{\text{Fs}} \left(\text{GFM}_{\boldsymbol{\theta}^*}, \mathcal{B}_{\text{Fs}}, \mathcal{V}_{\text{Fs}}, \mathcal{Y}_{\text{Fs}}\right), \quad \mathcal{B}_{\text{Fs}} \leftarrow \mathcal{B}^*_{\text{Pre}} \tag{1}$$

Remark 1: Pre-training-entangled GFM Prompts. Generally, prompts are typically additive or multiplicative vectors applied to nodes or embeddings (Liu et al., 2023b; Chen et al., 2025; Yu et al., 2024b; Wang et al., 2025; Yu et al., 2024c; 2025), e.g., in GCOPE (Zhao et al., 2024a), $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{b}$, while in SAMGPT (Yu et al., 2025), $\boldsymbol{h}' = \boldsymbol{h} \odot \boldsymbol{b}$, where $\odot$ denotes element-wise multiplication. Most existing methods learn prompts on source domains during pre-training and reuse or fine-tune them by the few-shot from target domain (Wang et al., 2025; Yu et al., 2024c; 2025), which corresponds to using $\mathcal{B}^*_{\text{Pre}}$ to initialise or replace $\mathcal{B}_{\text{Fs}}$ where $\mathcal{B}^*_{\text{Pre}}$ is optimised with the GFM by a pre-training loss $\mathcal{L}_{\text{Pre}}$. Such prompts are entangled with a specific set of source domains and pre-training strategy, limiting their generalisability to arbitrary target domains and GFMs pre-trained by other objectives.

Remark 2: Passive Utilisation of Unlabelled Nodes in GFM Prompts. Although few-shot learning for graph prompt tuning typically and only utilises the supervision signals from the few labelled nodes, the unlabelled nodes in the target graph $\mathcal{G}_{\text{Tar}}$ are still required to be accessible to provide neighbourhood information for the labelled nodes (Wang et al., 2025; Yu et al., 2024c; 2025).

Discussion on Existing Pre-training-entangled Prompts: Existing pre-training-entangled prompt design may suffer from several limitations in GFM cross-domain scenarios:

(i) Limited cross-domain generalisability. Existing prompts assume the source domain contains useful information for the target domain. This assumption does not always hold in GFM cross-domain scenarios, as the target domain is unseen. For example, MDGPT (Yu et al., 2024c) assumes the target domain is closely related to the source domain, thus pre-trains source domain tokens, which may not generalise well on an unseen target domain with a different distribution.

(ii) Limited cross-model generalisability. Existing prompts are often tied to specific model architectures and pre-training strategies, limiting their generalisability to other GFMs. For example, the domain prompts in SAMGPT (Yu et al., 2025) are pre-trained with a GCN using a graph contrastive task and cannot work with models pre-trained by link prediction tasks.

(iii) Vulnerability to source domain bias. Existing prompts are jointly trained with the backbone and may suffer from training issues such as source domain imbalances, resulting in over-fitting to the large-scale source domain and making it hard to generalise to unseen target domains. In contrast, a pre-training-agnostic prompt, constructed and adapted solely at test time on the target domain, can avoid source domain bias and the dominance issues present in previous methods.

(iv) Neglect of unlabelled target domain data. Existing GFM prompt methods (Yu et al., 2024c; 2025; Wang et al., 2025) directly apply the conventional few-shot fine-tuning in previous single-domain prompting methods (Liu et al., 2023b) to the GFM cross-domain setting while ignoring the valuable information contained in the abundant unlabelled data in the target domain. This neglect leaves the train-test distribution shift unresolved, hindering the effectiveness of GFM adaptation.

Empirical Observations of Current GFM Prompts: For existing GFM prompts, two key issues arise during the downstream stage on target domains, limiting their cross-domain generalisability.

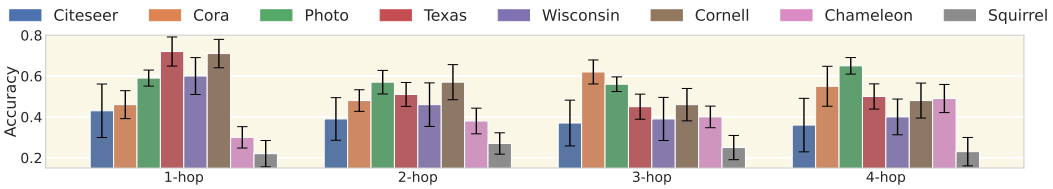
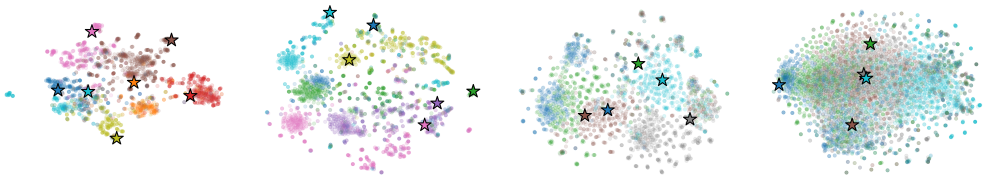


Figure 2: Performance of the embeddings extracted from each layer of a fixed pre-trained GFM varies significantly across different domains.

(i) Hop-aggregation performance variation across target domains. A key issue in cross-domain GFM prompt methods is that different target domain graphs exhibit distinct node neighbourhood patterns, leading to substantially varying performance. As illustrated in Figure 2, existing GFM prompts, which are pre-training-entangled and conditioned on source domains, fail to account for these data-level variations in target domains. Therefore, there is an urgent need for pre-training-agnostic GFM prompts that prioritise learning from the target domain to improve generalisability.



(a) Cora. (b) Citeseer. (c) Squirrel. (d) Chameleon.

Figure 3: Target domain node embedding visualisation by t-SNE. Stars denote one-shot embeddings from the existing GFM prompt method SAMGPT, which do not align with the test distribution. The GFM is pre-trained on all domains except the one for the target domain in a cross-domain scenario.

(ii) Train-test distribution shift within target domain. Existing GFM prompt methods, represented by SAMGPT (Yu et al., 2025), classify target-domain nodes by embedding similarity with encoded few shots (denoted as stars) as in Figure 3. It can be observed that in unseen target domains, the GFM with pre-training entangled prompts leads to unaligned embeddings between the few shots and the testing samples. This results in classification error, indicating that the generalisation of pre-training entangled prompts from source domains to unseen target domains is not satisfactory. Moreover, existing few-shot prompt fine-tuning is highly sensitive to distribution shift, since the limited few shots may inadequately capture the underlying distribution of the target domain data. While existing GFM prompt methods fail to account for distribution shifts in the target domain, learning pre-training-agnostic prompts from both few-shot and test data in the target domain can better capture the underlying test data distribution, thereby improving test-time performance.

These observations motivate our method design of pre-training-agnostic prompts on the target domain and the test-time learning objectives to capture the test distribution for effective prompt tuning.

3 METHOD: GFMATE

This section introduces the GFMate framework in Figure 4, with focus on the following problem:

Definition 2: Test-time Prompt Tuning for GFMs in Few-shot Classification (Figure 1 (b)). Given a pre-trained GFM$_{\boldsymbol{\theta}^*}$ optimised on source domain graphs $\mathcal{G}_{\text{Pre}}$ by pre-training task $\mathcal{L}_{\text{Pre}}$, test-time prompt tuning for GFMs aims to minimise the downstream loss $\mathcal{L}_{\text{Te}}$ by a set of learnable prompts $\mathcal{B}_{\text{Te}}$, with both few shots $(\mathcal{V}_{\text{Fs}}, \mathcal{Y}_{\text{Fs}})$ and unlabelled samples $\mathcal{V}_{\text{Tar}} \setminus \mathcal{V}_{\text{Fs}} = \{v \in \mathcal{V}_{\text{Tar}} \mid v \notin \mathcal{V}_{\text{Fs}}\}$ in the target domain graph $\mathcal{G}_{\text{Tar}}$ be exploited during GFM test-time. The objective is defined as:

$$\arg\min_{\mathcal{B}_{\text{Te}}} \mathcal{L}_{\text{Te}} \left(\text{GFM}_{\boldsymbol{\theta}^*}, \mathcal{B}_{\text{Te}}, \mathcal{V}_{\text{Fs}}, \mathcal{Y}_{\text{Fs}}, \mathcal{V}_{\text{Tar}} \setminus \mathcal{V}_{\text{Fs}}\right) \tag{2}$$

Remark 3: Pre-training-agnostic GFM Prompts. The prompt initialisation and training are unnecessary during pre-training, unlike existing approaches under Definition 1. The prompt relies on no prior assumption about the pre-trained model or strategies and is not constrained by the specific pre-training domain set, achieving better generalisability across domains and pre-trained GFMs.

Remark 4: Active Exploitation of Unlabelled Nodes. The unlabelled testing nodes are utilised to optimise the prompts with explicit learning strategies compared with existing work under Definition 1. This active exploitation of target domain testing data effectively mitigates the train-test distribution shift between the limited few-shot labelled nodes and the abundant testing nodes.

3.1 LABEL-FREE CROSS-DOMAIN PRE-TRAINING

GFMate does not require a specifically customised GFM architecture or pre-training strategies. In this work, a general and straightforward GFM is employed, built on standard GNN backbones such as GCN (Kipf & Welling, 2017) and GAT (Veličković et al., 2018). During pre-training, given a set of source domain graphs $\mathcal{G}_{\text{Pre}}$, Singular Value Decomposition (SVD) is adopted to align node features into a latent space with unified dimension following prior work (Zhao et al., 2024a). The model can then be pre-trained in a label-free manner by various self-supervised objectives, such as link prediction (Lu et al., 2021), graph contrastive learning (You et al., 2020) and deep graph infomax (Velickovic et al., 2019). Unless otherwise specified, the GFM is pre-trained by link prediction (Lu et al., 2021), eliminating the need for extensive prompt tuning during pre-training compared to existing GFM prompt methods (Yu et al., 2025; 2024c; Wang et al., 2025; Zhao et al., 2024a).

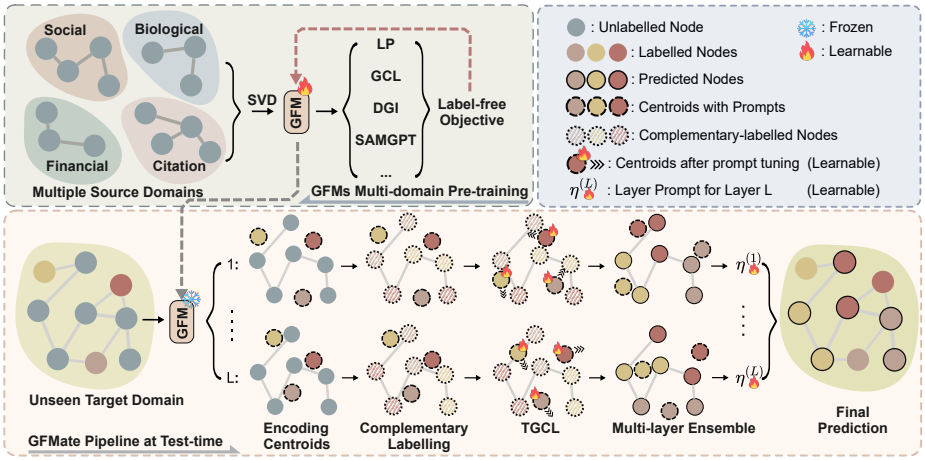


Figure 4: The overall framework for GFMate. The fixed GFMs can be pre-trained with self-supervised objectives in a label-free manner on source domains. GFMate is then applied at test-time to tune the pre-training-agnostic prompts, specifically the centroid prompts and the layer prompts, enabling the GFM to achieve better prediction results on unseen target domain graphs.

3.2 Centroids for Few-shot Classification

To perform downstream classification, centroids, which indicate the cluster centres of different classes of nodes, are used to compute similarities with testing samples and obtain classification results. Assume that the pre-trained GFM includes an L-layer GNN with optimal fixed parameters θ^* in hidden dimension d. At layer l, the target domain graph node embedding matrix is defined as $\boldsymbol{H}^{(l)} \in \mathbb{R}^{N\times d}$ where $\boldsymbol{H}^{(l)} = \text{GFM}^{(l)}_{\theta^*}(\boldsymbol{X}, \boldsymbol{A})$. The embedding of few-shot nodes is denoted by $\boldsymbol{h}^{(l)}_{\text{Fs}} \in \mathbb{R}^d$, and the embedding of testing nodes is denoted by $\boldsymbol{h}^{(l)}_{\text{Te}} \in \mathbb{R}^d$. The centroid $\boldsymbol{e}^{(l)}_c \in \mathbb{R}^d$ for class c at layer l can be initialised as the mean embedding of the few shots in class c, defined as:

$$\boldsymbol{e}^{(l)}_c = \frac{1}{|\mathcal{V}_{\text{Fs},c}|} \sum\nolimits_{i\in\mathcal{V}_{\text{Fs},c}} \boldsymbol{h}^{(l)}_{\text{Fs},i}. \tag{3}$$

The centroid matrix $\boldsymbol{E}$ collects all centroids $\boldsymbol{e}^{(l)}_c$ across layers l and classes c, with $\boldsymbol{E} \in \mathbb{R}^{L\times C\times d}$, and is used to compute embedding similarities for classification: $\hat{y}_i = \arg\max_c \text{sim}(\boldsymbol{h}^{(l)}_i, \boldsymbol{E}^{(l)})$ where sim denotes cosine similarity function. To be noticed, it is slightly overloaded to represent similarity between a node embedding vector and each row vector from the centroid matrix.

3.3 Pre-training-agnostic Prompt Design

This section introduces the prompt design in GFMate. Existing methods construct prompts by adding to or multiplying input graph features or node embeddings within GFMs, which tightly entangle them with specific GFM pre-training loss and source domains. Motivated by empirical observation in Section 2, GFMate develops pre-training-agnostic prompts that can effectively enhance GFM downstream adaptation without prior assumptions about source domains or pre-training strategies. The key idea is to avoid interfering with node embeddings trained during pre-training and instead focus on test-time prediction on the target domain. Specifically, GFMate constructs two pre-training-agnostic prompts: the centroid prompt and the layer prompt $\mathcal{B}_{\text{Te}} = (\boldsymbol{\beta}, \boldsymbol{\eta})$.

Centroid Prompts for Target-domain Centroid Movement. The centroid prompts are designed to adapt the centroid representations for improved test-time performance on the target domain. The intuition is to adjust the centroids derived from few-shot nodes toward directions that better align with the classification task in the unseen target domain. Let $\boldsymbol{\beta}^{(l)}_c \in \mathbb{R}^d$ denote a d-dimension learnable prompt for the centroid of class c at layer l, which is randomly initialised and added element-wise to the centroid $\boldsymbol{e}^{(l)}_c$, resulting in an refined centroid $\widetilde{\boldsymbol{e}}^{(l)}_c$, which can be defined as:

$$\widetilde{\boldsymbol{e}}^{(l)}_c = \boldsymbol{e}^{(l)}_c + \boldsymbol{\beta}^{(l)}_c. \tag{4}$$

With the centroids initialised by few-shot samples, the additive centroid prompt $\boldsymbol{\beta}$ provides the possibility **for centroids to move towards the real centre** of a cluster in target domains.

