Test-time Adaptation on Graphs via Adaptive Subgraph-based Selection and Regularized Prototypes

Yusheng Zhao [1] **Qixin Zhang** [2] **Xiao Luo** [# 3] **Junyu Luo** [1] **Wei Ju** [1] **Zhiping Xiao** [# 4] **Ming Zhang** [# 1]

Abstract

Test-time adaptation aims to adapt a well-trained model using test data only, without accessing training data. It is a crucial topic in machine learning, enabling a wide range of applications in the real world, especially when it comes to data privacy. While existing works on test-time adaptation primarily focus on Euclidean data, research on non-Euclidean graph data remains scarce. Prevalent graph neural network methods could encounter serious performance degradation in the face of test-time domain shifts. In this work, we propose a novel method named Adaptive Subgraph-based SElection and Regularized Prototype SupervisIon (ASSESS) for reliable test-time adaptation on graphs. Specifically, to achieve flexible selection of reliable test graphs, ASSESS adopts an adaptive selection strategy based on fine-grained individual-level subgraph mutual information. Moreover, to utilize the information from both training and test graphs, ASSESS constructs semantic prototypes from the well-trained model as prior knowledge from the unknown training graphs and optimizes the posterior given the unlabeled test graphs. We also provide a theoretical analysis of the proposed algorithm. Extensive experiments verify the effectiveness of ASSESS against various baselines.

[1]School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China [2]College of Computing and Data Science, Nanyang Technological University, Singapore [3]Department of Computer Science, University of California, Los Angeles, CA, USA [4]Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, WA, USA. Correspondence to: Xiao Luo <xiaoluo@cs.ucla.edu>, Zhiping Xiao <patxiao@uw.edu>, Ming Zhang <mzhang_cs@pku.edu.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

1. Introduction

Learning from non-Euclidean graph-structured data has recently received increasing attention, with a wide range of applications in knowledge graph representation learning (Chen et al., 2020; Xu et al., 2020), social network analysis (Fan et al., 2020; Singh et al., 2024), molecular property prediction (Godwin et al., 2021; Cai et al., 2022; Stärk et al., 2022), novel drug discovery (Jiang et al., 2021; Bongini et al., 2021), and traffic flow forecasting (Zhao et al., 2023; Ju et al., 2024b). Graph neural networks (GNNs) (Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018) have achieved promising performance on learning representations on graph-structured data via message passing. Subsequently, graph-level features are obtained by pooling operations, enabling a plethora of downstream tasks.

Despite their superior performance, GNNs mostly assume that test graphs come from the same distribution as training graphs, which is often violated in real-world scenarios (Gui et al., 2022). Distribution shifts during test time are often inevitable, and some prior works (Wu et al., 2020; Zhang et al., 2021b; Yin et al., 2022; Liu et al., 2024) investigate the problem of unsupervised graph domain adaptation that attempt to transfer the knowledge from the labeled training graphs to unlabeled test graphs via domain discrepancy minimization. However, these methods typically require labeled source graphs to perform domain alignment, which is often impractical when facing data privacy issues (Tan et al., 2023). It is a common practice for institutions to disclose their pretrained models for downstream tasks, while keeping the training dataset private. Towards this end, this paper investigates a more realistic yet under-explored problem of test-time adaptation on graphs, which aims to adapt the off-the-shelf well-trained model using test-time data only, without accessing training data.

Previously, several solutions have been proposed to tackle the problem of test-time adaptation (Sun et al., 2020b; Liang et al., 2020; Nado et al., 2020; Gao et al., 2023; Litrico et al., 2023; Karmanov et al., 2024). These methods often adopt self-training with pseudo-labels or use data-centric methods on the test data. However, these works focus mainly on Euclidean data, while research on non-Euclidean graph-structured data remains scarce. Specifically, test-time adap-

tation on graphs faces two important challenges: *(1) How to overcome the label scarcity of test graphs more reliably in the face of structural shifts?* As the adaptation of the model is performed during test time, the labels of graphs are unknown. Existing methods (Liang et al., 2020; Wang et al., 2020a) often utilize pseudo labels or entropy minimization for supervision. However, the shift on both node attributes and graph structures would deteriorate the performance of GNNs (Bao et al., 2025), making the model's prediction unreliable and pseudo labels noisy. *(2) How to utilize the knowledge from unknown training graphs and the information from unlabeled test graphs more effectively?* Adapting the model during test time faces a dilemma: when the model flexibly adapts to self-training signals, it may easily overfit and lose the knowledge learned from the unknown training graphs. On the other hand, when the model is not flexible enough, it may not fully utilize the information from the unlabeled test graphs.

To tackle these two challenges, this paper proposes a novel method named Adaptive Subgraph-based SElection and Regularized Prototype SupervSion (ASSESS) for test-time adaptation on graphs. The idea behind ASSESS is to first select reliable test graphs and then use them for effective self-supervised learning balancing prior knowledge from unknown training graphs and posterior information from unlabeled test graphs. More specifically, to achieve reliable test graph selection, we adopt a fine-grained, individual-level and graph-specific selection strategy named Adaptive Subgraph-based Selection (ASBS) that considers both the model's confidence and individual-level subgraph structure information. Prior works (Xu et al., 2021; Zhang et al., 2021a; Guo & Li, 2022) typically select confident test data using a single threshold or class-wise thresholds. In contrast, we propose a more fine-grained selection strategy that utilizes the mutual information between subgraphs and original graphs to construct individual-level and structure-aware thresholds. This allows us to flexibly differentiate hard testing samples whose inherent graph structures are difficult for the model to classify. Thus, reliable test graphs can be selected and used for self-supervised learning.

Moreover, to achieve effective self-training that preserves the prior knowledge from the unknown training graphs and adapts to the unlabeled test graphs, we propose a novel method called Regularized Prototype Supervision (RPS). Concretely, we construct semantic prototypes of each class according to the well-trained model and use them as prior knowledge from the unknown training graphs. Then, we optimize the posterior of the prototypes given the unlabeled test graphs. The maximum a posteriori objective is then approximated by a self-supervised objective with regularization of the prototypes. This leads to a trade-off between fully utilizing the information of unlabeled testing graphs and retaining the knowledge of unknown training graphs. While the idea of prototype adjustment has been used in previous literature (Iwasawa & Matsuo, 2021), we formulate this from the theoretical foundation of Bayesian theory.

The contribution of this paper is summarized as follows. **(1)** We explore a practical yet under-explored problem of test-time adaptation on graphs, which aims to adapt the model with unlabeled test graphs without accessing training data. **(2)** We propose ASSESS that introduces adaptive subgraph-based selection (ASBS) for choosing reliable test graphs, and regularized prototype supervision (RPS) for balancing the prior knowledge from unknown training graphs and the information of unlabeled test graphs. **(3)** We provide theoretical analysis of the proposed ASSESS. **(4)** We perform extensive experiments to show that ASSESS outperforms state-of-the-art baselines, confirming its effectiveness.

2. Related Works

Graph Neural Networks. Graph neural networks (GNNs) (Kipf & Welling, 2017; Hamilton et al., 2017; Chen et al., 2018) is a powerful tool for learning representations of graph structured data, with a wide range of applications in recommendation (Wang et al., 2020b; Chang et al., 2021), social network analysis (Yoo et al., 2022), molecule property prediction (Lu et al., 2019; Kim et al., 2023), and traffic flow forecasting (Zhao et al., 2023; 2024). The mainstream of GNNs is message passing neural networks that aggregate the neighborhood information to form representations of each node (Ju et al., 2024a). Subsequently, the feature vectors of nodes are aggregated using various pooling techniques (Zhang et al., 2019; Bianchi et al., 2020) to form graph-level representations. Despite their superior performance, they usually assume that the same distribution is shared across the training graphs and test graphs (Li et al., 2019; Zhao et al., 2025b), which fails to hold in various real-world scenarios (Shi et al., 2023; Dai et al., 2022). When distribution shifts become inevitable among test graphs, test-time adaptation emerges as a viable tool for adapting the model to the test graphs. Therefore, in this work, we investigate the problem of test-time adaptation on graphs and propose a novel method named ASSESS to solve this important problem.

Test-time Adaptation. Test-time adaptation (Chen et al., 2022; Zhang et al., 2022; Karmanov et al., 2024; Zhang et al., 2024; Zhao et al., 2025a), sometimes referred to as source-free domain adaptation (Kundu et al., 2020; Liu et al., 2021; Yang et al., 2021), has received increasing attention due to its ability to adapt to test data distributions without accessing the training data. This is a more realistic setting since training data is not always available, considering data privacy issues (Wang et al., 2020a; Tan et al., 2023). Most existing methods of test-time adaptation roughly fall into one of the two categories, *i.e.* self-training and data-

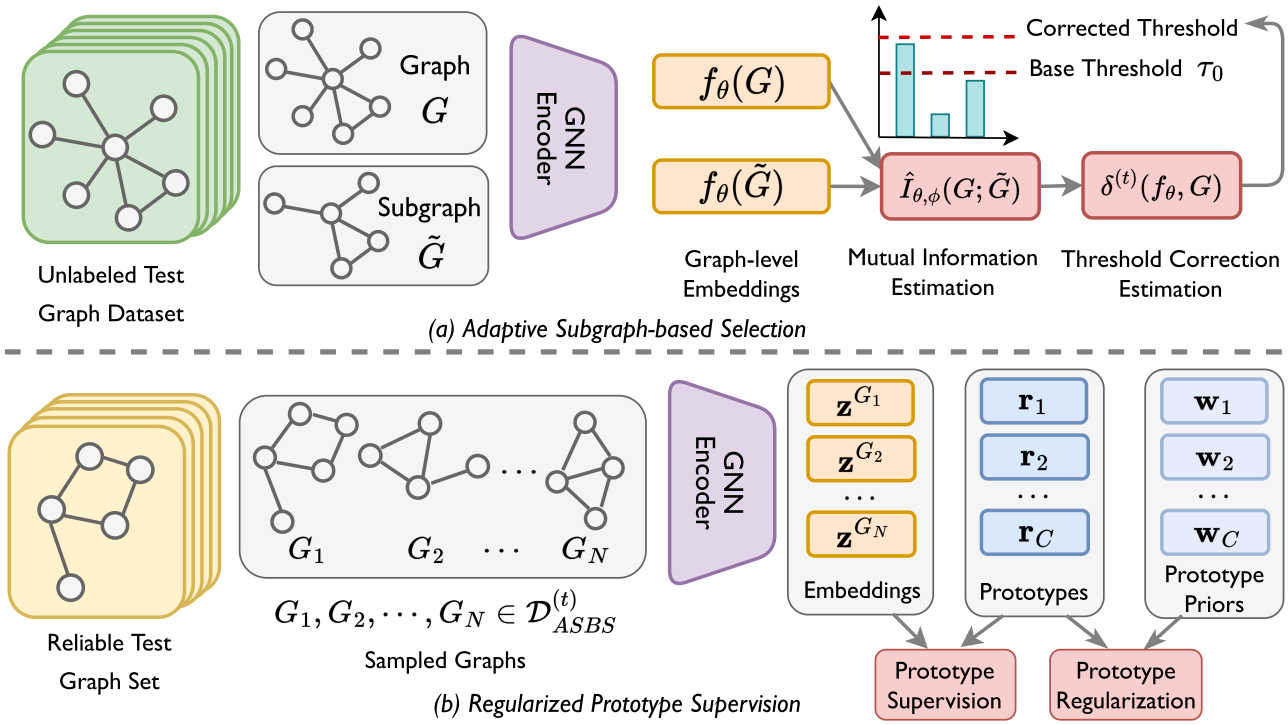

Figure 1. The overall framework of the proposed method. We first select reliable test graphs in the unlabeled test graph dataset using adaptive subgraph-based selection (ASBS), where we utilize mutual information to generate structure-aware, individual-level thresholds. Subsequently, we utilize these graphs for self-training with regularized prototype supervision (RPS), where the prototypes are regularized by prior knowledge and used for supervising the learned embedding of test graphs.

centric approaches. Self-training approaches (Sun et al., 2020b; Liang et al., 2020; Chen et al., 2022) aim to construct self-supervising signals via contrastive learning or pseudo-labeling. These methods often involve selecting reliable data for efficient self-training. By comparison, data-centric methods (Mocerino et al., 2021; Zhang et al., 2022; Tomar et al., 2023) aim to construct virtual samples related to the test distribution as data augmentation. In spite of their effectiveness in Euclidean data, test-time adaptation is still under-explored in non-Euclidean graph-structured data. Therefore, in this paper, we focus on test-time adaptation on graphs and propose ASSESS that adaptively selects reliable test graphs based on subgraph mutual information and uses them for effective self-training.

