Comparison of Test Time Adaptation Techniques for node classification using Graph Neural Networks

Paria Mehrbod
Computer Science, Concordia University
paria.mehrbod@mila.quebec

February 8, 2025

Abstract

Test-time adaptation (TTA) has emerged as a promising approach for addressing distribution shifts in Graph Neural Networks (GNNs) without requiring access to source training data. This report presents a comprehensive comparison of three TTA methods—GraphTTA, HomoTTT, and SOGA—for node classification tasks. We evaluate these methods across different domain adaptation scenarios using citation network datasets. The analysis reveals varying effectiveness across different types of distribution shifts: SOGA excels in natural domain shifts but shows vulnerability to feature corruption, while HomoTTT demonstrates more consistent but moderate improvements. We identify key limitations in current approaches, including sensitivity to label imbalance and optimization challenges during adaptation. Additionally, we provide insights into how dataset characteristics and adaptation duration impact performance. These findings highlight the need for more robust TTA methods and theoretical frameworks to understand their convergence properties.

1 Introduction

Graph Neural Networks (GNNs) have changed the way we analyze graph-structured data, showing impressive results in tasks such as node classification in citation networks, social networks, and financial transaction systems. However, these models often face difficulties when used on graphs that come from different distributions than those in their training data. This change in distribution appears in practical situations, such as in citation networks where research topics and citation patterns develop over time, or in social networks where user interaction patterns vary across different platforms or time periods. The decline in performance during these shifts presents a notable challenge, especially when access to the original training data is limited because of privacy issues or the costs associated with retraining the model. This paper addresses several key research questions regarding test-time adaptation for GNNs:

- How do different TTA methods compare in performance across various domain shift scenarios, particularly in node classification tasks?

- What are the key limitations and strengths of existing TTA approaches when dealing with natural domain shifts versus artificial feature corruption?
- How do dataset characteristics, such as size and label distribution, impact the effectiveness of different TTA methods?
- What role does adaptation duration play in the performance of different TTA methods, and how can we optimize it without access to target labels?

These questions arise from the practical challenges faced when deploying GNN models in real-world scenarios, where distribution shifts are common and access to training data may be restricted. By systematically investigating these questions, we aim to provide insights for developing more robust and effective TTA methods for GNNs.

2 Related Work

2.1 Graph Domain Adaptation and Self-Supervised Learning

Early approaches to addressing distribution shifts in graphs focused on unsupervised domain adaptation (UDA). Methods like DANE[1] and UDAGCN[2] employ domain adversarial training to align feature distributions between domains. Concurrently, graph self-supervised learning has emerged as a powerful paradigm for learning transferable representations. However, these approaches typically require access to source data or labeled target data for fine-tuning.

2.2 Test Time Adaptation in Computer Vision

Test-time adaptation has shown promising results in computer vision, offering a framework for model adaptation without source data access. Tent[3] introduced entropy minimization as a self-supervised objective during testing. TTT+++[4] utilizes offline feature extraction and online feature alignment to conduct regularization adaptation without the need to revisit the training data. BN Adapt[5] updates the batch normalization statistics according to the test samples. SHOT[6] exploits both information maximization and self-supervised pseudo-labeling during testing. These methods demonstrate the potential of test-time adaptation but are specifically designed for image data.

2.3 Test Time Adaptation for Graph Neural Networks

Recent studies have started to apply TTA principles to graph-structured data, but this area is still not widely investigated. GTrans[7] was the first to introduce test-time adaptation for graphs, emphasizing the adaptation of the test graph data instead of altering the models. The method uses a graph transformation strategy that changes both input feature matrices and adjacency matrices during testing, without needing particular model architectures. GT3[8] expanded this research for graph classification tasks by introducing a framework that uses both local and global graph information to enhance performance. GraphTTA[9] uses

a learnable augmenter that works in an adversarial manner within its framework for graph contrastive learning, which also focuses on the task of graph classification.