Layer Prompts for Multi-layer Ensemble Prediction. During test-time, the performance of neighbourhood aggregation across different hops in a pre-trained GFM varies substantially between target domains as observed in Section 2, leading to inconsistent layer-wise prediction accuracy. To adapt a pre-trained GFM to an arbitrary target domain, the predictions from all layers are ensembled by a layer prompt $\boldsymbol{\eta} \in \mathbb{R}^L$, which dynamically adjusts the contribution of each layer to exploit hop-aggregation patterns across different target domains. Specifically, given the refined centroid matrix by the centroid prompts for all classes at layer l, $\widetilde{\boldsymbol{E}}^{(l)}$, the final multi-layer ensemble prediction $\hat{y}$ for node i in an unseen target domain is defined by the class with the maximum ensemble probability:

$$\hat{y}_i = \arg\max_c \left[\text{Softmax} \left(\sum\nolimits_{l=0}^{L} \eta^{(l)} \cdot \text{sim}(\boldsymbol{h}_i^{(l)}, \widetilde{\boldsymbol{E}}^{(l)}) \right) \right]. \tag{5}$$

The layer prompt $\boldsymbol{\eta}$ allows our method to **adaptively learn from the hop-aggregation patterns of the target domain**, effectively ensembling the layer-wise predictions corresponding to different hop-aggregated representations to improve the GFM performance.

3.4 Test-time Graph Complementary Learning

Effective exploitation of few-shot and test data requires carefully designed test-time objectives. Directly tuning prompts with pseudo labels often leads to severe degradation due to inaccuracy (Appendix E.8). To address this, GFMate introduces a test-time graph complementary learning (TGCL) objective that jointly trains on few shots and complementary-labelled test nodes. The intuition is to learn from the predicted least similar class $\overline{y}$ rather than the most similar one $\hat{y}$. Since prediction performance varies across layers for target domains, complementary labels are derived by a layer-wise entropy-based strategy. Specifically, the layer-wise entropy score $H_i^{(l)}$ for test node i is:

$$H_i^{(l)} = -\sum\nolimits_{c=0}^{C} p_{i,c}^{(l)} \log p_{i,c}^{(l)}, \quad p_i^{(l)} = \text{Softmax}(\text{sim}(\boldsymbol{h}_i^{(l)}, \boldsymbol{E}^{(l)})). \tag{6}$$

The layer with the lowest average entropy score on the target domain graph, indicating the highest prediction confidence, is selected as the pivot layer $\hat{l}$. Then, the complementary label $\overline{y}$ is defined as the class with the lowest similarity at the pivot layer: $\overline{y}_i = \arg\min_c \text{sim}(\boldsymbol{h}_i^{(\hat{l})}, \boldsymbol{E}^{(\hat{l})})$, which is evaluated in the Appendix E.9 in comparison to the pseudo-labels by the fixed output layer. Once all testing nodes are complementary-labelled, the test-time learning loss $\mathcal{L}_{\text{Te}}$ is defined as:

$$\mathcal{L}_{\text{Te}} = -\sum_{l=0}^{L} \frac{1}{|\mathcal{V}_{\text{Te}}|} \sum_{(v_i, \overline{y}_i) \in \mathcal{V}_{\text{Te}}} \log \left(1 - p_{\overline{y}_i}^{(l)}\right), \text{ where } p_{\overline{y}_i}^{(l)} = \frac{\exp\left(\eta^{(l)} \cdot \text{sim}(h_i^{(l)}, \widetilde{e}_{\overline{y}_i}^{(l)})/\tau\right)}{\sum_{c=1}^{C} \exp\left(\eta^{(l)} \cdot \text{sim}(h_i^{(l)}, \widetilde{e}_c^{(l)})/\tau\right)} \tag{7}$$

where $\boldsymbol{\eta}$ is the learnable layer prompt and τ is a temperature that controls the smoothness of the probability distribution. $\mathcal{V}_{\text{Te}}$ refers to the testing nodes with test-time complementary labels.

Intuitively, the test-time learning loss **encourages centroids to be distant from testing samples being predicted to the most dissimilar class** (the complementary class $\overline{y}$), which in turn promotes increased similarity between centroids and testing samples from the other more similar classes. To evaluate the generalisation capability of the proposed test-time learning loss on complementary-labelled testing nodes, an excess risk bound is established:

Proposition 1 (Excess Risk Bound for Test-time Learning Loss). *Let $\mathcal{F}$ be the hypothesis space, where each predictor model $f \in \mathcal{F}$ refers to a GFM with prompts, defined as $f = (GFM_{\theta^*}, \mathcal{B})$ with pre-trained model parameters θ^* and learnable GFM prompts $\mathcal{B}$. Let $\overline{\mathcal{L}}$ be the test-time learning loss over all testing samples x and their complementary labels $\overline{y}$. The population risk can be defined as:*

$$R_{\overline{\mathcal{L}}}(f) = \mathbb{E}_{(x,\overline{y})}\left[\overline{\mathcal{L}}(f(x), \overline{y})\right].$$

Define the empirical risk minimiser as $\hat{f} = \arg\min_{f \in \mathcal{F}} \hat{R}_{\overline{\mathcal{L}}}(f)$. Then, with probability at least $1 - \delta$, the following generalisation bound on the excess risk of the test-time learning loss holds:

$$R_{\overline{\mathcal{L}}}(\hat{f}) - \min_{f \in \mathcal{F}} R_{\overline{\mathcal{L}}}(f) \leq 4C\ell_\rho \mathfrak{R}(\mathcal{F}) + 2\sqrt{\tfrac{\log(1/\delta)}{2N}}.$$

Here, C denotes the number of classes, ℓ_ρ is the Lipschitz constant of the complementary loss function $\overline{\mathcal{L}}$, and $\mathfrak{R}(\mathcal{F})$ is the Rademacher complexity (Shalev-Shwartz & Ben-David, 2014) of the hypothesis class $\mathcal{F}$, which consists of all predictor models with the same GFM parameter θ^* but varying prompts $\mathcal{B}$. $N = |\mathcal{V}_{\text{Te}}|$ is the number of complementary-labelled test samples, and δ is the confidence level. The proof and more detailed definition are deferred to Appendix D.

Intuitively, Proposition 1 suggests that a smaller number of classes or a greater number of complementary-labelled samples leads to a tighter risk upper bound of our GFM prompt. In light of this insight, GFMate exploits all testing samples in the test-time complementary learning process. These theoretical insights are further validated by empirical experiments in Appendix E.3 and E.4.

Meanwhile, by complementing the test-time learning loss on the testing samples, the few-shot learning loss $\mathcal{L}_{\text{Fs}}$ is introduced to learn from the few-shot labelled samples $\mathcal{V}_{\text{Fs}}$, defined as:

$$\mathcal{L}_{\text{Fs}} = -\sum_{l=0}^{L} \frac{1}{|\mathcal{V}_{\text{Fs}}|} \sum_{(v_i, y_i) \in \mathcal{V}_{\text{Fs}}} \log p_{y_i}^{(l)}, \quad \text{where} \quad p_{y_i}^{(l)} = \frac{\exp\left(\eta^{(l)} \cdot \text{sim}(h_i^{(l)}, \widetilde{e}_{y_i}^{(l)})/\tau\right)}{\sum_{c=1}^{C} \exp\left(\eta^{(l)} \cdot \text{sim}(h_i^{(l)}, \widetilde{e}_{c}^{(l)})/\tau\right)}. \tag{8}$$

Intuitively, the few-shot learning loss leverages the few-shot supervision signals to guide prompt optimisation by **maximising the similarity between centroids and labelled data with the same class while pushing the centroids away from different class samples.** Together, the final TGCL objective is defined as a convex combination of the few-shot and test-time learning losses to exploit all data from the target domain, which is optimised as:

$$\mathcal{L}_{\text{TGCL}} = \gamma \mathcal{L}_{\text{Te}} + (1 - \gamma)\mathcal{L}_{\text{Fs}}, \tag{9}$$

where γ is a hyperparameter in range $(0, 1)$ that regulates the relative contribution of each loss function. The centroid prompts and the layer prompts for layer l can be then optimised by:

$$\boldsymbol{\beta}_c^{(l)} = \boldsymbol{\beta}_c^{(l)} - \alpha \nabla_{\boldsymbol{\beta}_c^{(l)}} \mathcal{L}_{\text{TGCL}}(\boldsymbol{\beta}_c^{(l)}), \quad \eta^{(l)} = \eta^{(l)} - \alpha \nabla_{\eta^{(l)}} \mathcal{L}_{\text{TGCL}}(\eta^{(l)}), \tag{10}$$

where α is the learning rate. In this way, the centroid and layer prompts are jointly learned on both the few shots and the testing samples from the target domain, adequately capturing the target domain distribution, thereby facilitating effective GFM downstream adaptation.

3.5 Extend GFMate to Graph Classification

GFMate can be easily extended from node-level to graph-level classification by designing a subgraph classification task following (Zhao et al., 2024a). The subgraph embedding can be computed by averaging the node embeddings, while the subgraph label is determined by the central node. This process enables seamless integration of GFMate into both node- and graph-level classification tasks.

3.6 Complexity Analysis

The time complexity of GFMate consists of two components: (i) Test-time Graph Complementary Learning and (ii) Multi-layer Ensemble Prediction. Both require $\mathcal{O}(dLNC)$, where N is the number of nodes, E the number of edges, L the number of model layers, C the number of classes, and d the hidden dimension. The overall complexity of GFMate is $\mathcal{O}(dL(N + E + NC))$. In practice, because $C \ll d$, this can be approximated as linear in the target domain graph size $\mathcal{O}(dL(N + E))$.

4 Experiments

Datasets. Twelve graph benchmark datasets across diverse domains are adopted, varying in size up to one hundred thousand nodes and millions of edges. For **node classification**: (i) Social Network: Cornell, Texas, Wisconsin, Chameleon, and Squirrel, where nodes represent social entities; (ii) Citation Network: Cora, Citeseer, and Arxiv-year, where nodes denote academic papers and edges indicate citation relationships; (iii) Commercial System: Amazon-photo, where nodes represent products and edges denote co-purchase relationships. For **graph classification**: (i) Biological Network: BZR, COX2, and PROTEINS, where nodes are biological entities and edges represent interactions. Detailed dataset statistics are provided in the Appendix B. The cross-domain GFM node classification setting follows a one-versus-all setting in GCOPE (Zhao et al., 2024a), where one dataset serves as the unseen target domain and all remaining datasets serve as the source domains. Graph classification maintains fairness in comparisons.

Table 1: Cross-domain transfer learning performance of one-shot node classification. Results with different shots are in Appendix F. Average accuracy (%) over five runs is reported. The upper half shows single-domain training and testing setting, while the bottom half is cross-domain setting. ProNoG* refers to ProNoG that is implemented in a multi-domain pre-training and cross-domain testing setup (using SVD to unify dimensions) for a fair comparison with GFM methods. Results are marked as **best** and runner-up. OOM denotes out-of-memory on a 48 GB A6000 GPU, and "–" denotes that the official code has not been released for implementation on these datasets.

	Methods	Texas	Cornell	Wisconsin	Chameleon	Squirrel	Arxiv-year	Cora	Citeseer	Photo
Single-domain Training and Testing										
SL	GCN	38.82±9.79	24.58±10.09	43.36±13.17	24.30±6.59	19.96±5.89	18.64±6.91	29.85±8.98	33.39±11.86	47.09±5.81
	GAT	39.96±9.62	25.84±12.26	45.99±10.86	22.81±7.38	20.77±4.48	19.93±5.40	33.25±9.72	35.51±9.70	47.33±4.97
	SAGE	40.38±8.21	32.55±13.36	47.41±11.36	31.29±8.87	22.34±4.32	20.75±4.99	35.76±8.89	38.80±9.73	49.72±3.81
	GPR	37.60±9.54	30.77±13.42	49.56±15.58	28.44±6.65	21.06±6.14	10.81±9.94	38.99±15.77	29.77±13.22	43.39±4.66
	H2GCN	47.75±12.89	35.58±15.02	45.55±10.61	28.01±9.50	21.10±3.06	15.54±7.33	30.90±9.98	30.91±12.79	45.81±6.09
SSL+FT	LP+FT	36.93±11.90	23.88±11.43	41.37±14.22	25.34±7.19	20.04±5.61	17.94±6.06	35.59±9.74	34.92±12.08	48.82±6.55
	DGI+FT	34.46±12.92	22.89±11.32	38.89±15.77	25.78±7.34	20.85±6.57	18.06±6.22	32.38±8.86	33.96±11.57	45.17±7.34
	GCL+FT	28.81±15.32	20.56±13.36	35.70±17.73	24.69±8.82	19.97±5.62	15.53±8.89	33.27±9.50	36.05±13.44	47.33±5.89
SSL+Prompt	GPPT	44.47±10.88	29.74±8.32	35.16±9.07	29.91±6.48	21.16±5.95	19.96±7.63	40.62±8.69	39.79±10.67	50.19±7.74
	GPF	47.34±12.72	52.15±14.53	40.19±14.49	30.95±9.18	22.71±4.87	20.89±8.20	45.75±9.61	40.51±12.79	49.38±6.56
	ProNoG	55.31±12.92	48.49±11.54	46.29±17.74	31.19±8.09	24.25±4.79	OOM	56.54±12.33	37.79±13.35	47.72±6.60
	GraphPrompt	53.27±9.95	55.13±13.39	45.03±11.83	33.29±9.19	23.02±4.89	22.61±4.66	49.77±8.82	38.69±13.98	46.65±6.53
	DAGPrompt	68.27±10.02	59.11±10.04	50.49±11.59	37.79±6.62	25.67±6.34	23.08±8.14	54.88±9.24	47.24±9.59	52.96±6.07
	All-In-One	63.79±15.91	57.24±12.70	55.35±18.36	27.94±6.31	21.18±7.06	15.29±7.55	49.92±11.75	40.69±15.88	52.25±7.33
Multi-domain Pre-training and Cross-domain Fine-tuning then Testing										
GFM Methods	ProNoG*	48.25±9.60	37.18±10.04	39.70±12.32	26.48±8.69	22.10±5.52	OOM	40.07±13.35	35.56±10.63	45.28±7.02
	GCOPE	64.76±14.84	60.98±14.66	54.66±12.86	30.58±7.44	22.16±5.77	17.98±5.51	39.06±12.52	42.26±14.19	55.69±4.68
	MDGPT	59.76±12.44	54.19±13.08	50.40±15.07	28.04±4.28	24.41±7.01	OOM	44.52±11.39	41.98±12.24	54.96±10.25
	SAMGPT	66.79±10.77	59.34±9.82	52.29±14.40	38.12±8.90	25.75±6.29	OOM	52.83±12.04	47.76±10.55	56.33±9.04
	MDGFM	–	40.77±5.96	–	28.36±3.65	24.30±3.26	–	44.83±7.41	42.18±6.41	–
	BRIDGE	53.35±11.90	49.28±12.56	48.85±13.35	32.75±6.62	19.89±9.02	24.47±6.68	44.08±9.54	38.89±8.72	58.79±11.58
	RiemannGFM	58.60±15.27	46.35±11.92	47.32±16.58	29.68±9.95	20.13±8.58	OOM	37.91±16.13	38.02±9.58	49.69±13.32
	GFMate	**76.63**±7.81	**79.67**±8.47	**63.01**±10.78	**47.25**±6.11	**27.02**±6.22	**30.19**±3.65	**59.68**±5.37	**56.25**±13.33	**58.85**±2.17

Baselines. Twenty-one baselines are compared with GFMate. These include: (i) **Single domain supervised learning (SL)** such as GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), SAGE (Hamilton et al., 2017), H2GCN (Zhu et al., 2020a), and GPR (Chien et al., 2021); (ii) **Single domain self-supervised pre-training and fine-tuning (SSL + FT)**, including Link Prediction (LP) (Lu et al., 2021), DGI (Velickovic et al., 2019), and GCL (You et al., 2020). Fine-tuning (FT) is used to train the predictor; (iii) **Single domain pre-training and prompt tuning (SSL + Prompt)**, such as GPPT (Sun et al., 2022), All-In-One (Sun et al., 2023), ProNoG (Yu et al., 2024b), DAGPrompt (Chen et al., 2025), GraphPrompt (Liu et al., 2023b), and GPF (Fang et al., 2024). (iv) **Cross domain GFM pre-training and prompt tuning (GFMs)**, including prompt-based GCOPE (Zhao et al., 2024a), MDGFM (Wang et al., 2025), MDGPT (Yu et al., 2024c), SAMGPT (Yu et al., 2025), BRIDGE (Yuan et al., 2025), as well as manifold-based RiemannGFM (Sun et al., 2025). GraphAny (Zhao et al., 2025) and GTrans (Jin et al., 2023) are designed in full-shot settings in Appendix F. LLM-based GFMs are not applicable since our datasets cover various non-text situations.