3. Methodology

Problem Formulation. We denote a graph as $G = (V, \boldsymbol{A}, \boldsymbol{X})$, where V is the set of nodes, $\boldsymbol{A} \in \mathbb{R}^{|V|\times|V|}$ is the adjacency matrix, and $\boldsymbol{X} \in \mathbb{R}^{|V|\times d^f}$ is the node attribute matrix with d^f as the number of attributes. We denote labeled training graph dataset as $\mathcal{D}^{tr} = \{(G_i^{tr}, y_i^{tr})\}_{i=1}^{N_{tr}}$ where G_i^{tr} is the i-th training graph, y_i^{tr} is its label and N_{tr} is the number of training graphs. The unlabeled test graph dataset is $\mathcal{D}^{te} = \{G_j^{te}\}_{j=1}^{N_{te}}$ where G_j^{te} denotes the jth test graph, and N_{te} is the number of test graphs. The label set is $\{1, 2, ..., C\}$. In test-time adaptation, the model is initially trained with labeled training (source) data $\mathcal{D}^{tr}$, and then adapted to the test (target) data $\mathcal{D}^{te}$ under distribution shifts, without access to $\mathcal{D}^{tr}$. We denote the GNN backbone and the following MLP as f_θ, and the graph features are generated as $\boldsymbol{z}^G = f_\theta(G)$. Then, class probabilities are generated as $\boldsymbol{p}^G = \mathrm{softmax}(\boldsymbol{W}^T \boldsymbol{z}^G)$. The entire model is denoted as $\mathcal{M}$, *i.e.* $\boldsymbol{p}^G = \mathcal{M}(G)$.

Overview. During test-time adaptation, the model first selects reliable test graphs based on the model's ability to handle the inherent structure of test data. To achieve this, we propose adaptive subgraph-based selection that utilizes mutual information of subgraph structures and the entire graph to measure this ability. Then the confidence threshold for each graph is adjusted accordingly for fine-grained, individual-level, and structure-aware selection. Subsequently, we adopt self-training supervised by regularized prototypes balancing prior knowledge from unknown training graphs and the posterior information from unlabeled test graphs. The selection-supervision process is performed iteratively. The overall framework is illustrated in Figure 1. The upper part displays the adaptive subgraph-based selection (ASBS), while the lower part shows the regularized prototype supervision (RPS).

3.1. Adaptive Subgraph-based Selection

Adapting the model during test time often involves self-training on unlabeled test data, in which the output prediction of the model is often used to construct self-supervising signals. However, as the model inevitably yields inaccurate predictions, it is crucial to carefully select reliable data, which can promise the success of these self-training approaches, especially for graph data that involves complex structures. Existing methods (Xu et al., 2021; Zhang et al., 2021a; Guo & Li, 2022) often adopt a shared threshold or class-wise thresholds to select data , *i.e.*

$$s^G_{conf} = \max_{i\in\{1,\cdots,C\}} p^G_i, \;\; \mathcal{D}^{conf} = \{G \in \mathcal{D}^{te} | s^G_{conf} > \tau\}, \tag{1}$$

where s^G_{conf} is the confidence score of graph G, p^G_i is the predicted probability of graph G belonging to class i, and τ is the threshold, which can be a shared threshold $\tau = \tau_0$ or class-wise threshold $\tau = \tau^c$.

However, this simple strategy can be problematic, as the model can be inaccurate in exploring the unlabeled test data, especially when it comes to graphs with complex inherent structures. Therefore, it is reasonable to use different selection thresholds for graph structures with different levels of complexity. Graphs with complex structures are hard to learn for the model, and the model tends to be inaccurate. Thus, a higher threshold should be used, and vice versa. Towards this end, we propose an adaptive selection strategy that considers both the model's confidence and the model's ability to handle the inherent structures of the graph. Specifically, instead of a shared threshold, we adopt a graph-specific, individual-level threshold, *i.e.*

$$\tau^G = \tau_0 + \delta(G, f_\theta), \tag{2}$$

where τ^G is the confidence threshold for graph G, τ_0 is the base threshold, and $\delta(G, f_\theta)$ is the correction function taking into account the model's ability to yield an accurate prediction from G.

In order to measure this ability, we propose to use mutual information (MI) between graph G and its subgraph $\tilde{G} = (V_{idx}, \boldsymbol{A}_{idx,idx}, \boldsymbol{X}_{idx,:})$. The idx is the randomly selected subgraph indices, which is a simple yet effective strategy. A high MI between a graph and its subgraphs indicates that the graph representation encodes information shared across its subgraphs. In other words, the encoder is able to handle the inherent structure of this graph. Once the subgraph $\tilde{G}$ is obtained, we approximate the MI between G and its subgraph $\tilde{G}$ as follows, using Jensen-Shannon MI estimator (Sun et al., 2020a),

$$\begin{aligned} I_{\theta,\phi}(G;\tilde{G}) = &-\mathbb{E}_{G\sim\mathcal{D}^{te}}[\mathrm{sp}(-g_\phi(f_\theta(G), f_\theta(\tilde{G}))] \\ &-\mathbb{E}_{G\sim\mathcal{D}^{te},G'\sim\mathcal{D}^{te}}[\mathrm{sp}(g_\phi(f_\theta(G'), f_\theta(\tilde{G})], \end{aligned} \tag{3}$$

where g_ϕ is the discriminator instantiated with a multi-layer perceptron, and $\mathrm{sp}(x) = \log(1+e^x)$ is the softplus function. The estimated mutual information $\hat{I}_{\theta,\phi}(G;\tilde{G})$ is then used for calculating the threshold correction function as follows,

$$\delta(G, f_\theta) = -\omega \hat{I}_{\theta,\phi}(G;\tilde{G}), \tag{4}$$

where ω is a hyperparameter balancing the relative importance of mutual information in the threshold.

Moreover, as accurate and stable estimation of mutual information requires sampling a number of graphs from the test graphs $\mathcal{D}^{te}$, which can be computationally expensive, we adopt the temporal ensembling technique (Hao et al., 2015; Laine & Aila, 2016) that gradually updates the threshold correction function as follows,

$$\delta^{(t)}(G, f_\theta) = \omega \sum_{i=0}^{t-1} \beta(1-\beta)^i \hat{I}^{(t-i)}_{\theta,\phi}(G;\tilde{G}), \tag{5}$$

where $\delta^{(t)}(G, f_\theta)$ is the threshold correction at epoch t, $\hat{I}^{(t)}_{\theta,\phi}(G;\tilde{G})$ is the estimated mutual information at epoch t, and β is the parameter controlling the speed of updates, which is set to 0.2. The above formula can be computed recursively, similar to the exponential moving average, which leads to a stable estimation of mutual information. With this, reliable test graphs can be selected as follows:

$$\mathcal{D}^{(t)}_{ASBS} = \{G \in \mathcal{D}^{te} | s^G_{conf} > \tau^G = \tau_0 + \delta^{(t)}(G, f_\theta)\}. \tag{6}$$

3.2. Regularized Prototype Supervision

Once reliable test graphs are selected, self-training can be subsequently performed. While previous methods directly fine-tune the model's parameters on the test data using self-supervising signals (*e.g.* pseudo labels, entropy minimization), it can drastically change the model and overfit the self-training objectives. Self-training faces a trade-off between preserving prior knowledge of the unknown training graphs and the posterior information of unlabeled test graphs. Liang et al. (Liang et al., 2020) freeze the classifier head of the model so as to avoid forgetting prior knowledge. In this subsection, we propose a more flexible alternative that combines prior knowledge and posterior information.

Specifically, we first construct semantic prototypes of each class as $\boldsymbol{r}_i$ and initialize them with the weight matrix of the last layer of the model, *i.e.* $\boldsymbol{r}_i = \boldsymbol{w}_i$. We denote the stacked prototypes as $\boldsymbol{R} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \cdots, \boldsymbol{r}_C] \in \mathbb{R}^{d\times C}$, where d is the dimension of prototypes. The goal is to optimize the prototypes given the information about the test graphs. This objective can be written as:

$$\boldsymbol{R}^* = \underset{\boldsymbol{R}}{\mathrm{argmax}} \log p(\boldsymbol{R}|G), \; G \in \mathcal{D}^{(t)}_{ASBS}. \tag{7}$$

Using the Bayes rule, this objective can be further decomposed into prior and likelihood, which is

$$\log p(\boldsymbol{R}|G) = \log p(\boldsymbol{R}) + \log p(G|\boldsymbol{R}) + C_0, \quad (8)$$

where C_0 is a constant. For the first term, we assume that the prototypes $\boldsymbol{R}$ follow isotropic Gaussian distribution, *i.e.* $\boldsymbol{R} \sim \mathcal{N}(\boldsymbol{R}; \boldsymbol{W}, \sigma_p^2 \boldsymbol{I})$, where the mean is set to the initial values of the prototypes, and σ_p^2 is the shared variance of the prototypes. Under this assumption, the prior knowledge constraint of the prototypes can be written as:

$$\begin{aligned}\log p(\boldsymbol{R}) &= \log \Big(\frac{1}{\sqrt{(2\pi\sigma_p^2)^d}} \exp\big(-\frac{1}{2\sigma_p^2} ||\boldsymbol{R} - \boldsymbol{W}||_2^2 \big) \Big) \\ &= C_1 ||\boldsymbol{R} - \boldsymbol{W}||_2^2 + C_2,\end{aligned} \quad (9)$$

where C_1 and C_2 are constants.

For the second term, we transform it into a self-supervised objective. Inspired by the mixture models (McLachlan & Basford, 1988; Reynolds et al., 2009), we have:

$$\begin{aligned}\log p(G|\boldsymbol{R}) &= \log p(G|\{\boldsymbol{r}_i\}_{i=1}^C) = \log \sum_{i=1}^C q(G,i) p(G|\boldsymbol{r}_i) \\ &\geq \sum_{i=1}^C q(G,i) \log p(G|\boldsymbol{r}_i)\end{aligned} \quad (10)$$

where $q(G,i)$ denotes the assignment of graph G belonging to prototype $\boldsymbol{r}_i$, and the inequality comes from Jensen's inequality. Thus, we have a lower bound for the likelihood. For the estimation of $q(G,i)$, a common approach is to compute the similarity between the representation of the graph $\boldsymbol{z}^G$ and the prototypes $\boldsymbol{r}_i$. Denoting $\boldsymbol{q}^G = [q(G,1), q(G,2), \cdots q(G,C)]^T$, this simple objective can be written as

$$\underset{\boldsymbol{q}^G}{\operatorname{argmin}} \langle \boldsymbol{q}^G, \boldsymbol{R}^T \boldsymbol{z}^G \rangle, \text{ s.t.} \langle \boldsymbol{q}^G, \boldsymbol{1}_C \rangle = 1, \quad (11)$$

where $\langle \cdot, \cdot \rangle$ denotes inner product measuring the similarity, and $\boldsymbol{1}_C \in \mathbb{R}^C$ is an all-one vector. However, this simple objective can result in unbalanced estimation, where the number of assignments to some classes is substantially larger than others. To solve this problem, another constraint is needed. Concretely, given a set of graphs $G_1, G_2, \cdots, G_N$, where N is the number of graphs, we hope to achieve a balanced assignment, which can be written as

$$\underset{\boldsymbol{Q}}{\operatorname{argmin}} \operatorname{tr}(\boldsymbol{Q}^T \boldsymbol{R}^T \boldsymbol{Z}), \text{ s.t.} \boldsymbol{Q}\boldsymbol{1}_N = \frac{N}{C}\boldsymbol{1}_C, \ \boldsymbol{Q}^T \boldsymbol{1}_C = \boldsymbol{1}_N, \quad (12)$$

