Structural Alignment Improves Graph Test-Time Adaptation

Hans Hao-Hsun Hsu* Georgia Tech

Shikun Liu* Georgia Tech

Han Zhao UIUC

Pan Li Georgia Tech

Abstract

Graph-based learning excels at capturing interaction patterns in diverse domains like recommendation, fraud detection, and particle physics. However, its performance often degrades under distribution shifts, especially those altering network connectivity. Current methods to address these shifts typically require retraining with the source dataset, which is often infeasible due to computational or privacy limitations. We introduce Test-Time Structural Alignment (TSA), a novel algorithm for Graph Test-Time Adaptation (GTTA) that adapts a pretrained model to align graph structures during inference without the cost of retraining. Grounded in a theoretical understanding of graph data distribution shifts, TSA employs three synergistic strategies: uncertainty-aware neighborhood weighting to accommodate neighbor label distribution shifts, adaptive balancing of self-node and aggregated neighborhood representations based on their signal-to-noise ratio, and decision boundary refinement to correct residual label and feature shifts. Extensive experiments on synthetic and real-world datasets demonstrate TSA's consistent outperformance of both non-graph TTA methods and state-of-the-art GTTA baselines. Code is available at https://github.com/GraphCOM/TSA.

1 INTRODUCTION

Graph-based methods have become indispensable in handling structured data across a wide range of real-world applications (Duvenaud et al., 2015; Bronstein et al., 2017; Zhang et al., 2019a; Stokes et al., 2020), achieving significant success when training and testing data originate from similar distributions. However, these methods struggle to generalize to test-time data in a different domain, where variations in time, location, or experimental conditions result in distinct graph connection patterns. Although some literature on graph domain adaptation (GDA) (Wu et al., 2020; You et al., 2023; Zhu et al., 2021; Liu et al., 2023) seeks to bridge these gaps by aligning labeled source distributions with target distributions, such approaches are often infeasible in practice due to the cost of retraining.

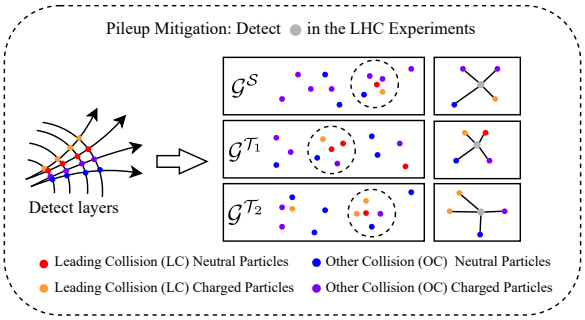

Figure 1: Example of the distribution shifts of neighborhood information due to different experimental conditions. Pileup mitigation aims to detect LC neutral particles using kNN graphs (dashed circles) that leverage nearby particles for inference. The model is trained on $\mathcal{G}^{\mathcal{S}}$ but needs to generalize to $\mathcal{G}^{\mathcal{T}_1}$ and $\mathcal{G}^{\mathcal{T}_2}$. The inferred nodes within the circles are the LC neutral particles, but their neighborhood node label ratios change in $\mathcal{G}^{\mathcal{T}_1}$ and $\mathcal{G}^{\mathcal{T}_2}$. Both cases represent *neighborhood shift*, which this work aims to address. A more formal definition of these shifts is discussed in Sec. 3.

Real-world applications may operate under constraints such as lightweight computation, limited storage, and strict privacy requirements, which make reprocessing of source data generally infeasible (Wang et al., 2021c; Wu et al., 2021). For example, Fig. 1 illustrates pileup mitigation based on graph learning at the Large Hadron Collider (LHC) (Shlomi et al., 2020; Highfield, 2008; Miao et al., 2024), where the connections between particles differ as experimental conditions change. To address this, the detector is expected to be capable of adapting on-the-fly (Li et al., 2023; Komiske et al.,

*Equal contribution. Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300. Copyright 2026 by the author(s).

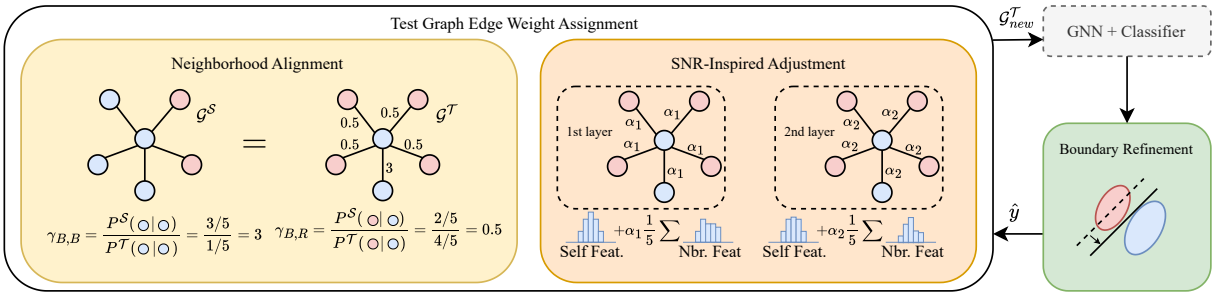

Figure 2: TSA utilizes neighborhood alignment to address neighborhood shift and SNR-inspired adjustment to mitigate SNR shift. It further adjusts the decision boundary to get refined predictions $\hat{y}$. The refined soft pseudo-labels are used to estimate the parameter $\boldsymbol{\gamma}$ for neighborhood alignment and to optimize α for combining self representations with neighborhood-aggregated representations.

2017), potentially on embedded hardware. Similarly, in fraud detection for financial networks (Clements et al., 2020; Wang et al., 2021b), privacy-sensitive raw training data may not be accessed during testing, while transaction patterns may change between different regions (Wang et al., 2019; Dou et al., 2020). These scenarios necessitate graph test-time adaptation (GTTA), which enables models to adjust to new test domains without incurring the overhead of full retraining.

Accurate node classification in graph-structured data relies heavily on effectively leveraging neighborhood information. Graph neural networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018; Hamilton et al., 2017; Hamilton, 2020) have achieved significant success across a wide range of applications by utilizing this neighborhood context, yet they remain sensitive to distribution shifts in node neighborhoods (Gui et al., 2022; Ji et al., 2023). In the above LHC example illustrated by Fig. 1, distribution shifts lead to distinct neighborhood connections among center particles in the same class, causing a neighborhood shift. Existing methods for test-time adaptation (TTA) often focus on adjusting only the classifier's final layer (Iwasawa & Matsuo, 2021), modifying its outputs (Boudiaf et al., 2022), or adapting normalization statistics (Wang et al., 2021a; Niu et al., 2023), primarily targeting image-based tasks with independently labeled data points. Consequently, they may not adequately address neighborhood shifts for node classification in graphs. While recent works on GTTA attempt to address the structural changes using graph augmentation (Jin et al., 2023), self-supervised learning (Zhang et al., 2024; Mao et al., 2024), these approaches mainly rely on augumentation heuristics and homophily consistency, failing to fully address changes in neighboring node connections in principle.

In this work, we propose a model-agnostic framework with test-time structural alignment, named TSA, as illustrated in Fig. 2, and supported by theoretical analysis. We begin by examining the generalization gap in GTTA for a pretrained source model. Our theoretical findings, further validated by empirical studies, indicate the importance of aligning the neighborhood distributions across the source and target domains. To this end, TSA employs a neighborhood weighting strategy to recalibrate the influence of neighboring nodes during message aggregation. This strategy is a form of first-order alignment, as it corrects for shifts in the neighborhood connection expectation. However, the weight assignment relies on the knowledge of node labels, which are unavailable in the target graph. Assigning weights based on pseudo labels can degrade performance. To mitigate this, TSA introduces an uncertainty-aware assignment strategy that aligns only node pairs with more reliable test-time pseudo labels.

Additionally, TSA goes beyond aligning first-order statistics by optimizing the test-time combination of self-node representations and neighborhood-aggregated representations based on their second-order signal-to-noise ratio (SNR). Our experiments show that neighborhood-aggregated representations tend to have higher SNR and are better denoised when the neighbor set size and the GNN layer depth increase. Under both conditions, TSA adjusts the combination by assigning greater weight to neighborhood-aggregated representations than to self-node representations to enhance model performance.

Lastly, TSA integrates non-graph TTA methods to refine the decision boundary, mitigating mismatches caused by additional label and feature shifts once neighborhood shifts have been addressed.

We conduct extensive experiments on synthetic and four real-world datasets, including those from high-energy physics and citation networks, with multiple GNN backbones, to demonstrate the effectiveness of TSA. TSA outperforms non-graph TTA baselines by up to 21.25% on synthetic datasets and surpasses all non-graph baselines. Compared to existing GTTA baselines, TSA achieves an average improvement of 10% on all real-world datasets.

2 PRELIMINARIES

2.1 Notations and Problem Setup

We use upper-case letters, such as Y to represent random variables, and lower-case letters, such as y to represent their realizations. Calligraphic letters, such as $\mathcal{Y}$ denote the domain of random variables. We use bold capital letters such as $\mathbf{Y}$ to represent the vectorized counterparts, i.e., collections of random variables. The probability of a random variable Y for a realization is expressed as $\mathbb{P}(Y = y)$.

Graph Neural Networks (GNNs). We let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote an undirected and unweighted graph with the symmetric adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ and the node feature matrix $\boldsymbol{X} = [x_1, \ldots, x_N]^T$. GNNs utilize neighborhood information by encoding $\boldsymbol{A}$ and $\boldsymbol{X}$ into node representations $\{h_v^{(k)}, v \in \mathcal{N}_u\}$. With $h_u^{(1)} = x_u$, the message passing in standard GNNs for node u and each layer $k \in [L] \coloneqq \{1, \ldots, L\}$ can be written as

$$h_u^{(k+1)} = \mathrm{UPT}(h_u^{(k)}, \mathrm{AGG}(\{h_v^{(k)}, v \in \mathcal{N}_u\})) \quad (1)$$

where $\mathcal{N}_u$ denotes the set of neighbors of node u, and $|\mathcal{N}_u|$ represents the node degree d_u. The AGG function aggregates messages from the neighbors, and the UPT function updates the node representations.

Test-Time Adaptation (TTA). Assume the model consists of a feature encoder $\phi : \mathcal{X} \to \mathcal{H}$ and a classifier $g : \mathcal{H} \to \mathcal{Y}$ and each domain $\mathcal{U} \in \{\mathcal{S}, \mathcal{T}\}$ consists of a joint feature and label distribution $\mathbb{P}^{\mathcal{U}}(X, Y)$. In TTA, the model is first trained on a labeled source dataset $\{(x_i, y_i)\}_{i=1}^N$, IID sampled from the source domain. In the target dataset $\{x_i\}_{i=1}^M$, the objective is to minimize the test error $\varepsilon^{\mathcal{T}}(g \circ \phi) = \mathbb{P}^{\mathcal{T}}(g(\phi(X)) \neq Y)$ given the source-pretrained model.

Graph Test-Time Adaptation (GTTA). Assume the model consists of a GNN feature encoder $\phi : \mathcal{X} \to \mathcal{H}$ and a classifier $g : \mathcal{H} \to \mathcal{Y}$. The model is trained on source graph $\mathcal{G}^{\mathcal{S}} = (\mathcal{V}^{\mathcal{S}}, \mathcal{E}^{\mathcal{S}})$ with node labels $\mathbf{y}^{\mathcal{S}}$ and the goal is to enhance model performance on a test graph $\mathcal{G}^{\mathcal{T}} = (\mathcal{V}^{\mathcal{T}}, \mathcal{E}^{\mathcal{T}})$ with distribution shifts that will be defined in Sec. 3.1. For node classification tasks, we aim to minimize the test error $\varepsilon^{\mathcal{T}}(g \circ \phi) = \mathbb{P}^{\mathcal{T}}(g(\phi(X_u, \mathbf{A})) \neq Y_u)$ given the source-pretrained model.

2.2 Related Works

Test-time Adaptation. Test-time training (TTT) approaches necessitate modifications to both the training pipeline and network architecture to accommodate customized self-supervised tasks, such as rotation prediction (Sun et al., 2020), contrastive learning (Liu et al., 2021; Bartler et al., 2022), and clustering (Hakim et al., 2023). These requirements, however, can limit their broader applicability. In contrast, our proposed setup aligns with the test-time adaptation (TTA) category, which does not alter the original training pipeline (Liang et al., 2024). One prominent TTA method, Tent (Wang et al., 2021a), adapts batch normalization parameters by minimizing entropy. This approach has inspired subsequent research focusing on normalization layer adaptation (Gong et al., 2022; Zhao et al., 2023; Lim et al., 2023; Niu et al., 2023). Other TTA strategies directly modify the classifier's predictions. For instance, LAME (Boudiaf et al., 2022) refines the model's soft predictions by applying regularization in the feature space. T3A (Iwasawa & Matsuo, 2021) classifies test data based on its distance to pseudo-prototypes derived from pseudo-labels. Building upon T3A, TAST (Jang et al., 2022) and PROGRAM (Sun et al., 2024) aim to construct more reliable prototype graphs. However, the effectiveness of these methods can be compromised when significant domain shifts occur (Mirza et al., 2023; Kang et al., 2023). To address larger domain shifts, some methods leverage stored, lightweight source statistics, such as label distributions, to perform test-time alignment (Mirza et al., 2023; Niu et al., 2024; Kojima et al., 2022; Eastwood et al., 2022; Adachi et al., 2024; Zhou et al., 2023). Act-MAD (Mirza et al., 2023), for example, aligns target feature statistics with stored source statistics across multiple layers. More recently, FOA (Niu et al., 2024) focuses on aligning the statistics of classification tokens in vision transformers. Note that all the aforementioned methods are primarily designed for image-based applications and may not adequately address shifts in the neighborhood information characteristic of graphs.

Graph Test-time Adaptation. Studies on GTTA are in two categories - node and graph classification. Graph classification problems can treat each graph as an i.i.d. input, allowing more direct extension of image-based TTA techniques to graphs (Chen et al., 2022; Wang et al., 2022). Our work focuses on node classification (Jin et al., 2023; Zhang et al., 2024; Mao et al., 2024; Ju et al., 2024). GTrans (Jin et al., 2023) proposes to augment the target graph at test time by optimizing a contrastive loss by generating positive views from DropEdge (Rong et al., 2020) and negative samples from the features of the shuffled nodes (Veličković et al., 2019). HomoTTT (Zhang et al., 2024) combines self-supervised learning with homophily-based DropEdge and prediction selection. SOGA (Mao et al., 2024) designs a self-supervised loss that relies on mutual information maximization and the homophily assumption. However, these works address the symptoms of the distribution shift (i.e., poor representations) by transforming the test graph or model parameters using a self-supervised contrastive loss, which relies on the choice of augmentation heuristics. In contrast, TSA is the first method that directly targets the root cause

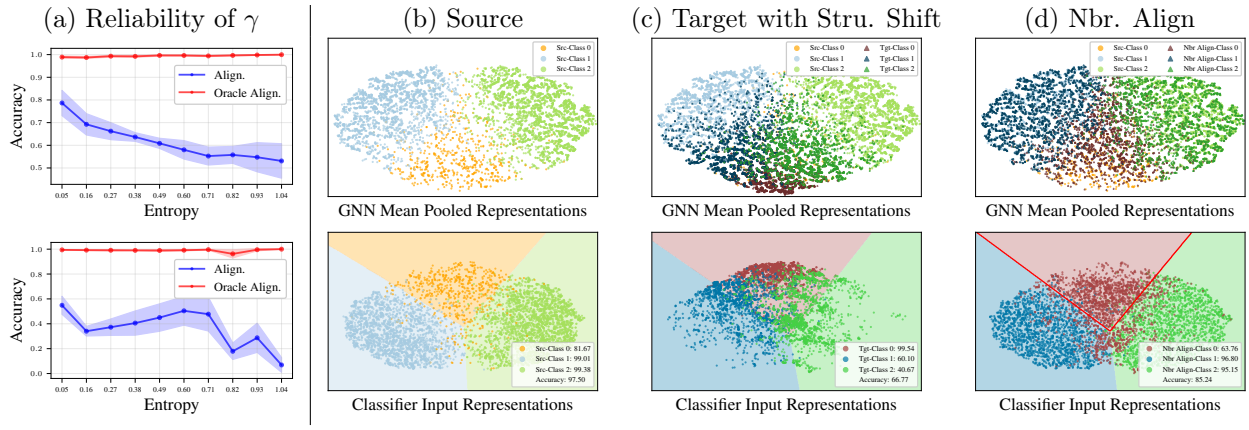

Figure 3: (a) Comparison of neighborhood alignment with $\boldsymbol{\gamma}$ from model prediction and Oracle on the CSBM graphs (Deshpande et al., 2018). The top (or bottom) subfigures represent the assignment under neighbor shift (or neighbor shift plus label shift, respectively). Nodes are grouped by the entropy of their soft pseudo labels and the y axis shows the accuracy after assigning $\boldsymbol{\gamma}$. Ideally, a correct assignment (red) would lead to near-perfect accuracy. However, the assignment based on pseudo labels is far from optimal (blue). Figure (b) to (d) present t-SNE visualizations of node representations during GTTA. The model is trained on the source domain in (b) (CSBM with label distribution $[0.1, 0.3, 0.6]$) and evaluated on the target domain in (c) (CSBM with label distribution $[0.3, 0.3, 0.3]$). Figure (d) illustrates the qualitative result of the neighborhood alignment. Despite well-aligned neighborhood representations, a mismatched decision boundary (red) leads to class 0 misclassifications. Node colors represent the ground-truth labels. The top subfigures show the output given by the GNN encoder, while the bottom subfigures show the classifier decision boundaries.