Implementation. For all methods, GCN is the default backbone model. Analyses of GFMate on GFMs by different backbones are presented in Section E.2. For each target dataset, after obtaining the few-shot training samples, the remaining data are randomly split in a 1:9 ratio for validation and testing, following prior GFMs (Zhao et al., 2024a). This ensures a fair and consistent setting for all methods. Implementation details, including hyperparameter settings and pseudo-codes, are provided in the reproducibility statement in the Appendix B.

4.1 Overall Performance

The overall performance of GFMate is first evaluated on one-shot node classification tasks against all baselines. GFMate is integrated into GFMs with a default GCN backbone with LP pre-training. **Overall, GFMate outperforms all baselines across all domains.** (i) For multi-domain pre-training and cross-domain adaptation settings, in comparison with existing GFM-based methods, GFMate achieves the highest performance without requiring a delicate design of the underlying GFMs. Furthermore, GFMate prompts are pre-training-agnostic and do not require prior assumptions on source

domains and GFM pre-training strategies. **To be noticed, directly extending the traditional single-domain graph prompts to the GFM cross-domain setting results in suboptimal performance, as shown by ProNoG*.** (ii) Compared with traditional single-domain methods where the source and target domains are the same, applying prompt tuning on top of supervised training can enhance performance, as observed in DAGPrompt and All-In-One. More analyses are provided as follows: efficiency in Section 4.2 and Appendix E.1; GFMate with different backbones and pre-training strategies in Section 4.4 and Appendix E.2; robustness under domain and test-time distribution shifts in Appendix E.5; hyperparameter sensitivity in Appendix E.7; different few-shot settings, additional baselines, and graph classification in Section 4.6 and Appendix F.1 and F.2, respectively.

4.2 Efficiency Analysis

GFMate is highly efficient in both the convergence time and the GPU memory usage. To ensure a fair comparison, convergence time is measured only for downstream adaptation for all methods, after GFM pre-training has been completed and the target data are available. The prompts in GFMate are only integrated with centroids and the multi-layer predictions, offering a less complicated and lightweight design compared to existing methods that require learnable prompts for each source domain and target domain data (Yu et al., 2024c; 2025). Furthermore, the substantially reduced number of tunable parameters leads to lower memory utilisation. The maximum improvements in time and memory efficiency reach **98.24%** and **97.18%**, respectively, over state-of-the-art methods SAMGPT. With such significant efficiency gains, GFMate emerges as the most effective approach among existing GFM-based methods, highlighting the superiority of its test-time design. More detailed statistics are provided in the Appendix E.1.

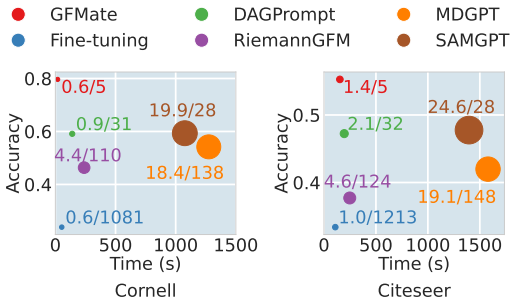


Figure 5: Efficiency analysis. The numbers in the figure indicate "GPU peak memory (k MB)/number of tunable parameters (k)". GFMate (red dot) has superior efficiency in time and memory while achieving higher performance.

Table 2: Robustness upon different test noise ratios (Left: Feature Shift, Right: Structural Shift).

	Texas	Cornell	Citeseer	Cora	Texas	Cornell	Citeseer	Cora
GFMate	76.63±7.81	79.67±8.47	56.25±13.33	59.68±5.37	76.63±7.81	79.67±8.47	56.25±13.33	59.68±5.37
w/ 10%	76.97±6.96	79.52±6.94	54.83±13.09	59.49±5.00	76.52±7.31	79.64±7.03	54.06±12.52	59.47±5.62
w/ 30%	72.58±9.68	79.10±7.91	54.48±12.50	59.29±5.59	75.96±8.24	79.62±7.15	53.82±12.30	59.06±6.13
w/ 50%	71.24±9.77	79.07±8.38	54.42±14.12	58.97±5.54	74.61±10.46	79.52±7.36	51.59±12.41	58.98±5.82

4.3 Robustness to Test-time Distribution Shift.

Experiments with both feature noise and structural perturbations at varying ratios on the target graphs are conducted to evaluate GFMate's capability in addressing distribution shifts at test time. Feature noise is introduced by randomly shuffling testing node features, and structural perturbations are introduced by randomly dropping edges connected to testing nodes, both disturbing the embeddings encoded by the fixed GFM. As shown in Table 2, even with a 50% perturbation rate, GFMate still maintains strong performance, especially on Cornell, Citeseer, and Cora, verifying its effectiveness in mitigating train–test distribution shift using both labelled and unlabelled target data.

4.4 On Generalisability

Applying GFMate on GFMs pre-trained with various methods can significantly enhance their performance. As a pre-training-agnostic prompt tuning method, GFMate can be seamlessly integrated with GFMs pre-trained by different objectives. In Table 3, with SSL-based pre-trained models

Table 3: Plug-in GFMate on different pre-training methods.

	Texas		Cornell		Citeseer		Cora	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
LP+FT	36.93±11.90	38.58±8.84	23.88±11.43	35.79±10.33	34.92±12.08	44.36±7.63	35.59±9.74	42.26±5.48
+GFMate	76.63±7.81	83.29±1.52	79.67±8.47	88.57±0.87	56.25±13.33	74.08±1.72	59.68±5.37	70.51±2.11
DGI+FT	34.46±12.92	40.33±7.81	22.89±11.32	34.48±9.97	33.96±11.57	41.77±7.90	32.38±8.86	39.98±6.33
+GFMate	74.80±9.97	81.38±3.94	75.06±10.37	88.19±4.04	52.31±12.96	73.70±5.42	54.08±9.04	62.25±3.52
GCL+FT	28.81±15.32	34.40±9.05	20.56±13.36	29.88±9.72	36.05±13.44	45.52±8.86	33.27±9.50	40.18±5.66
+GFMate	69.82±11.36	76.54±2.92	73.29±10.20	84.33±3.25	53.30±13.19	73.95±3.72	58.76±8.89	68.49±3.60
MDGPT	59.76±12.44	66.81±6.39	54.19±13.08	55.76±6.58	41.98±12.24	45.59±9.64	44.52±11.39	52.88±6.72
+GFMate	75.89±9.60	82.15±3.61	75.47±10.06	85.24±4.19	54.70±10.62	72.56±3.41	57.20±7.95	69.04±3.24
SAMGPT	66.79±10.77	70.44±7.61	59.34±9.82	72.25±7.67	47.76±13.06	49.58±8.71	52.83±12.04	63.39±7.71
+GFMate	72.54±8.89	80.60±2.03	76.34±9.17	85.72±3.21	55.73±12.25	73.30±4.52	58.49±7.93	70.10±4.63

like LP, DGI and GCL, GFMate can increase the performance over the common fine-tuning (FT). The GFMs from MDGPT and SAMGPT are pre-trained using specifically designed link prediction and GCL objectives, with prompts injected into the pre-training process. It is clear that applying GFMate to these various pre-trained GFMs leads to significant performance improvements, regardless of their pre-training strategies and backbone model, demonstrating the strong plug-in ability and generalisability of the proposed pre-training-agnostic prompts.

4.5 Ablation Study

The ablation study verifies the effectiveness of each key component in GFMate. In Figure 6, the three key modules, centroid prompt tuning (Centroid), layer prompt tuning (Layer), and test-time graph complementary learning (TGCL) are evaluated. "w/o Layer" denotes mean layer performance, and "w/o TGCL" denotes few-shot prompt tuning. The results verify that all three components substantially contribute to GFMate's overall performance.

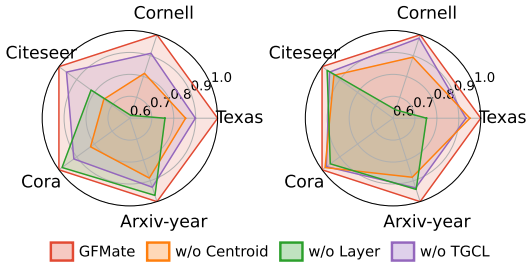


Figure 6: Ablation studies. One-shot (left) and three-shot (right) node classification results.

4.6 Different Few-shot Scenarios

With different numbers of labelled data (shots) in few-shot tuning, GFMate consistently outperforms other GFM methods. As in Figure 7, GFMate is evaluated using 1, 3, 5, 10 and all labelled data for prompt tuning. GFMate, represented by the red line, consistently outperforms existing GFM methods with relatively small standard deviations, particularly on graph classification datasets. Even in the full-shot setting, where all labelled data are available for model tuning, GFMate remains consistently more effective than other GFM baselines. More results are provided in the Appendix F.

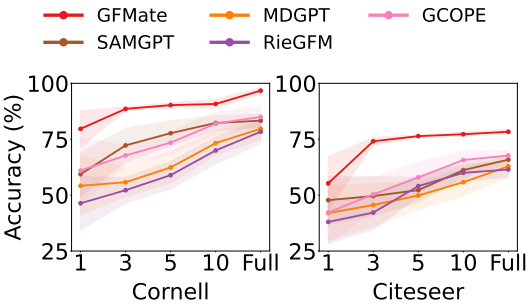


Figure 7: GFMate in different few-shot settings. GFMate consistently outperforms existing GFMs. RieGFM is RiemannGFM. The shaded regions indicate standard deviation.

5 Related Work

Our research domain intersects with three major areas: (i) **Graph Foundation Models (GFMs)**, which pre-train models on large-scale source domain graphs to capture transferable representations, including LLM-based GFMs for text-attributed graphs (Liu et al., 2023a; Li et al., 2024; Chen et al., 2024a; Xia et al., 2024; Chen et al., 2024b; Kong et al., 2025) and GNN-based GFMs for general types of graphs without text attributes (Zhao et al., 2024a; Yu et al., 2024c; 2025; Sun et al., 2025; Wang et al., 2025); (ii) **Graph Prompt Tuning**, consists of single-domain graph prompts, which train the prompts within a single domain, and cross-domain GFM prompts (Yu et al., 2024c; 2025; Wang et al., 2025), which learns prompts during pre-training on source domains and fine-tune them on unseen target domains; (iii) **Test-time Methods for Graphs**, consist of test-time training (Sun et al., 2020; Wang et al., 2021; Liu et al., 2021; Zhang et al., 2022a;b; Chen et al., 2022; Wang et al., 2022; Zhang et al., 2024c;b; Zheng et al., 2024; Bao et al., 2024) which fine-tune the model, and test-time graph transformation (Jin et al., 2023; Ju et al., 2023) which modify the test graphs; More detailed related work and comparison with GFMate are provided in Appendix C due to page limit.

6 Conclusion

This paper presents GFMate, a novel pre-training-agnostic test-time prompt tuning framework designed for GFMs. GFMate proposes pre-training-agnostic centroid and layer prompts, enhancing generalisability across different domains and pre-trained GFMs. Moreover, a novel test-time complementary learning objective is introduced to exploit both labelled and unlabelled target domain data, mitigating the target domain distribution shifts. Experiments on 12 benchmark datasets from diverse domains demonstrate superior performance with notable efficiency gains.

REFERENCES

Wenxuan Bao, Zhichen Zeng, Zhining Liu, Hanghang Tong, and Jingrui He. Adarc: Mitigating graph structure shifts during test-time. *CoRR*, abs/2410.06976, 2024.

Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21, 2005.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020.

Guanzi Chen, Jiying Zhang, Xi Xiao, and Yang Li. Graphtta: Test time adaptation on graph neural networks. *CoRR*, abs/2208.09126, 2022.

Nuo Chen, Yuhan Li, Jianheng Tang, and Jia Li. Graphwiz: An instruction-following language model for graph problems. *arXiv preprint arXiv:2402.16029*, 2024a.

Qin Chen, Liang Wang, Bo Zheng, and Guojie Song. Dagprompt: Pushing the limits of graph prompting with a distribution-aware graph prompt tuning approach. *arXiv preprint arXiv:2501.15142*, 2025.

Runjin Chen, Tong Zhao, Ajay Jaiswal, Neil Shah, and Zhangyang Wang. Llaga: Large language and graph assistant. *arXiv preprint arXiv:2402.08170*, 2024b.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *ICLR*, 2021.

Taoran Fang, Yunchao Zhang, Yang Yang, Chunping Wang, and Lei Chen. Universal prompt tuning for graph neural networks. In *NeurIPS*, 2024.

Chun-Mei Feng, Kai Yu, Yong Liu, Salman Khan, and Wangmeng Zuo. Diverse data augmentation with diffusions for effective test-time prompt tuning. In *ICCV*, 2023.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In *SIGKDD*, 2022.

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

Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. Gpt-gnn: Generative pre-training of graph neural networks. In *SIGKDD*, 2020b.

Takashi Ishida, Gang Niu, Weihua Hu, and Masashi Sugiyama. Learning from complementary labels. In *NeurIPS*, 2017.

Takashi Ishida, Gang Niu, Aditya Krishna Menon, and Masashi Sugiyama. Complementary-label learning for arbitrary losses and models. In *ICML*, 2019.

Wei Jin, Tong Zhao, Jiayuan Ding, Yozen Liu, Jiliang Tang, and Neil Shah. Empowering graph representation learning with test-time graph transformation. In *ICLR*, 2023.

Mingxuan Ju, Tong Zhao, Wenhao Yu, Neil Shah, and Yanfang Ye. Graphpatcher: Mitigating degree bias for graph neural networks via test-time augmentation. In *NeurIPS*, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Lecheng Kong, Jiarui Feng, Hao Liu, Chengsong Huang, Jiaxin Huang, Yixin Chen, and Muhan Zhang. GOFA: A generative one-for-all model for joint graph language modeling. In *ICLR*, 2025.