where $\boldsymbol{Q} = [\boldsymbol{q}^{G_1}, \boldsymbol{q}^{G_2}, \cdots, \boldsymbol{q}^{G_N}] \in \mathbb{R}^{C \times N}$ is the stacked assignments, and $\boldsymbol{Z} = [\boldsymbol{z}^{G_1}, \boldsymbol{z}^{G_2}, \cdots, \boldsymbol{z}^{G_N}] \in \mathbb{R}^{d \times N}$ is the stacked graph representations. Solving this problem, however, requires solving a large linear program (Genevay et al., 2019), and a common workaround is to add entropy regularization that encourages the diversity of assignment. A Lagrangian objective can be written as follows

$$\begin{aligned}\mathcal{E} = \operatorname{tr}(\boldsymbol{Q}^T \boldsymbol{R}^T \boldsymbol{Z}) + \epsilon \mathcal{H}(\boldsymbol{Q}) + \langle \boldsymbol{a}, \boldsymbol{Q}\boldsymbol{1}_N - \frac{N}{C}\boldsymbol{1}_C \rangle \\ + \langle \boldsymbol{b}, \boldsymbol{Q}^T \boldsymbol{1}_C - \boldsymbol{1}_N \rangle,\end{aligned} \quad (13)$$

where $\mathcal{E}$ is the objective to be minimized, ϵ is the temperature parameters controlling the degree of regularization, $\boldsymbol{a}$ and $\boldsymbol{b}$ are Lagrangian multipliers, $\mathcal{H}(\boldsymbol{Q}) = -\sum_{i,j} \boldsymbol{Q}_{i,j}(\log \boldsymbol{Q}_{i,j} - 1)$ is the entropy regularization. We take the derivative with respect to $\boldsymbol{Q}_{i,j}$, which gives the following result:

$$\frac{\partial \mathcal{E}}{\partial \boldsymbol{Q}_{i,j}} = [\boldsymbol{R}^T \boldsymbol{Z}]_{i,j} - \epsilon \log \boldsymbol{Q}_{i,j} + \boldsymbol{a}_i + \boldsymbol{b}_j. \quad (14)$$

Therefore, the closed-form solution can be written as:

$$\boldsymbol{Q}^* = \operatorname{diag}\left(\exp\left(\frac{\boldsymbol{a}}{\epsilon}\right)\right) \exp\left(\frac{\boldsymbol{R}^T \boldsymbol{Z}}{\epsilon}\right) \operatorname{diag}\left(\exp\left(\frac{\boldsymbol{b}}{\epsilon}\right)\right). \quad (15)$$

In practice, we utilize the iterative Sinkhorn-Knopp algorithm (Asano et al., 2019; Caron et al., 2020; Zheng et al., 2021) to solve this problem (more details in Appendix B). Specifically, repeated row normalization and column normalization are conducted on $\exp\left(\frac{\boldsymbol{R}^T \boldsymbol{Z}}{\epsilon}\right)$, and the algorithm converges quickly. Previous studies (Genevay et al., 2019; Luo et al., 2023) indicate satisfactory performance within 3 iterations, and the soft assignment of the entropy is beneficial.

Then, we provide an estimation of $p(G|\boldsymbol{r}_i)$. Although this probability takes the form of generative modeling, actually training a generative model can be costly, and when facing insufficient data, it can be inaccurate. Therefore, we estimate this term in the latent space using the feature extractor f_θ with isotropic Gaussian distribution, which is $p(G|\boldsymbol{r}_i) \propto \exp\left(-\frac{1}{2\sigma_z^2}||f_\theta(G) - \boldsymbol{r}_i||_2^2\right)$, where σ_z^2 is the shared variance of the latent features. Combining the two terms, the loss function can be written as

$$\mathcal{L}_{RPS} = \sum_{i=1}^C q(G,i)||f_\theta(G) - \boldsymbol{r}_i||_2^2 + \alpha_1 ||\boldsymbol{R} - \boldsymbol{W}||_2^2, \quad (16)$$

where $q(G,i)$ is obtained via the Sinkhorn-Knopp iterations within a batch of graphs, and α_1 is the hyperparameter balancing the two terms. The first term serves as a self-training objective utilizing the posterior information from the unlabeled data, whereas the second term serves as regularization from the prior knowledge of the unknown training graphs. As the prototypes $\boldsymbol{r}_i$ (with their matrix form $\boldsymbol{R}$) are used to provide supervision signals in the self-training and regularized by the prior knowledge, we call this method regularized prototype supervision.

Algorithm 1 Optimization Algorithm of ASSESS

Input: The well-trained GNN-based model on the training graphs $\mathcal{M}^{(0)}(\cdot)$, test graphs $\mathcal{D}^{te}$,
Output: GNN-based model after T epochs $\mathcal{M}^{(T)}(\cdot)$

```
Initialize the prototypes R ← W;
Initialize the reliable test graph set D^(0)_ASBS = D^conf using Eq. 1;
for i = 1, 2, ..., T do
    for each batch do
        Sample a mini-batch of target graphs;
        Calculate L_RPS using Eq. 16 over D^(i)_ASBS;
        Calculate L_MI using Eq. 3 over D^te;
        Calculate the loss objective using Eq. 17;
        Update parameters of M^(i) through back-propagation;
    end for
    Update the threshold correction function using Eq. 5;
    Obtain reliable graphs D^(i+1)_ASBS using Eq. 6;
end for
```

3.3. Summarization

The algorithm is performed iteratively. In each iteration, we first adopt adaptive subgraph-based selection to obtain reliable test graph set $\mathcal{D}^{(t)}_{ASBS}$. Subsequently, regularized prototype supervision training is performed on the reliable graphs. The two steps are performed iteratively. As accurate estimations of mutual information require optimized parameters θ, ϕ, we also put the mutual information as part of the objective, and the final loss is:

$$\mathcal{L} = \mathcal{L}_{RPS} + \alpha_2 \mathcal{L}_{MI}, \tag{17}$$

where α_2 is the hyperparameter balancing the two terms and $\mathcal{L}_{MI} = -I_{\theta,\phi}(G; \tilde{G})$ is the objective of optimizing the mutual information. The first term $\mathcal{L}_{RPS}$ is calculated with the reliable graphs $\mathcal{D}^{(t)}_{ASBS}$, while the second term is calculated with the entire test graph set $\mathcal{D}^{te}$ to encourage participation from graphs that are initially regarded as unreliable. Moreover, we also adopt a curriculum learning strategy that gradually increase τ_0, since we want more participation during the initial stages of adaptation to fully utilize unlabeled test graphs and more reliability when the adaptation proceeds to avoid noisy signals. The learning algorithm is summarized in Algorithm 1. To demonstrate the efficiency of the proposed algorithm, we provide a comprehensive time complexity analysis (which shows that the proposed algorithm has the same time complexity as most GNNs) and running time information in Appendix C.

3.4. Theoretical Analysis

In this subsection, we aim to provide a detailed theoretical analysis of our proposed ASSESS method. For simplicity, we extract the pivotal elements of the ASSESS algorithm and make them more math-friendly. We start by defining ξ as the data pair, namely, $\xi \triangleq (G, y)$ where the symbol G is an input graph and y represents the corresponding label of G. The training graph set $\mathcal{D}^{tr}$ is supposed to follow an underlying distribution $\mathcal{P}$. The objective of our ASSESS algorithm is to reduce the expected loss function $\mathcal{L}(\mathbf{w})$ as much as possible, that is to say, we hope to solve the following stochastic optimization:

$$\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) := \mathbf{E}_{\xi \sim \mathcal{P}}[l(\mathbf{w}; \xi)], \tag{18}$$

where $\xi \sim \mathcal{P}$ denotes the graph-label tuple $\xi \triangleq (G, y)$ comes from the distribution $\mathcal{P}$, $\mathbf{w}$ represents the model parameters and $l(\mathbf{w}; \xi)$ is the loss objective associated with the data pair ξ. For the test pair from the unlabeled graphs $\mathcal{D}^{te}$, we assume it originates from a mixture of two distributions $\mathcal{P}$ (with a probability of q) and $\mathcal{Q}$ (with $1-q$). Formally, for any test pair $\xi = (G, y)$, $\xi \sim q\mathcal{P} + (1-q)\mathcal{Q}$. The class y in test pair ξ can be considered as a pseudo label generated by the pretrained model. To distinguish the former two distributions, we adopt the Tsybakov noisy condition:

Assumption 1 (Tsybakov Condition (Tsybakov, 2004))**.** *For any model parameter* $\mathbf{w}$*, if the expected loss* $\mathcal{L}(\mathbf{w})$ *is less than a threshold "*a*", the following inequality holds:*

$$\mathbf{E}_{\xi \sim \mathcal{Q}}[I_{\{\xi : l(\mathbf{w};\xi) \leq b\mathcal{L}(\mathbf{w})\}}(\xi)] \leq \mathcal{L}^m(\mathbf{w}), \tag{19}$$

where the symbol $I_S(\xi)$ *stands for the standard indicator function that equals* 1 *if* $\xi \in S$ *and 0 otherwise,* b *is a positive scaling constant and* $m \geq 1$ *is a fixed constant.*

Next, we make some frequently used assumptions about the smoothness and boundedness of (stochastic) gradient, i.e.,

Assumption 2 (Boundedness)**.** *The stochastic gradient* $\nabla l(\boldsymbol{w}; \xi)$ *is bounded, i.e.,* $\|\nabla l(\boldsymbol{w}; \xi)\| \leq G$ *where the symbol* $\|\cdot\|$ *denotes the Euclidean distance.*

Assumption 3 (Smoothness)**.** $\mathcal{L}(\boldsymbol{w})$ *is smooth with an* L*-Lipchitz continuous gradient, i.e.,* $\mathcal{L}(\boldsymbol{w})$ *is differentiable and there exists a positive constant* L *such that*

$$|\nabla \mathcal{L}(\boldsymbol{w}) - \nabla \mathcal{L}(\boldsymbol{u})| \leq L\|\boldsymbol{w} - \boldsymbol{u}\|. \tag{20}$$

Assumption 4 (Polyak-Łojasiewicz condition (Polyak, 1963))**.** *There exists a constant* $\mu > 0$ *such that*

$$2\mu\big(\mathcal{L}(\boldsymbol{w}) - \mathcal{L}(\boldsymbol{w}^*)\big) \leq \|\nabla \mathcal{L}(\boldsymbol{w})\|^2,$$

where $\boldsymbol{w}^*$ *is a optimal solution of problem* $\min_{\mathbf{w}} \mathcal{L}(\mathbf{w})$.

The Polyak-Łojasiewicz(PL) condition has garnered attention in deep learning research. Particularly, (Allen-Zhu et al., 2019) offers theoretical proof of its ability to guarantee linear convergence of gradient-based methods in non-convex

Table 1. The classification accuracy (in %, training → test) on FRANKENSTEIN (F0, F1, F2) and Mutangenicity (M0, M1, M2).