(the statistical change in connectivity patterns) to address test-time graph structure shift in a principled way, grounded in formal error analysis.

3 TEST ERROR ANALYSIS

In this section, we characterize the generalization error between the source and target graphs and explicitly attribute it to different types of shifts in graph data, which further motivate our TSA algorithm.

3.1 Distribution Shifts on Graphs

Distribution shifts on graphs have been formally studied in previous GDA works (Wu et al., 2020; You et al., 2023; Zhu et al., 2021; Wu et al., 2022; Liao et al., 2021; Zhu et al., 2023). Following their definitions, we categorize shifts in graphs into two types: feature shift and structure shift. For simplicity, our analysis is based on a data generation process: $\mathbf{X} \leftarrow \mathbf{Y} \rightarrow \mathbf{A}$, where the graph structure and node features are both conditioned on node labels.

Definition 3.1. (Feature Shift). Assume node features $x_u, u \in \mathcal{V}$ are i.i.d sampled given labels y_u, then we have $\mathbb{P}(\mathbf{X}|\mathbf{Y}) = \prod_{u\in\mathcal{V}} \mathbb{P}(X_u|Y_u)$. We then define feature shift as $\mathbb{P}^{\mathcal{S}}(X_u|Y_u) \neq \mathbb{P}^{\mathcal{T}}(X_u|Y_u)$.

Definition 3.2. (Structure Shift). As the graph structure involves the connecting pattern between nodes including their labels, we consider the joint distribution of the adjacency matrix and labels $\mathbb{P}(\mathbf{A}, \mathbf{Y})$, where *structure shift*, denoted by $\mathbb{P}^{\mathcal{S}}(\mathbf{A}, \mathbf{Y}) \neq \mathbb{P}^{\mathcal{T}}(\mathbf{A}, \mathbf{Y})$, can be decomposed into *conditional structure shift (CSS)* $\mathbb{P}^{\mathcal{S}}(\mathbf{A}|\mathbf{Y}) \neq \mathbb{P}^{\mathcal{T}}(\mathbf{A}|\mathbf{Y})$ and *label shift (LS)* $\mathbb{P}^{\mathcal{S}}(\mathbf{Y}) \neq \mathbb{P}^{\mathcal{T}}(\mathbf{Y})$. Here, CSS reflects the shift in neighborhood information (neighborhood shift) in Fig. 1.

3.2 Theoretical Analysis

Let $\varepsilon^{\mathcal{S}}(g \circ \phi)$ denote the error of a pretrained GNN on the source domain and $\varepsilon^{\mathcal{T}}(g \circ \phi)$ the error of the model when applied to a test graph in the target domain. Inspired by Tachet des Combes et al. (2020), we provide an upper bound on the error gap between the source and target domains, showing how a pretrained GNN (e.g., ERM) can be influenced. We denote the *balanced error rate* of a pretrained node predictor $\hat{Y}_u$ on the source domain as $\mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) \coloneqq \max_{j\in\mathcal{Y}} \mathbb{P}^{\mathcal{S}}(\hat{Y}_u \neq Y_u | Y_u = j)$.

Theorem 3.3 (Error Decomposition Theorem (Informal))**.** *Suppose $\mathcal{G}^{\mathcal{S}}$ and $\mathcal{G}^{\mathcal{T}}$, we can decouple both graphs into independent ego-networks (center nodes and 1-hop neighbors). For any classifier g with a GNN encoder ϕ in node classification tasks, we have the following upper bound on the error gap between source and target under feature shift and structure shift:*

$$
\begin{aligned}
&|\varepsilon^{\mathcal{S}}(g \circ \phi) - \varepsilon^{\mathcal{T}}(g \circ \phi)| \\
&\leq \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) \cdot \{\underbrace{(2\,TV(\mathbb{P}^{\mathcal{S}}(Y_u), \mathbb{P}^{\mathcal{T}}(Y_u))}_{\textit{Label Shift}} \\
&+ \underbrace{\mathbb{E}^{\mathcal{T}}_{Y_u}[\max_{k\in\mathcal{Y}} |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u, v \in \mathcal{N}_u)}|]}_{\textbf{\textit{Neighborhood shift}}}\} + \underbrace{\Delta_{CE}}_{\textit{Feature shift}}
\end{aligned}
$$

where $TV(\mathbb{P}^{\mathcal{S}}(Y_u), \mathbb{P}^{\mathcal{T}}(Y_u))$ is the total variation distance between the source and target label distributions. and Δ_{CE} is the error gap that exists if and only if feature shift exists.

Our bound highlights the impact of conditional structure shift, label shift, and feature shift on generalization to the target graph.

4 TEST-TIME STRUCTURAL ALIGNMENT

Motivated by our theoretical analysis, we propose Test-Time Structural Alignment (TSA) to address GTTA in a principled way. To address structure shift, TSA first conducts neighborhood alignment via weighted aggregation and then properly balances self and neighboring node information combination based on the SNRs of these representations. Lastly, as a generic approach for structure shift, TSA can be combined with non-graph TTA methods to refine the decision boundary.

4.1 Neighborhood Alignment

The conditional structure shift (CSS) alters the node label ratios in the aggregated neighborhood information, causing a shift in the GNN-encoded representations. Based on Theorem 3.3, we align the target neighborhood distribution with the source domain, leveraging the fact that the pre-trained model is optimized for the source distribution. Specifically, the neighborhood distribution determines the ratio of messages passed from a neighboring class j to the center class i. To adjust for distributional differences, this ratio can be rescaled by assigning weights to edges from class j to class i, effectively acting as an up/down-sampling mechanism during message aggregation. To ensure that message aggregation in the target domain aligns with the expected behavior in the source domain, TSA incorporates the following adjustment:

Definition 4.1. Let $\mathbb{P}^{\mathcal{T}}(Y_v = j | Y_u = i, v \in \mathcal{N}_u) > 0, \forall i, j \in \mathcal{Y}$, we have $\boldsymbol{\gamma} \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{Y}|}$ as:

$$[\boldsymbol{\gamma}]_{i,j} = \frac{\mathbb{P}^{\mathcal{S}}(Y_v = j | Y_u = i, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{T}}(Y_v = j | Y_u = i, v \in \mathcal{N}_u)}, \forall i, j \in \mathcal{Y} \quad (2)$$

To estimate $\boldsymbol{\gamma}$, we estimate $\mathbb{P}^{\mathcal{T}}(Y_v = j | Y_u = i, v \in \mathcal{N}_u)$ based on target pseudo labels. $\mathbb{P}^{\mathcal{S}}(Y_v = j | Y_u = i, v \in \mathcal{N}_u)$ can be computed directly from the source graph during model training and added as extra model parameters for the test time direct usage. During test time, we weight the message with a properly assigned $[\boldsymbol{\gamma}]_{i,j}$ computed based on the ratio of the two quantities. This idea aligns with the general principle of test-time alignment (Mirza et al., 2023; Niu et al., 2024; Kojima et al., 2022), but is novel in the aspect of dealing with the structure shift for graph data. Assigning edge weights to reduce structure shifts was also studied in training-time GDA by PairAlign (Liu et al., 2024). However, PairAlign is inapplicable in the GTTA setting as it assumes concurrent access to both the labeled source graph and the unlabeled target graph to train a model from scratch. This allows the model to learn and adjust to the weighted messages simultaneously, leading to better coordination between the model and the weights. In contrast, TSA addresses the more restrictive and practical inference-time GTTA problem. The model is pre-trained and frozen. To compensate for this inability to co-train, TSA needs a more reliable assignment strategy specified in the following.

Reliable Assignment of $\boldsymbol{\gamma}$. The ratio $\boldsymbol{\gamma}$ should be assigned to edge weights based on the label pairs of the central and neighboring nodes. In training-time GDA, this assignment is straightforward, as it relies on the ground-truth labels of the source graph. However, in GTTA, the node labeling information is unavailable. A naive approach is to use target pseudo labels, i.e., $[\boldsymbol{\gamma}]_{i,j}$ for the message from a node with pseudo label j to a node with pseudo label i, but this often results in significant mismatches. TSA addresses this by quantifying the uncertainty of target pseudo labels (Zhang et al., 2019b; Stadler et al., 2021; Hsu et al., 2022b,a). In particular, TSA assigns $[\boldsymbol{\gamma}]_{i,j}$ only to node pairs $v \to u$ where both of their soft target pseudo labels $\hat{y}$ have low entropy $H(\hat{y}) = -\sum_{i \in \mathcal{Y}} [\hat{y}]_i \ln([\hat{y}]_i) \leq \rho_1 \cdot \ln(|\mathcal{Y}|)$. Here, ρ_1 is a hyperparameter and $\ln(|\mathcal{Y}|)$ is the maximum entropy for a $|\mathcal{Y}|$ class prediction. In Fig. 3 (a), nodes with low-entropy soft predictions are more reliable, resulting in higher accuracy after the assignment of $\boldsymbol{\gamma}$.

4.2 SNR-Inspired Adjustment

Going beyond aligning the first-order statistics, we further optimize the signal-to-noise ratio (SNR) of node representations to enhance performance. The SNR is defined as the ratio of inter-class representation variance to intra-class representation variance (Yuan et al., 2019), formally expressed as $\text{SNR} \coloneqq \sigma^2_{inter} / \sigma^2_{intra}$ with

$$\sigma^2_{inter} = \sum_{i \in \mathcal{Y}} \frac{n_i \|\mu_i - \mu_*\|_2^2}{|\mathcal{Y}|}, \quad \sigma^2_{intra} = \sum_{i \in \mathcal{Y}} \sum_{h \in \mathcal{H}_i} \frac{\|h - \mu_i\|_2^2}{|\mathcal{Y}|}$$

where n_i is the number of samples in class i, μ_i is the centroid representation of class i, μ_* is the overall centroid of all nodes, and $\mathcal{H}_i$ represents the set of representations in class i. A higher inter-class variance σ^2_{inter} indicates informative and well-separated representations, while a higher σ^2_{intra} indicates a higher noise in the representations.

Optimizing SNR complements the neighborhood alignment approach. Even if neighborhood label distributions are perfectly aligned, variations in neighbor

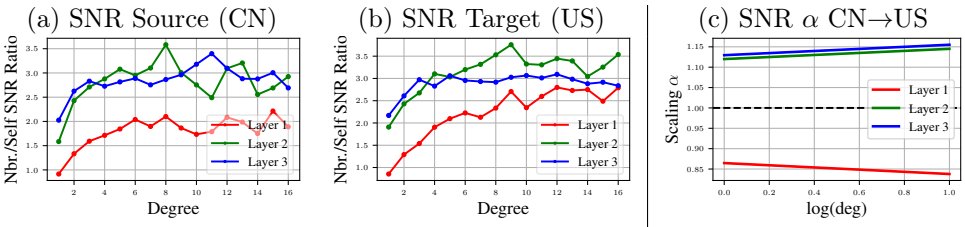

Figure 4: (a) and (b) compare the SNR of neighborhood-aggregated representations to that of self-node representations across node degrees and GNN layers, based on the ratio of the two quantities. A higher ratio indicates that the neighborhood-aggregated representations exhibit higher SNR. (c) Analysis of the SNR adjustment α across different layers and node degrees. Details are discussed in Sec. 5.2.

set sizes between the source and target graphs can impact the SNR of aggregated neighboring node representations. To compare the SNR of neighborhood-aggregated representations to that of self-node representations, we compute SNR ratio $= \frac{\text{Nbr. Aggr. SNR}}{\text{Self-Node SNR}}$ which exhibits different trends as a function of node degree (aka neighbor set size) and layer depth, as shown in Fig. 4 (a) and (b). In both figures, deeper layers and higher degree values generally lead to a higher SNR ratio, indicating aggregated representations are better denoised. However, the rate and shape of SNR ratio growth vary across source and target domains. Consequently, the combination of these aggregated representations with self-node representations in a typical GNN pipeline (Eq. 1) should be properly adjusted as well in test-time adaptation. To implement the SNR-inspired adjustment, we introduce a parameter to perform the weighted combination of self-node representations and neighborhood-aggregated representations at each layer, adapting to node degrees as follows:

Definition 4.2. Let $\tilde{d}_u = \left(\frac{\ln(d_u+1)}{\ln(\max_{v\in\mathcal{V}} d_v+1)}\right)$ denote log-normalized degree of node u and let $\text{MLP}^{(k)}$ and $b^{(k)}$ to be learnable parameters for adjusting k-th layer combination, we define the weights for combination $\alpha \in \mathbb{R}^L$ as:

$$[\alpha]_k = \sigma(\text{MLP}^{(k)}(\tilde{d}_u)) - 0.5 + b^{(k)},\ \forall k \in [L] \quad (3)$$

where σ is the sigmoid function. During initialization $[\alpha]_k$ is set to 1. Degree values are taken in the logarithmic domain to handle their often long-tailed distribution.

Together, $[\alpha]_k \cdot [\boldsymbol{\gamma}]_{i,j}$ is used to reweight the GNN messages, adjusting the influence from the node with a highly certain pseudo label j to the node with a highly certain pseudo label i.

Remark. Neighborhood alignment alone does not address potential shifts in SNR. This is because the alignment approach, inspired by Thm. 3.3, focuses solely on population-level performance, whereas SNR also incorporates the noise (variance) induced by empirical sampling from the population. Thus, these two aspects complement each other.

Algorithm 1 Test-Time Structural Alignment (TSA)

Input: A GNN ϕ and a classifier g pretrained on source graph $\mathcal{G}^{\mathcal{S}}$; Test-time target graph $\mathcal{G}^{\mathcal{T}}$; Source statistics $\mathbb{P}^{\mathcal{S}}(Y_v|Y_u, v \in \mathcal{N}_u) \in \mathbb{R}_+^{|\mathcal{Y}|\times|\mathcal{Y}|}$
Initialize $b^{(k)} \leftarrow 1$ and $\text{MLP}^{(k)}$ parameters $\leftarrow 0$ for the k-th layer.
Perform boundary refinement based on embeddings from $g \circ \phi(\mathcal{G}^{\mathcal{T}})$ and get soft pseudo-labels $\hat{y}$
Get $\mathcal{G}^{\mathcal{T}}_{new}$ by assigning edge weights:
  Compute γ using $\hat{y}$ via Eq. 2
  Assign γ only if node pairs $H(\hat{y}) \leq \rho_1 \cdot \ln(|\mathcal{Y}|)$
  Assign the parameterized α via Eq. 3
Update α's parameters $b^{(k)}$ and $\text{MLP}^{(k)}$ via Eq. 4
return $\hat{y}_{\text{final}}$ after another boundary refinement

4.3 Decision Boundary Refinement

In GTTA, even when CSS is fully addressed and high-SNR node representations are obtained, the decision boundary may still suffer from a mismatch due to label and feature shifts (Fig. 3 (d)).

A straightforward approach to refining the decision boundary at test time is to adjust the classifier's batch normalization parameters (TENT (Wang et al., 2021a)) or directly modify its output (T3A (Iwasawa & Matsuo, 2021) and LAME (Boudiaf et al., 2022)). We integrate these techniques into our framework for two folds: (1) their refined pseudo-labels provide a more reliable assignment of $\boldsymbol{\gamma}$ and can supervise the update of SNR adjustment. (2) Reciprocally, better alignment of neighborhood information can further refine the decision boundary.

Matcha (Bao et al., 2025) is a very recent work on GTTA that also integrates non-graph TTA methods with an approach for addressing structure shift. However, it considers only degree shift and homophily shift, where homophily shift is merely a special case of CSS in our context. Consequently, Matcha does not introduce the parameter $\boldsymbol{\gamma}$ and does not fully address the structure shift.

