# Remedying Coarsening-Based GNN Training under Heterophily via Adaptive Complementary Enhancement

**Guoming Li**[1] **Jian Yang**[2] **Xukun Wang**[3] **Zixiao Wang**[4] **Shangsong Liang**[5] **Yifan Chen**[†1]

[1]Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China
[2]School of Information, Renmin University of China, Beijing, China
[3]Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China
[4]Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE
[5]School of Computer Science, Sun Yat-sen University, Guangzhou, China

## Abstract

Coarsening-based training for graph neural networks (GNNs), i.e. training on coarsened graphs rather than the original large ones, has become a promising direction for scaling GNNs to massive graphs. However, prior work has been evaluated almost exclusively on *homophilic* graphs, leaving the more challenging *heterophilic* settings underexplored. We show, both empirically and theoretically, that existing coarsening-based training methods suffer significant performance degradation on heterophilic graphs due to inevitable loss of graph information during coarsening. To address this, we propose **A**daptive **C**omplementary **E**nhancement, a plug-and-play, model-agnostic strategy that reintegrates the information discarded in coarsening: ACE learns a projector for re-constructing original node features and applies *anisotropic structural regularization* to embed local heterophily. We further adopt *homoscedastic uncertainty weighting* to adaptively balance the combined training objective of primary coarsened-graph training loss and full-graph auxiliary loss with augmented node features re-constructed by the heterophily-aware projector. Extensive experiments show that ACE drives consistent gains on heterophilic benchmarks while preserving competitive results on homophilic graphs with minimal computational overhead. Code is available at the GitHub repository: https://github.com/vasile-paskardlgm/ACE.

## 1 INTRODUCTION

Graph neural networks (GNNs) have emerged as powerful tools for capturing structural information from graph signals, achieving prominent performance across a broad spectrum of structured and graph-based applications [Wu et al., 2022]. However, as real-world graphs scale to millions of nodes and billions of edges, the computational complexity of mainstream propagation algorithms results in formidable training costs [Zhou et al., 2022]. In response, various scalable training strategies have been proposed, including *model-centric* graph sampling [Serafini and Guan, 2021] and *data-centric* graph reduction [Hashemi et al., 2024].

Among data-centric methods, **graph coarsening** has emerged as a widely adopted paradigm [Huang et al., 2021, Kumar et al., 2023b, Kataria et al., 2024, Xia et al., 2025, Dickens et al., 2024]. By training GNNs on a reduced graph to produce predictions for the original scale, these methods achieve significant efficiency. Unlike *graph condensation* [Jin et al., 2022], coarsening is *model-agnostic*, allowing a single coarsened graph to support a broad family of GNN architectures [Huang et al., 2021, Kataria et al., 2024].

Despite this empirical success, existing coarsening methods have been developed almost exclusively for **homophilic** graphs, leaving more challenging **heterophilic** graphs (distinct from *heterogeneous* graphs; Wang et al. 2023) largely unexplored [Gong et al., 2024]. In homophilic graphs, neighboring nodes tend to share similar labels or features, whereas this assumption breaks down in heterophilic graphs; therein neighbors often possess different labels or representations.

Our preliminary analysis (in Section 2.2) reveals that coarsening-based training suffers a much greater performance drop on heterophilic graphs relative to full-graph training, while the degradation on homophilic graphs is notably smaller. We turn to explain this disparity through the lens of mutual information, showing that graph heterophily significantly exacerbates the *mutual information gap* between models trained via coarsening and those trained on the full graph. These results suggest that heterophily poses a fundamental challenge to coarsening-based training.

In response, this work proposes **A**daptive **C**omplementary **E**nhancement (ACE), a plug-and-play, model-agnostic framework for enhancing coarsening-based GNN training

[†]Corresponding author

*Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence* (UAI 2026), PMLR 337:3738–3761.

under heterophily. The central principle of ACE is to reintegrate the graph information discarded during coarsening via a heterophily-aware auxiliary loss. In particular, ACE begins by learning a refined **projector** for re-constructing original node features with averaged supernode feature; this learning process is further features *anisotropic structural regularization* (c.f. Eq. (10)) for identifying heterophily. This regularization stems from anisotropic graph diffusion and were primarily adapted to supervised GNN training settings [Chamberlain et al., 2021, Elhag et al., 2022]; here, we formulate it as an unsupervised data augmentation mechanism to embed local heterophily into the reconstructed features as well as the projector.

The resulting projector lifts GNN predictions from the coarsened graph back to the full-graph scale to define an auxiliary loss with respect to the full-graph labels. To further stabilize training, we adopt *homoscedastic uncertainty* [Kendall et al., 2018] to adaptively weigh the auxiliary objective against the primary coarsening loss. ACE's modularity enables seamless integration into existing pipelines, where our empirical studies verify consistent performance gains. To summarize, our main contributions are as follows:

- We recognize the heterophily challenge in coarsening-based GNN training, and provide empirical evidence and theoretical explanations highlighting this underexplored limitation.
- We propose **A**daptive **C**omplementary **E**nhancement, a plug-and-play, model-agnostic appraoch that augments coarsening-based training with an heterophily-aware auxiliary loss.
- We conduct extensive experiments on large-scale heterophilic and homophilic benchmarks, demonstrating that ACE consistently improves coarsening training by a notable margin.

## 2 PRELIMINARIES AND CHALLENGES

We first introduce key notations and background knowledge, then identify the heterophily challenge in existing coarsening training via empirical and theoretical analysis.

### 2.1 NOTATIONS AND PRELIMINARIES

**Notations.** Let $\mathcal{G} = (A, X)$ represent a graph, where $A \in \{0,1\}^{n\times n}$ denotes the adjacency matrix and $X \in \mathbb{R}^{n\times d}$ is the node features/signals. The normalized graph Laplacian is defined as $L = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ [Chung, 1997], where $I$ is the identity matrix and $D$ is the degree matrix. Additionally, we denote $Y \in \{0,1\}^{n\times c}$ as the one-hot encoded label matrix, where $c$ is the number of node classes. On graph coarsening, we follow prior literature [Liu et al., 2018, Chen et al., 2022a], and let $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_{n'}$ be a partition of the node set into $n' < n$ disjoint clusters, where each cluster corresponds to a *supernode* in the coarsened graph. The partition matrix $P \in \{0,1\}^{n'\times n}$ satisfies $P_{i,j} = 1$ if node $v_j \in \mathcal{C}_i$. The coarsened graph $\mathcal{G}' = (A', X')$ is given by: $A' = PAP^T$, $X' = C^{-1}PX$, and $Y' = C^{-1}PY$, where $C = \text{diag}(|\mathcal{C}_1|, |\mathcal{C}_2|, ..., |\mathcal{C}_{n'}|)$.

**Coarsening-based GNN Training.** Following prior works (e.g., [Huang et al., 2021, Xia et al., 2025]), coarsening training seeks to improve efficiency by training on a coarsened graph $\mathcal{G}'$, while aiming to retain comparable downstream performance to models trained on the full graph $\mathcal{G}$. Formally, this paradigm can be defined as follows:

**Definition 2.1** (Coarsening-based GNN Training)**.** Let $f(;\Theta)$ denote a GNN with parameter set $\Theta$ and a softmax output layer. Coarsening-based GNN training minimizes the task loss on the coarsened graph: $\mathcal{L}(f(A', X'; \Theta), Y')$, which serves as a surrogate objective for the original graph loss: $\mathcal{L}(f(A, X; \Theta), Y)$.

In this work, we focus on the *node classification* task, where $\mathcal{L}$ instantiated as the *cross-entropy* loss, following prior studies [Huang et al., 2021, Kumar et al., 2023b, Kataria et al., 2024, Xia et al., 2025].