Methods	FRANKENSTEIN							Mutangenicity						
	F0→F1	F0→F2	F1→F0	F1→F2	F2→F0	F2→F1	**AVG**	M0→M1	M0→M2	M1→M0	M1→M2	M2→M0	M2→M1	**AVG**
GCN	58.5±0.6	52.2±0.8	56.2±0.7	55.6±2.0	54.2±2.3	55.8±1.8	55.4±1.4	72.7±1.3	59.2±1.2	74.3±0.2	70.7±1.8	66.3±2.4	72.4±1.3	69.3±1.4
GraphSAGE	61.3±1.8	56.5±0.6	58.0±1.9	56.7±2.9	55.2±1.8	58.8±1.5	57.7±1.7	74.6±1.6	58.0±1.6	75.4±1.6	66.4±0.8	63.8±4.0	72.8±5.9	68.5±2.6
GIN	61.7±1.4	54.1±1.5	57.7±1.3	58.1±1.2	56.2±1.4	60.1±4.6	58.0±1.9	75.1±3.2	59.7±0.8	74.9±1.1	67.6±1.1	65.3±2.8	70.8±1.4	68.9±1.7
GAT	58.1±2.1	51.3±1.5	58.0±2.2	57.0±1.6	53.2±1.4	58.1±2.5	56.0±1.9	73.1±0.3	59.2±1.7	73.6±2.0	67.1±2.3	59.3±1.8	70.6±2.0	67.1±1.7
MeanTeacher	61.2±1.0	58.1±1.7	58.7±0.6	59.6±2.9	55.7±1.4	54.0±1.9	57.9±1.6	67.9±4.0	61.0±4.1	71.5±4.3	68.1±2.4	70.7±1.6	74.8±1.2	69.0±2.9
GraphCL	60.6±1.7	56.5±1.3	54.7±3.0	57.0±1.8	53.7±2.5	60.9±2.2	57.2±2.1	70.9±4.3	64.0±1.3	66.7±5.7	71.5±1.1	65.4±2.9	73.4±4.0	68.6±3.2
SHOT	60.4±1.7	59.7±2.3	55.6±0.9	57.2±1.8	55.7±1.9	59.1±2.5	58.0±1.8	74.4±1.5	60.7±2.8	72.0±2.4	63.0±4.0	69.2±2.3	72.0±4.1	68.6±2.8
TAST	62.6±1.3	55.4±0.7	52.9±5.9	52.9±3.7	57.0±1.8	59.4±3.0	56.7±2.8	74.2±3.2	59.7±0.4	73.4±0.5	68.5±2.0	70.6±0.8	77.2±0.5	70.6±1.2
RNA	62.4±0.9	55.5±1.3	59.1±1.6	59.1±8.1	57.8±0.3	60.4±2.4	59.0±2.4	71.9±1.7	58.8±4.9	72.3±2.0	67.5±1.7	61.6±4.4	73.2±0.3	67.6±1.8
Ours	64.1±1.4	56.1±1.9	59.2±1.0	61.4±2.1	59.3±1.3	61.8±3.9	**60.3±1.9**	78.9±1.6	63.1±1.6	77.0±1.6	72.5±2.0	68.7±1.3	80.9±0.9	**73.5±1.5**

optimization. Moreover, (Yuan et al., 2019) have also furnished empirical evidence of the PL condition's presence during the training of deep neural networks.

The above assumptions are common in machine learning (Akhavan et al., 2024; Yuan et al., 2019; Allen-Zhu et al., 2019). With these standard assumptions of the loss function, we now show the convergence for ASSESS. Prior to that, let us present a more detailed characterization of the algorithm: i) At each epoch $t \in \{1, 2, \ldots, T\}$, we suppose the total size of the sampled target graphs is $n_t = n_1\gamma^{t-1}$ and we also use $\mathcal{D}_t^{te}$ to denote the set of considered test graphs at t; ii) The loss function $\mathcal{L}$ incorporates the mutual information for subgraph selection, as detailed in Section 3.3. Thus, we can, approximate the graph selection strategy as follows:

$$\mathcal{D}_{ASBS}^{(t)} \approx \{G \in \mathcal{D}^{te} | l(\mathbf{w}; \xi(G)) \leq \rho_t, \xi(G) = (G, y)\}, \tag{21}$$

where ρ_t is the dynamic threshold and y is the pseudo label of the test graph G; iii) At each iteration $t \in \{1, 2, \ldots, T\}$, we assume the model $\mathbf{w}_t$ is updated via the stochastic gradient descent (SGD), *i.e.*, $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g}_t$ where $\eta > 0$ is the step size and $\mathbf{g}_t$ is the average gradient over all reliable and considered graphs. With these detailed characterizations and assumptions, we have the following theorem:

Theorem 3.1 (Proofs are deferred to Appendix A)**.** *Under Assumption 1-4, if we suppose $\mathcal{L}(\mathbf{w}^*) = 0$ and $l(\mathbf{w}; \xi) \in [0, 1]$ for any parameter $\mathbf{w}$ and data pair ξ, when $\mathcal{L}(\mathbf{w}_1) \leq a$, $\rho_t \in [c\mathcal{L}(\mathbf{w}_t), b\mathcal{L}(\mathbf{w}_t)]$, $n_t = n_1\gamma^{t-1}$ and $\eta L \leq 1$ where $\gamma > 1$ and $n_1 = \left\lceil \max\left(\frac{\log(2/\delta)}{2q^2}, \frac{\log(2/\delta)}{2(1-q)^2}, \frac{\log(2/\delta)}{q(1-c^{-1})^2}\right)\right\rceil$ for any $\delta \in (0, 1), c \in (0, b)$, we can show that the final model parameter $\mathbf{w}_{T+1}$ produced by our ASSESS algorithm satisfies that, with a probability $1 - (4T + 1)\delta$, $\delta \in (0, 1)$,*

$$\mathcal{L}(\mathbf{w}_{T+1}) = O\big(\gamma^{-T}\big). \tag{22}$$

Remark 3.2. Theorem 3.1 provides a theoretical validation that when the pretrained model $\mathbf{w}_1$ satisfies a certain generalization property, such as $\mathcal{L}(\mathbf{w}_1) \leq a$, if we can continuously adjust the batch size (e.g. $n_t = n_1\gamma^{t-1}$) and efficiently select the high-quality test graphs (e.g. the dynamic threshold ρ_t is within the range $[c\mathcal{L}(\mathbf{w}_t), b\mathcal{L}(\mathbf{w}_t)]$), our ASSESS algorithm can progressively reduce the loss to adapt to the test-time distribution shifts. More specifically, after $O(\frac{\log(\frac{1}{\epsilon})}{\log(\gamma)})$ iterations, our ASSESS algorithm is capable of attaining an ϵ optimization error.

Remark 3.3. It is important to emphasize that the proof of Theorem 3.1 primarily draws from Theorem 1 presented in (Xu et al., 2021). However, in sharp contrast with Theorem 1 of (Xu et al., 2021), our Theorem 3.1 incorporates a more precise Tsybakov condition and a threshold range assumption of $\rho_t \in [c\mathcal{L}(\mathbf{w}_t), b\mathcal{L}(\mathbf{w}_t)]$, which enables it to cover a broader range of application scenarios. Furthermore, the refined Tsybakov condition and the specified range of ρ_t allow us to better adapt to probabilistic inequalities in different directions, e.g., Eq.(25) and Eq.(28) in Appendix A.

4. Experiment

4.1. Experimental Settings

Datasets. We perform experiments on a number of real-world datasets with the test-time adaptation setting. Concretely, we choose five representative datasets, including FRANKENSTEIN (Orsini et al., 2015), Mutangenicity (Kazius et al., 2005), PROTEINS (Borgwardt et al., 2005), NCI1 (Wale et al., 2008), and IMDB-BINARY (Yanardag & Vishwanathan, 2015). The datasets are split into three subsets according to graph density, leading to distribution shifts across splits. The datasets cover a wide range of fields, including chemistry, biology, and social networks. We evaluate our methods under the offline test-time adaptation setting (Wang et al., 2022), where all test inputs are available for adaptation. More details about the datasets are shown in Appendix D.

Compared Baselines. We compare our method with a wide range of baselines listed as follows: (a) Graph neural networks, including GCN (Kipf & Welling, 2017), GraphSAGE (Hamilton et al., 2017), GIN (Xu et al., 2019), and GAT (Veličković et al., 2018). (b) Unsupervised / Semi-supervised training methods, including Mean-Teacher (Tarvainen & Valpola, 2017) and GraphCL (You et al., 2020). (c) Test-time adaptation methods, including SHOT (Liang et al.,

Table 2. The classification accuracy (in %, training → test) on PROTEINS (P0, P1, P2) and NCI1 (N0, N1, N2).

Methods	PROTEINS							NCI1						
	P0→P1	P0→P2	P1→P0	P1→P2	P2→P0	P2→P1	**AVG**	N0→N1	N0→N2	N1→N0	N1→N2	N2→N0	N2→N1	**AVG**
GCN	57.0±4.8	41.9±0.7	64.8±2.0	54.7±3.3	62.5±1.3	61.7±3.5	57.1±2.6	70.1±3.1	54.8±1.3	52.6±2.0	56.3±1.9	50.1±3.3	63.0±1.5	57.8±2.2
GraphSAGE	53.9±5.3	41.5±0.5	70.3±3.2	57.3±1.9	69.0±0.9	62.1±2.1	59.0±2.3	69.2±1.0	55.8±0.8	51.2±1.2	59.5±2.7	48.8±0.8	61.6±3.7	57.7±1.7
GIN	58.9±1.1	41.7±0.6	64.2±1.9	53.1±2.2	59.2±0.7	68.2±2.4	57.6±1.5	73.0±2.5	56.9±1.5	55.6±4.7	59.1±2.2	50.3±0.9	64.5±1.9	59.9±2.3
GAT	63.2±4.3	41.5±0.5	77.1±2.9	63.8±4.1	64.4±2.9	61.0±1.8	61.8±2.8	70.5±6.9	55.6±0.4	51.9±1.4	57.4±1.9	50.7±1.2	63.9±2.9	58.3±2.5
MeanTeacher	72.0±7.2	65.3±5.2	70.5±2.8	66.5±7.3	73.3±5.1	67.2±2.6	69.1±5.0	71.5±8.6	61.9±3.7	64.7±10.5	75.4±2.5	59.4±12.3	66.1±1.5	66.5±6.5
GraphCL	79.2±6.0	73.7±6.4	75.2±5.2	73.9±4.1	73.5±5.1	77.0±7.4	75.4±5.7	76.2±3.3	67.6±1.3	69.0±1.1	77.3±1.5	66.2±3.2	73.1±3.2	71.6±2.3
SHOT	68.2±5.4	62.9±4.2	71.6±4.2	65.9±7.0	67.8±6.0	68.8±7.0	67.5±5.6	64.4±4.4	56.6±2.6	73.9±1.1	65.5±3.4	69.6±2.7	64.0±2.8	65.7±2.8
TAST	68.9±7.5	41.9±1.0	75.1±2.2	62.1±2.6	63.8±2.5	61.9±1.6	62.3±2.9	72.2±5.1	56.5±0.8	55.2±2.0	57.7±2.3	48.9±0.7	66.0±3.7	59.4±2.4
RNA	61.8±9.1	65.8±4.1	79.1±2.3	71.1±9.1	74.2±3.1	69.3±1.9	70.2±4.9	72.0±1.8	61.3±4.7	76.4±3.7	72.4±1.1	65.7±4.4	74.9±0.5	70.5±2.7
Ours	81.9±3.7	74.7±4.1	86.7±1.7	77.3±2.8	82.4±4.0	70.4±6.6	**78.9±3.8**	79.6±2.1	69.1±2.5	81.9±2.0	78.5±1.1	73.4±2.0	70.3±1.1	**75.5±1.8**

Table 3. The classification results (in %, training → test) on IMDB-BINARY (I0, I1, I2).

Methods	I0→I1	I0→I2	I1→I0	I1→I2	I2→I0	I2→I1	**AVG**
GCN	78.5±2.0	66.0±2.9	60.6±1.8	62.4±1.7	55.2±1.3	57.0±3.3	63.3±2.2
GraphSAGE	78.2±1.8	60.6±4.1	58.5±3.2	63.9±1.5	56.7±0.9	56.7±2.7	62.4±2.4
GIN	75.8±1.7	66.6±2.2	58.8±5.6	64.5±1.7	54.3±0.7	56.7±2.1	62.8±2.4
GAT	77.3±1.1	64.5±1.7	63.6±0.7	62.1±2.0	54.9±2.2	58.8±0.7	63.5±1.4
MeanTeacher	70.1±6.8	65.2±2.5	63.2±4.3	70.1±2.4	59.7±2.4	61.2±4.2	64.9±3.8
GraphCL	66.2±3.1	70.6±3.1	61.7±1.9	70.1±2.4	61.7±4.9	46.3±4.4	62.8±3.3
SHOT	68.9±5.1	66.6±4.5	65.7±3.4	67.8±6.8	61.5±7.3	60.6±9.4	65.2±6.1
TAST	80.3±2.2	65.1±3.8	58.2±2.1	64.2±1.9	57.6±0.7	58.2±2.5	63.9±2.2
RNA	64.2±3.2	67.2±4.9	59.7±3.2	69.2±2.5	61.7±3.5	68.7±3.2	65.1±3.4
Ours	82.1±3.7	66.7±3.7	63.7±1.9	63.7±1.9	69.7±3.1	61.2±3.2	**67.8±2.9**

2020), TAST (Jang et al., 2023) and RNA (Luo et al., 2024). More details about the baseline methods are in Appendix E.