Table 1: Synthetic CSBM results (accuracy). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model. First six: imbalanced source training. Last two: balanced source training.

	Nbr. Shift		Nbr.+ SNR Shift		Struct. Shift (Imbal.→ Bal.)		Struct. Shift (Bal.→ Imbal.)	
ERM	82.70±4.45	61.11±10.81	77.03±3.99	61.93±6.44	50.41±4.88	39.12±4.71	68.27±5.00	61.39±1.89
GTrans	86.67±3.59	72.37±4.06	79.55±1.23	68.69±3.27	59.24±2.12	47.42±3.73	79.29±2.71	66.77±2.52
HomoTTT	85.08±2.87	65.79±9.07	78.84±3.42	65.31±5.69	52.34±4.69	40.68±5.24	70.11±5.08	62.19±2.51
SOGA	86.09±3.89	70.39±7.96	79.75±3.20	69.00±5.28	56.60±3.88	44.09±5.32	73.52±5.30	63.76±3.45
ActMAD	84.95±4.04	65.17±10.19	78.66±3.70	64.95±5.99	54.78±3.94	42.22±4.83	72.41±5.67	63.39±3.33
TENT	87.48±2.86	77.14±4.64	81.04±2.72	72.51±3.39	76.48±6.21	59.12±5.06	77.21±5.53	62.36±6.83
LAME	83.96±5.35	61.44±11.33	77.56±4.92	62.58±7.24	50.33±4.78	39.05±4.67	68.33±5.02	61.26±1.97
T3A	77.05±7.10	59.83±10.50	71.44±6.11	56.56±7.50	48.13±5.64	38.19±3.72	68.50±4.81	61.63±1.81
TSA-TENT	**88.78±1.37**	**80.51±2.39**	**83.19±1.46**	**76.41±1.25**	**88.68±4.99**	**66.25±7.75**	**81.20±8.18**	**70.15±2.30**
TSA-LAME	**88.96±1.66**	**80.02±5.44**	**83.51±0.55**	**79.56±1.82**	**65.09±2.34**	**52.90±6.11**	**82.20±5.17**	**63.15±2.58**
TSA-T3A	**89.96±1.33**	**81.08±2.73**	**84.23±1.24**	**76.89±2.02**	**65.59±2.57**	**52.34±7.19**	**82.55±5.06**	**64.88±3.30**

Table 2: MAG results (accuracy). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

Method	US→CN	US→DE	US→JP	US→RU	US→FR	CN→US	CN→DE	CN→JP	CN→RU	CN→FR
ERM	31.86±0.83	32.22±1.16	41.77±1.27	29.22±1.64	24.80±0.88	37.41±1.01	21.54±0.79	30.12±0.72	19.19±1.12	16.92±0.58
GTrans	31.77±0.91	32.14±1.05	41.55±1.23	29.74±1.57	25.03±0.85	36.17±0.89	21.07±0.93	29.08±0.82	19.68±1.14	16.78±0.62
HomoTTT	25.89±1.98	29.65±1.41	33.49±1.78	29.84±1.68	27.38±0.62	24.39±2.12	17.36±2.82	21.00±1.90	23.20±1.34	15.02±1.42
SOGA	21.54±2.52	25.48±0.93	36.24±3.31	29.07±4.14	24.34±0.91	38.95±3.35	25.75±1.14	38.25±1.42	29.86±1.71	23.50±0.67
ActMAD	31.38±0.74	31.93±1.18	41.53±1.22	29.68±1.73	25.09±0.93	36.04±0.97	20.59±0.78	29.01±0.67	19.37±1.07	16.30±0.53
TENT	26.72±1.33	32.73±0.63	40.80±0.91	32.26±0.95	28.32±0.66	27.21±0.88	15.66±0.86	24.62±0.48	21.37±0.73	13.84±0.62
LAME	35.75±0.85	33.64±1.70	44.97±1.15	30.19±1.64	24.17±1.84	40.08±1.13	22.64±1.14	33.00±1.48	17.80±0.55	17.43±0.93
T3A	41.47±1.15	45.36±2.15	50.34±0.94	46.41±0.84	40.26±1.69	46.50±1.26	38.62±1.03	46.10±0.38	43.11±0.76	29.95±1.36
TSA-TENT	**27.30±1.61**	**32.84±0.78**	**40.82±0.99**	**32.53±0.91**	**28.62±0.63**	**27.89±0.98**	**16.22±1.17**	**24.87±0.56**	**22.05±0.84**	**14.20±0.71**
TSA-LAME	**37.95±0.97**	**36.29±1.65**	**46.86±1.13**	**32.86±2.23**	**27.22±1.48**	**44.83±0.88**	**28.51±0.44**	**39.80±0.99**	**24.54±0.87**	**22.39±0.30**
TSA-T3A	**41.65±0.99**	**47.01±2.08**	**51.65±0.90**	**46.61±0.88**	**43.45±0.81**	**48.09±0.60**	**39.18±1.87**	**46.50±0.25**	**43.70±1.38**	**30.89±2.13**

TSA Overview. Note that $\boldsymbol{\gamma}$ is obtained from the initial pseudo labels. Only the parameters in $\text{MLP}^{(k)}$ and $b^{(k)}$ for α estimation in Eq. 3 need to be optimized at test time according to Eq. 4.

$$\mathcal{L}_{CE} = \frac{1}{|\mathcal{V}^{\mathcal{T}}|} \sum_{u \in \mathcal{V}^{\mathcal{T}}} \text{cross-entropy}(y'_u, \hat{y}_u) \tag{4}$$

where y'_u is the hard pseudo label refined by the procedure in Sec. 4.3 and $\hat{y}_u$ is the soft prediction from the original model. After updating α, TSA makes predictions on the newly weighted graphs and then further adapts the boundaries as described in Sec. 4.3. Our algorithm is summarized in Alg. 1.

5 EXPERIMENTS

We evaluate TSA on synthetic datasets and 5 real-world datasets. Additional results and discussions (e.g., hyperparameter sensitivity, ablation studies, computation efficiency) are provided in Appendix.

5.1 Datasets and Baselines

Synthetic Data. We use the CSBM model (Deshpande et al., 2018) to generate three-class datasets, focusing on structure shifts while keeping feature distributions unchanged. Specifically, we evaluate performance under three conditions: (1) CSS, (2) CSS plus SNR (induced by degree changes) shift, and (3) full structure shift (CSS + label shift) plus SNR shift. This experimental setup is motivated by Thm. 3.3 and the observations in Sec.4.2. For each shift scenario, we examine two levels of severity, with the left column in Table 1 corresponding to the smaller shift.

MAG (Liu et al., 2024) is a citation network extracted by OGB-MAG (Hu et al., 2020). Distribution shifts arise from partitioning papers into distinct graphs based on their countries of publication. The task is to classify the publication venue of each paper. Our model is pretrained on graphs from the US and China and subsequently adapted to graphs from other countries.

Pileup Mitigation (Liu et al., 2023) is a dataset curated for the data-denoising step in high-energy physics (Bertolini et al., 2014). Particles are generated by proton-proton collisions in LHC experiments. The task is to classify leading-collision (LC) neutral particles from other-collision (OC) particles. Particles are connected via kNN graphs if they are close in the $\eta - \phi$ space shown in Fig. 1. Distribution shifts arise from pile-up (PU) conditions (primarily structure shift), where PU level indicates the number of OC in the beam, and from the particle generation processes gg→qq and qq→gg (primarily feature shift).

Arxiv (Hu et al., 2020) is a citation network between all Computer Science (CS) arXiv papers. Distribution shifts originate from different time periods. Our model is pretrained on the earlier time span 1950 to 2007/ 2009/ 2011 and tested on later periods 2014-2016 and 2016-2018.

DBLP and ACM (Tang et al., 2008; Wu et al., 2020) are two citation networks. The model is trained on one network and adapted to the other to predict the research topic of a paper (node).

Baselines. We compare TSA with six baselines. For non-graph TTA methods, we include TENT (Wang et al., 2021a), LAME (Boudiaf et al., 2022), T3A (Iwasawa & Matsuo, 2021), and ActMAD (Mirza et al., 2023). GTrans (Jin et al., 2023), HomoTTT (Zhang

Table 3: Pileup results (f1-scores). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

Method	PU10→30	PU30→10	PU10→50	PU50→10	PU30→140	PU140→30	gg→qq	qq→gg
ERM	57.98±0.66	65.40±2.17	47.66±1.47	67.81±1.70	19.42±2.59	57.49±3.02	69.35±0.81	67.90±0.46
GTrans	57.37±1.49	63.66±2.43	48.13±2.11	65.74±1.95	28.41±4.01	57.65±1.97	69.17±0.82	67.37±0.56
HomoTTT	58.29±1.34	62.50±3.35	49.07±1.32	65.18±1.76	28.63±2.34	56.11±3.83	70.31±0.58	68.85±0.29
SOGA	60.13±0.75	69.71±0.92	51.68±0.83	67.47±1.42	37.16±1.15	56.59±3.86	70.83±0.66	68.84±0.97
ActMAD	58.54±0.81	64.59±2.38	49.39±1.35	67.12±1.74	20.88±2.66	50.97±5.14	69.20±0.79	68.04±0.46
TENT	58.25±1.85	58.73±4.80	48.27±2.58	58.41±7.89	30.59±3.07	0.04±0.08	68.72±0.48	68.00±0.60
LAME	57.77±0.66	66.29±2.42	47.31±1.58	68.24±1.61	11.96±3.08	58.81±2.00	69.25±0.52	67.94±0.66
T3A	58.74±1.09	70.02±2.33	49.61±1.03	70.85±0.64	30.33±3.66	54.79±1.52	69.45±0.97	68.45±0.61
TSA-TENT	**58.27±1.84**	**60.11±3.67**	**48.44±2.67**	**58.79±7.01**	**36.22±2.04**	**6.13±5.93**	**69.26±0.54**	**68.80±0.78**
TSA-LAME	**60.76±0.71**	**66.94±1.62**	**50.72±1.49**	**68.53±1.55**	**36.43±2.57**	**59.36±2.53**	**69.85±0.66**	**68.31±0.54**
TSA-T3A	**61.03±1.19**	**70.42±1.66**	**52.39±0.86**	**70.92±0.64**	**37.33±2.59**	**57.73±1.98**	**69.73±1.09**	**68.75±0.66**

Table 4: Arxiv and DBLP/ACM results (accuracy). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

	1950-2007		1950-2009		1950-2011		DBLP & ACM	
Method	2014-2016	2016-2018	2014-2016	2016-2018	2014-2016	2016-2018	D→A	A→D
ERM	41.04±0.50	40.48±1.45	44.80±1.96	42.38±3.64	53.40±1.14	51.68±1.76	28.95±4.50	52.45±6.81
GTrans	40.92±0.32	40.25±1.69	45.31±1.99	43.83±3.15	53.68±0.95	52.57±1.23	34.47±2.99	49.68±5.44
HomoTTT	36.33±0.95	32.83±1.15	41.22±1.15	38.86±1.42	48.18±1.38	44.61±2.90	31.25±3.96	52.90±6.25
SOGA	34.11±2.91	28.94±4.68	41.59±2.03	39.61±3.22	50.12±1.38	43.03±3.70	35.74±5.20	58.59±8.35
ActMAD	41.02±0.53	40.43±1.44	44.88±1.94	42.39±3.45	53.43±1.13	51.78±1.78	29.47±4.46	52.75±6.89
TENT	40.77±0.32	39.58±1.30	45.74±1.26	43.98±1.90	54.45±1.03	53.19±1.31	36.59±4.26	53.38±6.60
LAME	40.91±0.88	41.02±1.81	45.13±2.49	43.23±4.42	53.63±1.21	52.07±1.62	28.02±5.08	52.74±7.08
T3A	39.57±1.06	38.98±1.78	43.34±1.51	41.17±2.88	50.10±1.70	48.00±2.82	35.29±5.58	52.50±6.83
TSA-TENT	**40.92±0.34**	**39.78±1.21**	**46.68±1.24**	**45.08±1.71**	**54.78±0.80**	**53.61±1.24**	**37.06±3.93**	**54.06±6.63**
TSA-LAME	**41.34±0.83**	**41.23±1.70**	**45.42±2.45**	**43.71±4.99**	**54.05±1.05**	**52.76±1.47**	**28.66±5.27**	**52.90±7.30**
TSA-T3A	**40.03±1.03**	**39.70±1.49**	**44.09±1.53**	**42.17±2.91**	**50.96±1.68**	**49.21±2.67**	**35.30±5.57**	**52.55±6.92**

et al., 2024), SOGA (Mao et al., 2024), and Matcha (Bao et al., 2025) are direct comparisons in GTTA. Matcha is limited to GPRGNN due to its design. We present the results for GraphSAGE (Hamilton et al., 2017) in the main paper, while the results for GPRGNN (Chien et al., 2020) (including Matcha), GCN (Kipf & Welling, 2017) are provided in the Appendix.

5.2 Result Analysis

Comparison with Baselines. TSA achieves strong performance across all datasets and GNN backbones. In all cases, the best results are obtained by GTTA methods (underlined across Tables 1–4), highlighting the importance of addressing structure shift in graph data. In the synthetic dataset (Table 1), TSA outperforms GTTA baselines under plain CSS, demonstrating neighborhood alignment as an effective approach to handling CSS. The same table also showcases that dataset imbalance severely affects source training, which aligns with the worst-case error described in Theorem 3.3. In real-world datasets, TSA achieves an average improvement of 10% over existing GTTA baselines. GTrans relies on self-supervised learning to augment graphs. Although it shows improvements on CSBM, its performance is limited on real-world datasets (Tables 2–4). SOGA and HomoTTT depend on the homophily assumption, which does not always hold in real-world graphs (Luan et al., 2024), leading to unstable performance. In contrast, TSA makes no assumptions and consistently delivers improvements by structure shifts.

Different TSA-version. Overall, TSA-variants consistently yield performance gains of up to 21.25% over the corresponding non-graph TTA baselines (**bold** across Table 1- 4). In particular, the performance of the TSA variants differs in how the boundary refinement addresses the label shift. TSA-T3A is less sensitive to the dominance of the majority class, as it refines the boundary based on distances to prototypes, offering increased robustness (Zhang et al., 2022). Therefore, TSA-T3A works well on imbalanced datasets such as MAG (Table 2) and Pileup (Table 3). We observe TSA-TENT and TSA-LAME perform well on Arxiv and ACM/DBLP (Table 4), as they directly maximize the most probable class, with TSA-LAME introducing a regularization overhead.

Analysis of α. The learned α depends on node degrees and GNN layer depth. The trend of α in Fig. 4 (c) aligns with our expectations: earlier GNN layers require less attention to neighborhood-aggregated representations (small α), while deeper layers increasingly rely on them. Additionally, the plot reveals that nodes with higher degrees (and thus higher SNR ratios) place greater weight on neighborhood-aggregated messages. However, the overall effect of different GNN layers appears to be greater than that of varying node degrees. Lastly, the learned α exhibits a trend similar to the source SNR ratio shown in Fig. 4 (a), assigning lower weights in layer 1 and nearly identical weighting patterns in layer 2 and 3.

6 LIMITATIONS

While TSA effectively addresses distribution shifts in many settings, we identify specific limitations where its assumptions or mechanisms face challenges. We openly discuss these boundaries to clarify the method's scope to inspire future work.

Severe Feature Shifts. We observe that TSA's benefits can be marginal when feature shift significantly dominates over structure shift, as evidenced in the last two columns of the Pileup dataset. This is primarily because the neighborhood alignment strategy is not robust in such extreme scenarios. Furthermore, our current estimation of $\boldsymbol{\gamma}$ heavily depends on the accuracy of refined soft pseudo-labels. While confusion-matrix-based methods (Lipton et al., 2018; Azizzadenesheli et al., 2019; Alexandari et al., 2020) can provide more robust estimations, they often require additional adversarial training, which is incompatible with GTTA constraints. Exploring robust, GTTA-compatible approaches to estimating $\boldsymbol{\gamma}$ under severe feature shifts remains a crucial direction for future work.

Violation of the Markov Assumption. TSA calculates edge weights based on local pairwise conditional probabilities. If the graph exhibits strong long-range dependencies or subgraph correlations that extend beyond the 1-hop neighborhood, the pairwise term becomes an insufficient statistic. In such cases, the model fails to account for latent variables or hidden structures that co-determine the labels of interconnected nodes, leading to biased alignment weights.