**Graph with Heterophily.** Heterophilic graphs—distinct from *heterogeneous* graphs [Wang et al., 2023]—have become a central focus in modern node-classification research [Zheng et al., 2024, Gong et al., 2024]. Unlike homophilic graphs, where neighboring nodes typically share similar labels or attributes, heterophilic graphs connect dissimilar nodes, violating key assumptions behind message-passing GNNs and often exacerbating *over-smoothing* effects [Rusch et al., 2023, Yan et al., 2022]. Such structures commonly arise in real-world networks (e.g., financial, communication, and certain molecular graphs) and pose distinct challenges for GNN architectures [Lim et al., 2021, Platonov et al., 2023]. Despite their importance, the impact of heterophily has remained largely unexamined in coarsening-based GNN training, leaving a notable gap that we explore in this work.

**Dirichlet Energy on Graphs.** Let $E$ be the edge set of graph $\mathcal{G}$. The Dirichlet energy of node features $X$ is typically defined as

$$\mathcal{E}(\mathcal{G}) = \frac{1}{2}\sum_{(i,j)\in E} A_{ij}\|\frac{X_i}{\sqrt{D_{ii}}} - \frac{X_j}{\sqrt{D_{jj}}}\|_2^2 = \text{trace}(X^\top LX)\,. \tag{1}$$

Dirichlet energy serves as a fundamental descriptor of smoothness over the graph and is central to understanding diffusion, propagation, and convolution behavior on graphs [Giovanni et al., 2023, Han et al., 2023]. This concept closely tied to the aforementioned key phenomena: heterophily, where label signals exhibit high Dirichlet energy [Maskey et al., 2023], and over-smoothing, where repeated graph propagation drives the energy of node representations toward zero [Cai and Wang, 2020].

### 2.2 HETEROPHILY CHALLENGE

**Empirical Observations.** We examine how existing coarsening-based GNN training methods behave under heterophily. Specifically, we evaluate three representative approaches—SCAL [Huang et al., 2021], FGC [Kumar et al., 2023b], and SGBGC [Xia et al., 2025]—using a 10% coarsening ratio across large heterophilic graphs (Genius, Gamers, Wiki [Lim et al., 2021]) and homophilic graphs (Ogbn-arxiv, Ogbn-products [Hu et al., 2020]). The experiments are conducted with three backbone GNNs: GCN [Kipf and Welling, 2017], LINKX [Lim et al., 2021], and GloGNN [Li et al., 2022], where the latter two are explicitly designed for heterophilic settings. Full details follow Section 4.1.

Results in Table 1 reveals a striking trend: **all** coarsening-based methods suffer substantially larger performance degradation on heterophilic graphs compared to homophilic ones—often nearly **four times** larger—and this occurs consistently across all backbone models, including heterophily-specialized architectures (LINKX and GloGNN).

Notably, these heterophily-oriented models exhibit even larger performance drops, underscoring that the difficulty stems not from the model itself but from a more **fundamental limitation of coarsening-based training under heterophilic settings**. More empirical results supporting this observation are presented in the main experiments in Section 4 and Appendix D.4.

**Theoretical Explanation.** The empirical results on heterophilic graphs reveal a pronounced and consistent performance gap between models trained on the coarsened graph and those trained on the full graph, indicating that the degradation arises from the coarsening process itself. Motivated by this observation, we study the heterophily challenge from an information-theoretic perspective to understand how and why this gap arises.

Let $\mathcal{G} \setminus \mathcal{G}'$ denote the *discarded graph information* during coarsening, consisting of the intra-supernode structural details (illustrated in Figure 1). Using this notion, we characterize the discrepancy between the full-graph–trained model and the coarsened-graph–trained model by quantifying the mutual information between $\mathcal{G} \setminus \mathcal{G}'$ and the label $Y$, leading to the following proposition:

**Proposition 2.2.** *Let the GNN trained on the original graph be $f(;\Theta^{**})$ and the one trained on the coarsened graph be $f(;\Theta^{*})$, where*

$$\Theta^{*} \triangleq \arg\min_{\Theta} \mathcal{L}(f(A, X; \Theta), Y) \,, \tag{2}$$

$$\Theta^{**} \triangleq \arg\min_{\Theta} \mathcal{L}(f(A', X'; \Theta), Y') \,. \tag{3}$$

*When both models are evaluated on the original graph, the mutual information between their outputs satisfies*

$$I(f(A, X; \Theta^{*}); f(A, X; \Theta^{**})) \leq I(f(A, X; \Theta^{*}); Y) - \Omega \,, \tag{4}$$

*where $\Omega = I(g(\mathcal{G} \setminus \mathcal{G}'); Y | f(A, X; \Theta^{**}))$, and $g : \mathcal{G} \setminus \mathcal{G}' \mapsto \mathcal{Y}$ is arbitrary neural network that maps $\mathcal{G} \setminus \mathcal{G}'$ to the node label space $\mathcal{Y}$ and maximizes the conditional mutual information $I(g(\mathcal{G} \setminus \mathcal{G}'); Y | f(A, X; \Theta^{**}))$.*

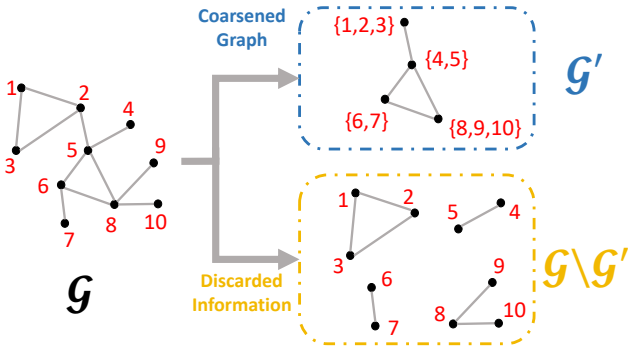



Figure 1: Illustration for graph coarsening: $\mathcal{G}'$ encodes inter-supernode relations; $\mathcal{G} \setminus \mathcal{G}'$ holds intra-supernode information discarded during coarsening.

Proof is provided in Appendix A. This proposition provides an **explanatory clarification** by linking the model performance gap to the information loss induced by graph coarsening. In the bound of Eq. (4), the gap between the two models is expressed by the mutual information $I(f(A, X; \Theta^{*}); f(A, X; \Theta^{**}))$, which measures how similar the outputs of the full-graph-trained model and the coarsened-graph-trained model are. Since the full-graph model $f(A, X; \Theta^{*})$ is unaffected by coarsening, the term $I(f(A, X; \Theta^{*}); Y)$ remains fixed. Therefore, the magnitude of the model gap is determined entirely by the quantity $\Omega = I(g(\mathcal{G} \setminus \mathcal{G}'); Y | f(A, X; \Theta^{**}))$, which refers to the graph information lost during coarsening. A larger $\Omega$ forces $I(f(A, X; \Theta^{*}); f(A, X; \Theta^{**}))$ to be smaller, producing a larger discrepancy between the full-graph and coarsened-graph models. In heterophilic graphs, the discarded region $\mathcal{G} \setminus \mathcal{G}'$ contains diverse labels and informative local structure; thus making $\Omega$ large. Consequently, the mutual information between the two models becomes small, yielding the substantial performance drop observed empirically. In contrast, homophilic graphs discard much less label-relevant content, resulting in a much smaller gap.

## 3 PROPOSED METHOD: ADAPTIVE COMPLEMENTARY ENHANCEMENT

We now present our proposed method, Adaptive Complementary Enhancement (ACE). We begin with a motivating toy experiment and then introduce the framework in detail.