For node classification specifically, very few methods have been proposed to date. SOGA[10] introduced a novel source-free framework that combines information maximization with structural consistency objectives to adapt GNNs without access to source data. Meanwhile, HomoTTT[11] leveraged graph homophily properties through a contrastive learning approach with adaptive augmentation strategies. The small number of methods focused on test time adaptation, especially in the context of node classification, points to a notable gap in existing research.

3 Background

3.1 TTA Framework

The objective of this study is the node classification task, where the aim is to predict the labels of nodes within a graph. Consider that we have a training graph G_{Tr} and a test graph G_{Te}, each with corresponding sets of node attribute information and label information, noted as $G_{Tr} = (X_s, Y_s)$ and $G_{Te} = (X_t, Y_t)$, respectively. Note that in both graphs the dimensionality of node features should be the same. Additionally, the GNN model, which learns node representations of an input graph, is denoted by $f_\theta(\cdot)$. Typically, a model $f_\theta(x)$ is trained on source data (X_s, Y_s) with parameter θ_f and then used to do inference on the target data (X_t, Y_t). When there is a distribution shift between the source data and the target data, the model $f_\theta(x)$ may fail to perform ideally on the target data X_t, Y_t without having access to label Y_t or the training phase.

3.2 Overview of selected TTA Methods for graphs

- GraphTTA utilizes an adversarial learning approach where a GNN-based augmentor and classifier work against each other. The augmentor, initialized from the classifier, generates structure-aware graph augmentations for contrastive learning. Through a min-max game, the classifier tries to minimize the contrastive loss while the augmentor tries to maximize it by creating challenging but meaningful positive samples. This helps adapt the model to the target domain while preserving important graph properties.

- HomoTTT performs node-level contrastive learning using edge dropping based on homophily scores - edges connecting similar nodes are more likely to be preserved. Negative samples are created by shuffling node features. A key innovation is their model selection mechanism that identifies which nodes would benefit from test-time adaptation and selectively applies HomoTTT only to those nodes, while using the original model for others.

- SOGA combines two objectives: information maximization and structure consistency. The first objective maintains prediction diversity while increasing prediction certainty on the target graph. The second objective leverages the target graph structure by considering both local neighborhood relationships and structural roles of nodes. The

method adapts the source model to the target domain without requiring access to source data.

4 Goals and Objectives

This project focuses on comparing different methods for adapting Graph Neural Networks (GNNs) during test time in response to distribution shifts using a single benchmark. The goal is to evaluate their performance against each other and identify the strengths and weaknesses of each method. SOGA and HomoTTT are two methods that have been specifically created for the purpose of node classification. GraphTTA was first introduced for the purpose of graph classification.

For a more comprehensive comparison, I have developed a new method for node classification inspired by GraphTTA, which adpots the same contrastive learning setup and could be a suitable candidate for comparing to HomoTTT. However, there is no available implementation for the HomoTTT, therefore, I have implemented a simpler version of their method which includes the homophily-based Graph Contrastive Learning and Adaptive Augmentation Strategy and excludes the model selection mechanism for simplicity. The objectives I aim to address are as follows:

- Adapting and implementing GraphTTA for the node classification task
- Implementing HomoTTT
- Analyzing and evaluating current research on test time adaptation for node classification in GNNs using a consistent setup

I believe that comparing these methods under the same setting would better clarify the challenges of this problem and would lead to a better insight for future work.

5 Methodology

5.1 Datasets

Datasets DBLPv8 and ACMv9 are collected by [12] from Aminer[13]. They are available at https://github.com/GRANDLab/UDAGCN. The statistics of experimental datasets are shown in Table 1. Cora[14] is a citation network where nodes correspond to papers, while the edges represent citation relationships among the papers. To simulate an OOD scenario, an artificial noise is added to the node features by randomly flipping node features with probability 0.05.