Non-Homogeneity of Neighborhood Distributions. The current formulation estimates a single global $\boldsymbol{\gamma}$ for the entire graph. If the target graph is heavily non-homogeneous—meaning different clusters exhibit distinctly different structural shifts—this global expectation will be insufficient. A single global parameter fails to capture local variations in neighborhood structures, which results in improved alignment within dominant clusters while inadvertently degrading performance in minority or structurally distinct regions.

7 BROADER IMPACT

Our work studies structure shift and proposes TSA for graph test-time adaptation (GTTA). The goal of GTTA is to enhance model performance under distribution shifts with computation and privacy constraints. We do not foresee any direct negative societal impacts.

8 CONCLUSION

In this work, we theoretically analyze the impact of various graph shifts on the generalization gap in GTTA and highlight their empirical effects on test-time performance degradation. Based on these insights, we propose TSA, a method that addresses structure shift through neighborhood alignment, SNR-inspired adjustment, and boundary refinement. Extensive experiments validate the effectiveness and efficiency of TSA.

9 ACKNOWLEDGMENT

H. Hsu, S. Liu, and P. Li are partially supported by NSF awards PHY-2117997, IIS-2239565, IIS-2428777, and CCF-2402816; DOE award DE-FOA-0002785; JPMC faculty awards; and Microsoft Azure Research Credits for Generative AI.

References

Adachi, K., Yamaguchi, S., Kumagai, A., and Hamagami, T. Test-time adaptation for regression by subspace alignment. *International Conference on Learning Representations*, 2024.

Alexandari, A., Kundaje, A., and Shrikumar, A. Maximum likelihood with bias-corrected calibration is hard-to-beat at label shift adaptation. In *International Conference on Machine Learning*, 2020.

Azizzadenesheli, K., Liu, A., Yang, F., and Anandkumar, A. Regularized learning for domain adaptation under label shifts. *arXiv preprint arXiv:1903.09734*, 2019.

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

Bartler, A., Bühler, A., Wiewel, F., Döbler, M., and Yang, B. Mt3: Meta test-time training for self-supervised test-time adaption. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Bertolini, D., Harris, P., Low, M., and Tran, N. Pileup per particle identification. *Journal of High Energy Physics*, 2014.

Boudiaf, M., Mueller, R., Ben Ayed, I., and Bertinetto, L. Parameter-free online test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 2017.

Chen, G., Zhang, J., Xiao, X., and Li, Y. Graphtta: Test time adaptation on graph neural networks. *arXiv preprint arXiv:2208.09126*, 2022.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized pagerank graph neural network. *International Conference on Learning Representations*, 2020.

Clements, J. M., Xu, D., Yousefi, N., and Efimov, D. Sequential deep learning for credit risk monitoring with tabular financial data. *arXiv preprint arXiv:2012.15330*, 2020.

Deshpande, Y., Sen, S., Montanari, A., and Mossel, E. Contextual stochastic block models. *Advances in Neural Information Processing Systems*, 2018.

Dou, Y., Liu, Z., Sun, L., Deng, Y., Peng, H., and Yu, P. S. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM international conference on information & knowledge management*, 2020.

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 2015.

Eastwood, C., Mason, I., Williams, C. K., and Schölkopf, B. Source-free adaptation to measurement shift via bottom-up feature restoration. *International Conference on Learning Representations*, 2022.

Fey, M. and Lenssen, J. E. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Gong, T., Jeong, J., Kim, T., Kim, Y., Shin, J., and Lee, S.-J. Note: Robust continual test-time adaptation against temporal correlation. *Advances in Neural Information Processing Systems*, 2022.

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

Hakim, G. A. V., Osowiechi, D., Noori, M., Cheraghalikhani, M., Bahri, A., Ben Ayed, I., and Desrosiers, C. Clust3: Information invariant test-time training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6136–6145, 2023.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 2017.

Hamilton, W. L. *Graph representation learning*. Morgan & Claypool Publishers, 2020.

Highfield, R. Large hadron collider: Thirteen ways to change the world. *The Daily Telegraph. London. Retrieved*, 2008.

Hsu, H. H.-H., Shen, Y., and Cremers, D. A graph is more than its nodes: Towards structured uncertainty-aware learning on graphs. *arXiv preprint arXiv:2210.15575*, 2022a.

Hsu, H. H.-H., Shen, Y., Tomani, C., and Cremers, D. What makes graph neural networks miscalibrated? *Advances in Neural Information Processing Systems*, 2022b.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 2020.

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

Jang, M., Chung, S.-Y., and Chung, H. W. Test-time adaptation via self-training with nearest neighbor information. *International Conference on Learning Representations*, 2022.

Ji, Y., Zhang, L., Wu, J., Wu, B., Li, L., Huang, L.-K., Xu, T., Rong, Y., Ren, J., Xue, D., et al. Drugood: Out-of-distribution dataset curator and benchmark for ai-aided drug discovery–a focus on affinity prediction problems with noise annotations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Jin, W., Zhao, T., Ding, J., Liu, Y., Tang, J., and Shah, N. Empowering graph representation learning with test-time graph transformation. *International Conference on Learning Representations*, 2023.

Ju, M., Zhao, T., Yu, W., Shah, N., and Ye, Y. Graphpatcher: mitigating degree bias for graph neural networks via test-time augmentation. *Advances in Neural Information Processing Systems*, 2024.

Kang, J., Kim, N., Donghyeon, K., Ok, J., and Kwak, S. Leveraging proxy of training data for test-time adaptation. In *International Conference on Machine Learning (ICML)*, July 2023.

Kingma, D. P. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

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

Kojima, T., Matsuo, Y., and Iwasawa, Y. Robustifying vision transformer without retraining from scratch by test-time class-conditional feature alignment. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pp. 1009–1016, 7 2022. URL `https://doi.org/10.24963/ijcai.2022/141`.

Komiske, P. T., Metodiev, E. M., Nachman, B., and Schwartz, M. D. Pileup mitigation with machine learning (pumml). *Journal of High Energy Physics*, 2017.

Li, T., Liu, S., Feng, Y., Paspalaki, G., Tran, N. V., Liu, M., and Li, P. Semi-supervised graph neural networks for pileup noise removal. *The European Physical Journal C*, 2023.

Liang, J., He, R., and Tan, T. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, 2024.

Liao, P., Zhao, H., Xu, K., Jaakkola, T., Gordon, G. J., Jegelka, S., and Salakhutdinov, R. Information obfuscation of graph neural networks. In *International conference on machine learning*, 2021.

Lim, H., Kim, B., Choo, J., and Choi, S. Ttn: A domain-shift aware batch normalization in test-time adaptation. *arXiv preprint arXiv:2302.05155*, 2023.

Lipton, Z., Wang, Y.-X., and Smola, A. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, 2018.

Liu, S., Li, T., Feng, Y., Tran, N., Zhao, H., Qiu, Q., and Li, P. Structural re-weighting improves graph domain adaptation. In *International Conference on Machine Learning*, pp. 21778–21793. PMLR, 2023.

Liu, S., Zou, D., Zhao, H., and Li, P. Pairwise alignment improves graph domain adaptation. *International Conference on Machine Learning*, 2024.

Liu, Y., Kothari, P., Van Delft, B., Bellot-Gurlet, B., Mordan, T., and Alahi, A. Ttt++: When does self-supervised test-time training fail or thrive? *Advances in Neural Information Processing Systems*, 2021.

Luan, S., Hua, C., Lu, Q., Ma, L., Wu, L., Wang, X., Xu, M., Chang, X.-W., Precup, D., Ying, R., et al. The heterophilic graph learning handbook: Benchmarks, models, theoretical analysis, applications and challenges. *arXiv preprint arXiv:2407.09618*, 2024.

Mao, H., Du, L., Zheng, Y., Fu, Q., Li, Z., Chen, X., Han, S., and Zhang, D. Source free graph unsupervised domain adaptation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2024.

Miao, S., Lu, Z., Liu, M., Duarte, J., and Li, P. Locality-sensitive hashing-based efficient point transformer with applications in high-energy physics. *International Conference on Machine Learning*, 2024.

Mirza, M. J., Soneira, P. J., Lin, W., Kozinski, M., Possegger, H., and Bischof, H. Actmad: Activation matching to align distributions for test-time-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Niu, S., Wu, J., Zhang, Y., Wen, Z., Chen, Y., Zhao, P., and Tan, M. Towards stable test-time adaptation in dynamic wild world. *International Conference on Learning Representations*, 2023.

Niu, S., Miao, C., Chen, G., Wu, P., and Zhao, P. Test-time model adaptation with only forward passes. In *The International Conference on Machine Learning*, 2024.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

Pei, H., Wei, B., Chang, K. C.-C., Lei, Y., and Yang, B. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

Rong, Y., Huang, W., Xu, T., and Huang, J. Dropedge: Towards deep graph convolutional networks on node classification. *International Conference on Learning Representations*, 2020.

Shlomi, J., Battaglia, P., and Vlimant, J.-R. Graph neural networks in particle physics. *Machine Learning: Science and Technology*, 2020.

Stadler, M., Charpentier, B., Geisler, S., Zügner, D., and Günnemann, S. Graph posterior network: Bayesian predictive uncertainty for node classification. *Advances in Neural Information Processing Systems*, 2021.

Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackermann, Z., et al. A deep learning approach to antibiotic discovery. *Cell*, 2020.

Sun, H., Xu, L., Jin, S., Luo, P., Qian, C., and Liu, W. Program: Prototype graph model based pseudo-label learning for test-time adaptation. *International Conference on Learning Representations*, 2024.

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 *International Conference on Machine Learning*, 2020.

Tachet des Combes, R., Zhao, H., Wang, Y.-X., and Gordon, G. J. Domain adaptation with conditional distribution matching and generalized label shift. *Advances in Neural Information Processing Systems*, 2020.

Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., and Su, Z. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008.

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

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. Deep graph infomax. *International Conference on Learning Representations*, 2019.

Wang, D., Lin, J., Cui, P., Jia, Q., Wang, Z., Fang, Y., Yu, Q., Zhou, J., Yang, S., and Qi, Y. A semi-

supervised graph attentive network for financial fraud detection. In *IEEE International Conference on Data Mining*, 2019.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. *International Conference on Learning Representations*, 2021a.

Wang, J., Zhang, S., Xiao, Y., and Song, R. A review on graph neural network methods in financial applications. *arXiv preprint arXiv:2111.15367*, 2021b.

Wang, Y., Feng, B., Li, G., Li, S., Deng, L., Xie, Y., and Ding, Y. {GNNAdvisor}: An adaptive and efficient runtime system for {GNN} acceleration on {GPUs}. In *15th USENIX symposium on operating systems design and implementation (OSDI 21)*, 2021c.

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.

Wu, C., Wu, F., Cao, Y., Huang, Y., and Xie, X. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925*, 2021.

Wu, M., Pan, S., Zhou, C., Chang, X., and Zhu, X. Unsupervised domain adaptive graph convolutional networks. In *Proceedings of The Web Conference 2020*, 2020.

Wu, Q., Zhang, H., Yan, J., and Wipf, D. Handling distribution shifts on graphs: An invariance perspective. In *International Conference on Learning Representations*, 2022.

You, Y., Chen, T., Wang, Z., and Shen, Y. Graph domain adaptation via theory-grounded spectral regularization. In *International Conference on Learning Representations*, 2023.

Yuan, T., Deng, W., Tang, J., Tang, Y., and Chen, B. Signal-to-noise ratio: A robust distance metric for deep metric learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4815–4824, 2019.

Zhang, J., Wang, Y., Yang, X., and Zhu, E. A fully test-time training framework for semi-supervised node classification on out-of-distribution graphs. *ACM Transactions on Knowledge Discovery from Data*, 18 (7):1–19, 2024.

Zhang, S., Yao, L., Sun, A., and Tay, Y. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, 2019a.

Zhang, Y., Pal, S., Coates, M., and Ustebay, D. Bayesian graph convolutional neural networks for semi-supervised classification. In *Proceedings of the AAAI conference on artificial intelligence*, 2019b.

Zhang, Z., Chen, W., Cheng, H., Li, Z., Li, S., Lin, L., and Li, G. Divide and contrast: Source-free domain adaptation via adaptive contrastive learning. *Advances in Neural Information Processing Systems*, 2022.

Zhao, B., Chen, C., and Xia, S.-T. Delta: degradation-free fully test-time adaptation. *International Conference on Learning Representations*, 2023.

Zhou, Z., Guo, L.-Z., Jia, L.-H., Zhang, D., and Li, Y.-F. Ods: Test-time adaptation in the presence of open-world data shift. In *International Conference on Machine Learning*, 2023.

Zhu, Q., Ponomareva, N., Han, J., and Perozzi, B. Shift-robust gnns: Overcoming the limitations of localized graph training data. *Advances in Neural Information Processing Systems*, 2021.

Zhu, Q., Jiao, Y., Ponomareva, N., Han, J., and Perozzi, B. Explaining and adapting graph conditional shift. *arXiv preprint arXiv:2306.03256*, 2023.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] Please see Section 4.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] We provide a runtime complexity analysis in Appendix.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes] We provide anonymized source code in the supplementary materials.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes] Please see Theorem 3.3.
   (b) Complete proofs of all theoretical results. [Yes] Please see Appendix.
   (c) Clear explanations of any assumptions. [Yes] Please see Theorem 3.3.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] We provide anonymized source code in the supplementary materials.

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes] Please see Appendix Experimental Setup.

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes] Please see Appendix Experimental Setup and Section 5.

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] Please see Appendix Experimental Setup.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes] We include citations for existing assets.

   (b) The license information of the assets, if applicable. [Yes]

   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes] We provide anonymized source code in the supplementary materials.

   (d) Information about consent from data providers/curators. [Yes]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [Not Applicable]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Appendix for Structural Alignment Improves Graph Test-Time Adaptation

A OMITTED PROOFS

A.1 Proof of Theorem 3.3

Lemma A.1. *Assume $\mathbb{P}^{\mathcal{M}} := \alpha_j \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) + \beta_j \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)$ to be a mixture conditional probability of source and target domain. The mixture weight is given by α_j and β_j where $\forall \alpha_j, \beta_j \geq 0$ and $\alpha_j + \beta + j = 1$, the following upper bound holds:*

$$
\begin{aligned}
&|\gamma^{\mathcal{S}}_{u_j} \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \gamma^{\mathcal{T}}_{u_j} \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j| \\
&\leq |\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j}| \cdot \mathbb{P}^{\mathcal{M}} + (\gamma^{\mathcal{S}}_{u_j}\beta_j + \gamma^{\mathcal{T}}_{u_j}\alpha_j)|\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)|
\end{aligned}
$$

Proof. To simplify the derivation we first abuse the notation by letting $\mathbb{P}^{\mathcal{S}}$ denote $\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j, W_u = k)$ and $\mathbb{P}^{\mathcal{T}}$ denote $\mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j, W_u = k)$.

$$
\begin{aligned}
&|\gamma^{\mathcal{S}}_{u_j} \mathbb{P}^{\mathcal{S}} - \gamma^{\mathcal{T}}_{u_j} \mathbb{P}^{\mathcal{T}}| - |\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j}| \cdot \mathbb{P}^{\mathcal{M}} \\
&\leq |\gamma^{\mathcal{S}}_{u_j} \mathbb{P}^{\mathcal{S}} - \gamma^{\mathcal{T}}_{u_j} \mathbb{P}^{\mathcal{T}} - (\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j})\mathbb{P}^{\mathcal{M}}| \\
&= |\gamma^{\mathcal{S}}_{u_j}(\mathbb{P}^{\mathcal{S}} - \mathbb{P}^{\mathcal{M}}) - \gamma^{\mathcal{T}}_{u_j}(\mathbb{P}^{\mathcal{T}} - \mathbb{P}^{\mathcal{M}})| \\
&\leq \gamma^{\mathcal{S}}_{u_j}|\mathbb{P}^{\mathcal{S}} - \mathbb{P}^{\mathcal{M}}| + \gamma^{\mathcal{T}}_{u_j}|\mathbb{P}^{\mathcal{T}} - \mathbb{P}^{\mathcal{M}}|
\end{aligned}
$$

In order to simplify the first term we substitute the definition of $\mathbb{P}^{\mathcal{M}} := \alpha_j \mathbb{P}^{\mathcal{S}} + \beta_j \mathbb{P}^{\mathcal{T}}$:

$$
|\mathbb{P}^{\mathcal{S}} - \mathbb{P}^{\mathcal{M}}| = |\mathbb{P}^{\mathcal{S}} - \alpha_j \mathbb{P}^{\mathcal{S}} - \beta_j \mathbb{P}^{\mathcal{T}}| = |\beta_j \mathbb{P}^{\mathcal{S}} - \beta_j \mathbb{P}^{\mathcal{T}}|
$$