Yuhan Li, Peisong Wang, Zhixun Li, Jeffrey Xu Yu, and Jia Li. Zerog: Investigating cross-dataset zero-shot transferability in graphs. *arXiv preprint arXiv:2402.11235*, 2024.

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*, 2023a.

Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S. Yu, and Chuan Shi. Graph foundation models: Concepts, opportunities and challenges. *TPAMI*, 2025.

Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. TTT++: when does self-supervised test-time training fail or thrive? In *NeurIPS*, 2021.

Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. GraphPrompt: Unifying pre-training and downstream tasks for graph neural networks. In *WWW*, 2023b.

Yuanfu Lu, Xunqiang Jiang, Yuan Fang, and Chuan Shi. Learning to pre-train graph neural networks. In *AAAI*, 2021.

Xiaosong Ma, Jie Zhang, Song Guo, and Wenchao Xu. Swapprompt: Test-time prompt adaptation for vision-language models. In *NeurIPS*, 2023.

Nagarajan Natarajan, Inderjit S. Dhillon, Pradeep Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *NeurIPS*, 2013.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *ICLR*, 2020.

Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *SIGKDD*, 2020.

Ryan Rossi and Nesreen Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *J. Complex Networks*, 9, 2021.

Jameel Abdul Samadh, Hanan Gani, Noor Hussein, Muhammad Uzair Khattak, Muzammal Naseer, Fahad Shahbaz Khan, and Salman H. Khan. Align your prompts: Test-time prompting with distribution alignment for zero-shot generalization. In *NeurIPS*, 2023.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014.

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

Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. In *NeurIPS 2022*, 2022.

Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000*, 2019.

Li Sun, Zhenhao Huang, Suyang Zhou, Qiqi Wan, Hao Peng, and Philip S. Yu. Riemanngfm: Learning a graph foundation model from riemannian geometry. In *WWW*, 2025.

Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. Gppt: Graph pre-training and prompt tuning to generalize graph neural networks. In *SIGKDD*, 2022.

Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. All in one: Multi-task prompting for graph neural networks. In *SIGKDD*, 2023.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *ICLR*, 2019.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A. Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*, 2021.

Shuo Wang, Bokui Wang, Zhixiang Shen, Boyan Deng, and Zhao Kang. Multi-domain graph foundation models: Robust knowledge transfer via topology alignment. *CoRR*, abs/2502.02017, 2025.

Yiqi Wang, Chaozhuo Li, Wei Jin, Rui Li, Jianan Zhao, Jiliang Tang, and Xing Xie. Test-time training for graph neural networks. *CoRR*, abs/2210.08813, 2022.

Lianghao Xia, Ben Kao, and Chao Huang. Opengraph: Towards open graph foundation models. *arXiv preprint arXiv:2403.01121*, 2024.

Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016.

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

Xingtong Yu, Zhenghao Liu, Yuan Fang, Zemin Liu, Sihong Chen, and Xinming Zhang. Generalized graph prompt: Toward a unification of pre-training and downstream tasks on graphs. *TKDE*, 2024a.

Xingtong Yu, Jie Zhang, Yuan Fang, and Renhe Jiang. Non-homophilic graph pre-training and prompt learning. *arXiv preprint arXiv:2408.12594*, 2024b.

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

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 *WWW*, 2025.

Haonan Yuan, Qingyun Sun, Junhua Shi, Xingcheng Fu, Bryan Hooi, Jianxin Li, and Philip S. Yu. How much can transfer? bridge: Bounded multi-domain graph foundation model with generalization guarantees. In *ICML*, 2025.

Jiaqi Zeng and Pengtao Xie. Contrastive self-supervised learning for graph classification. In *AAAI*, 2021.

Dingchu Zhang, Zhi Zhou, and Yufeng Li. Robust test-time adaptation for zero-shot prompt tuning. In *AAAI*, 2024a.

Jiaxin Zhang, Yiqi Wang, Xihong Yang, Siwei Wang, Yu Feng, Yu Shi, Ruicaho Ren, En Zhu, and Xinwang Liu. Test-time training on graphs with large language models (llms). *CoRR*, abs/2404.13571, 2024b.

Jiaxin Zhang, Yiqi Wang, Xihong Yang, and En Zhu. A fully test-time training framework for semi-supervised node classification on out-of-distribution graphs. *TKDD*, 18, 2024c.

Marvin Zhang, Sergey Levine, and Chelsea Finn. MEMO: test time robustness via adaptation and augmentation. In *NeurIPS*, 2022a.

Yifan Zhang, Bryan Hooi, Lanqing Hong, and Jiashi Feng. Self-supervised aggregation of diverse experts for test-agnostic long-tailed recognition. In *NeurIPS*, 2022b.

Haihong Zhao, Aochuan Chen, Xiangguo Sun, Hong Cheng, and Jia Li. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. In *SIGKDD*, 2024a.

Jianan Zhao, Zhaocheng Zhu, Mikhail Galkin, Hesham Mostafa, Michael Bronstein, and Jian Tang. Fully-inductive node classification on arbitrary graphs. *arXiv preprint arXiv:2405.20445*, 2025.

Shuai Zhao, Xiaohan Wang, Linchao Zhu, and Yi Yang. Test-time adaptation with CLIP reward for zero-shot generalization in vision-language models. In *ICLR*, 2024b.

Xin Zheng, Dongjin Song, Qingsong Wen, Bo Du, and Shirui Pan. Online GNN evaluation under test-time graph distribution shifts. In *ICLR*, 2024.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *NeurIPS*, 2020a.

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

Chenyi Zi, Haihong Zhao, Xiangguo Sun, Yiqing Lin, Hong Cheng, and Jia Li. Prog: A graph prompt learning benchmark. In *NeurIPS*, 2024.

A SUPPLEMENTARY MATERIAL OVERVIEW

In the Appendix, additional supplementary material to the main paper is provided. The structure is:

- The reproducibility statement is provided in Appendix B, including:
  - More detailed baseline method descriptions and categories in Table 4.
  - More detailed dataset descriptions and statistics in Table 5 and 6.
  - A Pseudo-code is provided in Algorithm 1.
  - More details on the implementation are included in the reproducibility statement.

- More detailed related work is provided in Appendix C, the structure is:
  - Appendix C.1 provides more detailed related work for pre-training methods on graphs.
  - Appendix C.2 provides more detailed related work for graph foundation models.
  - Appendix C.3 provides more detailed related work for prompting methods on graphs.
  - Appendix C.4 provides more detailed related work for test-time methods on graphs.
  - Appendix C.5 provides more detailed related work for test-time prompt tuning in other domains.

- Appendix D provides the theoretical analysis of the proposed test-time complementary learning.

- More in-depth analysis of GFMate is provided in Appendix E, the structure is:
  - Appendix E.1 provides the efficiency analysis of GFMate in terms of adaptation time, memory consumption, and parameter count.
  - Appendix E.2 provides the effectiveness analysis of GFMate on different backbone models.
  - Appendix E.3 provides the effectiveness analysis of GFMate in a binary classification setting.
  - Appendix E.4 evaluates the effectiveness of GFMate when less testing data are accessible.
  - Appendix E.5 provides the robustness analysis of GFMate under pre-training domain shift.
  - Appendix E.6 provides a t-SNE visualisation for the centroid with and without GFMate.
  - Appendix E.7 provides the sensitivity analysis of GFMate to hyperparameters.
  - Appendix E.8 provides the comparison between GFMate and few-shot prompt tuning and pseudo label-based prompt tuning.
  - Appendix E.9 evaluates the accuracy of the test-time complementary labels by the layer-wise entropy-based strategies for the testing nodes.

- More detailed experimental results are provided in Appendix F, the structure is:
  - Appendix F.1 provides results for more baseline methods and results in 1, 3, 5, 10, and full-shot node classification scenarios.
  - Appendix F.2 provides results for more baseline methods on the graph classification task.

- A discussion of comparison between pre-training-entangled prompt and pre-training-agnostic prompt is provided in Appendix G.
- A discussion of comparison between existing few-shot prompt tuning and our test-time prompt tuning for GFMs is provided in Appendix H.

B REPRODUCIBILITY STATEMENT

To promote reproducible research, we summarise our efforts as follows:

- **Baselines.** We adopt baseline methods from existing GFM and prompt-based methods, including SAMGPT (Yu et al., 2025), DAGPrompt (Chen et al., 2025), and ProG benchmark (Zi et al., 2024), and carefully tune their hyperparameters via random search to optimise for a fair comparison. All the existing methods are categorised and summarised in Table 4.
  - For GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), SAGE (Hamilton et al., 2017), H2GCN (Zhu et al., 2020a) and GPR (Chien et al., 2021), the hidden dimension is selected from $\{32, 64, 128, 256\}$, with the number of layers searched in the range of 2 to 4. The dropout rate is tuned within the range of 0 to 1.
  - For Link Prediction (Lu et al., 2021), DGI (Velickovic et al., 2019) and GCL (You et al., 2020), GCN with a hidden dimension of 256 is used as the default backbone. PReLU is employed, following prior work (Yu et al., 2024b; 2025). Edge dropping with a drop rate of 0.2 is employed as the default augmentation, following SAMGPT (Yu et al., 2025).
  - For GPPT (Sun et al., 2022), a 2-layer SAGE with hidden dimension 256 and mean aggregator is adopted as the backbone.
  - For GPF (Fang et al., 2024), link prediction(Lu et al., 2021) is used as the default pre-training task with a 3-layer GCN as the default backbone, following DAGPrompt (Chen et al., 2025).
  - For GraphPrompt (Liu et al., 2023b), a 3-layer GCN is used for Wisconsin, Squirrel, Chameleon, and Cornell, while a 2-layer architecture is adopted for Cora, Citeseer, ENZYMES, PROTEINS, COX2, and BZR, with hidden dimensions set to 256, following SAMGPT (Yu et al., 2025).
  - For ProNoG (Yu et al., 2024b), a 2-layer GCN is used as the backbone for graph contrastive learning pre-training on Wisconsin, Squirrel, Chameleon, and Cornell (hidden dimension 256), while a 1-layer GCN with the same hidden dimension is used for Cora, Citeseer, BZR, and COX2, and a 1-layer GCN with hidden dimension 64 is used for PROTEINS on the link prediction task, following the recommendation in their paper.
  - For DAGPrompt (Chen et al., 2025), a 256-dimensional GCN is used as the default backbone, and the hyperparameter r for GLoRA is randomly searched in the range $\{8, 16, 32\}$ as recommended in their paper.
  - For All-In-One (Sun et al., 2023), a GCN with 2 layers and a hidden dimension of 100 is adopted, with a token number set to 10 for the prompt graphs as recommended in their paper.
  - For GCOPE (Zhao et al., 2024a), a 2-layer GCN with 100 hidden dimensions is used as the backbone. Downstream adaptation is conducted via fine-tuning, following their original implementation. As recommended by GCOPE, the feature dimension of all source datasets is aligned to 100 via SVD for all GFM methods to make the setting consistent.
  - For RiemannGFM (Sun et al., 2025), the default number of Riemannian layers is 2. The parameterised scalar map in cross-geometry attention is a multi-layer perception with one hidden layer, whose dimension is set to 256, as described in their paper.
  - For MDGPT (Yu et al., 2024c) and SAMGPT (Yu et al., 2025), a 3-layer GCN is used as the base model for all datasets, with the hidden dimensions set to 256, as recommended in their paper.
  - For BRIDGE (Yuan et al., 2025), since their original implementation assumes a setting where the source domain contains only two datasets, we adapt their code to include more source domains so that it is consistent with our other GFM methods. A single-layer MLP is implemented as the learnable routing network and a two-layer GNN is used as the default pre-trained model, as recommended in their paper.

Table 4: Summary of existing methods for learning on graphs.

Generalisability	Categories	Representative Methods	Graphs
Non-GFMs	**Supervised GNNs**	GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), SAGE (Hamilton et al., 2017) H2GCN (Zhu et al., 2020a), GPR (Chien et al., 2021)	Text-free
	Supervised + Test-time	GTrans (Jin et al., 2023), GraphPatcher (Ju et al., 2023)	
	Linear GNNs	GraphAny (Zhao et al., 2025)	
	Pre-training + Fine-tuning	Link Prediction (Lu et al., 2021), DGI (Velickovic et al., 2019), GCL (You et al., 2020)	
	Pre-training + Prompt	GPPT (Sun et al., 2022), All-In-One (Sun et al., 2023), ProNoG (Yu et al., 2024b) DAGPrompt (Chen et al., 2025), GraphPrompt (Liu et al., 2023b), GPF (Fang et al., 2024)	
GFMs		GCOPE (Zhao et al., 2024a), MDGFM (Wang et al., 2025), MDGPT (Yu et al., 2024c) SAMGPT (Yu et al., 2025), BRIDGE (Yuan et al., 2025)	
	Other GFMs	RiemannGFM (Sun et al., 2025)	
	GFMs via LLMs	OFA (Liu et al., 2023a), ZeroG (Li et al., 2024), GraphWiz (Chen et al., 2024a)	TAGs

- **Datasets.** We utilise 12 publicly available graph benchmark datasets from different domains, as in Table 5 and 6, including:
  - Cora and Citeseer (Yang et al., 2016) are citation networks in which nodes represent academic publications, and edges denote citation relationships.
  - Amazon-Photo (Shchur et al., 2018) (Photo) is a co-purchasing network where nodes correspond to products, and edges indicate frequent co-purchases.
  - Arxiv-year (Hu et al., 2020a) contains 169,343 computer science papers, forming a large-scale citation network. Each paper is represented by a 128-dimensional feature vector, obtained by averaging the embeddings of words in its title and abstract.
  - Chameleon and Squirrel (Rozemberczki et al., 2021) are two page-page networks derived from Wikipedia. In these datasets, nodes represent web pages, edges signify mutual hyperlinks, and node features encode informative nouns extracted from Wikipedia pages.
  - Cornell (Pei et al., 2020) is a webpage network. It comprises 183 nodes, each symbolising a webpage, and 295 edges, which represent the hyperlinks between these pages. The node features are obtained from a bag-of-words representation of the webpages.
  - Wisconsin (Pei et al., 2020) is also a webpage network consisting of nodes representing individual webpages and edges indicating hyperlinks between them. Node features are derived from a bag-of-words representation, and the webpages are classified into categories such as student, project, course, staff, and faculty.
  - Texas (Pei et al., 2020) is a webpage network containing nodes representing university webpages and edges denoting hyperlinks. Features are derived from a bag-of-words representation, and nodes are classified into categories such as student, project, course, staff, and faculty.
  - PROTEINS (Borgwardt et al., 2005) is a dataset of protein graphs where nodes represent secondary structures and edges indicate spatial or sequential proximity. Node features capture structural and biochemical properties. Nodes are classified into three types, and each graph belongs to one of two classes.
  - BZR (Rossi & Ahmed, 2015) is a dataset of ligand graphs associated with the benzodiazepine receptor, where each graph represents a molecule. These molecules are divided into two categories.
  - COX2 (Rossi & Ahmed, 2015) is a dataset of molecular graphs representing cyclooxygenase-2 inhibitors. In these graphs, nodes correspond to atoms and edges denote different types of chemical bonds. The molecules are classified into two categories.

Table 5: Datasets statistics for node classification.