Datasets	# Nodes	# Edges	# Features	# Labels
DBLP	5578	7341	7537	6
ACM	7410	11135	7537	6
CORA	2708	5429	1433	7

Table 1: Dataset Statistics

5.2 Algorithms and Tools

In the experiments, we use a GCN architecture consisting of 3 layers with hidden dimensions of 256 and 128, followed by a linear classifier layer that maps the node representations to the number of classes. The source model was trained for 100 epochs on the source dataset, and the model achieving the best validation accuracy was selected for adaptation to the target domain.

5.2.1 Adapting GraphTTA for node classification

The training process maintains the adversarial min-max optimization from GraphTTA but adapts it for node-level features. The most significant adaptation of this work lies in the modification of the contrastive learning framework. The original GraphTTA methodology generates positive pairs by creating augmented versions of each input graph and identifies negative samples from graphs that have been assigned different pseudo-labels during the adaptation process. This implementation takes a different approach while maintaining the core concept of graph-wide augmentation. The process begins by applying multiple augmentations to the input graph, creating several modified versions of the original structure. Subsequently, each node across all these augmented versions is assigned a pseudo-label through the current state of the model. When constructing the contrastive learning pairs, any node from any augmented version that shares the same pseudo-label as a given anchor node is considered a positive sample for that anchor. Conversely, all nodes that have been assigned different pseudo-labels are treated as negative samples.

To ensure stability, nodes that do not possess any positive pairs are skipped. Additionally, node features are normalized before computing the loss. The augmentation strategy follows the original paper, but implements a parameter-free version of the augmenter. Instead of using an MLP for sampling the score of keeping each edge, this implementation employs normalized inner products to determine the importance of each edge. This decision was made to reduce the number of parameters to be trained at test time given the small size of the datasets. Moreover, we keep the parameters of the encoder classifier head fixed during adaptation, which provides cleaner gradient flow during adaptation.

5.2.2 Implementation details for HomoTTT

The implementation of HomoTTT follows the paper's instructions as much as available details and information allowed. However, for simplicity, the implementation omits the Homophily-based model selection component described in the paper, meaning all nodes use the adapted model's predictions rather than selectively choosing between original and adapted models based on Homophily scores. Regarding the contrastive learning setup, one positive view (augmented graph) and one negative view (shuffled features) are generated per graph, the contrastive loss computes similarities between each node's features and its counterparts in negative and positive samples. For computing homophily scores, each node is assigned a pseudo-label by using k-means clustering, where k equals the number of classes.

6 Results

The experimental results show varying effectiveness of test-time adaptation methods across different domain adaptation scenarios. The accuracy results show the best performance across 10 epochs of test time training and are aggregated across different seeds. The performance comparison of different methods is presented in Table 3.

6.1 Cross-Domain Adaptation

For cross-domain adaptation between DBLP and ACM citation networks, all methods showed improvement over the source model baseline, with SOGA achieving the strongest performance. In DBLP→ACM adaptation, SOGA attained a 5.6% improvement in Macro F1 over the source model, while in ACM→DBLP, both SOGA and HomoTTT demonstrated substantial gains with Macro F1 scores of 94.24% and 88.52% respectively. It is worth noting that DBLP and ACM were the original evaluation datasets in SOGA's development, which may partially explain its superior performance in these scenarios.

6.2 Corrupted Feature Adaptation

For adaptation to artificially corrupted data (Cora→Cora-C), we observed markedly different behavior. HomoTTT demonstrated the strongest performance with 71.95% Macro F1, showing significant improvement over the source model. This robust performance on Cora is particularly noteworthy as it was HomoTTT's primary evaluation dataset during development. In contrast, SOGA's performance degraded substantially to 29.11%, suggesting vulnerability to artificial feature corruption. These results highlight how different types of distribution shifts may require different adaptation strategies, and how method performance can be influenced by the specific challenges present in their development datasets.