Similarly for the second term:

$$
|\mathbb{P}^{\mathcal{T}} - \mathbb{P}^{\mathcal{M}}| = |\mathbb{P}^{\mathcal{T}} - \alpha_j \mathbb{P}^{\mathcal{S}} - \beta_j \mathbb{P}^{\mathcal{T}}| = |\alpha_j \mathbb{P}^{\mathcal{T}} - \alpha_j \mathbb{P}^{\mathcal{S}}|
$$

Using the two identities, we can proceed with the derivation:

$$
\begin{aligned}
&|\gamma^{\mathcal{S}}_{u_j} \mathbb{P}^{\mathcal{S}} - \gamma^{\mathcal{T}}_{u_j} \mathbb{P}^{\mathcal{T}}| - |\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j}| \cdot \mathbb{P}^{\mathcal{M}} \\
&\leq \gamma^{\mathcal{S}}_{u_j}\beta_j|\mathbb{P}^{\mathcal{S}} - \mathbb{P}^{\mathcal{T}}| + \gamma^{\mathcal{T}}_{u_j}\alpha_j|\mathbb{P}^{\mathcal{T}} - \mathbb{P}^{\mathcal{S}}| \\
&= (\gamma^{\mathcal{S}}_{u_j}\beta_j + \gamma^{\mathcal{T}}_{u_j}\alpha_j)|\mathbb{P}^{\mathcal{S}} - \mathbb{P}^{\mathcal{T}}|
\end{aligned}
$$

□

Theorem 3.3 (Error Decomposition Theorem (Informal))**.** *Suppose $\mathcal{G}^{\mathcal{S}}$ and $\mathcal{G}^{\mathcal{T}}$, we can decouple both graphs into independent ego-networks (center nodes and 1-hop neighbors). For any classifier g with a GNN encoder ϕ in node classification tasks, we have the following upper bound on the error gap between source and target under feature shift and structure shift:*

$$
\begin{aligned}
&|\varepsilon^{\mathcal{S}}(g \circ \phi) - \varepsilon^{\mathcal{T}}(g \circ \phi)| \\
&\leq \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) \cdot \{\underbrace{(2TV(\mathbb{P}^{\mathcal{S}}(Y_u), \mathbb{P}^{\mathcal{T}}(Y_u))}_{\textit{Label Shift}} \\
&+ \underbrace{\mathbb{E}^{\mathcal{T}}_{Y_u}[\max_{k \in \mathcal{Y}} |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u, v \in \mathcal{N}_u)}|]}_{\textbf{\textit{Neighborhood shift}}}\} + \underbrace{\Delta_{CE}}_{\textit{Feature shift}}
\end{aligned}
$$

where $TV(\mathbb{P}^{\mathcal{S}}(Y_u), \mathbb{P}^{\mathcal{T}}(Y_u))$ is the total variation distance between the source and target label distributions. and Δ_{CE} is the error gap that exists if and only if feature shift exists.

Proof. We start by converting the average error rate into individual error probabilities. For sufficiently large $|\mathcal{V}|$ we have $|\mathcal{V}| \approx |\mathcal{V}| - 1$. Given $|\mathcal{V}^{\mathcal{S}}| \to \infty$ and $|\mathcal{V}^{\mathcal{T}}| \to \infty$, we can have $|\mathcal{V}| \approx |\mathcal{V}^{\mathcal{S}}| \approx |\mathcal{V}^{\mathcal{T}}|$. By applying the triangle inequality and assuming that the graphs are sufficiently large, we obtain the following inequality:

$$
\begin{aligned}
&|\varepsilon^{\mathcal{S}}(h \circ g) - \varepsilon^{\mathcal{T}}(h \circ g)| \\
=&|\mathbb{P}^{\mathcal{S}}(g(\phi(X_u, \mathbf{A})) \neq Y_u) - \mathbb{P}^{\mathcal{T}}(g(\phi(X_u, \mathbf{A})) \neq Y_u)| \\
=&|\mathbb{P}^{\mathcal{S}}(\hat{Y}_u \neq Y_u) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u \neq Y_u)|
\end{aligned}
$$

To simplify the notation, define $\gamma^{\mathcal{U}}_{u_j} := \mathbb{P}^{\mathcal{U}}(Y_u = j)$, $\pi^{\mathcal{U}}_{v_k|u_j} := \mathbb{P}^{\mathcal{U}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)$, and $\omega^{\mathcal{U}}_{u_i|u_j,v_k} := \mathbb{P}^{\mathcal{U}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)$ for $\mathcal{U}) \in \{\mathcal{S}, \mathcal{T}\}$. Using the the law of total probability and assuming that the label prediction depends on the distribution of neighborhood labels, we can derive the following identity:

$$
\begin{aligned}
\mathbb{P}^{\mathcal{U}}(\hat{Y}_u \neq Y_u) &= \sum_{i \neq j} \mathbb{P}^{\mathcal{U}}(\hat{Y}_u = i, Y_u = j) = \sum_{i \neq j} \mathbb{P}^{\mathcal{U}}(Y_u = j)\mathbb{P}^{\mathcal{U}}(\hat{Y}_u = i|Y_u = j) \\
&= \sum_{i \neq j} \mathbb{P}^{\mathcal{U}}(Y_u = j)\Big(\sum_{k \in \mathcal{Y}} \mathbb{P}^{\mathcal{U}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)\mathbb{P}^{\mathcal{U}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)\Big) \\
&= \sum_{i \neq j} \gamma^{\mathcal{U}}_{u_j} \sum_{k \in \mathcal{Y}} \pi^{\mathcal{U}}_{v_k|u_j} \omega^{\mathcal{U}}_{u_i|u_j,v_k}
\end{aligned}
$$

Most GNNs marginalize the effect of degree magnitude, meaning that only the expectation of the neighboring nodes' label distribution and the label distribution of the central node itself influence the prediction. We use Lemma A.1 to bound the terms above. Since $\forall j \in \mathcal{Y}, \gamma^{\mathcal{S}}_{u_j}$ and $\gamma^{\mathcal{T}}_{u_j} \in [0, 1]$ and we have $\alpha_j + \beta_j = 1$, we obtain:

$$
\begin{aligned}
&|\varepsilon^{\mathcal{S}}(h \circ g) - \varepsilon^{\mathcal{T}}(h \circ g)| \\
&\leq \sum_{i \neq j} \gamma^{\mathcal{S}}_{u_j} \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j) \\
&\leq \sum_{i \neq j} |\gamma^{\mathcal{S}}_{u_j} \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \gamma^{\mathcal{T}}_{u_j} \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)| && \text{(a)} \\
&\leq \sum_{i \neq j} \Big(|\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j}| \cdot \mathbb{P}^{\mathcal{M}} + (\gamma^{\mathcal{S}}_{u_j}\beta_j + \gamma^{\mathcal{T}}_{u_j}\alpha_j)|\mathbb{P}^{\mathcal{S}} - \mathbb{P}^{\mathcal{T}}|\Big) && \text{(b)} \\
&= \sum_{i \neq j} \Big(|\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j}| \cdot \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) + \gamma^{\mathcal{T}}_{u_j}|\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)|\Big) && \text{(c)} \\
&= \sum_{i \neq j} |\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j}| \cdot \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) + \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j}|\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)| \\
&\leq \left(\sum_{j \in \mathcal{Y}} |\gamma^{\mathcal{S}}_{u_j} - \gamma^{\mathcal{T}}_{u_j}|\right) \cdot \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) + \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j}|\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)| && \text{(d)} \\
&= 2TV\big(\mathbb{P}^{\mathcal{S}}(Y_u = j), \mathbb{P}^{\mathcal{T}}(Y_u = j)\big) \cdot \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) + \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j}|\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)| && \text{(e)}
\end{aligned}
$$

By applying the triangle inequality, (a) is obtained. Following Lemma A.1 we have (b). (c) is obtained by choosing $\alpha_j = 1$ and $\beta_j = 0, \forall j \in \mathcal{Y}$. (d) is derived by applying Holder's inequality. (e) is based on the definition of total variation distance where $TV(\mathbb{P}^{\mathcal{S}}(Y_u = j), \mathbb{P}^{\mathcal{T}}(Y_u = j)) = \frac{1}{2}\sum_{j \in \mathcal{Y}} |\mathbb{P}^{\mathcal{S}}(Y_u = j) - \mathbb{P}^{\mathcal{T}}(Y_u = j)|$ is the total variation distance between the label distribution of source and target.

Similarly, the second term can be decomposed by Lemma A.1 while setting $\omega^{\mathcal{M}}_{u_i|u_j,v_k} := \alpha_j \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u) + \beta_j \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)$, $\omega^{\mathcal{S}}_{u_i|u_j,v_k} := \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)$, and $\omega^{\mathcal{T}}_{u_i|u_j,v_k} := \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)$. The decomposition is instead with respect to $\pi^{\mathcal{U}}_{v_k|u_j} := \mathbb{P}^{\mathcal{U}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)$.

$$
\begin{aligned}
&\sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} |\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j)| \\
&= \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} |\sum_{k \in \mathcal{Y}} \mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u) \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j, \{Y_v : v \in \mathcal{N}_u\}) \\
&\qquad - \sum_{k \in \mathcal{Y}} \mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u) \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j, \{Y_v : v \in \mathcal{N}_u\})| \\
&\leq \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} |\pi^{\mathcal{S}}_{v_k|u_j} \omega^{\mathcal{S}}_{u_i|u_j,v_k} - \pi^{\mathcal{T}}_{v_k|u_j} \omega^{\mathcal{T}}_{u_i|u_j,v_k}| \\
&\leq \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} \Big(|\pi^{\mathcal{S}}_{v_k|u_j} - \pi^{\mathcal{T}}_{v_k|u_j}| \cdot \omega^{\mathcal{M}}_{u_i|u_j,v_k} + (\pi^{\mathcal{S}}_{v_k|u_j} \beta_j + \pi^{\mathcal{T}}_{v_k|u_j} \alpha_j) |\omega^{\mathcal{S}}_{u_i|u_j,v_k} - \omega^{\mathcal{T}}_{u_i|u_j,v_k}| \Big) && \text{(a)} \\
&= \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} |\pi^{\mathcal{S}}_{v_k|u_j} - \pi^{\mathcal{T}}_{v_k|u_j}| \cdot \omega^{\mathcal{S}}_{u_i|u_j,v_k} + \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} \pi^{\mathcal{T}}_{v_k|u_j} \; |\omega^{\mathcal{S}}_{u_i|u_j,v_k} - \omega^{\mathcal{T}}_{u_i|u_j,v_k}| && \text{(b)} \\
&= \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} \pi^{\mathcal{S}}_{v_k|u_j} |1 - \frac{\pi^{\mathcal{T}}_{v_k|u_j}}{\pi^{\mathcal{S}}_{v_k|u_j}}| \cdot \omega^{\mathcal{S}}_{u_i|u_j,v_k} + \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} \pi^{\mathcal{T}}_{v_k|u_j} \; |\omega^{\mathcal{S}}_{u_i|u_j,v_k} - \omega^{\mathcal{T}}_{u_i|u_j,v_k}| \\
&\leq \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \left(\max |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}| \right) \cdot \mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) + \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} \pi^{\mathcal{T}}_{v_k|u_j} \; |\omega^{\mathcal{S}}_{u_i|u_j,v_k} - \omega^{\mathcal{T}}_{u_i|u_j,v_k}| && \text{(c)} \\
&\leq \mathbb{E}^{\mathcal{T}} \left[\left(\max |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}| \right) \right] \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) + \sum_{i \neq j} \gamma^{\mathcal{T}}_{u_j} \sum_{k \in \mathcal{Y}} \pi^{\mathcal{T}}_{v_k|u_j} \; |\omega^{\mathcal{S}}_{u_i|u_j,v_k} - \omega^{\mathcal{T}}_{u_i|u_j,v_k}| \\
&\leq \mathbb{E}^{\mathcal{T}}_{Y_u} \left[\left(\max |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}| \right) \right] \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) + \max_{i \neq j} \mathbb{E}^{\mathcal{T}} \left[|\omega^{\mathcal{S}}_{u_i|u_j,v_k} - \omega^{\mathcal{T}}_{u_i|u_j,v_k}| \right] \\
&\leq \mathbb{E}^{\mathcal{T}}_{Y_u} \left[\left(\max |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}| \right) \right] \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) \\
&\qquad + \max_{i \neq j} \mathbb{E}^{\mathcal{T}}_{Y_v} [|\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u) - \mathbb{P}^{\mathcal{T}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)|] \\
&\leq \mathbb{E}^{\mathcal{T}}_{Y_u} \left[\left(\max |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)}| \right) \right] \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) + \Delta_{CE} && \text{(d)}
\end{aligned}
$$

(a) comes from Lemma A.1. (b) comes from setting $\alpha_j = 1$ and $\beta_j = 0, \forall j \in \mathcal{Y}$ (c) The inequality comes from Holder's inequality, where we take the maximum-norm for k and take the L_1 norm to marginalize Y_v: $\mathbb{P}^{\mathcal{S}}(\hat{Y}_u = i|Y_u = j) = \sum_{k \in \mathcal{Y}} \mathbb{P}^{\mathcal{U}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u) \mathbb{P}^{\mathcal{U}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)$ (d) The last term in the previous line corresponds to the conditional error gap in (Tachet des Combes et al., 2020) but in a standard GNN's message passing fashion.

The term $\mathbb{P}^{\mathcal{U}}(\hat{Y}_u = i|Y_u = j, Y_v = k, v \in \mathcal{N}_u)$ can be interpreted as the probability of error prediction using a GNN encoder. By definition, Y_v represents a random variable of the neighborhood label ratio and Y_u denotes a random variable of self class label, their realization corresponds to the expectation of the neighboring nodes' label distribution and the label distribution of the central node itself. As a result, Δ_{CE} only exists if and only if feature shift exists. Putting everything together, we have

$$
\begin{aligned}
&|\varepsilon^{\mathcal{S}}(g \circ \phi) - \varepsilon^{\mathcal{T}}(g \circ \phi)| \\
&\leq \mathrm{BER}^{\mathcal{S}}(\hat{Y}_u \parallel Y_u) \cdot \{\underbrace{(2TV(\mathbb{P}^{\mathcal{S}}(Y_u), \mathbb{P}^{\mathcal{T}}(Y_u))}_{\text{Label Shift}} + \underbrace{\mathbb{E}^{\mathcal{T}}_{Y_u}[\max_{k \in \mathcal{Y}} |1 - \frac{\mathbb{P}^{\mathcal{T}}(Y_v = k | Y_u, v \in \mathcal{N}_u)}{\mathbb{P}^{\mathcal{S}}(Y_v = k | Y_u, v \in \mathcal{N}_u)}|]}_{\textbf{Conditional Structure Shift}}\} + \underbrace{\Delta_{CE}}_{\text{Feature Shift}}
\end{aligned}
$$

□

A.2 Finite-Sample Guarantees of Theorem 3.3

To extend the theoretical foundation of Theorem 3.3, we relax the asymptotic assumption ($|\mathcal{V}| \to \infty$) and provide a finite-sample guarantee. We demonstrate that for finite graphs consisting of independent ego-networks, the empirical error gap is bounded by the theoretical shift terms plus a statistical variance term that decays with sample size.