Table 1: Impact of heterophily on coarsening-based training. "-" denotes full-graph training, and $\Delta \downarrow$ indicates the average performance drop of coarsening training relative to full-graph training for each dataset.

| Backbone | GCN | | | | LINKX | | | | GloGNN | | | | $\Delta \downarrow$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | - | SCAL | FGC | SGBGC | - | SCAL | FGC | SGBGC | - | SCAL | FGC | SGBGC | |
| Genius | 87.42 | 67.47 | 70.13 | 71.22 | 90.77 | 60.62 | 65.49 | 62.68 | 90.66 | 59.47 | 63.81 | 67.33 | **26.26** |
| Gamers | 62.18 | 47.39 | 40.14 | 44.62 | 66.06 | 50.13 | 42.48 | 47.60 | 66.19 | 46.28 | 45.92 | 47.15 | **19.07** |
| Pokec | 75.45 | 50.44 | 53.81 | 55.48 | 82.04 | 56.48 | 57.63 | 54.18 | 83.00 | 61.58 | 66.91 | 63.77 | **22.35** |
| Arxiv | 71.74 | 63.42 | 65.52 | 64.12 | 69.54 | 57.71 | 59.38 | 61.55 | 72.68 | 57.09 | 61.73 | 61.24 | 7.79 |
| Products | 75.64 | 68.37 | 71.28 | 71.96 | 74.59 | 62.49 | 67.52 | 64.93 | 77.48 | 66.53 | 70.46 | 69.11 | 6.50 |

Table 2: Toy experiment results. Performance gains over the vanilla coarsening-based training are highlighted in **bold**.

| Model | Method | Genius | Gamers | Pokec |
|---|---|---|---|---|
| GCN | SCAL | 67.47 | 47.39 | 50.44 |
| | w/ $\tilde{\mathcal{L}}$ | 71.32 (**+3.85**) | 52.86 (**+5.47**) | 55.29 (**+4.85**) |
| | FGC | 70.13 | 40.14 | 53.81 |
| | w/ $\tilde{\mathcal{L}}$ | 75.67 (**+5.54**) | 49.26 (**+9.12**) | 57.22 (**+3.41**) |
| | SGBGC | 71.22 | 44.62 | 55.48 |
| | w/ $\tilde{\mathcal{L}}$ | 73.81 (**+2.59**) | 53.09 (**+8.47**) | 62.42 (**+6.94**) |
| GloGNN | SCAL | 59.47 | 46.28 | 61.58 |
| | w/ $\tilde{\mathcal{L}}$ | 67.41 (**+7.94**) | 53.11 (**+6.83**) | 67.74 (**+6.16**) |
| | FGC | 63.81 | 45.92 | 66.91 |
| | w/ $\tilde{\mathcal{L}}$ | 72.33 (**+8.52**) | 51.26 (**+5.34**) | 71.44 (**+4.53**) |
| | SGBGC | 67.33 | 47.15 | 63.77 |
| | w/ $\tilde{\mathcal{L}}$ | 73.26 (**+5.93**) | 54.31 (**+7.17**) | 69.36 (**+5.59**) |

### 3.1 MOTIVATING TOY EXPERIMENT: A SIMPLE AUXILIARY LOSS MATTERS

As demonstrated in Section 2.2, the heterophily challenge primarily arises from the loss of graph information during coarsening. A natural solution is therefore to **reintegrate the discarded information during training**. With this insight, we conduct a simple toy experiment: we augment the training via coarsening pipeline with an auxiliary loss that addresses the coarsening matrix $P^\top C$ and the original label $Y$. The resulting unified loss function is stated below, where $\lambda$ denotes the trade-off hyperparameter.

$$\tilde{\mathcal{L}} = \underbrace{\mathcal{L}\left(f(A', X'; \Theta), Y'\right)}_{\text{Primary Coarsened-Graph Training Loss}} + \lambda \cdot \underbrace{\mathcal{L}\left(P^\top C \cdot f(A', X'; \Theta), Y\right)}_{\text{Auxiliary Loss}} . \quad (5)$$

We follow the same experimental setup as in Section 2.2 and compare the augmented loss (5) against standard coarsening-based training pipelines. Remarkably, as shown in Table 2, the introduction of the simple auxiliary loss leads to substantial and consistent improvements across all coarsening-based methods and across diverse GNN backbones on heterophilic graphs. A more comprehensive investigation in Appendix C reveals that the auxiliary loss benefits homophilic graphs as well. In summary, these positive results highlight the strong potential of the auxiliary loss that compensates for information discarded during coarsening.

### 3.2 THE ACE FRAMEWORK

Motivated by the observations from the toy experiments, we develop a heterophily-aware enhancement to coarsening-based GNN training by incorporating an auxiliary loss. This leads to our proposed framework, ACE, which consists of two core modules: (*i*) ***Refined Projector Construction*** and (*ii*) ***Training With Unified Coarsening Loss***.

#### 3.2.1 Step i: Refined Projector Construction

**Parameterized Projector.** Although the simple auxiliary loss enhance training efficacy, its construction relies on the coarsening projector $P$, which is entirely dictated by the chosen graph coarsening algorithm: either classical [Purohit et al., 2014, Loukas and Vandergheynst, 2018, Jin et al., 2020, Fahrbach et al., 2020, Chen et al., 2023], or learnable [Kumar et al., 2023a,b, Joly and Keriven, 2024, Xia et al., 2025]. None of existing designs specifically incorporate heterophily-related cues into the construction of the so-called projector. We therefore seek to empower the projector with heterophily-awareness.

To this end, we first construct the *structural affinity* matrix $S$ and the *feature affinity* matrix $F$, both in $\mathbb{R}^{n \times n'}$:

$$S = AC^{-1}P\,, \quad (6)$$

$$F_{ij} = \exp(-\|X_i - \mu_j\|^2)\;;\; \mu_j = \frac{1}{|\mathcal{C}_j|}\sum_{i \in \mathcal{C}_j} X_i\,. \quad (7)$$

Following the intuition of *Algebraic Multigrid* (AMG; Ruge and Stüben), we note the term $S_{ij}$ measures how strongly node $i$ connects to supernode $j$, and the term $F_{ij}$ is a *Kernel*

*Regression* similarity [Nadaraya, 1964] between node $i$ and the centroid $\mu_j$. Together, they encode structural and feature connections while preserving coarsening structure. Using these two matrices, we devise a learnable, heterophily-aware projector:

$$P_{ij}(\phi) = \frac{\exp(\alpha_{ij})}{\sum_{k\in\mathcal{N}(i)}\exp(\alpha_{ik})}\,, \tag{8}$$

$$\alpha_{ij} := f_\phi([S_{ij}, F_{ij}])\,, \tag{9}$$

where $f_\phi$ is an MLP parametrized with $\phi$. Analogous to graph attention mechanism [Veličković et al., 2018], $P(\phi)$ learns how each fine node should associate with each supernode, balancing structural and feature signals through $\phi$.

**Heterophily-Aware Optimization.** To ensure that $P(\phi)$ correctly encodes heterophily, we study principled learning of the projector $P$. Conventional attributed graph coarsening specifies the projector through optimizing a combination of Dirichlet energy $\mathcal{E}(\mathcal{G})$ and feature reconstruction $\|X - PP^\top X\|^2$ [Kumar et al., 2023b, Kataria et al., 2024], thereby promoting smooth GNN predictions—an inductive bias suited for homophilic graphs. However, such objectives are inapplicable under heterophily, where neighboring nodes often carry contrasting information.

To overcome this issue, we draw inspiration from [Fu et al., 2023], which constructs a heterophily-aware Dirichlet energy via *anisotropic diffusion* [Perona and Malik, 1990], and propose the **Anisotropic Structural Regularization** term, defined as follows:

$$\mathcal{J}_{\text{ASR}}(\phi) = \underbrace{\sum_{i,j\in E}\omega_{ij}\cdot\|(P(\phi)\boldsymbol{\mu})_i - (P(\phi)\boldsymbol{\mu})_j\|^2}_{\text{Anisotropic Smoothness}} + \beta\cdot\underbrace{\|X - P(\phi)\boldsymbol{\mu}\|_F^2}_{\text{Reconstruction Cost}} \tag{10}$$

where $\omega_{ij} := \exp(-\|X_i - X_j\|^2)$, $\boldsymbol{\mu}$ represents the stacked centroids $\{\mu_1, \mu_2, ..., \mu_{n'}\}$ (see Eq. (7)), and $\beta$ is the trade-off coefficient.