Method	DBLP→ACM		ACM→DBLP		Cora→Cora-C	
	Macro F1	Micro F1	Macro F1	Micro F1	Macro F1	Micro F1
GraphTTA	58.12±1.55	65.68±0.74	73.89±5.69	73.11±5.45	52.25±2.13	58.06±3.62
HomoTTT	59.39±1.76	67.58±1.21	88.52±1.82	88.34±1.76	71.95±1.83	73.22±1.74
SOGA	63.32±0.32	71.85±0.46	94.24±0.14	93.99±0.15	29.11±12.58	34.42 ± 9.86
Source	57.72±2.44	66.53±2.36	70.10±2.44	69.24±2.75	61.35±9.79	63.53 ±8.75

Table 2: Best performance achieved during test-time adaptation across different datasets (within 10 epochs)

6.3 Impact of Test-Time Training Duration

The effectiveness of TTA methods is significantly influenced by hyperparameter choices, particularly the duration of adaptation. Without access to labeled validation data during adaptation, selecting optimal hyperparameters becomes challenging. Figure 1 shows the evolution of F1 scores during test-time adaptation across epochs. This reveals that extended

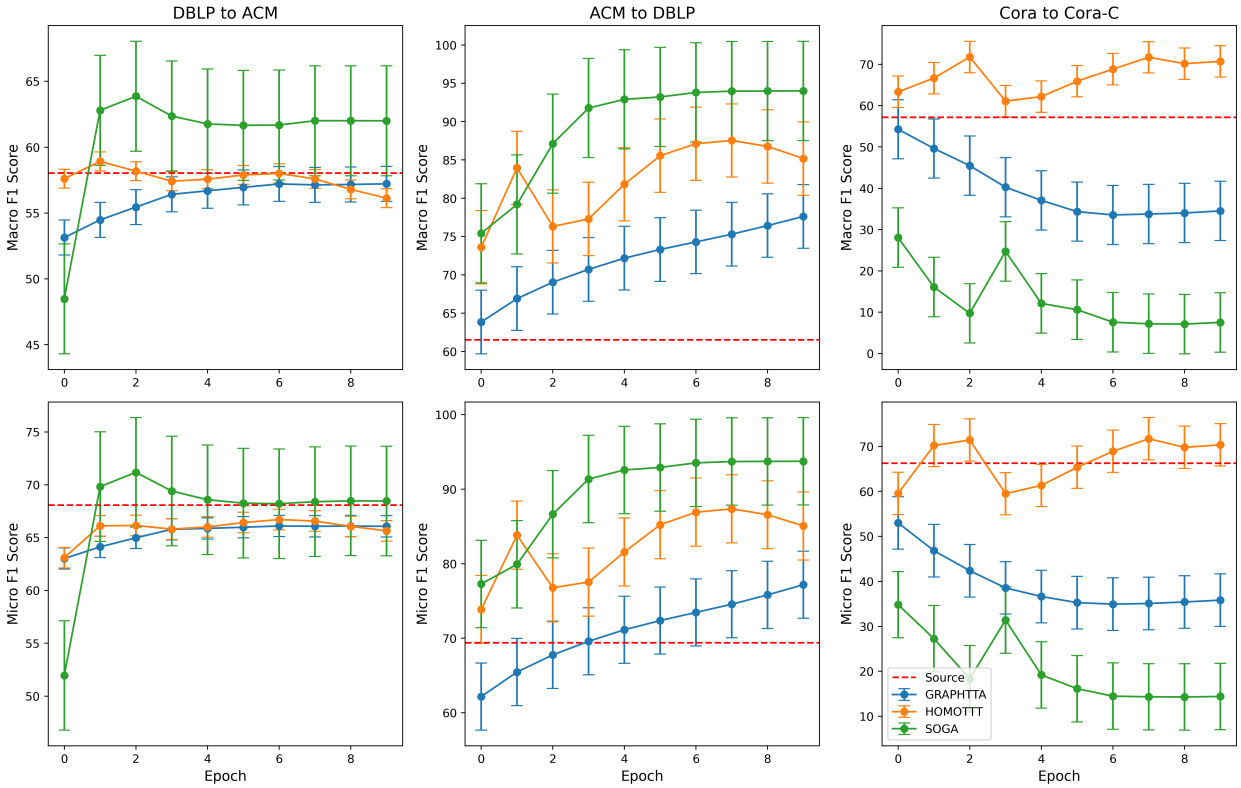

Figure 1: Evolution of F1 scores during test-time adaptation. The dashed line indicates source model baseline performance on target data.

training doesn't necessarily lead to better performance, with some methods showing unstable behavior or performance degradation over time.