Implementation Details. By default, we utilize two GIN (Xu et al., 2019) layers to learn graph representation, and then mean pooling is applied followed by a two-layer MLP to obtain the graph level features (*i.e.* $\boldsymbol{z}^G$). For the discriminator g_ϕ, the two inputs are concatenated and processed by a two-layer MLP to obtain a scalar-valued output. τ_0 is selected such that 80% of the samples are regarded as reliable in the first epoch and linearly decrease so that 60% of the samples are deemed reliable. As for hyperparameters, we set the following values by default. ω is set to 0.1, α_1 is set to 0.1, and α_2 is set to 10^{-2}. The learning rate is set to 10^{-4}. For optimization, we adopt Adam optimizer (Kingma & Ba, 2014), with learning rate of 10^{-4}. The experiments can be performed on an NVIDIA A40 GPU. The code is publicly available at https://github.com/YushengZhao/ASSESS.

4.2. Performance Comparison

The prediction accuracy on the test graphs is shown in Table 1, 2, and 3, from which we have the following observations: **Firstly**, ASSESS achieves a consistent improvement compared to all the baseline methods on average across all five datasets, which demonstrates the overall effectiveness of the proposed method. **Secondly**, the unsupervised / semi-supervised learning methods (Tarvainen & Valpola, 2017; You et al., 2020) are not the optimal solution. Although they experience improvement on some datasets like PROTEINS and NCI1, the performance gain on other datasets remains limited, which can be attributed to their lack of consideration of test-time distribution shifts. **Thirdly**, test-time adaptation methods designed for Euclidean data (Liang et al., 2020; Jang et al., 2023) are also not very satisfactory on graphs. Although they experience modest improvements on some datasets, their performance is still weaker than ASSESS, which can be attributed to their lack of consideration of complex structures of graph data.

Table 4. Ablation studies on three datasets, *i.e.* FRANKENSTEIN (FRAN), Mutangenicity (MUTA), and NCI1. Average accuracy over six settings (in %) is reported.

Experiments	FRAN	MUTA	NCI1	**AVG**
w/o ASBS	59.8±2.0	72.4±1.3	73.9±1.4	68.7
w/o RPS-a	59.4±1.5	71.3±1.1	74.0±1.7	68.2
w/o RPS-b	59.4±1.2	70.6±1.5	74.1±1.8	68.0
Full Model	60.3±1.9	73.5±1.5	75.5±1.8	69.8

4.3. Ablation Study

We now investigate the impact of the proposed adaptive subgraph-based selection (ASBS) and regularized prototype supervision (RPS). For RPS, we denote "RPS-a" as the first term of the loss function (Eq. 16), and "RPS-b" as its second term. The results are shown in Table 4, from which we have the following observations: **(1)** All of the proposed components contribute to the model's performance, since the improvement decreases without each of the ablated modules. **(2)** The performance drops by 1.1% on average without ASBS, showing the necessity of selecting reliable graphs adaptively during test time adaptation, as unreliable graphs might introduce noisy signals during self-training. **(3)** The model is suboptimal without supervision signals of the prototypes (removing the first term in Eq. 16, *i.e.* "w/o RPS-a"). Without proper supervision, the accuracy decreases by 1.6% on average, showing the importance of exploiting test graphs. **(4)** Removing the regularization on the prototypes (the second term in Eq. 16, "w/o RPS (2)") causes a considerable performance drop, demonstrating the importance of preserving prior knowledge of the model, which prevents catastrophic forgetting.

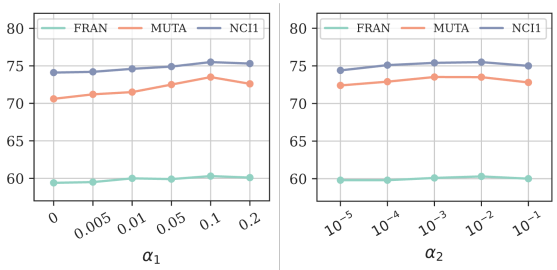

Figure 2. The model's sensitivity to hyperparameters (*i.e.* α_1 and α_2). The experiments are performed on FRANKENSTEIN (FRAN), Mutangenicity (MUTA), and NCI1. Average accuracy (in %) are reported over six settings.

4.4. Parameter Sensitivity

We then investigate the model's sensitivity to hyperparameters. Specifically, we focus on two of them: α_1, which balances the relative importance of prior prototype regularization in Eq. 16, and α_2, which is the weight of information maximization loss in Eq. 17. The results are shown in Figure 2, and we can see that the model is generally not sensitive to hyperparameters, as slight perturbations around optimal values yield similar accuracy. Specifically, setting α_1 to 0.1 yields the best performance. When α_1 is small, the weight of the prior prototype regularization is small, lacking regularization to the prototypes. Conversely, when α_1 is too large, strong constraint from the prior prevents adaptation to test graphs under distributional shifts. As for α_2, it balances the relative importance of $\mathcal{L}_{RPS}$ and $\mathcal{L}_{MI}$. This hyperparameter achieves its best at 10^{-3} or 10^{-2}. Generally, when α_2 is too small, it falls short in proper supervision of the MI training, leading to sub-optimal performance. Conversely, larger weights decrease the relative importance of the self-training loss, resulting in sub-optimal adaptation.

4.5. Visualization of Learned Representations

We also use t-SNE (Rauber et al., 2016) to visualize the learned representations of the test graphs. The results are shown in Figure 3, where we adapt the model initially trained on the P1 subset of the PROTEINS dataset to the P0 subset. The learned representations of the test graphs before and after adaptation are compared. As can be seen from the figure, the proposed method yields more clustered and condensed representations. This can be attributed to adaptive subgraph-based selection that filters out unreliable test graphs, and regularized prototype supervision that better utilizes unlabeled test graphs with self-training while preserving the knowledge learned from the inaccessible training graphs. More visualization about the learned representations can be found in Appendix F.

4.6. Analysis of Selection Results

In this subsection, we compare the selection results of adaptive subgraph-based selection (ASBS) with uniform threshold selection and class-wise threshold selection, and we show the selection accuracy (*i.e.* the accuracy of pseudo-labels of selected graphs) under varying epochs in Figure 4. As can be seen from the results, the selection accuracy of ASBS is significantly higher than those of others. The selection accuracy of the uniform threshold is much lower than ASBS, and it drops considerably after the 15th epoch. Using class-wise thresholds helps with the selection accuracy slightly, reaching 0.7. By comparison, ASBS reaches a high accuracy of nearly 0.8 and stabilizes as the adaptation proceeds, demonstrating that ASBS is better at identifying reliable test graphs during adaptation.

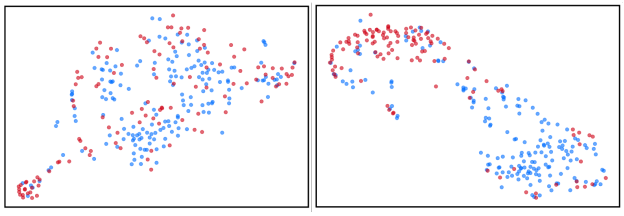

Figure 3. Visualization of learned representations on the PROTEINS dataset, before and after adaptation in the P1→P0 setting.

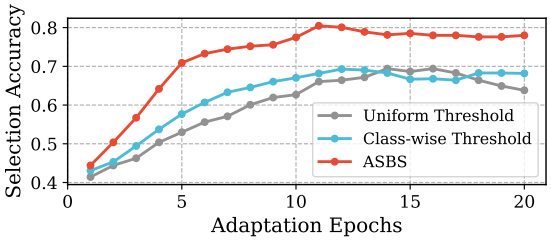

Figure 4. The selection accuracy using uniform threshold, class-wise threshold, and the proposed ASBS on the PROTEINS dataset.

5. Conclusion

This paper investigates test-time adaptation on graphs, and proposes a novel method named Adaptive Subgraph-based SElection and Regularized Prototype SupervisIon (ASSESS). To select reliable test graphs for self-training, ASSESS utilizes subgraph mutual information and selects test graphs adaptively. To fully utilize the information of unlabeled test graphs while preserving the knowledge from unknown training graphs, ASSESS adopts regularized prototype supervision that constructs semantic prototypes and performs self-training with regularization from the prior. Extensive experiments across various datasets validate the effectiveness of ASSESS.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgment

This paper is partially supported by grants from the National Key Research and Development Program of China with Grant No. 2023YFC3341203 and the National Natural Science Foundation of China (NSFC Grant Number 62276002). The authors are grateful to the anonymous reviewers for critically reading this article and for giving important suggestions to improve this article.

References

Akhavan, A., Chzhen, E., Pontil, M., and Tsybakov, A. B. Gradient-free optimization of highly smooth functions: improved analysis and a new algorithm. *Journal of Machine Learning Research*, 25(370):1–50, 2024.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*, pp. 242–252. PMLR, 2019.

Asano, Y. M., Rupprecht, C., and Vedaldi, A. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*, 2019.

Bao, W., Zeng, Z., Liu, Z., Tong, H., and He, J. Matcha: Mitigating graph structure shifts with test-time adaptation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Bianchi, F. M., Grattarola, D., and Alippi, C. Spectral clustering with graph neural networks for graph pooling. In *Proc. of International Conference on Machine Learning*, 2020.

Bongini, P., Bianchini, M., and Scarselli, F. Molecular generative graph neural networks for drug discovery. *Neurocomputing*, 450:242–252, 2021.

Borgwardt, K. M., Ong, C. S., Schönauer, S., Vishwanathan, S., Smola, A. J., and Kriegel, H.-P. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1): i47–i56, 2005.

Cai, H., Zhang, H., Zhao, D., Wu, J., and Wang, L. Fp-gnn: a versatile deep learning architecture for enhanced molecular property prediction. *Briefings in bioinformatics*, 23 (6):bbac408, 2022.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems*, 2020.

Chang, J., Gao, C., Zheng, Y., Hui, Y., Niu, Y., Song, Y., Jin, D., and Li, Y. Sequential recommendation with graph neural networks. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pp. 378–387, 2021.

Chen, D., Wang, D., Darrell, T., and Ebrahimi, S. Contrastive test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 295–305, 2022.

Chen, J., Ma, T., and Xiao, C. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*, 2018.

Chen, X., Jia, S., and Xiang, Y. A review: Knowledge reasoning over knowledge graph. *Expert systems with applications*, 141:112948, 2020.

Dai, Q., Wu, X.-M., Xiao, J., Shen, X., and Wang, D. Graph transfer learning via adversarial domain adaptation with graph convolution. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4908–4922, 2022.

Fan, W., Ma, Y., Li, Q., Wang, J., Cai, G., Tang, J., and Yin, D. A graph neural network framework for social recommendations. *IEEE Transactions on Knowledge and Data Engineering*, 34(5):2033–2047, 2020.

Gao, J., Zhang, J., Liu, X., Darrell, T., Shelhamer, E., and Wang, D. Back to the source: Diffusion-driven adaptation to test-time corruption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Genevay, A., Dulac-Arnold, G., and Vert, J.-P. Differentiable deep clustering with cluster size constraints. *arXiv preprint arXiv:1910.09036*, 2019.

Godwin, J., Schaarschmidt, M., Gaunt, A., Sanchez-Gonzalez, A., Rubanova, Y., Veličković, P., Kirkpatrick, J., and Battaglia, P. Simple gnn regularisation for 3d molecular property prediction & beyond. *arXiv preprint arXiv:2106.07971*, 2021.

Gui, S., Li, X., Wang, L., and Ji, S. Good: A graph out-of-distribution benchmark. *Advances in Neural Information Processing Systems*, 35:2059–2073, 2022.

Guo, L.-Z. and Li, Y.-F. Class-imbalanced semi-supervised learning with adaptive thresholding. In *International Conference on Machine Learning*, pp. 8082–8094. PMLR, 2022.

Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017.

Hao, L., Healey, C. G., and Bass, S. A. Effective visualization of temporal ensembles. *IEEE Transactions on*

Visualization and Computer Graphics, 22(1):787–796, 2015.

Iwasawa, Y. and Matsuo, Y. Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34: 2427–2440, 2021.

Jang, M., Chung, S.-Y., and Chung, H. W. Test-time adaptation via self-training with nearest neighbor information. In *The Eleventh International Conference on Learning Representations*, 2023.

Jiang, D., Wu, Z., Hsieh, C.-Y., Chen, G., Liao, B., Wang, Z., Shen, C., Cao, D., Wu, J., and Hou, T. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of cheminformatics*, 13: 1–23, 2021.