The core idea behind this design is the superior ability of graph anisotropic diffusion to model heterophilic relationships in graphs, which regular isotropic diffusion adopted in most GNNs cannot capture [Eliasof et al., 2021, Chen et al., 2022b, Dan et al., 2023, Han et al., 2023]. Notably, prior methods apply anisotropic diffusion in supervised GNN training, whereas we use it in an **unsupervised** manner, relying solely on the graph data. Despite lack of supervision, minimizing the proposed regularization term $\mathcal{J}_{\text{ASR}}(\phi)$ ensures that the projector $P(\phi^*)$ is *heterophily-aware*, as shown in the following proposition:

**Proposition 3.1.** *Let $\phi^*$ denote the minimizer of $\mathcal{J}_{ASR}(\phi)$. The resulting projector $P(\phi^*)$ acts as a conditional spectral filter that interpolates between two distinct regimes below based on local homophily:*

- ***Heterophilic Regime:*** *In regions where $\|X_i - X_j\| \to \infty$, $P(\phi^*)$ approaches the solution of identity reconstruction, prioritizing high-frequency signal fidelity over structural smoothness.*
- ***Homophilic Regime:*** *In regions where $\|X_i - X_j\| \to 0$, $P(\phi^*)$ approaches the solution of isotropic laplacian smoothing, enforcing local smoothness.*

The proof is provided in Appendix B. This proposition provides a **local optimality characterization** of the learned projector, demonstrating that ACE naturally adapts to both homophilic and heterophilic graph structures: In heterophilic regions, where $\|X_i - X_j\| \to \infty$, the weights $\omega_{ij}$ vanish, suppressing the smoothness term and allowing the projector to prioritize high-frequency, heterophily-specific information; In homophilic regions, where $\|X_i - X_j\| \to 0$, the smoothness constraint remains active, encouraging projector to preserve locally consistent, low-frequency structure.

By minimizing Eq. (10) with standard optimizers such as Adam [Kingma and Ba, 2014], we can easily learn the projector $P(\phi^*)$ adaptive to heterophilic graphs.

### 3.2.2 Step ii: Training With Joint Coarsening Loss

The optimized projector $P(\phi^*)$ can then be used to construct the auxiliary loss term in Eq. (5). However, the combined loss in Eq. (5) is inherently inflexible: it relies on a manually chosen tradeoff hyperparameter, which does not adapt to the dynamics of training and may vary across datasets or backbones. To overcome this inconvenience and better leverage the supervision signals available in coarsening-based GNN training, we further enhance the primitive combined loss (5) by employing *Homoscedastic Uncertainty weighting* [Kendall et al., 2018]. This technique interprets the primary and auxiliary objectives as a *multi-task learning* problem [Zhang and Yang, 2022], allowing their relative contributions to be balanced automatically via learnable task uncertainties. The resulting ACE objective is:

$$\mathcal{L}_{\text{ACE}} = \frac{1}{2\sigma_1^2}\underbrace{\mathcal{L}\left(P(\phi^*)^\top C\cdot f(A', X'; \Theta), Y\right)}_{\text{Auxiliary Loss}} + \frac{1}{2\sigma_2^2}\underbrace{\mathcal{L}\left(f(A', X'; \Theta), Y'\right)}_{\text{Primary Loss}} + \log(\sigma_1\sigma_2)\,, \tag{11}$$

where $\sigma_1$ and $\sigma_2$ are learnable uncertainty parameters that dynamically adjust weights of the two losses during training.

### 3.2.3 Complexity Analysis

We analyze the computational overhead introduced by ACE relative to vanilla training via coarsening, focusing on the cost of constructing the refined projector and the additional terms incurred by the ACE optimization framework. We follow the notations in Section 2.1.

**Complexity of Refined Projector Construction.** Constructing the refined projector $P(\phi^*)$ consists of a light pre-computation stage and an iterative optimization over $T$ epochs. The pre-computation refers to calculating $\omega_{ij}$ over all fine-level edges in $\mathcal{O}(|E|d)$ time, and forming the structure and feature affinities restricted by the coarsening assignment $\{\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_{n'}\}$ with $\mathcal{O}(nd)$ cost. An optimization step then includes a forward pass of the parametric MLP map $f_\phi$ over the sparse candidate pairs with $\mathcal{O}(n)$ cost; the sparse lifting operation $(P(\phi)\boldsymbol{\mu}$ are then performed with $\mathcal{O}(nd)$ cost; ultimately, obtaining the reconstruction and anisotropic smoothness terms costs $\mathcal{O}(nd)$ and $\mathcal{O}(|E|d)$ time, respectively.

Aggregating all the components, the total complexity of refined projector construction is $\boldsymbol{\mathcal{O}(T(nd+|E|d))}$, which is linear in the graph size and remains lightweight compared with a standard GNN layer.

**Overhead in Training with Joint Coarsening Loss.** Training with the ACE loss is compatible with the standard coarsened-graph pipeline on $\mathcal{G}'$, while adding a step to compute the auxiliary loss by lifting the coarsened logits back to the fine graph. This lifting step incurs an additional cost of approximately $\boldsymbol{\mathcal{O}(nc)}$ time. Since the feature dimension $d$ is typically much larger than the number of classes $c$, this overhead is negligible relative to the dominant cost of vanilla coarsening-based training, which is on the order of $\mathcal{O}(nd^2 + |E'|d)$ (let $E'$ be the edge set of $\mathcal{G}'$).

**Remark.** In summary, ACE improves performance by releveraging fine-grained supervision while maintaining the efficiency advantage of coarsening-based training. The inference complexity remains identical to the baseline, as the auxiliary branch is discarded after training. This claim is verified in the following section, where ACE is shown to improve performance with minimal computational overhead.

# 4 EXPERIMENTS

We conduct extensive experiments to evaluate the effectiveness of ACE across diverse coarsening-based GNN training pipelines. Our evaluation covers both large-scale heterophilic and homophilic benchmarks and multiple GNN backbones. We organize the study around the following research questions:

- **RQ1:** How much does ACE improve performance over vanilla coarsening training pipelines?
- **RQ2:** How does the refined projector $P(\phi^*)$ influence training behavior?
- **RQ3:** How does ACE influence the scalability and efficiency of coarsening-based training?
- **RQ4:** How does ACE narrow the performance gap between conventional methods and state-of-the-art ones?

## 4.1 EXPERIMENTAL SETUP

We evaluate four state-of-the-art coarsening-based GNN training pipelines: SCAL [Huang et al., 2021], FGC [Kumar et al., 2023b], UGC [Kataria et al., 2024], and SG-BGC [Xia et al., 2025]. For our ACE, we parameterize $f_\phi$ as a two-layer MLP with 500 hidden dimensions and tune the trade-off coefficient $\beta$ from {0.1,1,10} during projector optimization.

We use five large-scale heterophilic datasets from [Lim et al., 2021]: Genius, Gamers, Pokec, Snap, and arXiv-year, and two large homophilic datasets from the OGB [Hu et al., 2020]: Ogbn-arXiv and Ogbn-products. Full dataset statistics are provided in Appendix D.1.

We consider six GNN backbones: GCN [Kipf and Welling, 2017], LINKX [Lim et al., 2021], GLOGNN [Li et al., 2022], GPRGNN [Chien et al., 2021], SGFormer [Wu et al., 2023], and Polynormer [Deng et al., 2024], covering models specialized for heterophily, homophily, or both. **Due to space constraints**, we report results for three backbones (GCN, LINKX, and GLOGNN) in the main text, and **defer the remaining results to Appendix D.4**.