This instability is further evidenced in Table 3, which reports the performance at the final epoch rather than the best epoch shown previously. Notably, HomoTTT maintains relatively stable performance between its best and final epochs across all scenarios. In contrast, GraphTTA shows significant degradation on the corrupted Cora dataset, dropping from 52% to 37% Macro F1, indicating potential overfitting during prolonged adaptation. SOGA similarly exhibits some performance drop in the DBLP→ACM scenario while maintaining its strong performance on ACM→DBLP, highlighting how adaptation stability can vary across different domain shifts.

7 Discussion

7.1 Dataset Characteristics Impact

The consistent performance gap between ACM→DBLP and DBLP→ACM adaptations across all methods can be attributed to two key dataset characteristics. First, ACM's larger size (7410 nodes vs. DBLP's 5578) allows the source model to learn more comprehensive representations, facilitating easier adaptation to the smaller target domain. Second, ACM exhibits

2*Method	DBLP→ACM		ACM→DBLP		Cora→Cora-C	
	Macro F1	Micro F1	Macro F1	Micro F1	Macro F1	Micro F1
GraphTTA	57.55±1.38	64.70±1.06	73.89±5.69	73.11±5.45	37.29±17.75	37.29±17.75
HomoTTT	58.26±2.15	66.80±1.20	87.97±2.30	87.76±2.27	66.31±4.30	66.39±3.34
SOGA	61.96±0.89	69.10±1.04	94.21±0.13	93.96±0.14	32.08±10.47	19.31±9.94
Source	57.72±2.44	66.53±2.36	70.10±2.44	69.24±2.75	61.35±9.79	63.53 ±8.75

Table 3: Performance at the final epoch (epoch 10) of test-time adaptation across different datasets

significant label imbalance, with the largest category containing 2947 nodes compared to the smallest with 201 nodes. None of the evaluated methods explicitly address label imbalance during adaptation, leading to better performance when adapting from the imbalanced ACM to the more balanced DBLP dataset than vice versa.

7.2 Method-Specific Limitations

GraphTTA's performance is particularly sensitive to its min-max optimization framework. Analysis of gradient norms reveals a significant disparity between the augmenter and encoder components, making the augmenter training process challenging. This limitation becomes more pronounced in class-imbalanced scenarios, where the method's effectiveness heavily depends on initial pseudo-labels. The adaptation process begins with categorizing the target distribution using these potentially inaccurate initial predictions, which then determine the contrastive learning pairs. In label-imbalanced cases, this can lead to a negative feedback loop where the model increasingly favors dominant categories, reinforced through subsequent training iterations. The delicate balance required between augmenter and encoder learning rates, combined with the dependence on potentially faulty initial pseudo-labels, represents a fundamental limitation of the GraphTTA framework.

HomoTTT takes a different approach to contrastive learning by generating negative samples through feature shuffling rather than relying on pseudo-labels, which provides more reliable negative pairs. However, HomoTTT still faces challenges in class-imbalanced scenarios due to its reliance on k-means clustering for pseudo-label generation, which can be biased towards dominant classes. The method also requires careful tuning of the homophily threshold parameter, which can be dataset-dependent and difficult to optimize without access to target labels.

SOGA shows significant vulnerabilities to artificial feature corruption as evidenced by its poor performance on Cora-C. A key limitation stems from its information maximization loss, which encourages the model to make more confident predictions by minimizing prediction entropy. While this approach can enhance performance in natural domain shifts, it may lead to overconfident predictions on corrupted or highly shifted data. This entropy minimization strategy also explains SOGA's performance pattern in DBLP→ACM, where after initial improvement, the model's performance degrades in later epochs as it becomes increasingly confident in potentially incorrect predictions on the imbalanced target domain.