Dataset	#Nodes	#Edges	#Feature	#Classes	#Full-shot	Domain	Source
Cornell	183	554	1703	5	87	Social Network	(Pei et al., 2020)
Texas	183	558	1703	5	87	Social Network	(Pei et al., 2020)
Wisconsin	251	900	1703	5	120	Social Network	(Pei et al., 2020)
Chameleon	2277	36101	2325	5	1092	Social Network	(Rozemberczki et al., 2021)
Squirrel	5201	217073	2089	5	2496	Social Network	(Rozemberczki et al., 2021)
Cora	2708	10556	1433	7	140	Citation Network	(Yang et al., 2016)
Citeseer	3327	9104	3703	6	120	Citation Network	(Yang et al., 2016)
Arxiv-year	169343	1166243	128	5	100	Citation Network	(Hu et al., 2020a)
Amazon-photo	7650	238162	745	8	160	Commercial System	(Shchur et al., 2018)

Table 6: Datasets statistics for graph classification.

Dataset	#Graphs	Avg.#Nodes	Avg.#Edges	#Feature	#Classes	Domain	Source
BZR	405	35.8	38.4	3	2	Biochemical Molecules	(Rossi & Ahmed, 2015)
COX2	467	41.2	43.5	3	2	Biochemical Molecules	(Rossi & Ahmed, 2015)
PROTEINS	1113	39.1	72.8	3	2	Protein–Protein Interaction	(Borgwardt et al., 2005)

- **Implementation Details.** We utilise publicly available benchmark datasets, as shown in Table 5 and 6, fully adhering to their CC-BY 4.0 license. The datasets are sourced from PyTorch Geometric 2.6.1 and OGB 1.3.6, both under the MIT license. The experiments are conducted using Python 3.8.2 and PyTorch 2.4.1 with CUDA 12.2, on a single NVIDIA A6000 GPU with 48GB of memory. For all the baseline methods, the code is obtained from their official GitHub repositories, and the hyperparameters for each method are rigorously tuned to ensure a fair comparison with our proposed method. The hidden dimension is selected from $\{32, 64, 128, 256\}$, and the number of layers is searched within the range of 2 to 5. The Adam optimiser (Kingma & Ba, 2015) is adopted, with learning rates searched over $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$ for all loss terms. The

hyperparameter γ is randomly searched in the range of 0 to 1 with an interval of 0.01 based on performance on the validation graphs. The test data with the top-confidence predictions will be used with pseudo-labels. All experiments are repeated five times with five random data splits, and the mean accuracy with standard deviation is reported.

- **Algorithm.** Our GFMate framework is fully documented in the method section. In addition, we provide a detailed pseudo-code in Algorithm 1.

Algorithm 1: Pseudo-code of GFMate

Input: Node features $\boldsymbol{X}$, adjacency $\boldsymbol{A}$, node set from target domain graph $\mathcal{V}_{\text{Tar}}$, pre-trained GFM with fixed $\boldsymbol{\theta}^*$, loss weight γ
Output: Final predictions $\hat{\boldsymbol{y}}$
% Encoding Process
Encode few-shot and testing nodes by the fixed GFM.
% Pre-training-agnostic Prompt Design
Initialise centroid prompt $\boldsymbol{\beta}$.
for $l = 0$ **to** L **do**
  for $c = 1$ **to** C **do**
    $\widetilde{\boldsymbol{e}}_c^{(l)} \leftarrow \boldsymbol{e}_c^{(l)} + \boldsymbol{\beta}_c^{(l)}$;
Initialise layer prompt $\boldsymbol{\eta}$.
% Test-time Graph Complementary Learning
Compute layer-wise entropy to select the pivot layer.
Construct test-time complementary labels.
while *not converged* **do**
  Compute test-time learning loss on complementary-labelled data $\mathcal{D}_{\text{Te}}$.
  Compute few-shot loss on $\mathcal{D}_{\text{Fs}}$.
  Convex loss combination.
  Update centroid prompts & layer prompts.
% Prediction with Multi-layer Ensemble
Predict with the multi-layer ensemble.

- **Open Source.** The full code will be released upon acceptance for reproducibility.

C Detailed Related Work

C.1 Graph Pre-training Methods

Graph pre-training methods seek to capture the intrinsic characteristics of graphs, predominantly leveraging self-supervised learning techniques. Such approaches can be categorised into two main types: (i) Graph Reconstruction Based Methods, which aim to recover specific graph attributes (Hu et al., 2020b; Hou et al., 2022; Kipf & Welling, 2016; Lu et al., 2021); (ii) Graph Contrastive Learning Based Methods, which improve representation learning by contrastive learning on different views of representations (Zhu et al., 2020b; Zeng & Xie, 2021; You et al., 2020; Velickovic et al., 2019; Sun et al., 2019). Despite their effectiveness, these approaches struggle to mitigate the task objective discrepancy between pre-training and fine-tuning, thus constraining their generalisation capacity on different downstream tasks (Sun et al., 2023; Zhao et al., 2024a).

C.2 Graph Foundation Models

GFMs aim to learn generalisable graph representations by leveraging label-free pre-training on large-scale source graphs to adapt to a wide range of downstream tasks and target graphs in different domains (Liu et al., 2025). Current GFMs can be broadly categorised based on backbone architectures into (i) **LLM-based GFMs** (Liu et al., 2023a; Li et al., 2024; Chen et al., 2024a; Xia et al., 2024; Chen et al., 2024b; Kong et al., 2025) and (ii) **GNN-based GFMs** (Zhao et al., 2024a; Yu et al., 2024c; 2025; Sun et al., 2025; Wang et al., 2025; Yuan et al., 2025). LLM-based GFMs utilise textual information in graphs by harnessing the language modelling capabilities of large language models (LLMs) (Brown et al., 2020), but are inherently limited to text-attributed graphs (TAGs). In contrast, GNN-based GFMs operate in continuous feature spaces and are applicable to general text-free graphs (Zhao et al., 2024a; Sun et al., 2025; Wang et al., 2025; Yu et al., 2024c; 2025; Yuan et al., 2025). Therefore, GFMate focuses on enhancing the GNN-based GFMs due to their broader applicability.

C.3 Graph Prompt Tuning Methods

Conventional Single-domain Graph Prompt Tuning. Graph prompting seeks to reformulate diverse downstream tasks to align with the pre-training objective by introducing trainable prompts, without requiring fine-tuning of the model (Sun et al., 2023). Existing graph prompt tuning approaches primarily target different types of graphs, such as homophilic graphs (Sun et al., 2022), heterophilic graphs (Yu et al., 2024b; Chen et al., 2025) and (ii) other types, including graphs from biological domains (Liu et al., 2023b; Yu et al., 2024a; Fang et al., 2024). Such prompting methods typically assume that source and target graphs originate from the same domain (Hu et al., 2020b; Qiu et al., 2020), thereby overlooking scenarios in which pre-training and downstream tasks involve datasets drawn from distinct domains.

Cross-domain Graph Prompt Tuning for GFMs. Recent advancements in graph prompt tuning have extended to the domain of GFMs, aiming to enhance their cross-domain generalisation by designing diverse prompting strategies (Wang et al., 2025; Yu et al., 2024c; 2025). MDGPT (Yu et al., 2024c) develops a unifying prompt and a mixing prompt to align the target domains with source domains. SAMGPT (Yu et al., 2025) further develops structural tokens with dual prompts during both pre-training and downstream stages to enhance the structural alignment between source and target domains. BRIDGE (Yuan et al., 2025) further proposes a domain invariant aligner during pre-training and employs a mixture of expert networks for cross-domain downstream prompting.

C.4 Test-time Methods in Graph Domain

Test-time methods for graphs aim to enhance the pre-trained model's performance on test graphs. They can be broadly categorised into two types: (i) test-time training (Sun et al., 2020; Wang et al., 2021; Liu et al., 2021; Zhang et al., 2022a;b; Chen et al., 2022; Wang et al., 2022; Zhang et al., 2024c;b; Zheng et al., 2024; Bao et al., 2024), which adapt the model during inference without altering the input graph; and (ii) test-time graph transformation (Jin et al., 2023; Ju et al., 2023), which improve the test data by modifying the graph structure or node features without retraining the model. In contrast, GFMate diverges from both categories by performing test-time tuning of

the additional prompts, without modifying either the model or the test graph, thereby introducing a novel and unexplored direction.

C.5 Test-time Prompt Tuning in Other Domains

Test-time prompt tuning is a promising research direction to address distribution shifts during downstream tasks in domains such as computer vision (CV) (Shu et al., 2022; Ma et al., 2023; Samadh et al., 2023; Feng et al., 2023; Zhao et al., 2024b; Zhang et al., 2024a). These methods leverage test data to optimise prompts, which are concatenated with input data and passed to the fixed model to enhance test-time performance. In comparison, GFMate optimises the graph prompts to enhance downstream task performance without altering the input data, which significantly differs from test-time prompt tuning methods in the CV domain.

D Detailed Proof for Theoretical Analysis

In this section, we prove the excess risk bound for the hypothesis of the proposed test-time complementary learning. Our test-time graph complementary learning objective falls into the category of training-time noisy label learning (Natarajan et al., 2013), since the ground-truth complementary labels on test data are inaccessible during test-time.

Proposition 1 (Excess Risk Bound for Test-time Learning Loss). *Let $\mathcal{F}$ be the hypothesis space, where each predictor model $f \in \mathcal{F}$ refers to a GFM with prompts, defined as $f = (GFM_{\theta^*}, \mathcal{B})$ with pre-trained model parameters θ^* and learnable GFM prompts $\mathcal{B}$. Let $\overline{\mathcal{L}}$ be the test-time learning loss over all testing samples x and their complementary labels $\overline{y}$. The population risk can be defined as:*

$$R_{\overline{\mathcal{L}}}(f) = \mathbb{E}_{(x,\overline{y})}\left[\overline{\mathcal{L}}(f(x), \overline{y})\right].$$

Define the empirical risk minimiser as $\hat{f} = \arg\min_{f\in\mathcal{F}} \hat{R}_{\overline{\mathcal{L}}}(f)$. Then, with probability at least $1-\delta$, the following generalisation bound on the excess risk of the test-time learning loss holds:

$$R_{\overline{\mathcal{L}}}(\hat{f}) - \min_{f\in\mathcal{F}} R_{\overline{\mathcal{L}}}(f) \leq 4C\ell_\rho \mathfrak{R}(\mathcal{F}) + 2\sqrt{\tfrac{\log(1/\delta)}{2N}}.$$

Proof. To simplify the notation, let $\mathcal{F}$ be the hypothesis space, where each $f \in \mathcal{F}$ denotes a predictor model of the form $f = (\text{GFM}_{\theta^*}, \mathcal{B})$ with fixed pre-trained model parameters θ^* and learnable prompt parameters $\mathcal{B}$. Let $\mathcal{L}(\hat{y}, y)$ be an ℓ-Lipschitz loss function with respect to the predicted output $\hat{y}$ (for every label y). Then, with probability at least $1-\delta$, the generalisation error based on the Rademacher complexity (Shalev-Shwartz & Ben-David, 2014; Natarajan et al., 2013) satisfies:

$$\sup_{f\in\mathcal{F}} \left| R_{\mathcal{L}}(f) - \hat{R}_{\mathcal{L}}(f) \right| \leq 2\ell_\rho \mathfrak{R}(\mathcal{F}) + \sqrt{\frac{\log(1/\delta)}{2N}}, \tag{11}$$

where $\mathfrak{R}(\mathcal{F}) := \mathbb{E}_{X_i,\epsilon_i}\left[\sup_{\text{GFM}_{\theta^*},\mathcal{B}\in\mathcal{F}} \frac{1}{n}\sum \epsilon_i \text{GFM}_{\theta^*}, \mathcal{B}(X_i)\right]$ denotes the Rademacher complexity of $\mathcal{F}$, and $\ell_\rho \leq \frac{2\ell}{1-\rho_{+1}-\rho_{-1}}$ is the Lipschitz constant of the loss $\mathcal{L}$ under label noise. Here, ϵ_i are i.i.d. Rademacher variables, and n is the number of training samples.

Let $\hat{f} = \arg\min_{f\in\mathcal{F}} \hat{R}_{\mathcal{L}}(f)$ be the empirical risk minimiser, and let $f^* = \arg\min_{f\in\mathcal{F}} R_{\mathcal{L}}(f)$ denote the optimal hypothesis with respect to the true risk. Here, $R_{\mathcal{L}}(f)$ and $\hat{R}_{\mathcal{L}}(f)$ denote the true (expected) and empirical risks of a hypothesis f, respectively; $R_{\mathcal{L}}(\hat{f})$ and $\hat{R}_{\mathcal{L}}(\hat{f})$ are the true and empirical risks of the learned model $\hat{f}$; and $R_{\mathcal{L}}(f^*)$ and $\hat{R}_{\mathcal{L}}(f^*)$ denote the true and empirical risks of the optimal hypothesis f^*. Then, the following inequality holds:

$$\hat{R}_{\mathcal{L}}(\hat{f}) \leq \hat{R}_{\mathcal{L}}(f^*).$$

Because $\hat{R}_{\mathcal{L}}(\hat{f})$, defined as the empirical risk of $\hat{f}$, is the global minimiser of the empirical risk, it satisfies $\hat{R}_{\mathcal{L}}(\hat{f}) \leq \hat{R}_{\mathcal{L}}(f)$ for all $f \in \mathcal{F}$, including f^*. Therefore, the excess risk of $\hat{f}$ over f^* can

be bounded as:

$$\begin{aligned} R_{\mathcal{L}}(\hat{f}) - R_{\mathcal{L}}(f^*) &= R_{\mathcal{L}}(\hat{f}) - \hat{R}_{\mathcal{L}}(\hat{f}) + \hat{R}_{\mathcal{L}}(\hat{f}) - R_{\mathcal{L}}(f^*) \\ &\leq R_{\mathcal{L}}(\hat{f}) - \hat{R}_{\mathcal{L}}(\hat{f}) + \hat{R}_{\mathcal{L}}(f^*) - R_{\mathcal{L}}(f^*) \\ &\leq 2 \sup_{f \in \mathcal{F}} \left| R_{\mathcal{L}}(f) - \hat{R}_{\mathcal{L}}(f) \right| \\ &\leq 4\ell_\rho \mathfrak{R}(\mathcal{F}) + 2\sqrt{\frac{\log(1/\delta)}{2N}}. \end{aligned} \tag{12}$$

The above result is for the standard loss $\mathcal{L}$. To extend this to the complementary loss $\overline{\mathcal{L}}$, we note that our test-time loss aligns with the one-vs-all loss (Ishida et al., 2017), where the model is encouraged to distance itself from the complementary class while remaining close to others. Therefore, by Talagrand's contraction lemma (Ishida et al., 2017; 2019), the Rademacher complexity of the one-vs-all complementary loss $\overline{\mathcal{L}}$ is bounded by:

$$\overline{\mathfrak{R}}_n(\mathcal{F}_{\mathrm{OVA}}) \leq C\ell_\rho \mathfrak{R}_n(\mathcal{F}),$$

where C is the number of classes. Hence, with probability at least $1-\delta$, the excess risk for test-time complementary learning is bounded as:

$$R_{\overline{\mathcal{L}}}(\hat{f}) - \min_{f \in \mathcal{F}} R_{\overline{\mathcal{L}}}(f) \leq 4C\ell_\rho \mathfrak{R}(\mathcal{F}) + 2\sqrt{\frac{\log(1/\delta)}{2N}}. \tag{13}$$

In summary, the excess risk of the test-time learning loss is upper bounded by a term that increases with the number of classes and decreases at a rate of $\mathcal{O}(1/\sqrt{N})$, where N denotes the number of testing samples. Intuitively, Proposition 1 suggests that a smaller number of classes or a greater number of complementary-labelled samples leads to a tighter risk upper bound of the GFM prompt method. As GFMate utilises all unlabelled test data for learning, the bound becomes tighter, supporting the theoretical benefit of our test-time learning framework. □

E Further In-depth Analysis

This section presents a more in-depth analysis of GFMate, including the more detailed efficiency analysis in Appendix E.1 and more detailed effectiveness analysis in Appendix E.2.