Ju, W., Fang, Z., Gu, Y., Liu, Z., Long, Q., Qiao, Z., Qin, Y., Shen, J., Sun, F., Xiao, Z., et al. A comprehensive survey on deep graph representation learning. *Neural Networks*, pp. 106207, 2024a.

Ju, W., Zhao, Y., Qin, Y., Yi, S., Yuan, J., Xiao, Z., Luo, X., Yan, X., and Zhang, M. Cool: a conjoint perspective on spatio-temporal graph neural network for traffic forecasting. *Information Fusion*, 107:102341, 2024b.

Karmanov, A., Guan, D., Lu, S., El Saddik, A., and Xing, E. Efficient test-time adaptation of vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14162–14171, 2024.

Kazius, J., McGuire, R., and Bursi, R. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of medicinal chemistry*, 48(1):312–320, 2005.

Kim, S., Lee, D., Kang, S., Lee, S., and Yu, H. Learning topology-specific experts for molecular property prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Kundu, J. N., Venkat, N., Babu, R. V., et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4544–4553, 2020.

Laine, S. and Aila, T. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

Li, J., Rong, Y., Cheng, H., Meng, H., Huang, W., and Huang, J. Semi-supervised graph classification: A hierarchical graph perspective. In *The World Wide Web Conference*, pp. 972–982, 2019.

Liang, J., Hu, D., and Feng, J. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *Proc. of International Conference on Machine Learning*, 2020.

Litrico, M., Del Bue, A., and Morerio, P. Guiding pseudo-labels with uncertainty estimation for source-free unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Liu, M., Fang, Z., Zhang, Z., Gu, M., Zhou, S., Wang, X., and Bu, J. Rethinking propagation for unsupervised graph domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13963–13971, 2024.

Liu, Y., Zhang, W., and Wang, J. Source-free domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1215–1224, 2021.

Lu, C., Liu, Q., Wang, C., Huang, Z., Lin, P., and He, L. Molecular property prediction: A multilevel quantum interactions modeling perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

Luo, J., Xiao, Z., Wang, Y., Luo, X., Yuan, J., Ju, W., Liu, L., and Zhang, M. Rank and align: Towards effective source-free graph domain adaptation. *arXiv preprint arXiv:2408.12185*, 2024.

Luo, X., Zhao, Y., Qin, Y., Ju, W., and Zhang, M. Towards semi-supervised universal graph classification. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

McLachlan, G. J. and Basford, K. E. *Mixture models: Inference and applications to clustering*, volume 38. M. Dekker New York, 1988.

Mocerino, L., Rizzo, R. G., Peluso, V., Calimera, A., and Macii, E. Adaptta: adaptive test-time augmentation for reliable embedded convnets. In *2021 IFIP/IEEE 29th International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 1–6. IEEE, 2021.

Nado, Z., Padhy, S., Sculley, D., D'Amour, A., Lakshminarayanan, B., and Snoek, J. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020.

Orsini, F., Frasconi, P., and De Raedt, L. Graph invariant kernels. In *Proceedings of the twenty-fourth international joint conference on artificial intelligence*, volume 2015, pp. 3756–3762, 2015.

Polyak, B. T. Gradient methods for minimizing functionals. *Zhurnal vychislitel'noi matematiki i matematicheskoi fiziki*, 3(4):643–653, 1963.

Rauber, P. E., Falcão, A. X., and Telea, A. C. Visualizing Time-Dependent Data Using Dynamic t-SNE. In *EuroVis*, pp. 73–77, 2016.

Reynolds, D. A. et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

Shi, B., Wang, Y., Guo, F., Shao, J., Shen, H., and Cheng, X. Improving graph domain adaptation with network hierarchy. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 2249–2258, 2023.

Singh, S. S., Muhuri, S., Mishra, S., Srivastava, D., Shakya, H. K., and Kumar, N. Social network analysis: A survey on process, tools, and application. *ACM Computing Surveys*, 56(8):1–39, 2024.

Stärk, H., Beaini, D., Corso, G., Tossou, P., Dallago, C., Günnemann, S., and Liò, P. 3d infomax improves gnns for molecular property prediction. In *International Conference on Machine Learning*, pp. 20479–20502. PMLR, 2022.

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

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-time training with self-supervision for generalization under distribution shifts. In *Proc. of International Conference on Machine Learning*, 2020b.

Tan, Y., Liu, Y., Long, G., Jiang, J., Lu, Q., and Zhang, C. Federated learning on non-iid graphs via structural knowledge sharing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Tarvainen, A. and Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, 2017.

Tomar, D., Vray, G., Bozorgtabar, B., and Thiran, J.-P. Tesla: Test-time self-learning with automatic adversarial augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20341–20350, 2023.

Tsybakov, A. B. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135 – 166, 2004. doi: 10.1214/aos/1079120131. URL https://doi.org/10.1214/aos/1079120131.

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

Wale, N., Watson, I. A., and Karypis, G. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems*, 14: 347–375, 2008.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020a.

Wang, Y., Li, C., Jin, W., Li, R., Zhao, J., Tang, J., and Xie, X. Test-time training for graph neural networks. *arXiv preprint arXiv:2210.08813*, 2022.

Wang, Z., Wei, W., Cong, G., Li, X.-L., Mao, X.-L., and Qiu, M. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 169–178, 2020b.

Wu, M., Pan, S., Zhou, C., Chang, X., and Zhu, X. Unsupervised domain adaptive graph convolutional networks. In *WWW*, 2020.

Xu, J., Kim, S., Song, M., Jeong, M., Kim, D., Kang, J., Rousseau, J. F., Li, X., Xu, W., Torvik, V. I., et al. Building a pubmed knowledge graph. *Scientific data*, 7(1):205, 2020.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Xu, Y., Shang, L., Ye, J., Qian, Q., Li, Y.-F., Sun, B., Li, H., and Jin, R. Dash: Semi-supervised learning with dynamic thresholding. In *International Conference on Machine Learning*, pp. 11525–11536. PMLR, 2021.

Yanardag, P. and Vishwanathan, S. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1365–1374, 2015.

Yang, S., Wang, Y., Van De Weijer, J., Herranz, L., and Jui, S. Generalized source-free domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8978–8987, 2021.

Yin, N., Shen, L., Li, B., Wang, M., Luo, X., Chen, C., Luo, Z., and Hua, X.-S. Deal: An unsupervised domain adaptive framework for graph-level classification. In *Proc. of ACM International Conference on Multimedia*, 2022.

Yoo, J., Shim, S., and Kang, U. Model-agnostic augmentation for accurate graph classification. In *Proc. of ACM Web Conference*, 2022.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. 2020.

Yuan, Z., Yan, Y., Jin, R., and Yang, T. Stagewise training accelerates convergence of testing error over sgd. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhang, B., Wang, Y., Hou, W., Wu, H., Wang, J., Okumura, M., and Shinozaki, T. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. In *Advances in Neural Information Processing Systems*, 2021a.

Zhang, M., Liu, K., Li, Y., Guo, S., Duan, H., Long, Y., and Jin, Y. Unsupervised domain adaptation for person re-identification via heterogeneous graph alignment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 3360–3368, 2021b.

Zhang, M., Levine, S., and Finn, C. Memo: Test time robustness via adaptation and augmentation. *Advances in neural information processing systems*, 35:38629–38642, 2022.

Zhang, Z., Bu, J., Ester, M., Zhang, J., Yao, C., Yu, Z., and Wang, C. Hierarchical graph pooling with structure learning. *arXiv preprint arXiv:1911.05954*, 2019.

Zhang, Z., Liu, M., Wang, A., Chen, H., Li, Z., Bu, J., and He, B. Collaborate to adapt: Source-free graph domain adaptation via bi-directional adaptation. In *Proceedings of the ACM on Web Conference 2024*, pp. 664–675, 2024.

Zhao, Y., Luo, X., Ju, W., Chen, C., Hua, X.-S., and Zhang, M. Dynamic hypergraph structure learning for traffic flow forecasting. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pp. 2303–2316. IEEE, 2023.

Zhao, Y., Luo, X., Wen, H., Xiao, Z., Ju, W., and Zhang, M. Embracing large language models in traffic flow forecasting. *arXiv preprint arXiv:2412.12201*, 2024.

Zhao, Y., Luo, J., Luo, X., Huang, J., Yuan, J., Xiao, Z., and Zhang, M. Attention bootstrapping for multi-modal test-time adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 22849–22857, 2025a.

Zhao, Y., Wang, C., Luo, X., Luo, J., Ju, W., Xiao, Z., and Zhang, M. Traci: A data-centric approach for multi-domain generalization on graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 13401–13409, 2025b.

Zheng, K., Liu, W., He, L., Mei, T., Luo, J., and Zha, Z.-J. Group-aware label transfer for domain adaptive person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

A. Proofs of Theorem 3.1

Proof. Before going into the details, we firstly introduce some auxiliary constants, that is,

$$\begin{aligned}
\alpha &= \sqrt{\frac{\log(2/\delta)}{qn_1(1-c^{-1})^2}};\\
\beta &= \max\left(\sqrt{\frac{\log(2/\delta)}{2q^2n_1}}, \sqrt{\frac{\log(2/\delta)}{2(1-q)^2n_1}}\right);\\
a_0 &= (1-c^{-1})(1-\beta)(1-\alpha)q;\\
\tau &= \max\left(\mathcal{L}(\mathbf{w}_1), \frac{4G^2}{\mu}\left(\frac{1}{\delta a_0 m} + \frac{b_0}{a_0}\right)\right);\\
b_0 &= 2\left((1-q)(1+\beta)\tau^m + \log(1/\delta)\right).
\end{aligned}$$

Note that when $n_1 = \left\lceil \max\left(\frac{\log(2/\delta)}{2q^2}, \frac{\log(2/\delta)}{2(1-q)^2}, \frac{\log(2/\delta)}{q(1-c^{-1})^2}\right)\right\rceil$ for any $\delta \in (0,1)$ and $0 < c < b$, $\alpha \leq 1$ and $\beta \leq 1$.

Next, we prove the Theorem 3.1 by showing that $\mathcal{L}(\mathbf{w}_t) \leq \tau\gamma^{-(t-1)}$. At first, when $t = 1$, we have

$$\mathcal{L}(\mathbf{w}_1) \leq \tau = \max\left(\mathcal{L}(\mathbf{w}_1), \frac{4G^2}{\mu}\left(\frac{1}{\delta a_0 m} + \frac{b_0}{a_0}\right)\right).$$

This establishes the initial bound for the induction. For the inductive step, we suppose that at some iteration $t < T$, with a probability $1 - (4t+1)\delta$, the following inequality holds, namely, $\mathcal{L}(\mathbf{w}_t) \leq \tau\gamma^{-(t-1)}$. We will show that this implies the bound on $\mathcal{L}(\mathbf{w}_{t+1})$.

From the assumption of Theorem 3.1, we know that, at each time step t, the total number of test graphs under consideration is given by $n_t := n_1\gamma^{t-1}$. For the purpose of our analysis, we categorize all test graphs into two distinct classes based on their derivation. The first class, denoted as A_t, comprises samples originating from the distribution consists of samples drawn from the distribution $\mathcal{P}$, while the second class, B_t, consists of test graphs from the distribution $\mathcal{Q}$. Subsequently, we further refine both A_t and B_t by identifying those samples whose associated loss values fall below the predetermined threshold ρ_t. Mathematically, these subsets can be expressed as:

$$\begin{aligned}
A_t^{\rho_t} &= \{\xi \in A_t : l(\mathbf{w}_t;\xi) \leq \rho_t\},\\
B_t^{\rho_t} &= \{\xi \in B_t : l(\mathbf{w}_t;\xi) \leq \rho_t\}.
\end{aligned}$$

By leveraging the standard concentration inequalities in probability theory and recognizing the true that all sampling graphs used to compute the gradient $\mathbf{g}_t$ are derived from the elements of $A_t^{\rho_t}$ and $B_t^{\rho_t}$, we can show the following results:
With a probability $1 - 2\delta$,

$$|A_t| \geq qn_t\left(1 - \sqrt{\frac{\log(2/\delta)}{2q^2n_t}}\right), \tag{23}$$

$$|B_t| \leq (1-q)n_t\left(1 + \sqrt{\frac{\log(2/\delta)}{2(1-q)^2n_t}}\right). \tag{24}$$

Next, based on the well-known inequality $\Pr(X \geq a) \leq \frac{\mathbf{E}(X)}{a}$ for any nonnegative random variable X, we also can show that

$$\Pr_{\xi\sim\mathcal{P}}(l(\mathbf{w}_t;\xi) \geq \rho_t) \leq \frac{\mathbf{E}_{\xi\sim\mathcal{P}}[l(\mathbf{w}_t;\xi)]}{\rho_t} = \frac{\mathcal{L}(\mathbf{w}_t)}{\rho_t} \leq \frac{1}{c}, \tag{25}$$

where the final inequality follows from $\frac{\mathcal{L}(\mathbf{w}_t)}{\rho_t} \leq \frac{1}{c}$. From the Eq.(25), we also can show that $\Pr_{\xi\sim\mathcal{P}}(l(\mathbf{w}_t;\xi) \leq \rho_t) \geq 1 - \frac{1}{c}$.