Let $\epsilon^{\mathcal{S}}$ and $\hat{\epsilon}^{\mathcal{S}}$ denote the expected and empirical risk on the source, respectively (and similarly for target $\mathcal{T}$). By decomposing the error gap via triangle inequality, we have:

$$
|\hat{\epsilon}^{\mathcal{S}} - \hat{\epsilon}^{\mathcal{T}}| \leq \underbrace{|\hat{\epsilon}^{\mathcal{S}} - \epsilon^{\mathcal{S}}|}_{\text{Source Est. Error}} + \underbrace{|\epsilon^{\mathcal{S}} - \epsilon^{\mathcal{T}}|}_{\text{Distribution Shift (Thm 3.3)}} + \underbrace{|\epsilon^{\mathcal{T}} - \hat{\epsilon}^{\mathcal{T}}|}_{\text{Target Est. Error}}
$$

The middle term, $|\epsilon^{\mathcal{S}} - \epsilon^{\mathcal{T}}|$, is strictly bounded by the Label, Neighborhood, and Feature shift components derived in Theorem 3.3. We bound the estimation error terms using McDiarmid's Inequality. Following the formulation in Theorem 3.3, we assume the graph $\mathcal{G}$ consists of M independent samples (ego-networks) $\{Z_1, Z_2, ..., Z_M\}$, where $M = |\mathcal{V}|$. Let the empirical error be a function $f : \mathcal{Z}^M \to [0, 1]$ defined as the average 0-1 loss over the nodes:

$$
f(Z_1, ..., Z_M) = \frac{1}{M} \sum_{i=1}^{M} \mathbb{I}(g(Z_i) \neq y_i)
$$

Since the loss function $\mathbb{I}(\cdot)$ is bounded within $[0, 1]$, changing a single ego-network Z_i to Z_i' changes the value of f by at most $\frac{1}{M}$. Thus, applying McDiarmid's Inequality, with probability at least $1 - \delta$, the estimation error is bounded by:

$$
\eta = \sqrt{\frac{\ln(2/\delta)}{2|\mathcal{V}|}}
$$

Applying this bound to both the source and target estimation errors, we conclude that with probability at least $1 - 2\delta$:

$$
|\hat{\epsilon}^{\mathcal{S}} - \hat{\epsilon}^{\mathcal{T}}| \leq \underbrace{\text{Bound}_{\text{Thm 3.3}}}_{\text{Shift Components (CSS, Label, Feature)}} + \underbrace{\sqrt{\frac{\ln(2/\delta)}{2|\mathcal{V}^{\mathcal{S}}|}} + \sqrt{\frac{\ln(2/\delta)}{2|\mathcal{V}^{\mathcal{T}}|}}}_{\text{Finite Sample Noise}}
$$

This shows that for finite graphs, the error decomposition in Theorem 3.3 holds, subject to a standard complexity term $\mathcal{O}(1/\sqrt{|\mathcal{V}|})$ that vanishes as graph size increases.

A.3 Justification of the Ego-Network Assumption

Theorem 3.3 assumes that the joint distribution of the entire graph can be factorized into the product of local marginal distributions centered at each node. Namely, it treats the graph as a "bag" of independent training examples. While we agree that this assumption is not fully satisfied in real-world graphs, it is grounded in practical graph learning properties:

- **Decaying Correlation:** In many physical and social networks, the correlation between nodes decays as the graph distance increases. The influence of a distant node on a center node is statistically negligible compared to the influence of immediate neighbors.
- **Local Dominance:** The assumption captures the core characteristic of the data generation process for node classification: the immediate structural context (the ego-network) contains the highest density of discriminatory signals. Therefore, modeling the data as a collection of local environments is a justifiable approximation for analyzing distribution shifts, even if it ignores weaker high-order dependencies.

Our empirical results demonstrate that while the independence assumption is a simplification, it successfully captures the dominant factors driving distribution shift in practice on real graphs.

A.4 Clarification on Feature Shift (Δ_{CE})

Feature Shift Definition: We define Feature Shift as a divergence in the conditional feature distribution given the node label, expressed as $\mathbb{P}^{\mathcal{S}}(X_u|Y_u) \neq \mathbb{P}^{\mathcal{T}}(X_u|Y_u)$.

Correspondence to Δ_{CE}: In our proof (Appendix **??**), Δ_{CE} is derived as the discrepancy in prediction probabilities between domains, conditioned on both the center label Y_u and the neighborhood labels Y_v. Because we condition on the label topology and the pre-trained model parameters are frozen, the only variable remaining that can alter the prediction probability is the input feature distribution X. Thus, Δ_{CE} isolates the error gap caused specifically by feature shift.

B DISCUSSION OF USING SOURCE DOMAIN INFORMATION IN GTTA

In this section, we delve deeper into the role of source domain information within GTTA setting. We argue that a truly "source-agnostic" approach is fundamentally limited and that effective adaptation necessitates leveraging, at minimum, summary statistics or assumptions from the source domain. We will illustrate this point through conceptual examples, theoretical arguments, and empirical results, demonstrating how explicitly accounting for source-side properties leads to more robust and reliable adaptation.

B.1 The Necessity of Source Information as an Anchor for Adaptation

We clarify a fundamental point: a purely *source-agnostic* approach to GTTA is inherently limited. Effective adaptation requires anchoring the process with either explicit source statistics or an implicit assumption about the source domain (e.g., a clustering assumption). Attempting adaptation without such an anchor cannot sufficiently correct for the inherent biases learned during pre-training.

Illustrative Example (Fraud Detection). Consider two different models for financial fraud detection, both adapting to the same target domain of *new mobile payment transactions*:

- **Model A** is pre-trained on *Source A (E-commerce Data)*, where fraud is typically characterized by *high-frequency, low-value* transactions. This model is thus biased toward that pattern.
- **Model B** is pre-trained on *Source B (Wire Transfer Data)*, where fraud more often appears as *single, high-value* anomalous transfers. This model carries a different bias.

A single, source-agnostic TTA rule would apply the same adaptation logic to both models and therefore fail to correct their distinct pre-existing biases. In contrast, a method like TSA that uses explicit source statistics performs *tailored* adaptation, counterbalancing each model's specific bias. This intuition is formalized by our analysis and validated empirically.

Theoretical Demand. Theorem 3.3 decomposes the generalization error and makes explicit its dependence on both source and target shifts. The bound implies that to reduce this error, adaptation must account for properties of the source distribution.

Lightweight Information. The required source summary is extremely lightweight: For TSA, a single table of probabilities that consumes negligible space and can be shipped alongside model parameters.

B.2 The Implicit Source Assumptions in "Source-Agnostic" Methods

Table 5: GTTA performance when the source cluster assumption is satisfied (well-separated clusters) and when it is violated (highly imbalanced source clusters) on synthetic datasets, with the target domain held constant.

	ERM	GTrans	HomoTTT	SOGA	TSA-TENT	TSA-LAME	TSA-T3A
Assumption **satisfied**	82.70 ± 4.45	86.67 ± 3.59	85.08 ± 2.87	86.09 ± 3.89	88.78 ± 1.37	88.96 ± 1.66	$\mathbf{89.96 \pm 1.33}$
Assumption **violated**	64.38 ± 4.25	62.30 ± 3.20	60.44 ± 4.95	60.07 ± 6.16	$\mathbf{74.97 \pm 5.41}$	70.77 ± 4.76	$\mathbf{74.97 \pm 3.41}$

As demonstrated in our experiments (Table 5), methods often regarded as source-agnostic (e.g., GTrans, HomoTTT, SOGA) rely critically on implicit structural assumptions about the source domain. In contrast, TSA is more robust when those assumptions do not hold. Every method fails to transfer (accuracy decreases) except TSA.

B.3 TSA Robustness in the Absence of Explicit Source Statistics

Table 6: Performance on MAG for US→CN (top) and CN→US (bottom) when using a uniform distribution for the source prior. **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

ERM	GTrans	HomoTTT	SOGA	TENT	LAME	T3A	TSA-TENT	TSA-LAME	TSA-T3A
31.86 ± 0.83	31.77 ± 0.91	25.89 ± 1.98	21.54 ± 2.52	26.72 ± 1.33	35.75 ± 0.85	41.47 ± 1.15	$\mathbf{27.24 \pm 1.51}$	$\mathbf{37.83 \pm 1.30}$	$\underline{\mathbf{41.56 \pm 1.04}}$
37.41 ± 1.01	36.17 ± 0.89	24.39 ± 2.12	38.95 ± 3.35	27.21 ± 0.88	40.08 ± 1.13	46.50 ± 1.26	$\mathbf{27.82 \pm 0.88}$	$\mathbf{44.72 \pm 0.73}$	$\underline{\mathbf{47.98 \pm 0.89}}$

Even for third-party models where source statistics are unknown, TSA remains effective. Assuming a simple uniform source prior, our method still delivers consistent improvements on real-world datasets, as shown in Table 6. This is because the alignment is then driven primarily by target-graph statistics, providing useful regularization for an SNR-inspired adjustment.

Together, these arguments support the central claim: *effective GTTA requires lightweight source-domain information or, equivalently, explicit assumptions about it.* Methods that pretend to be fully source-agnostic inevitably encode—and are limited by—implicit source assumptions; making such information explicit enables more reliable, bias-aware adaptation.

C ADDITIONAL RESULTS

C.1 TSA Ablation Studies

We conduct an ablation study to examine the individual contributions of neighborhood alignment and SNR-based representation adjustments. For this analysis, we use synthetic datasets with known structural shifts and the real-world datasets exhibiting the highest conditional structure shift (CSS), as detailed in Table 20.

The results, presented in Table 7, consistently show that incorporating both modules yields the best performance. Removing neighborhood alignment leads to a significant performance drop of up to 19.9% on the synthetic dataset and 3.16% on the real-world dataset. The larger drop on the synthetic CSBM data is expected, given the more significant structural shift modeled. Meanwhile, removing SNR adjustments results in a performance decrease of up to 2.43% and 2.97%, respectively. These findings underscore that both modules are crucial for handling structure shift.

Table 7: TSA Ablation studies on CSBM and MAG.

Method	Ablation	Struct. Shift (Imbal.→ Bal.)		US→RU	US→FR
TSA-TENT	Base	88.68±4.99	66.25±7.75	32.53±0.91	28.62±0.63
	w/o Nbr. Align	78.45±5.51	61.12±4.86	32.26±0.95	28.41±0.59
	w/o SNR	86.93±5.05	63.82±8.06	32.52±0.90	28.57±0.59
TSA-LAME	Base	65.09±2.34	52.89±6.11	32.86±2.23	27.22±1.48
	w/o Nbr. Align	48.45±5.30	37.31±3.95	32.70±2.29	26.49±1.76
	w/o SNR	63.65±2.65	51.18±5.84	29.89±1.94	24.41±1.43
TSA-T3A	Base	65.59±2.57	52.34±7.19	46.61±0.88	43.45±0.81
	w/o Nbr. Align	45.69±5.44	36.75±3.09	46.44±0.80	42.09±1.26
	w/o SNR	64.17±2.11	51.04±6.79	46.51±0.94	41.35±1.09

C.2 Visualization of Distribution Shift

Figure 5 (a)-(e) investigates the impact of different types of distribution shifts on GNNs. We use a one-layer GraphSAGE followed by an MLP classifier to visualize node representations on synthetic datasets generated by the CSBM model.

- **Feature Shift:** Figure 5 (b) shows that the representations deviate from the source domain, but the clusters remain separable.
- **Conditional Structure Shift (CSS):** Figure 5 (c) shows that the embeddings largely overlap, making the representation space less discriminative.
- **Label Shift:** Figure 5 (d) illustrates a biased decision boundary, which leads to misclassification of the minority class.
- **SNR Shift:** Figure 5 (e) shows that the node representations remain roughly aligned with the source, but are more dispersed, as the SNR shift introduces noise into the representations.

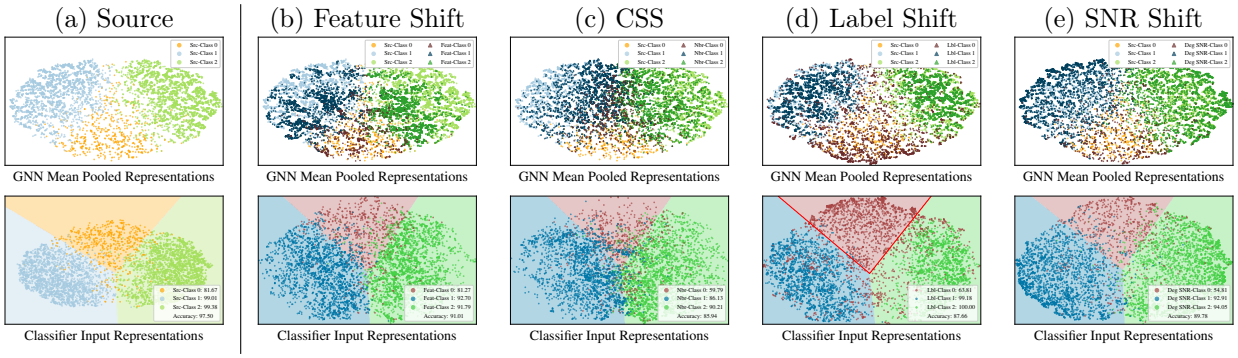

Figure 5: The t-SNE visualization of a one-layer GraphSAGE under different distribution shifts: (top) the output representations given by the GNN encoder and (bottom) the node representations with the classifier decision boundaries. (a) indicates the source domain where the model is pretrained. (b), (c), and (d) stand for feature shift, conditional structure shift (CSS), and label shift. (e) represent the impact of SNR shift induced by degree changes. The color of the nodes represents the ground-truth labels. The legends in the bottom plots show the accuracy and recall scores for each class. The red contours in the bottom (d) highlight the decision boundary of the minority class.

Table 8: Computation time on MAG (CN → US). TTA methods are recorded in adaptation time per epoch. The Wall-Clock Time (seconds) are measured on a single NVIDIA RTX A6000 GPU.

Method	Stage	Computation Time (sec)	Additional Time (%)
–	Inference	0.518±0.038	–
TSA-T3A	Adaptation	1.138±0.072	219.69%
	- Neighborhood Alignment	0.037±0.011	7.14%
	- SNR Adjustment	0.729±0.023	140.73%
	- Boundary Refinement (T3A)	0.368±0.046	71.04%
GTrans	Adaptation	4.309±0.171	831.85%
HomoTTT	Adaptation	2.014±0.815	388.80%
SOGA	Adaptation	9.264±0.218	1788.42%

C.3 Computational Efficiency

We quantify the computation time of TSA when adapting GraphSAGE on the MAG US dataset, containing the largest number of nodes and edges, and compare it with other GTTA methods:

- **Inference time** refers to the time required for the model to generate predictions on the target graph $\mathcal{G}^{\mathcal{T}}$. This is also equivalent to the computation time for ERM.
- **Adaptation time** refers to the time required per epoch *after* inference. It reflects the additional overhead introduced by the adaptation methods.

For a fair comparison, we report the computation time per epoch of adaptation. Note that GTrans alternates between optimizing adjacency structure and node features; in this case, we report the combined runtime of one graph feature update and one edge update as a single adaptation epoch. Table 8 shows that TSA is more efficient than other GTTA baselines. Notably, compared to GTTA baselines, TSA-T3A is approximately 2, 4 and 8 times more efficient than HomoTTT, GTrans and SOGA. The time complexity of neighborhood alignment is $O(1)$ as it is a one-step edge weight assignment. SNR adjustment updates a lightweight MLP and scalar bias terms. While it requires backpropagation, it only introduces minimal overhead compared to full model retraining or extensive graph augmentation.

C.4 Hyperparameter Sensitivity

Hyperparameter analysis on ρ_1. TSA employs neighborhood alignment based on the uncertainty of target pseudo labels. To evaluate the sensitivity of the hyperparameter ρ_1 (as defined in Section 4.1 by the entropy threshold $-\sum_{i\in\mathcal{Y}}[\hat{y}]_i \ln([\hat{y}]_i) \leq \rho_1 \cdot \ln(|\mathcal{Y}|)$), we vary ρ_1 within the range of $[-0.1, 0.1]$ centered around the value selected on the validation set. We set $\rho_1 = 1.0$ for the synthetic datasets and hence we vary it within $[0.9, 1.0]$. We perform evaluations on both synthetic and real-world datasets, using the best-performing variants: TSA-TENT, TSA-T3A and TSA-LAME for each case. TSA-TENT and TSA-T3A are evaluated with the GraphSAGE backbone, while TSA-LAME is evaluated with the GPRGNN backbone. Overall, Figure 6 shows that all tested values of ρ_1 outperform the corresponding non-graph TTA baselines.

Hyperparameter analysis on ρ_2. When optimizing α using the cross-entropy loss in Eq. 4, real-world datasets often produce unreliable hard pseudo-labels. To address this, we adopt the same strategy used for selecting ρ_1. Only hard pseudo-labels whose entropy satisfies $-\sum_{i\in\mathcal{Y}}[\hat{y}]_i \ln([\hat{y}]_i) \leq \rho_2 \cdot \ln(|\mathcal{Y}|)$ are used to optimize α. To further analyze the effect of ρ_2, we vary it within the range $[-0.1, 0.1]$ centered around the value selected on the validation set. We set $\rho_2 = 1.0$ for the synthetic datasets and hence we vary it within $[0.9, 1.0]$. Figure 7 shows that TSA is robust to variations in ρ_2, with all perturbations outperforming the corresponding non-graph TTA baselines.