Table 7: Efficiency Analysis. Downstream adaptation time in seconds for 5 repeat runs, GPU peak memory in MB and total number of tunable parameters. Average accuracy is also provided.

	Cornell				Citeseer				Cora			
	Time (s)	Memory (MB)	#Params	Acc (%)	Time (s)	Memory (MB)	#Params	Acc (%)	Time (s)	Memory (MB)	#Params	Acc (%)
GFMate	19	560	5124	79.67	153	1416	4611	55.27	269	1494	5379	59.68
Fine-tuning	53	646	1081350	23.88	109	994	1212934	33.39	102	936	1147142	29.85
DAGPrompt	140	876	31022	59.11	195	2100	31748	47.24	273	2096	32772	54.88
RiemannGFM	239	4364	109764	46.35	248	4599	123876	37.71	418	5721	146824	39.91
MDGPT	1275	18436	137645	54.19	1579	19108	147659	41.98	1494	20368	143742	44.52
SAMGPT	1076	19876	282360	59.34	1396	24599	283773	47.76	1267	13077	282929	52.83

E.1 Detailed Efficiency Analysis

This section presents a detailed efficiency analysis to evaluate downstream adaptation time, peak GPU memory usage, and the number of tunable parameters, in relation to the average accuracy over five repeated runs on one-shot node classification tasks. To ensure a fair comparison, all time measurements start after the GFM model pre-training is completed, with the model fixed and the target domain graph becoming available. As shown in Table 7, GFMate demonstrates significantly faster downstream adaptation time and reduced GPU memory consumption. The number of tunable parameters in GFMate is notably lower than that in existing GFM methods like SAMGPT, prompting-based methods like DAGPrompt, and full fine-tuning approaches. This highlights the lightweight nature of the pre-training-agnostic prompts in GFMate compared to existing prompt-based methods. Overall, GFMate consistently achieves superior time and space efficiency, while delivering the highest accuracy across all baseline methods.

Table 8: Effectiveness on GNN-based GFMs with different backbones. The default backbone is GCN, following prior work (Yu et al., 2025).

	Texas		Cornell		Citeseer		Cora	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
GFMate	76.63±7.81	83.29±1.52	79.67±8.47	88.57±0.87	56.25±13.33	74.08±1.72	59.68±5.37	70.51±2.11
w/ SAGE	78.19±8.82	82.76±1.21	80.05±8.24	88.24±1.26	57.39±11.40	73.63±1.89	60.08±5.37	69.35±2.94
w/ GAT	77.14±7.42	83.40±1.55	79.58±8.41	88.97±1.03	54.19±13.50	73.68±2.14	59.30±6.11	70.97±2.68
w/ H2GCN	82.31±9.92	85.56±2.29	81.89±9.43	90.07±1.58	52.38±15.52	70.74±3.25	58.75±5.64	65.82±3.47

E.2 GFMate with Different Backbone Models

In this section, GFMate is plugged into GFMs implemented with different GNN backbone models, including SAGE, GAT, and H2GCN. The default pre-training strategy is link prediction. As shown in Table 8, GFMate effectively improves GFMs across different backbones, with GFMate applied to the GCN backbone consistently achieving strong performance across all datasets, demonstrating its effectiveness on diverse GFM architectures.

Table 9: Comparison of performance gain of GFMate with SAMGPT on binary classification.

Methods	**Chameleon (5)**	**Chameleon (2)**	**Cora (7)**	**Cora (2)**
SAMGPT	38.12±8.90	60.61±10.36	52.83±12.04	71.19±7.80
GFMate	**47.25±6.11**	**78.88±7.56**	**59.68±5.37**	**81.77±3.51**
Δ	9.13	18.27	6.85	10.58
Methods	**Citeseer (6)**	**Citeseer (2)**	**Photo (8)**	**Photo (2)**
SAMGPT	47.76±10.55	55.57±12.04	56.33±9.04	73.36±4.52
GFMate	**56.25±13.33**	**69.05±8.40**	**58.85±2.17**	**84.37±1.79**
Δ	8.49	13.48	2.52	10.01

E.3 GFMate in Binary Classification Cases

The effectiveness of GFMate is further evaluated on binary classification cases. To establish a balanced classification setting, the datasets were modified as follows:

- For the **Cora** and **Amazon-Photo** datasets, four randomly selected classes were merged into a single group, with the remaining three classes treated as the second group.
- For the **Citeseer** and **Chameleon** datasets, three randomly selected classes were merged into one group, and the remaining classes constituted the second group.

This process successfully established a two-class, or binary, setting for subsequent evaluation. The performance comparison against the current state-of-the-art baseline, SAMGPT, is presented in Table 9. The results clearly demonstrate that GFMate achieves a significant performance advantage over the SOTA baseline. Crucially, the performance improvement achieved by GFMate over the baseline in the binary classification setting is consistently larger than the improvement observed in the multi-class classification setting. This enhanced result in the binary case aligns with the theoretical underpinnings of Proposition 1, which suggests that a reduced number of classes leads to improved generalisation capabilities.

E.4 GFMate with less testing data

In this section, the effectiveness of GFMate when fewer unlabelled testing data are available is evaluated. According to Table 10, a larger number of testing nodes leads to better performance for GFMate. This arises from the effective test time complementary learning mechanism in GFMate,

Table 10: GFMate with less testing data consistently demonstrates its effectiveness. The ratio indicates how many randomly selected testing nodes are available. GFMate at zero percent denotes the setting where GFMate uses few shot tuning only.

	Cornell		Citeseer		Cora		Arxiv-year	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
GFMate-0%	72.59±7.68	86.13±2.40	52.92±10.34	71.95±3.66	54.57±4.31	68.79±3.24	28.15±3.50	28.59±3.32
GFMate-20%	73.30±7.47	86.52±3.36	54.40±11.25	72.03±4.32	55.22±4.98	68.92±3.31	28.20±3.65	29.06±3.98
GFMate-50%	77.69±8.61	87.13±2.01	54.98±11.37	72.68±2.76	56.60±4.33	69.01±3.17	29.73±4.07	29.52±3.31
GFMate-80%	78.06±8.08	87.97±1.94	55.79±12.16	73.35±2.07	58.17±5.08	69.79±2.68	30.05±3.96	30.34±3.37
GFMate-Full	79.67±8.47	88.57±0.87	56.25±13.33	74.08±1.72	59.68±5.37	70.51±2.11	30.19±3.65	30.61±2.97

which successfully utilises the unlabelled target domain testing data to adapt the GFM. Moreover, this empirical observation aligns with the theoretical insight provided by Proposition 1, which indicates that a greater amount of testing data yields better generalisability for GFMate.

Table 11: Robustness to GFMs pre-training domain shift. The default GFM backbone is GCN, and the pre-training strategy is link prediction. Each category of domain refers to a set of pre-training graphs, as shown in Table 5, excluding the target domain graphs. Comme and Bioinfn refer to the graph in commercial and bio-information system domains.

	Texas		Cornell		Citeseer		Cora	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
GFMate	76.63±7.81	83.29±1.52	79.67±8.47	88.57±0.87	56.25±13.33	74.08±1.72	59.68±5.37	70.51±2.11
w/ Social	77.42±7.59	83.10±1.58	79.31±8.29	87.95±1.14	52.33±13.18	69.89±2.25	57.64±6.31	68.80±2.55
w/ Citation	75.97±8.24	81.15±2.06	77.49±8.76	85.52±1.35	54.68±12.16	73.79±2.01	59.19±5.87	70.55±2.30
w/ Comme	76.09±7.66	82.16±2.03	74.85±9.12	83.47±2.79	52.94±10.63	70.28±3.39	56.98±7.02	67.19±3.14
w/ Bioinfo	73.22±9.01	79.98±3.27	77.45±9.28	83.59±2.95	48.09±13.32	66.84±4.17	55.10±8.23	65.85±3.98

E.5 Robustness Analysis

In GFMate, the learnable layer-wise prompts capture graph patterns from the target domain and adaptively adjust the contribution of each layer's prediction to the final ensemble output. According to Table 11, GFMate demonstrates effectiveness across GFMs pre-trained on different domain graphs, while models pre-trained on all domains consistently achieve strong performance. These results showcase that the learnable layer prompts can mitigate domain shift by effectively adapting GFMs pre-trained on different domains to the target domain graph.

E.6 Visualisation of Centroids

A t-SNE visualisation is conducted to verify the effectiveness of GFMate in a one-shot node classification task on Cora. The GFM is fixed, and GFMate optimises the centroids via the proposed test-time prompt tuning modules without accessing ground-truth test labels. As shown in Figure 8, after applying GFMate, the centroids align more closely with the true class centres of the testing nodes, contributing to improved classification performance.

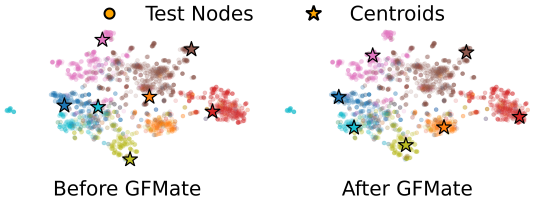


Figure 8: t-SNE visualisation of centroids in GFMate. After GFMate, the centroids become more aligned with the true class centres of the target domain test nodes.

E.7 Hyperparameter Sensitivity Analysis

In this section, the impact of the hyperparameters γ in GFMate is investigated. The parameter γ controls the relative contribution of the test-time learning loss and the few-shot loss. The results of varying γ in the one-shot node classification task are shown in Figure 9. Increasing γ initially improves performance. However, further increases lead to a performance drop on Cora, while other datasets remain stable. This suggests that both the test-time and few-shot learning loss are essential, which aligns

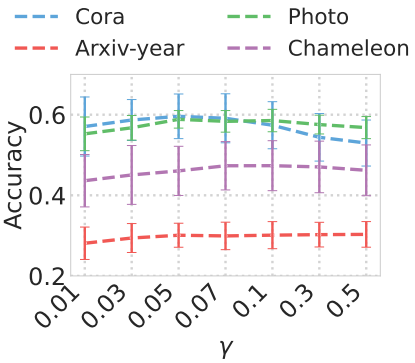


Figure 9: Sensitivity studies of γ.

with the insight that learning from both labelled and unlabelled data is important.

E.8 Comparison with Few-shot (Inductive Cases) and Pseudo Label Training

In this section, experiments are conducted to compare GFMate with conventional few-shot prompt tuning (using only the few-shot set to form a fully inductive setting where testing data is not accessible), as well as with the few-shot set augmented by test data with different ratios of pseudo labels, which are the most similar class from the last layer. According to Table 12, adopting pseudo labels for prompt tuning significantly degrades the performance of few-shot learning, owing to the inaccuracies in the pseudo labels. GFMate with test-time complementary learning substantially outperforms both few-shot prompt tuning and pseudo-label prompt tuning, validating the effectiveness of the proposed test-time graph complementary learning approach.

It should be noted that when the test data are not accessible or in an inductive setting (which is not consistent with the transductive setting of all our baseline prompt methods (Yu et al., 2025; 2024c; Wang et al., 2025)), the test-time prompt tuning setting in GFMate reduces to a conventional few-shot prompt tuning scenario for the proposed pre-training agnostic prompts. Even in this extreme case, the GFMate variant tuned using only the few-shot set still outperforms the baseline method from Table 1, demonstrating that GFMate retains its effectiveness when test data are fully unavailable or in an inductive setting during GFM downstream adaptation.

Table 12: Comparison with few-shot tuning (inductive setting) and pseudo-label tuning. GFMate with test-time complementary learning is more effective than prompt tuning based on few-shot and pseudo labels of testing nodes. Few-shot denotes GFMate with prompt tuning only on the original few-shot set. Pseu denotes GFMate tuned with both few-shot and pseudo-labels of testing nodes from the last layer.

	Cornell		Citeseer		Cora		Arxiv-year	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
GFMate	79.67±8.47	88.57±0.87	56.25±13.33	74.08±1.72	59.68±5.37	70.51±2.11	30.19±3.65	30.61±2.97
Few-shot	72.59±7.68	86.13±2.40	52.92±10.34	71.95±3.66	54.57±4.31	68.79±3.24	28.15±3.50	28.59±3.32
Pseu-All	57.95±9.36	58.05±4.27	13.79±8.91	52.91±6.43	18.29±12.35	35.97±10.64	12.16±8.04	15.61±8.63
Pseu-80%	58.10±9.27	58.39±3.91	22.58±7.72	59.50±4.28	25.53±9.81	39.49±8.28	13.77±8.40	16.48±7.25
Pseu-50%	60.37±8.96	62.33±3.65	35.84±7.65	63.27±5.04	38.39±9.47	44.71±7.66	18.52±6.58	20.29±7.50
Pseu-20%	66.95±8.81	67.72±4.31	45.27±9.89	69.96±4.37	46.61±7.15	59.75±9.54	24.07±5.39	24.32±7.11

E.9 Test-time Complementary labels Analysis

In this section, the test-time complementary labels in GFMate are evaluated. GFMate assigns labels to testing nodes using a layer-wise entropy-based strategy, selecting the least similar class in the pivot layer $\hat{l}$ based on the entropy scores. According to Table 13, the complementary labels in GFMate are more accurate compared to those obtained by simply selecting the least similar class from the last layer, which showcases the effectiveness of the proposed layer-wise entropy-based strategies and contributes to the proposed test-time graph complementary learning.

Table 13: Accuracy of the test-time complementary label. GFMate samples test-time complementary labels using a layer-based entropy strategy, which is more accurate than simply selecting the least similar class as pseudo-label from the final layer.

	Cornell		Citeseer		Cora		Arxiv-year	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
GFMate-comp	99.08±0.27	99.57±0.15	99.25±0.49	99.36±0.31	96.63±0.86	97.20±0.63	88.14±1.29	90.26±1.14
Pseudo-comp	90.31±0.52	93.05±0.64	91.26±0.77	93.98±0.61	92.07±1.26	93.18±1.35	80.29±1.77	83.57±1.68

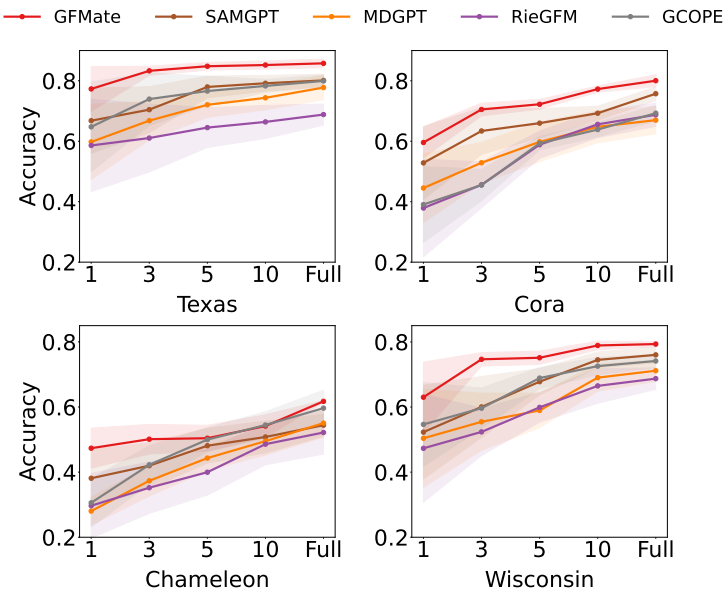


Figure 10: Node classification performance of GFMate under different shot settings for more datasets. RieGFM refers to RiemannGFM. GFMate consistently outperforms existing GFM methods across all scenarios.