As a result, we have that, with a probability $1-2\delta$,

$$\begin{aligned}|A_t^{\rho_t}| \geq & \mathrm{Pr}_{\xi\sim\mathcal{P}}(l(\mathbf{w}_t;\xi)\leq\rho_t)|A_t|\left(1-\sqrt{\frac{\log(2/\delta)}{2\left(\mathrm{Pr}_{\xi\sim\mathcal{P}}(l(\mathbf{w}_t;\xi)\right)^2|A_t|}}\right)\\ \geq & \left(1-\frac{1}{c}\right)|A_t|\left(1-\sqrt{\frac{\log(2/\delta)}{2\left(1-\frac{1}{c}\right)^2|A_t|}}\right).\end{aligned} \tag{26}$$

Merging Eq.(23) into Eq.(26), we then can show that

$$|A_t^{\rho_t}| \geq \left(1-\frac{1}{c}\right)\left(1-\sqrt{\frac{\log(2/\delta)}{2q^2n_t}}\right)\cdot\left(1-\sqrt{\frac{\log(2/\delta)}{qn_1(1-c^{-1})^2}}\right)qn_t \geq a_0n_1\gamma^{t-1}. \tag{27}$$

Then, according to the Tsybakov Condition, we can derive that

$$\begin{aligned}\mathrm{Pr}_{\xi\sim\mathcal{Q}}\left(l(\mathbf{w}_t;\xi)\leq\rho_t\right) = & \mathrm{Pr}_{\xi\sim\mathcal{Q}}\left(l(\mathbf{w}_t;\xi)\leq\frac{\rho_t}{\mathcal{L}(\mathbf{w}_t)}\mathcal{L}(\mathbf{w}_t)\right)\\ \leq & \mathrm{Pr}_{\xi\sim\mathcal{Q}}(l(\mathbf{w}_t;\xi)\leq b\mathcal{L}(\mathbf{w}_t))\\ = & \mathbf{E}_{\xi\sim\mathcal{Q}}\left[I\left(l(\mathbf{w}_t;\xi)\leq b\mathcal{L}(\mathbf{w}_t)\right)\right]\\ \leq & \mathcal{L}^m(\mathbf{w}_t)\\ \leq & \left(\tau\gamma^{-(t-1)}\right)^m,\end{aligned} \tag{28}$$

where the first inequality uses the assumption that $\rho_t \leq b\mathcal{L}(\mathbf{w}_t)$ and $\mathcal{L}(\mathbf{w}_t)\in[0,1]$, the second inequality utilizes the Tsybakov condition and the final inequality from the assumption of induction.

It is worth noting that, compared with Eq.(45) in (Xu et al., 2021), our former Eq.(28) utilizes a more precise Tsybakov condition and a threshold range assumption of $\rho_t \in [c\mathcal{L}(\mathbf{w}_t), b\mathcal{L}(\mathbf{w}_t)]$ to effectively bound the probability of random event $\{l(\mathbf{w}_t;\xi)\leq\rho_t\}$.

Like the Eq.(26), with Eq.(28), we also can show that

$$\begin{aligned}&\mathbf{E}_{\xi\sim\mathcal{Q}}[|B_t^{\rho_t}|]\\ =&\sum_{i=1}^{|B_t|}\mathbf{E}_{\xi_i\sim\mathcal{Q}}\left[I\left(l(\mathbf{w}_t;\xi_i)\leq\rho_t\right)\right]\\ =&\sum_{i=1}^{|B_t|}\mathrm{Pr}_{\xi_i\sim\mathcal{Q}}(l(\mathbf{w}_t;\xi_i)\leq\rho_t)\\ \leq&|B_t|\left(\tau\gamma^{-(t-1)}\right)^m.\end{aligned} \tag{29}$$

With this inequality Eq.(29), according to the concentration inequality, we also can show that, with a probability $1-2\delta$,

$$\begin{aligned}|B_t^{\rho_t}| \leq & \mathbf{E}_{\xi\sim\mathcal{Q}}[|B_t^{\rho_t}|]+\frac{1}{3}\log(1/\delta)+\sqrt{\frac{1}{9}\log^2(1/\delta)+2\log(1/\delta)\mathbf{E}_{\xi\sim\mathcal{Q}}[|B_t^{\rho_t}|]}\\ \leq & |B_t|\left(\tau\gamma^{-(t-1)}\right)^m+\frac{1}{3}\log(1/\delta)+\sqrt{\frac{1}{9}\log^2(1/\delta)+2\log(1/\delta)|B_t|\left(\tau\gamma^{-(t-1)}\right)^m}\\ \leq & 2|B_t|\left(\tau\gamma^{-(t-1)}\right)^m+2\log(1/\delta)\\ \leq & (1-q)n_1\gamma^{t-1}\left(1+\sqrt{\frac{\log(2/\delta)}{2(1-q)^2n_t}}\right)\cdot\left(\tau\gamma^{-(t-1)}\right)^m+2\log(1/\delta)\\ \leq & 2\left((1-q)\left(1+\beta\right)\tau^m+\log(1/\delta)\right)n_1\\ = & b_0n_1.\end{aligned} \tag{30}$$

Therefore, according to the results of both Eq.(27) and Eq.(30), with a probability $1-4\delta$, we can show that

$$|A_t^{\rho_t}| \geq a_0 n_1 \gamma^{(t-1)}, \quad |B_t^{\rho_t}| \leq b_0 n_1. \tag{31}$$

Our analysis of Eq.(31) reveals a striking pattern in the dynamics of the sample subsets. As iterations proceed, the size of the subset $|A_t^{\rho_t}|$ exhibits exponential growth, while the subset $|B_t^{\rho_t}|$ remains tightly constrained within a fixed upper bound. This observation underscores the efficacy of our dynamic thresholding mechanism in progressively identifying and incorporating highly relevant unlabeled examples that align closely with the labeled data distribution.

Moreover, the selection process demonstrates remarkable stability, with the number of potential misclassifications maintained at a constant level throughout iterations. This balance between expansion and control enables our optimization framework to leverage an increasing volume of informative unlabeled data without compromising the integrity of the learning process. In the subsequent analysis, we will demonstrate that these properties translate to significant improvements in model performance. Specifically, we will show that with high probability, the expected loss of the model parameters at each iteration satisfies the bound $\mathcal{L}(\mathbf{w}_{t+1}) \leq \tau\gamma^{-t}$.

Before that, we also introduce some new auxiliary notations:

$$\begin{aligned}
\mathbf{g}_t^a &= \frac{1}{|A_t^{\rho_t}|} \sum_{\xi \in A_t^{\rho_t}} \nabla l(\mathbf{w}_t; \xi); \\
\mathbf{g}_t^b &= \frac{1}{|B_t^{\rho_t}|} \sum_{\xi \in B_t^{\rho_t}} \nabla l(\mathbf{w}_t; \xi); \\
b_t &= \frac{|B_t^{\rho_t}|}{|A_t^{\rho_t}| + |B_t^{\rho_t}|} \leq \frac{|B_t^{\rho_t}|}{|A_t^{\rho_t}|} \leq \frac{b_0}{a_0}\gamma^{-(t-1)}.
\end{aligned}$$

With these new symbols, we can rewrite $\mathbf{g}_t$ as $\mathbf{g}_t = (1-b_t)\mathbf{g}_t^a + b_t\mathbf{g}_t^b$.

Following L-smoothness of Assumption 3, we can have that

$$\mathcal{L}(\mathbf{w}) - \mathcal{L}(\mathbf{u}) \leq \langle \nabla\mathcal{L}(\mathbf{u}), \mathbf{w}-\mathbf{u}\rangle + \frac{L}{2}\|\mathbf{w}-\mathbf{u}\|^2. \tag{32}$$

According to Eq.(32), we also can have that

$$\begin{aligned}
&\mathcal{L}(\mathbf{w}_{t+1}) - \mathcal{L}(\mathbf{w}_t) \\
\leq &\langle \nabla\mathcal{L}(\mathbf{w}_t), \mathbf{w}_{t+1}-\mathbf{w}_t\rangle + \frac{L}{2}\|\mathbf{w}_{t+1}-\mathbf{w}_t\|^2 \\
= &\frac{\eta}{2}\|\nabla\mathcal{L}(\mathbf{w}_t) - \mathbf{g}_t\|^2 - \frac{\eta}{2}\left(\|\nabla\mathcal{L}(\mathbf{w}_t)\|^2 + (1-\eta L)\|\mathbf{g}_t\|^2\right) \\
\leq &\frac{\eta}{2}\left((1-b_t)\|\nabla\mathcal{L}(\mathbf{w}_t) - \mathbf{g}_t^a\|^2 + b_t\|\nabla\mathcal{L}(\mathbf{w}_t) - \mathbf{g}_t^b\|^2\right) \\
&-\frac{\eta}{2}\left(\|\nabla\mathcal{L}(\mathbf{w}_t)\|^2 + (1-\eta L)\|\mathbf{g}_t\|^2\right) \\
\leq &\frac{\eta}{2}\left((1-b_t)\|\nabla\mathcal{L}(\mathbf{w}_t) - \mathbf{g}_t^a\|^2 + 4b_tG^2\right) - \eta\mu\mathcal{L}(\mathbf{w}_t),
\end{aligned}$$

where the first equality follows the update of $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\mathbf{g}_t$; the final inequality follows from Assumption 2, Assumption 4 and $\eta L \leq 1$. Then, we derive a bound for the expectation of $\|\nabla\mathcal{L}(\mathbf{w}_t) - \mathbf{g}_t^a\|$, namely,

$$\begin{aligned}
&\mathbf{E}_{\xi\sim\mathcal{P}}\left[\|\mathbf{g}_t^a - \nabla\mathcal{L}(\mathbf{w}_t)\|^2\right] \\
= &\mathbf{E}_{\xi\sim\mathcal{P}}\left[\left\|\frac{1}{|A_t^{\rho_t}|}\sum_{\xi\in A_t^{\rho_t}}\nabla l(\mathbf{w}_t;\xi) - \nabla\mathcal{L}(\mathbf{w}_t)\right\|^2\right] \\
= &\frac{1}{|A_t^{\rho}|^2}\sum_{\xi\in A_t^{\rho}}\mathbf{E}_{\xi\sim\mathcal{P}}\left[\|\nabla l(\mathbf{w}_t;\xi) - \nabla\mathcal{L}(\mathbf{w}_t)\|^2\right] \leq \frac{4G^2}{|A_t^{\rho_t}|}.
\end{aligned}$$

Algorithm 2 Sinkhorn-Knopp Algorithm for Regularized Prototype Supervision

Input: scores $\boldsymbol{S}$, temperature ϵ, number of iterations n,
Output: $\boldsymbol{Q}$ as an approximation of $\boldsymbol{Q}^*$

```
Initialize the assignment Q ← exp(S/ε);
for i = 1, 2, ···, n do
    Normalize each row of Q to sum to N/C;
    Normalize each column of Q to sum to 1;
end for
return Q
```

Then, according to concentration inequality, we have with a probability $1-5\delta$,

$$\left\|\mathbf{g}_t^a - \nabla\mathcal{L}(\mathbf{w}_t)\right\|^2 \leq \frac{4G^2}{\delta|A_t^{\rho_t}|} \leq \frac{4G^2}{\delta a_0 n_1 \gamma^{t-1}}.$$

As a result,

$$\begin{aligned}
&\mathcal{L}(\mathbf{w}_{t+1}) - \mathcal{L}(\mathbf{w}_t) \\
\leq &\frac{\eta}{2}\left((1-b_t)\frac{4G^2}{\delta a_0 n_1 \gamma^{t-1}} + 4b_t G^2\right) - \eta\mu\mathcal{L}(\mathbf{w}_t) \qquad (33)\\
\leq &\frac{\eta}{2}\left(\frac{4G^2}{\delta a_0 n_1 \gamma^{t-1}} + 4G^2\frac{b_0}{a_0}\gamma^{-(t-1)}\right) - \eta\mu\mathcal{L}(\mathbf{w}_t) \\
= &2\eta G^2\left(\frac{1}{\delta a_0 n_1} + \frac{b_0}{a_0}\right)\gamma^{-(t-1)} - \eta\mu\mathcal{L}(\mathbf{w}_t), \qquad (34)
\end{aligned}$$

where the second inequality follows from the $b_t \leq \frac{b_0}{a_0}\gamma^{-(t-1)}$.