We set a 10% coarsening ratio for all pipelines and finetune based on the recommended configurations from the original papers and benchmark repositories. All models are trained for 500 epochs using the Adam optimizer [Kingma and Ba, 2014], with early stopping triggered after 50 epochs of no validation improvement. Training follows a consistent configuration across all methods: learning rate 0.01, weight decay $5e-4$, and dropout 0.5. All reported numbers are averaged over 10 independent runs.

Additional training details and complementary experimental results are provided in Appendix D.

## 4.2 RQ1: PERFORMANCE GAINS FROM ACE

We examine how ACE improves the established coarsening training pipelines. As shown in Tables 3 and 8, adding ACE yields consistent and significant gains across all backbones and datasets. The gains are notably striking on heterophilic graphs—where coarsening tends to discard more essential graph information—**with increases of up to 14.86**. These confirm that ACE markedly enhances the efficacy of coarsening training, especially under heterophilic conditions.

Beyond these overall improvements, ACE also significantly outperforms the naive auxiliary loss used in the toy experiments. By using the refined projector, ACE achieves **roughly twice the performance gain** of the naive formulation. This highlights the value of our refinement: it not only strengthens the effect of reintegrating the discarded information but also validates the central premise that restoring lost information is essential for handling heterophily in

Table 3: Performance of integrating ACE into GCN, LINKX, and GloGNN. Improvements on heterophilic datasets are highlighted in **bold**. Standard deviations are reported in Table 7; results for the remaining pipelines are provided in Table 8.

| Backbone | Method | Heterophilic Graphs | | | | | Homophilic Graphs | |
|---|---|---|---|---|---|---|---|---|
| | | Genius | Gamers | Pokec | Snap | arXiv-year | ogbn-arXiv | ogbn-products |
| GCN | - | 87.42 | 62.18 | 75.45 | 45.65 | 46.02 | 71.74 | 75.64 |
| | SCAL | 67.47 | 47.39 | 50.44 | 22.66 | 27.59 | 63.42 | 68.37 |
| | + ACE | 77.63 (**+10.16**) | 55.63 (**+8.24**) | 58.12 (**+7.68**) | 33.47 (**+10.81**) | 34.53 (**+6.94**) | 66.03 (+2.61) | 70.38 (+2.01) |
| | FGC | 70.13 | 40.14 | 53.81 | 23.06 | 25.49 | 65.52 | 71.28 |
| | + ACE | 77.32 (**+7.19**) | 52.52 (**+12.38**) | 61.67 (**+7.86**) | 32.73 (**+9.67**) | 31.91 (**+6.42**) | 67.02 (+1.50) | 72.47 (+1.19) |
| | UGC | 69.37 | 42.70 | 51.74 | 21.49 | 29.18 | 63.06 | 68.92 |
| | + ACE | 75.28 (**+5.91**) | 53.83 (**+11.13**) | 59.37 (**+7.63**) | 30.82 (**+9.33**) | 33.49 (**+4.31**) | 64.87 (+1.81) | 70.19 (+1.27) |
| | SGBGC | 71.22 | 44.62 | 55.48 | 25.16 | 27.02 | 64.12 | 71.96 |
| | + ACE | 76.26 (**+5.04**) | 56.28 (**+11.66**) | 64.04 (**+8.56**) | 31.93 (**+6.77**) | 33.75 (**+6.73**) | 66.22 (+2.10) | 73.13 (+1.17) |
| LINKX | - | 90.77 | 66.06 | 82.04 | 61.95 | 56.00 | 69.54 | 74.59 |
| | SCAL | 60.62 | 50.13 | 56.48 | 40.13 | 42.66 | 57.71 | 62.49 |
| | + ACE | 73.29 (**+12.67**) | 57.11 (**+6.98**) | 65.77 (**+9.29**) | 47.64 (**+7.51**) | 47.18 (**+4.52**) | 61.02 (+3.31) | 64.84 (+2.35) |
| | FGC | 65.49 | 42.48 | 57.63 | 38.17 | 44.37 | 59.38 | 67.52 |
| | + ACE | 74.71 (**+9.22**) | 52.33 (**+9.85**) | 70.47 (**+12.84**) | 49.08 (**+10.91**) | 47.65 (**+3.28**) | 61.73 (+2.35) | 69.21 (+1.69) |
| | UGC | 63.61 | 44.55 | 52.38 | 41.22 | 41.49 | 55.46 | 64.37 |
| | + ACE | 71.08 (**+7.47**) | 50.69 (**+6.14**) | 67.24 (**+14.86**) | 47.33 (**+6.11**) | 46.21 (**+4.72**) | 60.28 (+4.82) | 66.52 (+2.15) |
| | SGBGC | 62.68 | 47.60 | 54.18 | 42.91 | 45.42 | 61.55 | 64.93 |
| | + ACE | 71.53 (**+8.85**) | 56.48 (**+8.88**) | 63.24 (**+9.06**) | 46.47 (**+3.56**) | 49.02 (**+3.60**) | 63.88 (+2.33) | 66.87 (+1.94) |
| GloGNN | - | 90.66 | 66.19 | 83.00 | 62.09 | 54.68 | 72.68 | 77.48 |
| | SCAL | 59.47 | 46.28 | 61.58 | 43.84 | 40.91 | 57.09 | 66.53 |
| | + ACE | 72.29 (**+12.82**) | 55.46 (**+9.18**) | 70.34 (**+8.76**) | 49.28 (**+5.44**) | 46.26 (**+5.35**) | 59.20 (+2.11) | 67.91 (+1.38) |
| | FGC | 63.81 | 45.92 | 66.91 | 42.70 | 43.28 | 61.73 | 70.46 |
| | + ACE | 74.77 (**+10.96**) | 53.42 (**+7.50**) | 74.27 (**+7.36**) | 50.38 (**+7.68**) | 46.62 (**+3.34**) | 63.78 (+2.05) | 71.93 (+1.47) |
| | UGC | 61.49 | 42.96 | 64.04 | 45.61 | 40.36 | 60.29 | 68.74 |
| | + ACE | 73.68 (**+12.19**) | 51.77 (**+8.81**) | 72.47 (**+8.43**) | 49.71 (**+4.10**) | 45.24 (**+4.88**) | 63.21 (+2.92) | 70.87 (+2.13) |
| | SGBGC | 67.33 | 47.15 | 63.77 | 45.26 | 41.27 | 61.24 | 69.11 |
| | + ACE | 78.43 (**+11.10**) | 56.29 (**+9.14**) | 74.55 (**+10.78**) | 52.82 (**+7.56**) | 47.50 (**+6.23**) | 63.63 (+2.39) | 72.18 (+3.07) |

coarsening-based training.

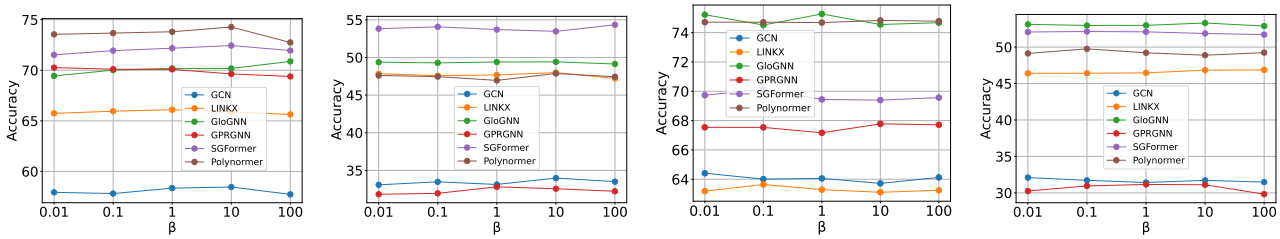
Moreover, ACE also benefits other graph-reduction methods beyond coarsening, such as *graph condensation* [Jin et al., 2022] (Table 9), and remains robust even under **extreme coarsening ratio** (Table 10), highlighting its generality and broad applicability.