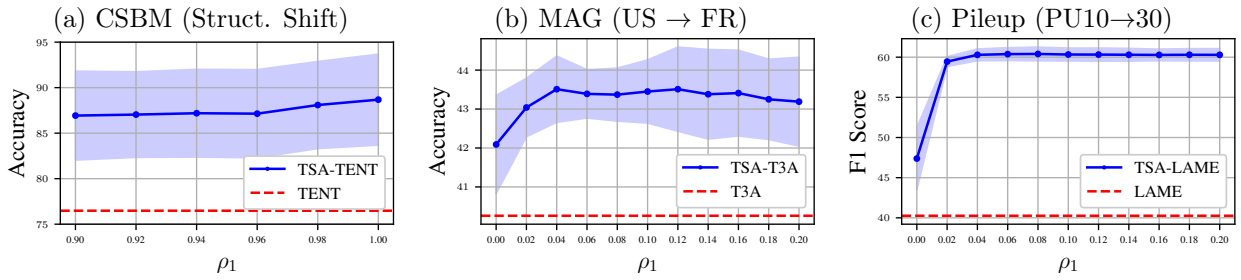

Figure 6: Hyperparameter sensitivity analysis of ρ_1.

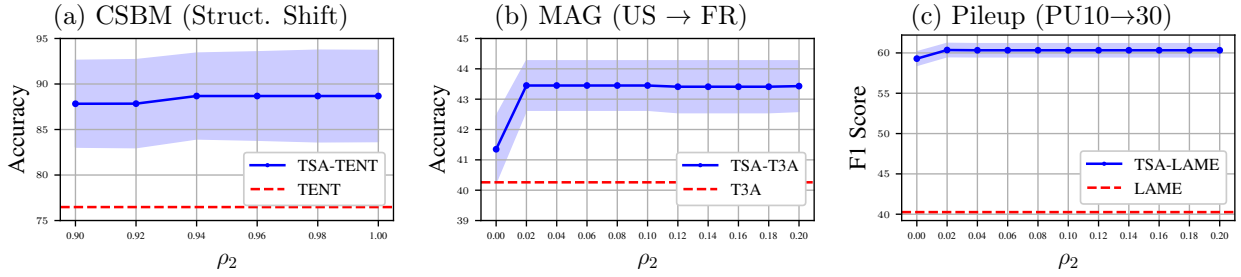

Figure 7: Hyperparameter sensitivity analysis of ρ_2.

C.5 Results of More GNN Architectures

TSA is Model-Agonistic. TSA employs neighborhood alignment and SNR adjustment on edge weights and do not need any assumption on the GNN architecture such as hop-aggregation parameters. We demonstrate that TSA is applicable to common GNN architectures such as GraphSAGE, GPRGNN, and GCN. While different backbones yield varying ERM results, we observe that TSA consistently improves model performance.

Comparison with Baselines. TSA exhibits strong outcomes on real-world datasets and with both GPRGNN and GCN backbones. In all cases, the best results are obtained by GTTA methods (underlined across Table9–14), highlighting the importance of addressing structure shift in graph data. For GPRGNN, TSA consistently outperforms GTrans and HomoTTT and surpasses SOGA by an average of 10.5% and Matcha by an average of 19.5%. For GCN, TSA similarly outperforms GTrans and HomoTTT, and exceeds SOGA by an average of 13.1%. Furthermore, TSA shows prevailing gains over the corresponding non-graph TTA baselines (**bold** across Table9–14). While Matcha sometimes demonstrates strong performance, we frequently observe unstable results that may even fall below ERM. This drop is attributed to the reliance of their proposed PIC loss on minimizing intra-variance, which depends on the quality of node hidden representations in the target domain. Under label shift, node representations can easily degrade, leading to instability in Matcha's performance.

Table 9: GPRGNN results on MAG (accuracy). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

Method	US→CN	US→DE	US→JP	US→RU	US→FR	CN→US	CN→DE	CN→JP	CN→RU	CN→FR
ERM	26.26±1.53	27.09±0.74	38.90±1.06	22.88±1.14	20.69±0.64	27.58±0.55	10.77±0.55	17.41±0.86	12.49±0.74	9.05±0.42
GTrans	26.04±1.41	26.96±0.72	38.71±0.96	23.07±1.08	20.80±0.70	27.49±0.46	10.97±0.56	17.41±0.78	12.58±0.75	9.08±0.41
HomoTTT	26.08±1.35	28.12±0.85	39.13±1.10	24.16±1.17	21.85±0.64	25.50±0.39	9.64±0.41	15.95±0.62	11.10±0.61	8.08±0.31
SOGA	34.70±0.71	35.18±0.33	48.47±0.30	33.63±0.73	27.98±0.42	35.16±0.57	18.50±0.45	28.35±1.04	20.11±0.71	14.73±0.58
ActMAD	25.14±1.41	26.77±0.69	37.97±1.09	22.30±0.99	20.52±0.61	26.89±0.57	10.35±0.52	16.90±0.85	12.03±0.77	8.71±0.42
TENT	13.79±0.72	25.67±0.50	29.13±0.80	18.00±0.14	20.85±0.26	16.29±0.82	7.59±0.73	13.81±0.74	7.82±0.74	5.94±0.34
LAME	24.54±2.55	22.50±0.85	35.93±1.42	17.61±1.46	15.71±0.92	28.02±0.55	9.08±0.61	15.58±1.02	10.87±0.70	8.01±0.38
T3A	40.58±0.23	39.00±0.85	48.96±1.00	46.13±1.56	32.35±1.29	38.92±2.23	38.07±1.45	46.23±1.73	42.82±1.52	28.65±0.65
Matcha-TENT	**13.90±0.71**	**25.87±0.47**	**29.30±0.82**	**18.14±0.19**	**21.03±0.26**	**16.38±0.80**	**7.62±0.73**	**13.84±0.74**	**7.85±0.81**	**5.97±0.36**
Matcha-LAME	8.93±4.76	8.18±0.78	15.46±1.34	6.25±1.43	6.69±0.81	1.35±2.69	**17.60±8.89**	**33.42±2.41**	**18.81±1.42**	4.23±8.02
Matcha-T3A	22.34±18.21	32.18±1.70	42.85±1.15	34.77±2.62	25.75±0.62	**45.27±2.57**	33.01±1.17	42.08±0.82	38.57±1.50	**29.30±0.47**
TSA-TENT	**14.51±0.66**	**25.86±0.54**	**29.47±0.73**	**18.76±0.39**	**21.48±0.23**	**16.96±0.74**	**7.89±0.72**	**14.32±0.70**	**8.35±0.81**	**6.16±0.38**
TSA-LAME	**26.58±2.09**	**22.75±0.85**	**35.95±0.98**	**19.30±1.99**	**17.49±0.79**	**32.30±0.67**	**14.86±1.12**	**21.88±0.95**	**15.02±0.86**	**11.13±0.51**
TSA-T3A	**40.77±0.15**	**39.03±0.86**	**49.38±0.89**	**47.81±1.81**	**32.43±1.24**	**40.94±2.82**	**38.12±1.42**	**46.26±1.81**	**43.70±1.33**	**29.07±0.46**

Table 10: GPRGNN results on Pileup (f1-scores). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

Method	PU10→30	PU30→10	PU10→50	PU50→10	PU30→140	PU140→30	gg→qq	qq→gg
ERM	50.60±3.52	68.09±0.74	32.60±5.55	64.59±1.16	0.68±0.40	24.95±5.32	71.00±0.61	69.56±1.06
GTrans	51.08±3.06	66.26±1.77	33.94±5.39	62.53±1.11	0.75±0.52	25.46±5.19	70.11±1.13	68.40±1.93
HomoTTT	52.05±3.09	67.46±0.94	35.84±4.98	63.59±1.11	0.75±0.43	23.03±5.46	71.06±0.50	69.72±1.10
SOGA	54.18±3.08	66.13±0.70	42.30±4.74	59.97±1.70	1.16±0.84	18.68±5.49	69.72±1.27	68.39±1.75
ActMAD	58.80±1.58	62.22±0.95	49.22±2.21	56.04±2.24	2.16±0.75	17.81±6.45	70.40±0.94	69.95±1.02
TENT	36.64±0.15	2.04±1.05	27.01±0.06	0.62±0.81	12.74±0.50	3.77±3.30	60.19±3.55	71.19±0.58
LAME	46.33±4.67	67.61±0.96	22.58±9.42	64.58±1.00	0.00±0.00	20.94±7.27	70.72±0.74	69.67±0.88
T3A	55.66±2.89	71.36±0.79	47.89±4.19	71.01±0.38	19.69±3.72	54.38±2.56	70.34±1.33	68.63±1.51
Matcha-TENT	36.58±0.17	0.00±0.00	27.01±0.06	0.00±0.00	9.17±0.94	0.00±0.00	0.72±0.76	20.30±4.03
Matcha-LAME	7.45±3.55	0.00±0.00	8.28±4.16	0.00±0.00	0.00±0.00	0.00±0.00	0.52±1.05	6.56±4.08
Matcha-T3A	15.48±2.91	0.00±0.00	14.15±1.86	0.00±0.00	0.00±0.00	9.60±19.19	17.20±14.72	22.77±5.23
TSA-TENT	**39.90±1.18**	**5.07±10.13**	**27.78±0.48**	0.59±0.75	**20.31±0.65**	3.65±3.22	**67.97±1.60**	**71.60±0.18**
TSA-LAME	**60.33±0.80**	**71.54±0.33**	**52.23±0.61**	**70.62±0.35**	**38.32±0.27**	20.49±7.76	**72.28±0.47**	**71.23±0.27**
TSA-T3A	**59.40±1.70**	**72.94±0.18**	**53.13±0.34**	**72.63±0.23**	**36.39±1.48**	**62.31±0.28**	**71.30±1.23**	**69.70±1.04**

Table 11: GPRGNN results on Arxiv and DBLP/ACM (accuracy). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

	1950-2007		1950-2009		1950-2011		DBLP & ACM	
Method	2014-2016	2016-2018	2014-2016	2016-2018	2014-2016	2016-2018	D→A	A→D
ERM	45.62±2.30	40.35±4.46	49.21±1.36	44.78±2.70	56.49±0.97	54.14±1.31	67.34±1.21	56.47±2.84
GTrans	45.25±2.25	40.51±4.31	48.56±1.23	44.57±2.55	55.81±0.99	53.59±1.34	59.61±0.98	59.27±1.79
HomoTTT	45.49±2.64	40.03±4.80	48.10±1.17	42.33±2.34	56.27±1.03	53.25±1.24	67.11±1.03	57.18±2.54
SOGA	48.40±2.29	46.01±4.00	52.59±1.38	50.64±1.63	58.61±1.15	57.68±0.88	67.00±1.03	55.42±3.03
ActMAD	45.47±2.28	40.12±4.42	49.42±1.31	44.96±2.59	56.63±1.00	54.25±1.25	67.07±1.11	55.83±2.76
TENT	44.73±2.63	40.58±3.44	48.47±0.94	42.37±1.52	57.15±0.98	52.45±1.75	61.56±0.77	56.84±1.04
LAME	45.75±3.96	41.35±7.20	49.52±2.24	50.18±3.36	55.65±1.09	54.93±1.72	66.92±0.99	56.58±2.83
T3A	32.73±1.63	31.97±1.75	36.99±2.01	34.10±1.32	40.53±2.88	39.04±3.86	67.21±1.31	65.32±1.69
Matcha-TENT	43.89±2.77	39.30±3.63	48.01±0.90	41.91±1.66	56.28±1.30	51.27±1.64	**64.43±0.90**	**78.61±1.38**
Matcha-LAME	32.93±3.30	24.13±5.06	37.41±2.06	24.16±2.92	48.29±3.27	44.74±4.20	**67.83±1.71**	**70.39±2.15**
Matcha-T3A	24.81±2.32	19.06±3.09	25.85±3.54	19.60±4.56	26.90±1.55	18.74±4.37	**67.59±1.87**	**72.91±2.69**
TSA-TENT	**45.17±2.84**	**41.07±3.67**	**49.13±0.84**	**43.19±1.24**	**57.66±1.00**	**53.00±1.65**	**62.77±0.84**	**61.71±0.99**
TSA-LAME	**49.75±4.12**	**48.67±7.74**	**53.82±0.53**	**55.18±1.21**	**57.67±0.98**	**57.14±1.83**	**67.71±1.10**	**65.20±1.70**
TSA-T3A	**39.11±1.73**	**39.40±3.69**	**41.15±6.33**	**44.65±2.31**	**44.14±2.63**	**40.82±5.28**	**68.02±1.40**	**74.64±1.38**

Table 12: GCN results on MAG (accuracy). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

Method	US→CN	US→DE	US→JP	US→RU	US→FR	CN→US	CN→DE	CN→JP	CN→RU	CN→FR
ERM	27.76±1.49	30.70±0.96	40.12±0.73	27.70±0.75	23.03±0.80	36.46±2.00	21.03±1.56	28.95±2.43	20.53±2.07	16.68±1.54
GTrans	27.44±1.48	30.59±0.99	39.65±0.97	27.79±0.72	22.95±0.88	35.88±2.09	20.80±1.59	28.13±2.49	20.53±2.06	16.52±1.51
HomoTTT	23.64±2.09	28.49±1.29	35.34±1.87	26.21±1.62	23.78±0.56	29.27±2.91	18.60±1.59	23.10±2.01	21.71±1.70	16.03±1.33
SOGA	28.36±2.23	35.61±1.50	45.29±2.08	36.79±1.18	29.30±0.90	44.09±1.92	26.78±1.26	37.93±2.05	33.24±1.49	24.23±1.01
ActMAD	28.11±1.34	31.25±0.90	40.71±0.65	28.61±0.63	23.82±0.74	36.07±2.01	20.73±1.50	28.76±2.33	20.92±2.04	16.65±1.46
TENT	26.66±0.57	33.38±0.32	41.84±0.59	33.01±0.54	29.17±0.20	28.82±0.85	16.35±0.76	25.20±0.85	23.45±0.87	14.96±0.56
LAME	28.20±3.24	30.90±1.77	42.25±0.97	27.20±2.17	20.83±1.08	35.89±2.53	19.31±2.11	30.01±4.10	17.46±2.39	14.24±2.02
T3A	41.53±0.93	45.41±1.76	51.11±1.18	48.02±1.99	40.09±1.53	43.72±2.31	39.68±2.25	47.31±0.84	45.93±1.14	30.95±1.20
TSA-TENT	**27.46±0.53**	**33.89±0.31**	**42.34±0.60**	**33.72±0.53**	**29.59±0.21**	**29.33±0.86**	**16.82±0.75**	**25.90±0.89**	**24.03±0.81**	**15.32±0.56**
TSA-LAME	**28.64±3.24**	**31.01±1.91**	**42.35±0.86**	**27.22±1.87**	**22.27±0.52**	**39.64±2.08**	**22.96±1.62**	**33.94±3.14**	**20.08±1.93**	**17.36±1.91**
TSA-T3A	**41.61±0.95**	**46.13±1.83**	**51.81±1.22**	**48.87±1.43**	**42.98±1.90**	**45.48±2.69**	**40.11±2.74**	**47.45±0.42**	**45.95±1.17**	**32.03±1.30**

Table 13: GCN results on Pileup (f1-scores). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

Method	PU10→30	PU30→10	PU10→50	PU50→10	PU30→140	PU140→30	gg→qq	qq→gg
ERM	8.30±1.79	29.20±4.57	2.27±0.70	26.61±6.79	0.33±0.13	16.08±10.46	23.71±2.49	26.73±4.79
GTrans	11.97±2.67	28.88±4.71	3.90±1.26	27.20±6.88	0.91±0.29	16.98±11.06	24.62±2.38	28.88±5.14
HomoTTT	8.94±2.05	31.21±4.79	2.54±0.81	27.59±6.18	0.55±0.18	16.77±11.23	24.84±3.27	27.48±4.95
SOGA	5.81±2.20	35.21±4.00	1.00±0.44	27.19±6.76	0.54±0.20	19.17±11.32	25.38±1.66	28.06±5.06
ActMAD	7.20±2.22	30.45±4.13	1.15±0.41	24.35±6.69	0.31±0.14	16.67±9.32	23.77±1.78	27.36±4.75
TENT	2.45±0.93	35.99±3.12	1.63±0.75	26.20±7.51	0.09±0.05	13.84±8.89	19.96±4.06	29.58±4.46
LAME	7.48±2.17	30.91±5.58	1.46±0.75	24.97±9.02	0.06±0.06	15.30±10.74	21.50±5.24	25.72±6.51
T3A	41.55±13.17	62.80±2.48	29.92±7.29	56.32±14.68	4.45±2.48	33.26±4.35	60.15±10.72	62.77±8.69
TSA-TENT	**2.96±1.12**	**38.45±3.83**	**2.17±0.82**	**26.85±7.89**	**0.24±0.16**	**15.20±10.17**	**24.22±3.88**	**33.47±4.43**
TSA-LAME	**12.71±2.29**	**32.37±8.02**	**4.14±1.48**	**27.14±10.60**	**0.38±0.19**	**16.53±12.07**	**29.29±2.32**	**27.72±4.22**
TSA-T3A	**43.86±13.14**	**64.00±1.56**	**33.31±8.50**	**60.22±8.38**	**7.12±4.19**	**35.72±4.56**	**62.03±8.01**	**64.02±6.87**

Table 14: GCN results on Arxiv and DBLP/ACM (accuracy). **Bold** indicates improvements in comparison to the corresponding non-graph TTA baselines. Underline indicates the best model.