F Extended Experimental Results

This section provides additional experimental results, incorporating more baseline methods such as GTrans and GraphAny. To comprehensively evaluate the effectiveness of GFMate, the one-shot classification setting is extended to 3-shot, 5-shot, 10-shot, and full-shot settings. The full-shot setting corresponds to conventional supervised training, where the entire training set is utilised.

F.1 Further Experimental Results on Node Classification

According to the detailed node classification results presented in Table 14, 15, 16, 17 and 18. "SL" refers to supervised learning-based GNN methods, while "TT" denotes test-time adaptation approaches. "Lin' indicates methods implemented using linear GNN architectures. "SSL+FT" represents self-supervised pre-training followed by supervised fine-tuning, whereas "SSL+Prompt" denotes self-supervised pre-training with prompt tuning. "GFMs" refer to methods designed for GFMs with cross-domain generalisability. GFMate demonstrates strong performance and consistently outperforms all baseline methods from 1-shot to full-shot settings, as demonstrated in Figure 10. Notably, when the target domain is the Cora or Amazon-Photo dataset, supervised training methods outperform all GFM and graph prompt tuning approaches, including GFMate. This observation aligns with prior findings that conventional supervised training becomes more effective as more task-specific labelled data are available (Yu et al., 2025). In general, among GFM-based methods where the target domains differ from the source domains, GFMate achieves the best performance across all datasets and settings.

F.2 Further Experimental Results on Graph Classification

In this section, the effectiveness of GFMate in the downstream graph classification task is evaluated. Methods such as GPPT, GTrans and GraphAny, which are not applicable to graph classification, are excluded from the comparison. According to Table 19, GFMate demonstrates the best performance compared to all baseline methods. GFMate also achieves substantially lower standard deviation, particularly on the COX2 and BZR datasets in biological target domains, where existing methods suffer from large standard deviations and consequently unstable performance. This indicates that GFMate's performance in graph classification is more stable than existing methods.

G Discussion between pre-training-entangled and agnostic GFM prompts

The proposed pre-training-agnostic prompt differs from existing pre-training-entangled prompts in the following aspects.

1. Conceptual perspectives

- **Pre-training-entangled methods** in prior work prioritise knowledge specific to certain source domains and pre-train their prompts accordingly, under the assumption that the target domain is closely related to the source domain, as discussed in MDGPT (Yu et al., 2024c).
- **Pre-training-agnostic methods** prioritise information specific to the target domain and employ test-time prompt tuning directly on the target domain, obviating any requirement for prompt pre-training. This approach is particularly appropriate in GFM cross-domain scenarios, where the target domain is unseen during pre-training and may exhibit a distribution distinct from the source domains.

2. Technical perspectives

- Prompts in **pre-training-entangled methods** are optimised jointly with a fixed backbone and a fixed pre-training objective and require retraining whenever a novel training domain is introduced.
- Prompts in **pre-training-agnostic methods** are learned entirely at test time, without any pre-training, and without dependence on a specific backbone architecture or pre-training strategy. This design ensures compatibility with a broad range of backbones and GFMs employing diverse pre-training procedures, as demonstrated in Section 4.4 and Table 3.

3. Advantages and Limitations

- **Pre-training-entangled methods**:
  1. Effective utilisation of source domain information when the target domain is closely related,
  2. Limited generalisability when the target domain differs substantially,
  3. Reduced computational efficiency due to the requirement to retrain prompts for new domains,
  4. Constrained compatibility across different GFM architectures and backbone choices.
- **Pre-training-agnostic methods**:
  1. Enhanced capability to exploit the testing distribution of unseen target domains and improved generalisability across diverse GFM backbones,
  2. Potential computational overhead arising from test-time prompt tuning.

Section 4.2 further evaluates the efficiency of GFMate, demonstrating that it substantially outperforms existing GFMs relying on fine-tuning under a fair comparison at the GFM downstream adaptation stage. The centroid and layer prompt designs eliminate the need to train a separate prompt for each testing sample, in contrast to instance-level prompts in existing GFMs which must be trained individually for each node.

H Discussion between few-shot and test-time prompt tuning in GFM setting

The proposed test-time graph prompt tuning is fundamentally distinct from the existing few-shot prompt tuning task, as discussed in Remarks in the main paper. This distinction primarily arises from how the abundant testing data is utilised.

- **Existing Few-Shot Prompt Tuning for GFMs (e.g., SAMGPT and MDGPT (Yu et al., 2024c; 2025)**
  - Operate in a transductive setting where abundant testing data is accessible. This testing information is utilised **passively** as neighbourhood context when encoding the few labelled nodes.

  - The substantial distribution gap between the limited labelled samples and the abundant unlabelled testing samples remains unaddressed.
- **Proposed Test-Time Prompt Tuning for GFMs**
  - Actively leverages the accessible testing samples, which previous methods overlook. This active utilisation enables the prompts to adapt effectively to the testing data in the unseen target domain, thereby enhancing the generalisation performance of the GFM.
  - Uses target domain data to **directly optimise the prompts during inference**, without reliance on prompt pre-training or model pre-training strategies.

Table 14: Full experimental results for node classification (Part 1). GraphAny and GTrans are evaluated only under the full-shot setting, as they are not designed for GNNs trained in few-shot scenarios. GraphAny (C) and GraphAny (W) denote GraphAny trained on Cora and Wisconsin, respectively. Results are highlighted by **best** and runner-up.

	Dataset	Texas					Cornell				
	Setting	1-shot	3-shot	5-shot	10-shot	full-shot	1-shot	3-shot	5-shot	10-shot	full-shot
SL	GCN	38.82±9.79	42.25±7.03	55.82±3.99	59.96±4.36	64.87±1.47	24.58±10.09	36.71±8.84	58.74±4.95	61.18±4.12	65.89±1.08
	GAT	39.96±9.62	41.19±6.64	55.72±2.68	59.17±3.32	65.41±1.08	25.84±12.26	37.11±10.61	60.88±4.99	65.53±4.10	68.92±5.51
	SAGE	40.38±8.21	44.39±8.11	68.81±3.55	70.25±2.24	81.32±3.59	32.55±13.36	40.81±8.89	70.41±5.33	80.04±3.99	82.83±3.15
	GPR	37.60±9.54	40.36±6.68	74.83±2.84	77.44±3.96	82.24±2.81	30.77±13.42	39.28±9.19	62.25±4.67	79.89±5.32	80.03±2.88
	H2GCN	47.75±12.89	56.69±7.83	70.66±3.59	76.99±4.51	80.65±3.69	35.58±15.02	42.29±9.60	69.90±5.53	80.33±5.57	84.40±4.37
TT	GTrans	—	—	—	—	67.79±4.33	—	—	—	—	65.72±5.84
Lin	GraphAny (C)	—	—	—	—	71.89±1.48	—	—	—	—	64.86±1.91
	GraphAny (W)	—	—	—	—	73.51±1.21	—	—	—	—	66.49±1.48
SSL+FT	LP	36.93±11.90	38.58±8.84	51.45±4.31	55.08±5.33	60.16±3.32	23.88±11.43	35.79±10.33	55.65±4.99	60.22±5.38	62.25±5.10
	DGI	34.46±12.92	40.33±7.81	50.73±4.39	54.46±4.98	59.85±3.34	22.89±11.32	34.48±9.97	52.26±6.60	61.32±5.34	63.35±4.89
	GCL	28.81±15.32	34.40±9.05	45.59±5.11	50.22±5.57	57.73±4.12	20.56±13.36	29.88±9.72	54.56±5.24	58.85±4.66	64.62±3.37
SSL+Prompt	GPPT	44.47±10.88	49.91±6.69	55.29±4.96	60.22±4.49	67.98±3.34	29.74±8.32	35.52±8.93	56.99±5.07	60.81±4.35	65.34±1.78
	GPF	47.34±12.72	55.87±8.62	61.49±4.33	65.02±4.17	68.86±3.58	52.15±14.53	59.07±11.48	69.92±7.74	74.08±7.90	82.25±5.88
	GraphPrompt	53.27±9.95	61.13±4.94	67.57±2.96	70.33±3.40	75.59±2.66	55.13±13.39	65.47±8.62	73.36±6.98	78.81±5.86	81.15±3.77
	ProNoG	55.31±12.92	57.99±8.48	72.24±5.98	77.36±3.77	80.36±4.43	48.49±11.54	69.91±7.62	79.94±6.68	82.25±5.03	85.59±3.31
	DAGPrompt	68.27±10.02	77.34±7.44	80.93±3.88	81.99±1.94	82.29±2.28	59.11±10.04	75.27±9.91	83.76±1.77	85.25±1.95	86.44±4.28
	All-In-One	63.79±15.91	69.44±9.19	74.92±5.79	75.22±2.54	78.11±2.25	57.24±12.70	65.35±6.48	75.50±6.33	80.64±5.44	82.29±3.90
GFMs	GCOPE	64.76±14.84	73.94±8.69	76.61±4.89	78.30±2.16	79.95±1.96	60.98±14.66	67.74±4.42	73.46±4.98	81.99±4.97	84.99±4.56
	RiemannGFM	58.60±15.27	61.06±11.24	64.51±6.58	66.39±5.47	68.82±3.49	46.35±11.92	52.20±5.79	58.93±6.14	69.97±6.08	78.36±5.21
	MDGPT	59.76±12.44	66.81±6.39	72.05±4.18	74.36±3.97	77.75±4.02	54.19±13.08	55.76±6.58	62.37±6.99	73.25±5.28	79.69±4.97
	SAMGPT	66.79±10.77	70.44±7.61	77.96±3.59	79.18±2.59	80.12±2.07	59.34±9.82	72.25±7.67	77.69±5.56	82.27±3.39	83.30±2.91
	GFMate	**76.63**±7.81	**83.29**±1.52	**84.81**±1.27	**85.21**±1.34	**85.77**±1.44	**79.67**±8.47	**88.57**±0.87	**90.25**±1.17	**90.77**±1.98	**96.73**±1.55

Table 15: Full experimental results for node classification (Part 2). GraphAny and GTrans are evaluated only under the full-shot setting, as they are not designed for GNNs trained in few-shot scenarios. GraphAny (C) and GraphAny (W) denote GraphAny trained on Cora and Wisconsin, respectively. Results are highlighted by **best** and runner-up.

	Dataset	Chameleon					Wisconsin				
	Setting	1-shot	3-shot	5-shot	10-shot	full-shot	1-shot	3-shot	5-shot	10-shot	full-shot
SL	GCN	24.30±6.59	27.81±5.64	30.66±3.32	35.58±3.07	42.06±1.16	43.36±13.17	47.28±7.70	53.37±4.33	55.89±3.45	56.90±3.51
	GAT	22.81±7.38	28.14±4.46	31.17±4.09	37.72±3.30	44.34±1.55	45.99±10.86	49.91±6.18	51.49±4.08	52.63±4.43	54.51±3.37
	SAGE	31.29±8.87	38.21±4.37	45.82±2.26	49.89±2.95	51.54±0.55	47.41±11.36	49.24±8.61	58.88±4.05	69.66±3.34	83.14±1.56
	GPR	28.44±6.65	36.77±5.85	43.88±3.74	48.60±4.59	49.96±1.17	49.56±15.58	58.95±6.92	71.42±3.28	73.24±2.58	77.81±3.10
	H2GCN	28.01±9.50	33.75±5.88	44.89±6.01	47.31±3.38	52.01±2.65	45.55±10.61	55.33±6.94	65.16±3.35	73.36±3.88	76.69±4.13
TT	GTrans	—	—	—	—	41.82±2.29	—	—	—	—	59.98±4.96
Lin	GraphAny (C)	—	—	—	—	61.49±1.88	—	—	—	—	61.18±5.08
	GraphAny (W)	—	—	—	—	60.09±0.93	—	—	—	—	71.77±5.98
SSL+FT	LP	25.34±7.19	28.86±6.03	31.17±4.39	35.62±3.33	41.98±1.46	41.37±14.22	45.41±8.80	53.08±4.59	55.92±3.47	57.21±3.79
	DGI	25.78±7.34	29.04±6.11	31.59±4.46	37.02±4.18	42.34±2.80	38.89±15.77	42.31±10.52	51.19±5.57	54.96±3.55	55.73±3.68
	GCL	24.69±8.82	27.03±7.44	29.96±5.65	36.40±4.09	37.12±3.17	35.70±17.73	39.88±9.29	48.86±6.61	52.77±5.10	53.19±4.33
SSL+Prompt	GPPT	29.91±6.48	34.49±3.72	38.04±2.57	42.29±2.55	47.58±2.74	35.16±9.07	45.82±8.89	53.09±4.53	54.88±3.90	55.72±4.07
	GPF	30.95±9.18	36.76±5.33	45.44±2.37	47.32±2.86	49.90±2.34	40.19±14.49	47.22±9.34	58.86±4.41	60.33±3.78	68.84±4.02
	GraphPrompt	33.29±9.19	42.13±6.07	47.81±4.11	51.96±3.09	53.84±2.69	45.03±11.83	52.11±6.24	65.08±3.17	68.80±2.93	72.55±2.45
	ProNoG	31.19±8.09	35.52±6.63	37.34±3.88	38.79±2.21	43.38±2.95	46.29±17.74	55.84±8.81	66.51±4.33	70.22±3.70	75.97±2.26
	DAGPrompt	37.79±6.62	41.26±5.89	49.96±1.98	55.32±2.39	60.79±3.35	50.49±11.59	72.21±3.17	74.08±3.06	76.22±2.34	77.29±1.89
	All-In-One	27.94±6.31	38.03±4.75	46.33±3.05	52.20±2.14	58.85±1.17	55.35±18.36	67.17±4.28	69.22±3.75	70.40±2.29	72.01±1.68
GFMs	GCOPE	30.58±7.44	42.25±5.82	49.98±3.66	54.46±4.09	59.66±5.52	54.66±12.86	59.64±6.20	68.89±3.25	72.56±2.66	74.14±1.99
	RiemannGFM	29.68±9.95	35.21±7.88	39.97±7.04	48.59±6.34	52.17±6.65	47.32±16.58	52.36±7.28	59.90±4.56	66.49±5.32	68.72±3.31
	MDGPT	28.04±4.28	37.34±4.71	44.31±3.82	49.52±4.28	55.06±4.35	50.40±15.07	55.46±8.79	58.95±5.30	69.02±4.18	71.15±3.27
	SAMGPT	38.12±8.90	41.98±7.04	48.11±5.34	50.79±4.88	54.43±3.37	52.29±14.40	60.03±9.36	67.78±4.03	74.48±2.40	76.02±2.45
	GFMate	**47.25**±6.11	**50.12**±4.55	**50.42**±3.96	**54.15**±3.49	**61.75**±0.24	**63.01**±10.78	**74.66**±2.07	**75.13**±2.03	**78.91**±1.32	**79.34**±0.77

Table 16: Full experimental results for node classification (Part 3). GraphAny and GTrans are evaluated only under the full-shot setting, as they are not designed for GNNs trained in few-shot scenarios. GraphAny (C) and GraphAny (W) denote GraphAny trained on Cora and Wisconsin, respectively. Results are highlighted by **best** and runner-up.