### 4.3 RQ2: IMPACT OF THE REFINED PROJECTOR

The optimization of Eq. (10) for obtaining $P(\phi^*)$ is governed by a trade-off hyperparameter $\beta$, which balances the reconstruction and anisotropic regularization terms and thus shapes the refined projector. To evaluate how sensitive the learned projector—and consequently the coarsening training—is to this choice, we conduct an ablation study over a wide range of $\beta$ within $\{0.01,0.1,1,10,100\}$. We perform this analysis using two strong pipelines (SCAL and SGBGC) and two large heterophilic benchmarks (Pokec and Snap). Additional results for UGC and FGC are provided in Appendix D.4.

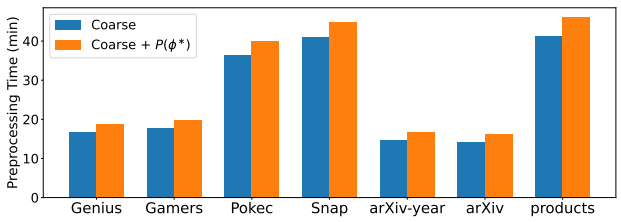
Figure 2 (and Figure 5) showcase that varying $\beta$ across two orders of magnitude produces only minor performance differences, consistently across datasets and backbone GNNs. This highlights that the learning dynamics of the refined projector $P(\phi^*)$ are highly robust to the choice of $\beta$. Although occasional small fluctuations can be observed, the overall trend remains stable, and setting $\beta$ be 1 already yields superior performance without further tuning. To sum up, such insensitivity to hyperparameter choice makes ACE **practically convenient**, requiring minimal effort to deploy while preserving strong performance across diverse settings.

**Beyond the Projection Module.** Readers are encouraged to consult the additional ablation studies in Appendix D.4 (Tables 12 and 13), which investigate the remaining components of ACE and offer an extensive understanding of their isolated contributions beyond the projector module.

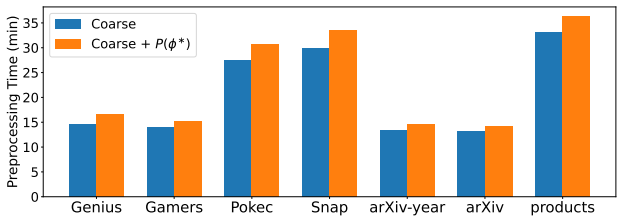



(a) SCAL on Pokec. (b) SCAL on Snap. (c) SGBGC on Pokec. (d) SGBGC on Snap.

Figure 2: Effect of $\beta$ across coarsening pipelines, GNN backbones, and datasets. **More results can be found in Figure 5.**

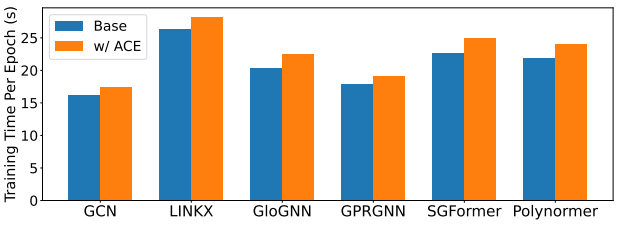



(a) Preprocessing time: SCAL vs. SCAL+ACE

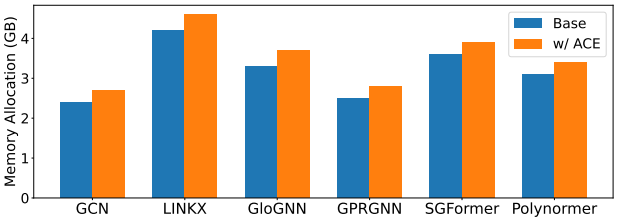



(b) Preprocessing time: SGBGC vs. SGBGC+ACE

Figure 3: Comparison of preprocessing time for SCAL, SGBGC, and their ACE-enhanced variants. **Complete results are provided in Figure 6.**




(a) SCAL training time per epoch on Pokec.




(b) SCAL training memory usage on Pokec.

Figure 4: Training memory usage and per-epoch runtime (on Pokec) of SCAL and SCAL+ACE across diverse backbone GNNs. **More results are provided in Tables 7 and 8.**

### 4.4 RQ3: EFFECTS OF ACE ON EFFICIENCY AND RESOURCE CONSUMPTION

**Preprocessing Overhead.** Coarsening training pipelines require a preprocessing stage to construct coarsened graphs, typically incurring substantial cost (often on the order of $n \cdot E$). ACE adds one additional step—learning the refined projector $P(\phi^*)$—but, as discussed in Section 3.2.3, this operation scales only with $n + E$ level and is therefore lightweight relative to the core coarsening procedure.

Figure 3 (with full results in Figure 6) confirms this analysis: ACE introduces only a negligible increase in preprocessing time across all pipelines and datasets. This validates that the refined projector construction is highly efficient, allowing ACE to improve coarsened representations without altering the overall preprocessing cost profile.

**Training Overhead.** We next assess the runtime and memory efficiency of ACE during model training, using the large-scale Pokec dataset and all six backbone GNNs. As shown in Figure 4 (with additional results in Figures 7 and 8), ACE adds only a small overhead: approximately **4%–7%** in both training time per epoch and GPU memory usage. Given that ACE delivers over 10% accuracy improvement on challenging heterophilic graphs, this modest overhead represents a highly favorable trade-off and highlights the practicality of integrating ACE into existing coarsening-based pipelines.

### 4.5 RQ4: COMPETITIVENESS OF CONVENTIONAL METHODS WITH ACE

To assess the position of ACE among existing scalable and efficient GNN training paradigms, we compare ACE-enhanced conventional coarsening methods—specifically FGC and UGC—with representative state-of-the-art approaches beyond coarsening, including the sampling-based method AGS [Das et al., 2024] and the graph condensation method GECC Gong et al. [2026]. Results using the GCN backbone are presented here, while those with GPRGNN backbone are deferred to Appendix D.4.

Table 4: Comparison with state-of-the-art scalable GNN training methods based on graph data reduction. Results on a **GCN** backbone with a 10% reduction ratio are reported in terms of test accuracy and total training time. **Complementary results are deferred to Table 11.**

| Category | Method | Pokec | | Snap | | Gamers | | Ogbn-product | |
|---|---|---|---|---|---|---|---|---|---|
| | | ACC | Time (min) | ACC | Time (min) | ACC | Time (min) | ACC | Time (min) |
| Coarsening | FGC | 53.81 | 24.73 | 23.06 | 31.66 | 40.14 | 12.39 | 71.28 | 32.71 |
| | UGC | 51.74 | 23.69 | 21.49 | 29.19 | 42.70 | 11.23 | 68.92 | 31.41 |
| Data-Adaptive | GECC | 60.66 | 33.82 | 29.31 | 38.77 | 50.66 | 18.43 | 71.69 | 45.29 |
| Subgraph-Sampling | AGS | 57.32 | 21.81 | 28.32 | 29.48 | 48.81 | 10.13 | 71.02 | 30.18 |
| **Ours** | FGC+ACE | **61.67** | 25.67 | **32.73** | 32.92 | **52.52** | 12.89 | **72.47** | 33.65 |
| | UGC+ACE | 59.37 | 24.34 | **30.82** | 31.25 | **53.83** | 11.72 | 70.19 | 32.24 |

As shown in Table 4 (with additional results in Table 11), the conventional coarsening methods FGC and UGC substantially underperform state-of-the-art approaches when used alone. However, after incorporating ACE, both methods exhibit significant performance gains, achieving results that are comparable to, and in several cases surpass, those of the state-of-the-art baselines. These findings suggest that ACE can substantially enhance the competitiveness of conventional coarsening-based training methods. Furthermore, ACE introduces only negligible computational overhead relative to the compared baselines, resulting in a more favorable performance-cost trade-off. Overall, the results highlight ACE as a practical and promising direction for revitalizing conventional coarsening-based training methods in the era of scalable graph learning.