	1950-2007		1950-2009		1950-2011		DBLP & ACM	
Method	2014-2016	2016-2018	2014-2016	2016-2018	2014-2016	2016-2018	D→A	A→D
ERM	49.25±1.58	50.64±3.11	52.61±2.67	51.92±4.12	58.83±0.83	57.73±1.27	62.56±3.32	47.59±1.92
GTrans	49.10±1.68	50.57±3.10	52.50±2.71	51.87±4.26	58.72±0.89	57.75±1.31	62.03±2.31	51.65±3.09
HomoTTT	46.22±1.54	46.11±2.79	48.96±3.13	46.83±5.00	55.10±1.63	53.18±2.52	63.21±3.31	48.45±1.68
SOGA	49.82±2.38	52.36±2.54	52.92±1.08	54.45±1.15	57.93±1.71	58.66±2.02	64.22±2.40	53.29±2.08
ActMAD	49.24±1.57	50.68±2.94	52.71±2.64	52.10±3.98	58.83±0.86	57.85±1.25	62.68±3.36	47.58±1.89
TENT	48.59±1.41	49.90±1.15	53.39±1.82	53.45±2.11	58.72±0.91	58.23±0.95	60.80±2.51	48.11±1.43
LAME	49.77±2.50	50.97±4.60	53.35±2.72	52.70±4.25	58.85±0.57	57.05±1.57	62.51±3.33	47.64±1.70
T3A	43.37±1.38	41.32±3.13	49.95±3.76	47.11±6.73	55.56±1.16	53.57±1.94	62.60±3.65	49.65±1.83
TSA-TENT	**49.63±1.26**	**50.78±0.74**	**53.78±1.57**	**53.67±1.85**	**58.90±0.94**	**58.27±0.95**	**61.31±2.39**	**53.14±2.01**
TSA-LAME	**50.95±2.05**	**53.04±2.86**	**54.49±1.81**	**54.13±3.35**	**59.26±0.70**	**57.55±1.30**	**63.92±2.76**	**55.19±4.50**
TSA-T3A	**45.37±0.76**	**45.46±1.92**	**51.57±2.73**	**49.70±5.44**	**56.14±1.02**	**54.84±1.17**	**63.31±3.50**	**54.53±3.07**

C.6 Sensitivity of γ to Noise in Pseudo-Labels

In this subsection, we investigate how sensitive γ is to noise in pseudo-labels. Specifically, we progressively add $x\%$ additional noise to the input features and evaluate the discrepancy between the estimated γ and the ground-truth value calculated from the labels. We report root mean squared error (RMSE). The results are shown in Table 15. Results indicate that the estimated γ values are robust: the relative increase in error is much smaller than the injected input noise. Even with 100% additional noise, the increase in γ estimation error is only about 2.6%$\sim$8.3% across methods.

Table 15: Sensitivity of γ estimation to pseudo-label noise on synthetic CSBMs. The percentages in parentheses denote the relative increase compared to the "No Add" setting.

Method	No Add	Add 10%	Add 50%	Add 100%
TSA-TENT	0.6938	0.6984 (+**0.66%**)	0.7159 (+**3.18%**)	0.7330 (+**5.64%**)
TSA-LAME	2.4489	2.5033 (+**2.22%**)	2.6856 (+**9.66%**)	2.6516 (+**8.28%**)
TSA-T3A	1.4768	1.4791 (+**0.16%**)	1.4943 (+**1.18%**)	1.5155 (+**2.69%**)

C.7 TSA on the Heterophilic Graphs

TSA is in principle applicable to heterophilic datasets because it makes no homophily-specific assumptions. We include an experiment adapting between homophilic and heterophilic CSBMs to demonstrate the model's adaptability empirically. Specifically, we adapt a source model first trained on a heterophilic dataset to a homophilic dataset, and vice versa. We control neighborhood shifts to synthesize data with high heterophily. The results are shown in Table 16. Across both directions, TSA consistently improves performance and attains the best accuracy. By contrast, SOGA relies on homophily assumptions and hence fails to adapt effectively. Its accuracy decreases under both settings.

Table 16: We report performance when adapting between heterophilic and homophilic synthetic CSBMs. A higher node homophily ratio (Pei et al., 2020) indicates that a larger fraction of each node's neighbors share its label.

Homophily ratio	ERM	GTrans	HomoTTT	SOGA	TSA-TENT	TSA-LAME	TSA-T3A
0.041 $\rightarrow$ 0.775	29.22±18.20	29.98±17.46	29.75±18.28	26.88±17.94	**49.47±16.55**	33.12±14.63	39.51±17.68
0.775 $\rightarrow$ 0.041	36.17±1.98	35.29±3.00	36.02±2.63	35.14±1.88	**39.11±3.88**	37.13±1.04	36.95±1.87

D DATASET DETAILS

D.1 Dataset Statistics

Here we summarize the statistics of the four real-world graph datasets in Table 17, 18, and 19. The edges are counted twice in the edge list as they are undirected graphs.

Table 17: Arxiv and ACM/DBLP dataset statistics

	ACM	DBLP	1950-2007	1950-2009	1950-2011	1950-2016	1950-2018
Nodes	7410	5578	4980	9410	17401	69499	120740
Edges	11135	7341	5849	13179	30486	232419	615415
Features	7537	7537	128	128	128	128	128
Classes	6	6	40	40	40	40	40

Table 18: MAG dataset statistics

	US	CN	DE	JP	RU	FR
Nodes	132558	101952	43032	37498	32833	29262
Edges	697450	285561	126683	90944	67994	78222
Features	128	128	128	128	128	128
Classes	20	20	20	20	20	20

Table 19: Pileup dataset statistics

	gg-10	qq-10	gg-30	gg-50	gg-140
Nodes	18611	17242	41390	60054	154750
Edges	53725	42769	173392	341930	2081229
Features	28	28	28	28	28
Classes	2	2	2	2	2

D.2 Dataset Shift Statistics

Below we provide two metrics indicating the neighborhood and label shift in real-world dataset. We measure the CSS by computing the weighted average of the total variation distance between of the neighborhood distribution:

$$\text{CSS} := \frac{1}{2}\sum_{j\in\mathcal{Y}}\sum_{k\in\mathcal{Y}} \mathbb{P}^{\mathcal{T}}(Y_u = j)|\mathbb{P}^{\mathcal{S}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u) - \mathbb{P}^{\mathcal{T}}(Y_v = k|Y_u = j, v \in \mathcal{N}_u)|$$

The label shift is measured as the total variation distance between the label distribution, which corresponds to the first term in Theorem 3.3.

$$\text{Label Shift} := \frac{1}{2}\sum_{j\in\mathcal{Y}} |\mathbb{P}^{\mathcal{S}}(Y_u = j) - \mathbb{P}^{\mathcal{T}}(Y_u = j)|$$

Table 20: MAG dataset shift metrics

	$US \to CN$	$US \to DE$	$US \to JP$	$US \to RU$	$US \to FR$	$CN \to US$	$CN \to DE$	$CN \to JP$	$CN \to RU$	$CN \to FR$
CSS	0.2137	0.1902	0.1476	0.2855	0.2238	0.1946	0.2053	0.1361	0.2025	0.2039
Label Shift	0.2734	0.1498	0.1699	0.3856	0.1706	0.2734	0.2691	0.1522	0.2453	0.2256

Table 21: Pileup dataset shift metrics

	Pileup Conditions						Physical Processes	
Domains	PU10 → 30	PU30 → 10	PU10 → 50	PU50 → 10	PU30 → 140	PU140 → 30	$gg \to qq$	$qq \to gg$
CSS	0.1493	0.1709	0.2039	0.2595	0.1293	0.1635	0.0214	0.0205
Label Shift	0.2425	0.2425	0.2982	0.2982	0.1377	0.1377	0.0341	0.0348

Table 22: Real dataset shift metrics

	1950-2007		1950-2009		1950-2011		DBLP/ACM	
Domains	2014 – 2016	2016 – 2018	2014 – 2016	2016 – 2018	2014 – 2016	2016 – 2018	$D \to A$	$A \to D$
CSS	0.3005	0.3450	0.2583	0.3120	0.1833	0.2567	0.1573	0.2227
Label Shift	0.2938	0.4396	0.2990	0.4552	0.2853	0.4438	0.3434	0.3435

E DETAILED EXPERIMENTAL SETUP

E.1 Datasets Setup

For Arxiv, MAG, and DBLP/ACM we follow the setting of (Liu et al., 2024). Hence we discuss the setting of synthetic dataset and pileup in the following:

Synthetic Dataset. The synthetic dataset is generated by the contextual stochastic block model (CSBM) (Deshpande et al., 2018). The generated graph contains 6000 nodes and 3 classes. We alter the edge connection probability matrix $\mathbf{B}$ and the label ratio $\mathbb{P}_Y$ Specifically, the edge connection probability matrix $\mathbf{B}$ is a symmetric matrix given by

$$\mathbf{B} = \begin{bmatrix} p & q & q \\ q & p & q \\ q & q & p \end{bmatrix}$$

where p and q indicate the intra- and the inter-class edge probability respectively. We assume the node features are sampled from a Gaussian distribution $X_u \sim \mathcal{N}(\boldsymbol{\mu}_u, \sigma \boldsymbol{I})$ where we set $\sigma = 0.3$ as well as $\boldsymbol{\mu}_u = [1, 0, 0]$ for class 0, $\boldsymbol{\mu}_u = [0, 1, 0]$ for class 1, and $\boldsymbol{\mu}_u = [0, 0, 1]$ for class 2.

- The source graph for setting 1-6 has $\mathbb{P}_Y = [0.1, 0.3, 0.6]$, with $p = 0.01$ and $q = 0.0025$.
- For neighborhood shift we fix the same class ratio but change the connection probability:
  - Condition 1: $p = 0.005$, $q = 0.00375$.
  - Condition 2: $p = 0.005$, $q = 0.005$.
- For condition 3 and 4 we introduce degree shift for SNR discrepancy.
  - Condition 3: $p = \frac{0.005}{2}$, $q = \frac{0.00375}{2}$.
  - Condition 4: $p = \frac{0.005}{2}$, $q = \frac{0.005}{2}$.
- For condition 5 and 6 we investigate structure shift under training from give training is from the imbalanced source label ratio.
  - Condition 5: $\mathbb{P}_Y = [1/3, 1/3, 1/3]$ $p = \frac{0.005}{2}$, $q = \frac{0.00375}{2}$.
  - Condition 6: $\mathbb{P}_Y = [1/3, 1/3, 1/3]$ $p = \frac{0.005}{2}$, $q = \frac{0.005}{2}$.
- The source graph for Condition 7-8 has $\mathbb{P}_Y = [1/3, 1/3, 1/3]$, with $p = 0.01$ and $q = 0.0025$.
- For condition 7 and 8 we investigate structure shift under training from balanced source ratio. The p and q in Condition 7 and 8 is the same as in Condition 5 and 6, respectively.
  - Condition 7: $\mathbb{P}_Y = [0.1, 0.3, 0.6]$, $p = \frac{0.005}{2}$, $q = \frac{0.00375}{2}$.
  - Condition 8: $\mathbb{P}_Y = [0.1, 0.3, 0.6]$, $p = \frac{0.005}{2}$, $q = \frac{0.005}{2}$.

Pileup. Under the study of different PU levels we use the data from signal gg. Compared to the Pileup datasets used in the (Liu et al., 2024), we explicitly label all types of particles to better align the neighborhood distribution. Note that the charged particle can achieve all most perfect recall as their label information is encoded in their feature, hence the four-class classification still reduces into a binary classification between LC neutral particles and OC neutral particles. For the study of different data generation process, we use PU10 from signal $gg \to qq$ and $qq \to gg$.

E.2 Pretraining Setup

Model Architecture and Pretraining. We use GraphSAGE as the model backbone, with a 3-layer mean pooling GNN as the encoder and a 2-layer MLP with a batch normalization layer as the classifier. The hidden dimension of the encoder is set to 300 for Arxiv and MAG, 50 for Pileup, 128 for DBLP/ACM, and 20 for the synthetic datasets. The hidden dimension of the classifier is set to 300 for Arxiv and MAG, 50 for Pileup, 40 for DBLP/ACM, and 20 for the synthetic datasets. For the GPRGNN backbone, we follow the same configuration but increase the number of encoder layers to 5. For the GCN backbone, we use a 3-layer GCN as the feature

encoder, followed by the same 2-layer MLP classifier as in the other backbones. All models are trained for 400 epochs with the learning rate set to 0.003. We use Adam (Kingma, 2015) as the optimizer for all models, and apply a learning rate scheduler with a decay rate of 0.9. All experiments are repeated 5 times under different initializations and data splits to ensure consistency.

Hardware and Frameworks. All experiments are conducted on a server with a single NVIDIA RTX 6000 GPU and an AMD EPYC 7763 64-Core Processor. We use PyTorch 2.2.2 (Paszke et al., 2017) with CUDA 12.1 as the framework. For graph neural networks, we use PyTorch Geometric 2.5.2 (Fey & Lenssen, 2019).

E.3 Evaluation and Metric

The source graph is splitted into train/val/test 60/20/20 during the pretraining stage. The target graph is splitted into labeled/unlabeled 3/97 during test time. We use the the labeled nodes in the target graph to do hyperparameter tuning and select the model with the optimal validation score. We follow the metrics in (Liu et al., 2024) for evaluation. MAG, Arxiv, DBLP/ACM, and synthetic datasets are evaluated with accuracy. For the MAG datasets, only the top 19 classes are evaluated. For the Pileup datasets, we evaluate the performance with f1 score.

E.4 Hyperparameter Tuning

Below we introduce the search space of the hyperparameters. LAME is a parameter-free approach so we do not tune it.

- TSA introduces three hyperparameters (1) the learning rate lr for optimizing α, (2) the ratio ρ_1 for reliable assignment of γ based on entropy $H(\hat{y}) \leq \rho_1 \cdot \ln(|\mathcal{Y}|)$, and (3) the ratio ρ_2 for filtering out unreliable hard pseudo-labels in Eq. 4 based on entropy $H(\hat{y}) \leq \rho_2 \cdot \ln(|\mathcal{Y}|)$. We select lr from $\{0.001, 0.01, 0.05, 0.1\}$. For real world datasets, we select ρ_1 from $\{0, 0.01, 0.1, 0.5\}$, and ρ_2 from $\{0.1, 1.0\}$. For synthetic datasets, we set ρ_1 and ρ_2 to 1.0. We present the results updated by one epoch. We observe that given the same domain shifts, the optimal hyperparameters of ρ_1 may differ due to different performance of the boundary refinement approaches (TENT, T3A, and LAME).
- Matcha introduces two hyperparameters learning rate lr and the number of epochs T. Based on their hyperparameter study, We select lr from $\{0.1, 1, 5, 10\}$ and T from $\{2, 10\}$.
- GTrans introduces four hyperparameters learning rate of feature adaptation lr_f, learning rate of structure adaptation lr_A, the number of epochs T, and the budget to update. We select lr_f from $\{0.001, 0.01\}$, $lr_A = \{0.01, 0.1, 0.5\}$ from $\{0.01, 0.05\}$, T from $\{5, 10\}$, and budget from $\{0.01, 0.05, 0.1\}$.
- HomoTTT introduces two hyperparameter lr and $k0$. The hyperparameter $k0$ is a scaling factor for the model selection score. We present the results updated by one epoch. We select lr from $\{0.001, 0.01\}$ and $k0$ from $\{0.001, 0.01\}$.
- SOGA introduces one hyperparameter lr. We present the results updated by one epoch and select lr from $\{0.001, 0.01\}$.
- TENT introduces one hyperparameter lr. We present the results updated by one epoch and select lr from $\{0.001, 0.01, 0.05\}$. Note that TENT is applied only to the classifier in our implementation, as GNNs typically do not include batch normalization layers.
- T3A introduces one hyperparameter M for deciding the number of supports to restore. We select M from $\{5, 20, 50, 100\}$
- ActMAD introduces one hyperparameter lr. We present the results updated by one epoch and select lr from $\{0.001, 0.01\}$. We adapt ActMAD to GNNs by aligning the outputs of each GNN layer before the classifier.