	Dataset	Squirrel					Amazon-photo				
	Setting	1-shot	3-shot	5-shot	10-shot	full-shot	1-shot	3-shot	5-shot	10-shot	full-shot
SL	GCN	19.96±5.89	20.14±2.37	22.08±2.22	24.49±1.08	28.88±1.15	47.09±5.81	58.38±4.08	61.09±4.34	73.35±3.39	89.75±0.77
	GAT	20.77±4.48	22.69±4.70	23.26±3.45	25.57±2.33	31.70±1.30	47.33±4.97	58.49±4.65	62.25±3.52	72.65±2.33	88.97±1.98
	SAGE	22.34±4.32	24.56±3.28	28.99±2.94	32.25±1.32	38.17±0.77	49.72±3.81	60.33±3.73	64.58±3.35	73.99±1.45	**90.61**±0.59
	GPR	21.06±6.14	23.31±1.89	27.50±0.94	31.68±1.03	32.99±0.87	43.39±4.66	52.66±3.44	59.91±3.47	72.89±3.60	85.39±1.57
	H2GCN	21.10±3.06	25.57±4.47	27.95±4.06	29.91±3.44	35.25±2.40	45.81±6.09	58.86±5.33	66.74±4.88	72.26±2.84	89.05±2.26
TT	GTrans	—	—	—	—	30.70±2.51	—	—	—	—	90.06±1.44
Lin	GraphAny (C)	—	—	—	—	48.49±0.98	—	—	—	—	90.14±0.93
	GraphAny (W)	—	—	—	—	42.34±3.46	—	—	—	—	90.18±0.91
SSL+FT	LP	20.04±5.61	21.11±3.69	22.48±3.35	25.07±2.09	28.64±1.98	48.82±6.55	56.88±5.36	61.18±4.50	73.72±3.31	87.64±0.96
	DGI	20.85±6.57	21.28±4.06	22.35±3.39	24.98±2.44	27.65±2.03	45.17±7.34	56.67±6.12	60.09±3.89	72.54±3.59	85.25±1.34
	GCL	19.97±5.62	20.88±3.39	21.58±3.32	23.34±2.60	25.53±2.71	47.33±5.89	58.34±4.91	61.16±4.43	74.08±4.47	89.07±3.43
SSL+Prompt	GPPT	21.16±5.95	22.59±2.87	24.93±2.38	27.94±2.45	29.08±2.70	50.19±7.74	54.65±6.24	59.87±5.08	73.40±3.73	69.98±3.36
	GPF	22.71±4.87	25.26±3.99	27.95±2.80	29.94±3.31	30.49±2.79	49.38±6.56	53.37±6.52	62.29±7.03	73.88±5.82	66.70±4.56
	GraphPrompt	23.02±4.89	26.10±5.17	30.08±3.43	32.28±2.99	34.40±2.56	46.65±6.53	55.62±7.08	60.74±6.50	67.99±4.78	64.40±4.34
	ProNoG	24.25±4.79	27.97±4.40	28.68±3.49	30.04±2.99	32.25±2.33	47.72±6.60	52.85±6.68	60.69±5.88	63.98±6.63	69.92±2.79
	DAGPrompt	25.67±6.34	28.79±4.01	30.99±3.27	34.62±2.27	37.33±3.34	52.96±6.07	60.72±7.14	65.77±5.28	66.04±4.99	71.66±3.98
	All-In-One	21.18±7.06	24.32±5.03	29.58±4.88	31.45±3.04	35.59±3.92	52.25±7.33	54.84±6.36	63.67±5.52	66.32±5.65	67.75±2.44
GFMs	GCOPE	22.16±5.77	23.25±5.65	30.79±4.43	34.45±2.88	36.01±2.12	55.69±4.68	61.99±4.11	64.29±3.77	70.79±4.31	72.33±2.20
	RiemannGFM	20.13±8.58	22.08±7.35	25.39±5.52	27.30±5.08	29.79±5.27	49.69±13.32	52.18±9.44	58.47±6.36	64.15±6.08	68.99±5.61
	MDGPT	24.41±7.01	26.62±6.94	28.85±5.69	32.15±4.57	37.30±4.45	54.96±10.25	58.99±10.28	61.59±7.68	65.52±6.33	69.12±5.07
	SAMGPT	25.75±6.29	29.02±4.96	31.17±4.03	34.49±3.04	39.98±4.35	56.33±9.04	63.95±8.89	67.27±6.82	73.95±5.65	77.82±4.49
	GFMate	**27.02**±6.22	**30.99**±4.75	**32.77**±3.58	**38.73**±2.06	**42.44**±1.73	**58.85**±2.17	**64.28**±4.72	**67.49**±2.97	**74.40**±0.95	78.06±2.69

Table 17: Full experimental results for node classification (Part 4). GraphAny and GTrans are evaluated only under the full-shot setting, as they are not designed for GNNs trained in few-shot scenarios. GraphAny (C) and GraphAny (W) denote GraphAny trained on Cora and Wisconsin, respectively. Results are highlighted by **best** and runner-up.

	Dataset	Cora					Citeseer				
	Setting	1-shot	3-shot	5-shot	10-shot	full-shot	1-shot	3-shot	5-shot	10-shot	full-shot
SL	GCN	29.85±8.98	35.26±4.77	50.48±3.39	58.79±2.14	81.09±1.18	33.39±11.86	43.66±6.80	45.71±5.52	53.39±2.10	68.18±0.97
	GAT	33.25±9.72	49.92±5.33	55.78±4.03	68.85±3.63	80.47±1.22	35.51±9.70	40.11±4.78	45.59±3.89	57.12±1.96	67.06±2.19
	SAGE	35.76±8.89	50.34±4.88	65.41±3.92	72.25±2.58	**81.98**±1.56	38.80±9.73	42.15±3.56	48.81±1.66	59.99±1.43	68.69±0.79
	GPR	38.99±15.77	50.49±9.49	64.25±2.44	70.36±1.85	79.43±2.06	29.77±13.22	33.89±7.75	39.79±4.88	48.97±3.25	61.33±1.03
	H2GCN	30.90±9.98	48.86±5.05	57.85±3.44	69.77±2.99	80.79±1.35	30.91±12.79	39.43±8.02	45.58±4.37	58.82±3.22	67.42±1.75
TT	GTrans	—	—	—	—	80.79±2.43	—	—	—	—	69.94±1.89
Lin	GraphAny (C)	—	—	—	—	79.98±0.36	—	—	—	—	68.90±0.07
	GraphAny (W)	—	—	—	—	77.82±1.15	—	—	—	—	67.50±0.44
SSL+FT	LP	35.59±9.74	42.26±5.48	59.98±3.79	64.30±2.84	79.38±2.75	34.92±12.08	44.36±7.63	49.79±5.65	57.64±3.27	68.78±2.15
	DGI	32.38±8.86	39.98±6.33	57.34±2.81	64.46±2.12	77.26±2.59	33.96±11.57	41.77±7.90	47.62±4.89	55.43±3.31	65.46±1.92
	GCL	33.27±9.50	40.18±5.66	55.37±3.32	62.65±2.58	75.58±3.36	36.05±13.44	45.52±8.86	50.49±5.61	58.84±4.28	66.35±2.44
SSL+Prompt	GPPT	40.62±8.69	49.40±4.71	63.39±2.02	69.84±1.92	71.19±1.78	39.79±10.67	47.65±6.83	55.48±5.16	59.89±3.55	64.49±3.13
	GPF	45.75±9.61	52.29±5.48	65.15±2.58	67.35±3.03	72.16±2.28	40.51±12.79	49.07±7.62	57.35±5.68	59.88±4.55	62.75±3.06
	GraphPrompt	49.77±8.82	58.84±4.98	70.34±2.69	71.76±2.27	74.51±1.99	38.69±13.98	45.16±6.67	49.95±6.60	55.83±5.44	59.98±5.14
	ProNoG	56.54±12.33	59.87±8.84	68.83±3.40	71.35±1.98	74.41±2.25	37.79±13.35	40.06±7.14	53.39±7.33	60.96±4.99	62.29±5.40
	DAGPrompt	54.88±9.24	62.59±5.78	70.19±2.08	72.98±0.99	75.08±1.13	47.24±9.59	60.99±6.82	62.57±4.96	66.72±2.77	68.16±3.25
	All-In-One	49.92±11.75	50.44±4.39	66.43±2.21	69.51±1.17	70.34±1.42	40.69±15.88	48.05±6.20	53.25±5.68	62.23±4.59	64.49±3.34
GFMs	GCOPE	39.06±12.52	45.51±5.17	59.32±2.16	63.86±2.39	69.28±2.50	42.26±14.19	50.35±7.79	57.91±6.77	65.70±4.02	67.72±2.25
	RiemannGFM	37.91±16.13	45.58±7.60	58.85±4.52	65.59±4.39	68.75±3.84	38.02±9.58	42.19±7.67	54.03±4.79	59.98±3.72	61.46±3.41
	MDGPT	44.52±11.39	52.88±6.72	59.79±6.55	64.77±5.28	66.98±4.55	41.98±12.24	45.59±9.64	49.82±6.33	55.79±6.17	62.88±4.82
	SAMGPT	52.83±12.04	63.39±7.71	65.98±4.83	69.27±2.31	75.72±2.24	47.76±13.06	49.58±8.71	52.25±5.58	61.17±3.29	65.87±1.98
	GFMate	**59.68**±5.37	**70.51**±2.11	**72.23**±1.61	**77.25**±1.03	80.02±1.97	**56.25**±13.33	**74.08**±1.72	**76.38**±0.52	**77.23**±0.88	**78.34**±1.07

Table 18: Full experimental results for node classification (Part 5) on large-scale dataset. GraphAny and GTrans are evaluated only under the full-shot setting, as they are not designed for GNNs trained in few-shot scenarios. GraphAny (C) and GraphAny (W) denote GraphAny trained on Cora and Wisconsin, respectively. Results are highlighted by **best** and runner-up.

	Dataset	Arxiv-year				
	Setting	1-shot	3-shot	5-shot	10-shot	full-shot
SL	GCN	18.64±6.91	20.26±4.74	23.17±2.11	27.39±1.96	33.44±1.98
	GAT	19.93±5.40	21.17±3.79	24.31±3.19	25.59±4.33	32.84±2.15
	SAGE	20.75±4.99	24.52±5.17	25.08±3.24	29.98±3.66	34.30±3.06
	GPR	10.81±9.94	15.45±5.57	20.37±4.51	24.46±3.39	28.77±2.58
	H2GCN	15.54±7.33	17.89±6.48	22.35±3.46	26.61±3.88	29.15±2.91
TT	GTrans	—	—	—	—	30.19±4.37
Lin	GraphAny (C)	—	—	—	—	31.47±2.58
	GraphAny (W)	—	—	—	—	30.91±1.76
SSL+FT	LP	17.94±6.06	18.84±5.81	22.74±2.88	26.70±1.65	31.59±1.89
	DGI	18.06±6.22	18.95±5.99	21.98±4.07	25.84±2.76	28.81±2.43
	GCL	15.53±8.89	17.66±7.44	19.95±5.89	23.47±4.43	26.08±2.69
SSL+Prompt	GPPT	19.96±7.63	20.15±4.39	24.40±2.75	24.98±2.56	25.80±2.06
	GPF	20.89±8.20	24.77±6.08	27.94±6.10	28.45±4.01	29.19±3.67
	GraphPrompt	22.61±4.66	23.72±3.89	26.50±2.77	27.89±3.25	29.03±2.70
	ProNoG	OOM	OOM	OOM	OOM	OOM
	DAGPrompt	23.08±8.14	25.31±5.82	29.79±3.14	30.36±2.09	31.17±3.09
	All-In-One	15.29±7.55	19.78±5.50	24.06±3.38	28.89±2.07	29.01±3.98
GFMs	GCOPE	17.98±5.51	21.04±4.89	25.75±3.06	30.09±1.82	31.15±2.48
	RiemannGFM	OOM	OOM	OOM	OOM	OOM
	MDGPT	OOM	OOM	OOM	OOM	OOM
	SAMGPT	OOM	OOM	OOM	OOM	OOM
	GFMate	**30.19**±3.65	**30.61**±2.97	**31.36**±2.90	**33.80**±2.55	**34.47**±2.69

Table 19: Cross-domain transfer learning performance of graph classification under one-shot and three-shot settings. Each dataset contains two columns corresponding to one-shot and three-shot results. The average accuracy (%) over five runs with standard deviation is reported. **Best** and Runner-up results are highlighted.

	Dataset	**PROTEINS**		**COX2**		**BZR**	
	Shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
SL	GCN	51.24±7.96	53.89±3.52	45.59±9.62	59.74±8.96	44.89±16.27	52.65±9.06
	GAT	50.89±8.04	52.53±4.60	50.71±7.26	62.29±8.47	46.58±15.83	55.34±9.22
	SAGE	51.39±6.92	53.58±3.96	55.74±5.98	63.36±6.52	52.39±12.98	58.85±7.63
	GPR	51.09±9.26	52.90±5.05	48.15±9.69	58.73±7.99	45.16±17.63	53.06±8.29
	H2GCN	52.28±8.75	54.41±3.69	53.67±7.82	60.08±7.81	49.88±13.35	54.40±8.81
SSL+FT	LP	52.29±7.50	53.92±3.57	53.91±12.24	65.49±10.16	52.29±16.73	55.57±9.68
	DGI	49.68±8.37	50.28±4.06	54.42±10.03	60.07±12.25	49.79±14.95	50.73±8.96
	GCL	50.75±7.86	51.17±4.43	46.61±9.85	58.84±10.75	50.48±16.89	53.59±9.57
SSL+Prompt	GPF	54.36±4.99	55.09±2.89	60.49±14.56	65.30±13.37	58.71±15.45	70.05±12.47
	GraphPrompt	55.79±9.62	55.82±3.04	62.13±9.38	65.76±11.82	55.07±13.18	58.89±9.92
	ProNoG	53.59±7.65	54.07±4.18	57.04±11.91	60.23±12.28	52.25±15.86	55.47±13.36
	DAGPrompt	56.57±4.84	57.45±3.96	56.28±8.34	63.39±11.75	55.47±17.58	65.69±9.31
	All-In-One	56.68±8.37	57.72±5.94	58.76±8.52	66.27±9.94	61.62±13.47	62.95±10.86
GFMs	GCOPE	53.21±6.44	55.58±4.45	54.90±13.25	62.29±10.71	58.15±16.09	60.08±11.70
	MDGPT	54.70±6.69	55.06±6.49	50.47±12.35	59.81±11.85	53.48±12.92	55.90±12.81
	SAMGPT	55.19±5.52	55.27±5.86	52.07±7.82	63.37±9.38	58.59±17.04	66.35±14.07
	GFMate	**59.47**±6.08	**61.24**±3.62	**66.39**±4.39	**69.82**±1.18	**65.77**±8.40	**77.06**±2.18