8 Conclusion

Our comparative analysis of test-time adaptation methods for GNNs reveals several key insights about their effectiveness and limitations. The methods demonstrate varying success across different domain adaptation scenarios, with each showing distinct strengths and vulnerabilities. SOGA excels in natural domain shifts but struggles with artificial feature corruption, while HomoTTT shows more robust but moderate improvements across different scenarios. GraphTTA's performance, while generally stable, is limited by its sensitivity to initial pseudo-labels and optimization challenges.

A critical finding is the significant impact of dataset characteristics on adaptation performance. The superior results in ACM→DBLP compared to DBLP→ACM highlight how source dataset size and label distribution influence adaptation success. None of the current methods explicitly address label imbalance during adaptation, suggesting an important direction for future work. Additionally, the performance degradation observed over extended adaptation periods points to the challenge of hyperparameter selection without access to target labels. These practical challenges also point to a deeper need for theoretical frameworks that can explain the convergence properties and limitations of different adaptation approaches, potentially leading to more principled design of test-time adaptation methods for GNNs.

References

[1] Yizhou Zhang, Guojie Song, Lun Du, Shuwen Yang, and Yilun Jin. Dane: Domain adaptive network embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4362–4368. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[2] Man Wu, Shirui Pan, Chuan Zhou, Xiaojun Chang, and Xingquan Zhu. Unsupervised Domain Adaptive Graph Convolutional Networks. In *Proceedings of The Web Conference 2020*, pages 1457–1467, Taipei Taiwan, April 2020. ACM.

[3] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully Test-Time Adaptation by Entropy Minimization. October 2020.

[4] Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. TTT++: When Does Self-Supervised Test-Time Training Fail or Thrive? In *Advances in Neural Information Processing Systems*, volume 34, pages 21808–21820. Curran Associates, Inc., 2021.

[5] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. In *Advances in Neural Information Processing Systems*, volume 33, pages 11539–11551. Curran Associates, Inc., 2020.

[6] Jian Liang, Dapeng Hu, and Jiashi Feng. Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6028–6039. PMLR, November 2020. ISSN: 2640-3498.

[7] Wei Jin, Tong Zhao, Jiayuan Ding, Yozen Liu, Jiliang Tang, and Neil Shah. Empowering Graph Representation Learning with Test-Time Graph Transformation, February 2023. arXiv:2210.03561.

[8] Yiqi Wang, Chaozhuo Li, Wei Jin, Rui Li, Jianan Zhao, Jiliang Tang, and Xing Xie. Test-Time Training for Graph Neural Networks, October 2022. arXiv:2210.08813.

[9] Guanzi Chen, Jiying Zhang, Xi Xiao, and Yang Li. GraphTTA: Test Time Adaptation on Graph Neural Networks, August 2022. arXiv:2208.09126.

[10] Haitao Mao, Lun Du, Yujia Zheng, Qiang Fu, Zelin Li, Xu Chen, Shi Han, and Dongmei Zhang. Source Free Graph Unsupervised Domain Adaptation. WSDM '24, pages 520–528, New York, NY, USA, March 2024. Association for Computing Machinery.

[11] Jiaxin Zhang, Yiqi Wang, Xihong Yang, and En Zhu. A Fully Test-time Training Framework for Semi-supervised Node Classification on Out-of-Distribution Graphs. *ACM Trans. Knowl. Discov. Data*, 18(7):172:1–172:19, June 2024.

[12] Man Wu, Shirui Pan, Chuan Zhou, Xiaojun Chang, and Xingquan Zhu. Unsupervised Domain Adaptive Graph Convolutional Networks. In *Proceedings of The Web Conference 2020*, pages 1457–1467, Taipei Taiwan, April 2020. ACM.

[13] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. ArnetMiner: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 990–998, New York, NY, USA, August 2008. Association for Computing Machinery.

[14] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting Semi-Supervised Learning with Graph Embeddings. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 40–48. PMLR, June 2016. ISSN: 1938-7228.