Finally, we have that

$$\mathcal{L}(\mathbf{w}_{t+1}) \leq (1-\eta\mu)\mathcal{L}(\mathbf{w}_t) + 2\eta G^2\left(\frac{1}{\delta a_0 n_1} + b_1\right)\gamma^{-(t-1)}. \qquad (35)$$

Then, if we set $\gamma = \frac{1}{1-\eta\mu/2}$, we know that

$$\begin{aligned}
\mathcal{L}(\mathbf{w}_{t+1}) \leq &\gamma(1-\eta\mu)\tau\gamma^{-t} + 2\eta\gamma G^2\left(\frac{1}{\delta a_0 n_1} + \frac{b_0}{a_0}\right)\gamma^{-t} \\
= &\left(1 - \frac{\eta\mu/2}{1-\eta\mu/2}\right)\tau\gamma^{-t} + \frac{\eta\mu/2}{1-\eta\mu/2}\frac{4G^2}{\mu}\left(\frac{1}{\delta a_0 n_1} + \frac{b_0}{a_0}\right)\gamma^{-t} \\
\leq &\tau\gamma^{-t},
\end{aligned}$$

where the first inequality follows from the induction assumption and the final inequality follows from $\tau \geq \frac{4G^2}{\mu}\left(\frac{1}{\delta a_0 n_1} + \frac{b_0}{a_0}\right)$. □

B. Details of the Sinkhorn-Knopp Algorithm

In this section, we provide an introduction of the Sinkhorn-Knopp algorithm. Specifically, we compute the matching score of the prototypes and the graph-level representations as $\boldsymbol{S} = \boldsymbol{R}^T\boldsymbol{Z}$, and initialize the assignment as $\boldsymbol{Q} = \exp(\boldsymbol{S}/\epsilon)$, where ϵ is the temperature. Then, we iteratively perform row normalization and column normalization of the assignment matrix such that it follows the constraints in Eq. 12. After n iterations, we obtain an approximation of the optimal assignment $\boldsymbol{Q}^*$ as $\boldsymbol{Q}$. The algorithm is summarized in Algorithm 2. Prior studies (Caron et al., 2020; Asano et al., 2019; Zheng et al., 2021) suggest that this algorithm converges quickly, and in practice setting n to 3 yields decent approximations.

Table 5. Adaptation time (s/epoch) and the number of epochs to converge of the proposed ASSESS.

Dataset	FRAN	MUTA	PROT	NCI1	IMDB
Time (s/epoch)	0.21	0.11	0.06	0.14	0.04
# Epochs	∼10	10∼20	10∼20	10∼20	∼30

C. Time Complexity and Efficiency Analysis

For the GNN backbone, the time complexity is $\mathcal{O}(N_{te}E)$, where N_{te} is the number of test graphs and E is the average number of edges. For ASBS, Eq. 10 has a time complexity of $\mathcal{O}(BN_{te})$, where B is the batch size. Then the model updates the threshold correction function recursively (Eq. 12), and selects reliable graphs (Eq. 13), which leads to $\mathcal{O}(N_{te})$. In RPS, Sinkhorn-Knopp algorithm has a time complexity of $\mathcal{O}(BC)$ for each batch, where C is the number of classes. Eq. 23 also has a time complexity of $\mathcal{O}(BC)$ for each batch. For each epoch, the time complexity is $O(CN_{te})$. The whole algorithm has a time complexity of $\mathcal{O}(N_{te}E)$, assuming constant batch size B and number of classes C, which is equivalent to most GNNs. In practice, the adaptation converges in a few seconds, as shown in Table 5.

D. Details of the Datasets and Dataset Splits

In this paper, we perform extensive experiments on five representative datasets, *i.e.* FRANKENSTEIN (Orsini et al., 2015), Mutangenicity (Kazius et al., 2005), PROTEINS (Borgwardt et al., 2005), NCI1 (Wale et al., 2008), and IMDB-BINARY (Yanardag & Vishwanathan, 2015). We split each dataset into three subsets, *i.e.* FRANKENSTEIN (F0, F1, F2), Mutangenicity (M0, M1, M2), PROTEINS (P0, P1, P2), NCI1 (N0, N1, N2), and IMDB-BINARY (I0, I1, I2). The datasets are partitioned according to the density of the graphs, which is defined as

$$D = \frac{2|E|}{|V|(|V|-1)}, \tag{36}$$

where D denotes density, $|E|$ denotes the number of edges, and $|V|$ denotes the number of nodes. The graphs are then sorted in ascending order according to the density and then partitioned into three dataset splits. Among the dataset partitions, F0, M0, P0, N0, and I0 have the smallest density values, while F2, M2, P2, N2, and I2 have the largest density values. We provide a detailed description of the datasets as follows:

- **FRANKENSTEIN** (Orsini et al., 2015) is a dataset created by combining the BURSI and MNIST datasets. FRANKENSTEIN modifies the BURSI dataset by removing bond type information and replacing the most common atom symbols with MNIST digit images. The original atom symbols are now encoded in the MNIST images' pixel intensity vectors, making it a challenging dataset.

- **Mutagenicity** (Kazius et al., 2005) is proposed by Kazius et al. and involves a wide range of molecular structures, each intricately associated with its respective Ames test data, totaling 4,337 molecular structures.

- **PROTEINS** (Borgwardt et al., 2005) is proposed by Borgwardt et al., and comprises protein data represented as graphs, with each graph's label identifying if a protein is a non-enzyme. In this dataset, amino acids are depicted as nodes, and edges are drawn between nodes when the distance between two amino acids is less than 6 angstroms.

- **NCI1** (Wale et al., 2008) is derived from the National Cancer Institute (NCI) database, which includes a large collection of compounds. Each graph in the NCI1 dataset has a label indicating whether a compound is active or inactive.

- **IMDB-BINARY** (Yanardag & Vishwanathan, 2015) consists of graphs derived from IMDB, the Internet Movie Database, where each graph represents the ego-network of a movie. In these graphs, nodes represent actors/actresses, and edges indicate that two actors appeared in the same movie. The primary objective with this dataset is to classify each graph into one of two categories based on the genre of the movie.

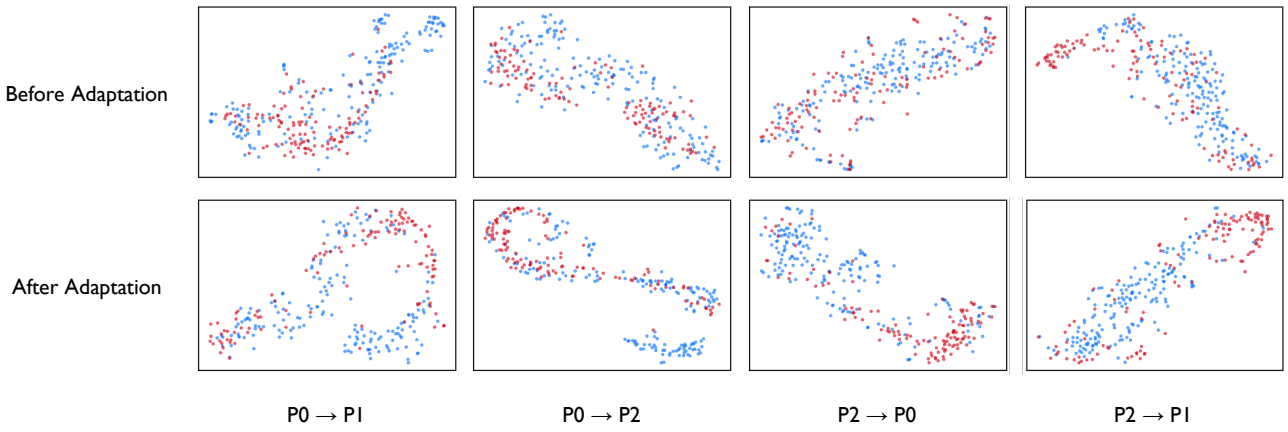

Figure 5. Additional visualization of learned representations of test graphs under distribution shifts on the PROTEINS dataset. The learned embeddings before and after adaptation is displayed.

E. Details of the Baseline Methods

The proposed method is compared to a wide range of baseline methods, including graph neural networks (GCN (Kipf & Welling, 2017), GraphSAGE (Hamilton et al., 2017), GIN (Xu et al., 2019), and GAT (Veličković et al., 2018)), unsupervised / semi-supervised training methods (Mean-Teacher (Tarvainen & Valpola, 2017) and GraphCL (You et al., 2020)), test-time adaptation methods (SHOT (Liang et al., 2020) and TAST (Jang et al., 2023)). We introduce these methods in more detail as follows:

- **GCN** (Kipf & Welling, 2017) is a foundational model that generalizes convolutional neural networks (CNNs) to graphs, efficiently aggregating neighborhood information to generate node representations,
- **GraphSAGE** (Hamilton et al., 2017) extends GCN by sampling and aggregating vectorized representations from the node's neighborhood, enabling it to handle large graphs dynamically.
- **GIN** (Xu et al., 2019) is proposed by Xu et al. and addresses the limitation of existing graph neural network models in distinguishing graph structures by learning to represent graphs in a way that preserves the graph isomorphism property, which effectively captures the topology of the graph data.
- **GAT** (Veličković et al., 2018) adopts the attention mechanism to the aggregation step in graph neural networks, enabling nodes to generate weights for the importance of their neighbors dynamically to achieve more effective feature integration.
- **MeanTeacher** (Tarvainen & Valpola, 2017) is a robust semi-supervised or unsupervised learning algorithm that utilizes a student model for making predictions and a teacher model to create training targets.
- **GraphCL** (You et al., 2020) introduces a graph contrastive learning framework aimed at advancing semi-supervised or unsupervised representation learning on graphs. It utilizes augmentations to embed various priors into the learning process.
- **SHOT** (Liang et al., 2020) (Source HypOthesis Transfer) fixes the classifier module (hypothesis) of the prediction model while learning a feature extraction module specific to the target domain. This process leverages information maximization and pseudo-labeling techniques for implicit alignment of the representations in target domains with the source hypothesis.
- **TAST** (Jang et al., 2023) (Test-time Adaptation via Self-Training with nearest neighbor information) gathers valuable information for classifying test data under domain shifts by leveraging information from nearest neighbors and employing various randomly initialized adaptation modules.

- **RNA** (Luo et al., 2024) (Rank and Align) ranks graph similarities to achieve robust semantic learning, and then aligns inharmonic graphs with harmonic graphs for subgraph extraction.

F. Additional Visualization of Learned Representations

In this section, we provide more visualization results of learned representations on the test graphs. More specifically, we use t-SNE (Rauber et al., 2016) in alignment with Section 4.5, and the results are shown in Figure 5. As can be seen from the results, the representations after adaptation are more clustered, which leads to better classification accuracy. Moreover, as we can see from the results, vanilla graph neural networks fall short in dealing with distribution shifts. The representations they learn from test graphs that experience distribution shifts are not satisfactory, in that the embeddings of two classes are mixed and hard to differentiate. By comparison, the proposed method adapt the model to the test graphs, without any ground truth labels from the test graphs, and improve the learned representations significantly.