## 5 CONCLUSION

**Paper Summary.** We study how graph heterophily affects GNN performance in coarsening-based GNN training. Guided by empirical observations and supported by our theoretical analysis, we interpret the degradation as that for heterophilic graphs coarsening discards more essential graph information than for homophilic graphs, which is an underexplored perspective in prior research. To address this issue, we introduced ACE, which first learns a heterophily-aware refined projector via Anisotropic Structural Regularization and incorporates it into an auxiliary loss; we then combine the designs and propose an improved optimization objective via Homoscedastic Uncertainty weighting, thus adaptively reintegrating discarded information to enhance coarsening-based training. Across diverse coarsening training pipelines, benchmarks, and backbone GNNs, ACE consistently achieves superior performance gains with minimal computational overhead, offering an efficient and easy-to-deploy enhancement for practical coarsening training.

**Limitations and Future Work.** Although ACE substantially improves coarsening-based training, a non-negligible performance gap remains compared to full-graph training. This is primarily because ACE recovers discarded information implicitly via the refined projector, rather than modeling explicit intra-supernode structure. While this design preserves scalability and efficiency, it may limit the achievable performance. A promising direction for future work is to develop more expressive mechanisms for integrating discarded information, extending the ACE framework to further close this gap without sacrificing scalability.


### Acknowledgements

Yifan Chen acknowledges funding from National Natural Science Foundation of China (NSFC) under grant 62502408, Research Grants Council (RGC) under grant 22303424, GuangDong Basic and Applied Basic Research Foundation under grant 2025A1515010259, and Guangdong and Hong Kong Universities "1+1+1" Joint Research Collaboration Scheme under grant 2025A050500.


### References


Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks, 2020. URL https://arxiv.org/abs/2006.13318.

Ben Chamberlain, James Rowbottom, Maria I Gorinova, Michael Bronstein, Stefan Webb, and Emanuele Rossi. Grand: Graph neural diffusion. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1407–1418. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/chamberlain21a.html.

Jie Chen, Yousef Saad, and Zechen Zhang. Graph coarsening: from scientific computing to machine learning. *SeMA Journal*, 79(1):187–223, 2022a.

Qi Chen, Yifei Wang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. Optimization-induced graph implicit nonlinear diffusion. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 3648–3661. PMLR, 17–23 Jul 2022b. URL https://proceedings.mlr.press/v162/chen22z.html.

Yifan Chen, Rentian Yao, Yun Yang, and Jie Chen. A gromov-Wasserstein geometric view of spectrum-preserving graph coarsening. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 5257–5281. PMLR, 23–29 Jul 2023.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=n6jl7fLxrP.

Fan Chung. *Spectral Graph Theory*, volume 92. CBMS Regional Conference Series in Mathematics, 1997. ISBN 978-0-8218-0315-8. doi: /10.1090/cbms/092.

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

Tingting Dan, Jiaqi Ding, Ziquan Wei, Shahar Kovalsky, Minjeong Kim, Won Hwa Kim, and Guorong Wu. Rethink and re-design graph neural networks in spaces of continuous graph diffusion functionals. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 59375–59387. Curran Associates, Inc., 2023.

Siddhartha Shankar Das, S M Ferdous, Mahantesh M. Halappanavar, Edoardo Serra, and Alex Pothen. Ags-gnn: Attribute-guided sampling for graph neural networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 538–549, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671940. URL https://doi.org/10.1145/3637528.3671940.

Chenhui Deng, Zichao Yue, and Zhiru Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hmv1LpNfXa.

Charles Dickens, Edward Huang, Aishwarya Reganti, Jiong Zhu, Karthik Subbian, and Danai Koutra. Graph coarsening via convolution matching for scalable graph neural network training. In *Companion Proceedings of the ACM on Web Conference 2024*, WWW '24, page 1502–1510, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400701726. URL https://doi.org/10.1145/3589335.3651920.

Ahmed A. A. Elhag, Gabriele Corso, Hannes Stärk, and Michael M. Bronstein. Graph anisotropic diffusion for molecules. In *ICLR2022 Machine Learning for Drug Discovery*, 2022. URL https://openreview.net/forum?id=MDYOh60QN94.

Moshe Eliasof, Eldad Haber, and Eran Treister. Pde-gcn: Novel architectures for graph neural networks motivated by partial differential equations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 3836–3849. Curran Associates, Inc., 2021.

Matthew Fahrbach, Gramoz Goranci, Richard Peng, Sushant Sachdeva, and Chi Wang. Faster graph embeddings via coarsening. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 2953–2963. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/fahrbach20a.html.

Guoji Fu, Mohammed Haroon Dupty, Yanfei Dong, and Wee Sun Lee. Implicit graph neural diffusion based on constrained dirichlet energy minimization. In *NeurIPS 2023 Workshop: New Frontiers in Graph Learning*, 2023. URL https://openreview.net/forum?id=a9loLCbWs1.

Francesco Di Giovanni, James Rowbottom, Benjamin Paul Chamberlain, Thomas Markovich, and Michael M. Bronstein. Understanding convolution on graphs via energies. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=v5ew3FPTgb.

Chenghua Gong, Yao Cheng, Xiang Li, Caihua Shan, and Siqiang Luo. Learning from graphs with heterophily: Progress and future, 2024. URL https://arxiv.org/abs/2401.09769.

Shengbo Gong, Mohammad Hashemi, Juntong Ni, Carl Yang, and Wei Jin. Scalable graph condensation with evolving capabilities. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '26, page 314–323, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400722585. doi: 10.1145/

3770854.3780217. URL https://doi.org/10.1145/3770854.3780217.

Andi Han, Dai Shi, Lequan Lin, and Junbin Gao. From continuous dynamics to graph neural networks: Neural diffusion and beyond, 2023. URL https://arxiv.org/abs/2310.10121.

Mohammad Hashemi, Shengbo Gong, Juntong Ni, Wenqi Fan, B. Aditya Prakash, and Wei Jin. A comprehensive survey on graph reduction: Sparsification, coarsening, and condensation. In Kate Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 8058–8066. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi: 10.24963/ijcai.2024/891. URL https://doi.org/10.24963/ijcai.2024/891. Survey Track.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22118–22133. Curran Associates, Inc., 2020.

Zengfeng Huang, Shengzhong Zhang, Chong Xi, Tang Liu, and Min Zhou. Scaling up graph neural networks via graph coarsening. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 675–684, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383325.

Wei Jin, Lingxiao Zhao, Shichang Zhang, Yozen Liu, Jiliang Tang, and Neil Shah. Graph condensation for graph neural networks. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=WLEx3Jo4QaB.

Yu Jin, Andreas Loukas, and Joseph JaJa. Graph coarsening with preserved spectral properties. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 4452–4462. PMLR, 26–28 Aug 2020.

Antonin Joly and Nicolas Keriven. Graph coarsening with message-passing guarantees, 2024. URL https://arxiv.org/abs/2405.18127.

Mohit Kataria, Sandeep Kumar, and Jayadeva. Ugc: Universal graph coarsening. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 63057–63081. Curran Associates, Inc., 2024.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Manoj Kumar, Anurag Sharma, and Sandeep Kumar. A unified framework for optimization-based graph coarsening. *Journal of Machine Learning Research*, 24(118): 1–50, 2023a. URL http://jmlr.org/papers/v24/22-1085.html.

Manoj Kumar, Anurag Sharma, Shashwat Saxena, and Sandeep Kumar. Featured graph coarsening with similarity guarantees. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17953–17975. PMLR, 23–29 Jul 2023b.

Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. Finding global homophily in graph neural networks when meeting heterophily. In *ICML*. PMLR, 2022. URL https://proceedings.mlr.press/v162/li22ad.html.

Derek Lim, Felix Matthew Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Prasad Bhalerao, and Ser-Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. Graph summarization methods and applications: A survey. *ACM Comput. Surv.*, 51(3), jun 2018. ISSN 0360-0300. URL https://doi.org/10.1145/3186727.

Andreas Loukas and Pierre Vandergheynst. Spectrally approximating large graphs with smaller graphs. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3237–3246. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/loukas18a.html.

Sitao Luan, Chenqing Hua, Qincheng Lu, Liheng Ma, Lirong Wu, Xinyu Wang, Minkai Xu, Xiao-Wen Chang, Doina Precup, Rex Ying, Stan Z. Li, Jian Tang, Guy Wolf, and Stefanie Jegelka. The heterophilic graph learning handbook: Benchmarks, models, theoretical analysis, applications and challenges, 2024. URL https://arxiv.org/abs/2407.09618.

Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. A fractional graph laplacian approach to oversmoothing. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 13022–13063. Curran Associates, Inc., 2023.

E. A. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964. doi: 10.1137/1109020. URL https://doi.org/10.1137/1109020.

P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990. doi: 10.1109/34.56205.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tJbbQfw-5wv.

Manish Purohit, B. Aditya Prakash, Chanhyun Kang, Yao Zhang, and V.S. Subrahmanian. Fast influence-based coarsening for large networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 1296–1305, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450329569. doi: 10.1145/2623330.2623701. URL https://doi.org/10.1145/2623330.2623701.

Zhenyue Qin, Dongwoo Kim, and Tom Gedeon. Rethinking softmax with cross-entropy: Neural network classifier as mutual information estimator, 2020. URL https://arxiv.org/abs/1911.10688.

J. W. Ruge and K. Stüben. *4. Algebraic Multigrid*, pages 73–130. doi: 10.1137/1.9781611971057.ch4. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611971057.ch4.

T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks, 2023. URL https://arxiv.org/abs/2303.10993.

Marco Serafini and Hui Guan. Scalable graph neural network training: The case for sampling. *SIGOPS Oper. Syst. Rev.*, 55(1):68–76, June 2021. ISSN 0163-5980. doi: 10.1145/3469379.3469387. URL https://doi.org/10.1145/3469379.3469387.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and Philip S. Yu. A survey on heterogeneous graph embedding: Methods, techniques, applications and sources. *IEEE Transactions on Big Data*, 9(2):415–436, 2023. doi: 10.1109/TBDATA.2022.3177455.

Lingfei Wu, Peng Cui, Jian Pei, Liang Zhao, and Le Song. *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer, 2022. ISBN 978-981-16-6054-2. doi: 10.1007/978-981-16-6054-2. URL https://doi.org/10.1007/978-981-16-6054-2.

Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. Sgformer: Simplifying and empowering transformers for large-graph representations. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 64753–64773. Curran Associates, Inc., 2023.

Shuyin Xia, Xinjun Ma, Zhiyuan Liu, Cheng Liu, Sen Zhao, and Guoyin Wang. Graph coarsening via supervised granular-ball for scalable graph neural network training. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(12):12872–12880, Apr. 2025. doi: 10.1609/aaai.v39i12.33404. URL https://ojs.aaai.org/index.php/AAAI/article/view/33404.

Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 1287–1292, 2022.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, 2022. doi: 10.1109/TKDE.2021.3070203.

Xin Zheng, Yi Wang, Yixin Liu, Ming Li, Miao Zhang, Di Jin, Philip S. Yu, and Shirui Pan. Graph neural networks for graphs with heterophily: A survey, 2024. URL https://arxiv.org/abs/2202.07082.

Yu Zhou, Haixia Zheng, Xin Huang, Shufeng Hao, Dengao Li, and Jumin Zhao. Graph neural networks: Taxonomy, advances, and trends. *ACM Transactions on Intelligent*

*Systems and Technology*, 13(1), jan 2022. ISSN 2157-6904. doi: 10.1145/3495161. URL https://doi.org/10.1145/3495161.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/58ae23d878a47004366189884c2f8440-Paper.pdf.

## A PROOF OF PROPOSITION 2.2

**Proposition 2.2.** *Let the GNN trained on the original graph be $f(;\Theta^{**})$ and the one trained on the coarsened graph be $f(;\Theta^{*})$, where*

$$\Theta^{*} \triangleq \arg\min_{\Theta} \mathcal{L}(f(A, X; \Theta), Y) \,, \tag{12}$$

$$\Theta^{**} \triangleq \arg\min_{\Theta} \mathcal{L}(f(A', X'; \Theta), Y') \,. \tag{13}$$

*When both models are evaluated on the original graph, the mutual information between their outputs satisfies*

$$I(f(A, X; \Theta^{*}); f(A, X; \Theta^{**})) \leq I(f(A, X; \Theta^{*}); Y) - \Omega \,, \tag{14}$$

*where $\Omega = I(g(\mathcal{G} \setminus \mathcal{G}'); Y | f(A, X; \Theta^{**}))$, and $g : \mathcal{G} \setminus \mathcal{G}' \mapsto \mathcal{Y}$ is arbitrary neural network that maps $\mathcal{G} \setminus \mathcal{G}'$ to the node label space $\mathcal{Y}$ and maximizes the conditional mutual information $I(g(\mathcal{G} \setminus \mathcal{G}'); Y | f(A, X; \Theta^{**}))$.*

*Proof.* We first introduce two lemmas that will serve as the foundation for our subsequent analysis.

**Lemma A.1** ((Theorem 1 [Qin et al., 2020] Restated))**.** *In standard neural network training, the infimum of the expected cross-entropy loss with a softmax output is equivalent to the mutual information between input and output variables, up to an additive $\log c$ term (where $c$ is the number of classes), provided that all class probabilities are bounded away from zero [Qin et al., 2020].*

**Lemma A.2.** *Let $Y$ be the ground-truth label, and let $A$ and $B$ be the outputs of two different models trained to predict $Y$. Assume that $A$ and $B$ are conditionally independent given $Y$, i.e. $A \perp B \mid Y$. Then*

$$I(A; B) \;\leq\; \min\{\, I(A; Y),\; I(B; Y) \,\}.$$

*Proof.* The conditional independence $A \perp B \mid Y$ implies the Markov chains

$$A \to Y \to B \qquad \text{and} \qquad B \to Y \to A.$$

By the data-processing inequality applied to the first Markov chain, any information shared between $A$ and $B$ must flow through $Y$, which yields

$$I(A; B) \leq I(A; Y).$$

Applying data processing to the second chain analogously gives

$$I(A; B) \leq I(B; Y).$$

Taking the minimum of the two bounds establishes

$$I(A; B) \leq \min\{I(A; Y),\; I(B; Y)\}.$$

Intuitively, since $A$ and $B$ can only correlate through the common target $Y$, their mutual information cannot exceed the amount of information that each individually retains about $Y$. The weaker predictor therefore determines the maximum possible dependence between them. □

Now we begin the proof. By Lemma A.1, the optimization objectives in Eq. (2) can be equivalently expressed in terms of mutual-information maximization. In particular,

$$\Theta^{*} \triangleq \arg\max_{\Theta} I(f(A, X; \Theta); Y) \,, \tag{15}$$

$$\Theta^{**} \triangleq \arg\max_{\Theta} I(f(A', X'; \Theta); Y') \,, \tag{16}$$

thus recasting the original loss-minimization problems into the perspective of mutual-information maximization.

Next, recall the chain-rule of mutual information [Cover, 1999]:

$$I(U, V; W) - I(U; W) = I(V; W | U) \tag{17}$$

Let $U = f(A, X; \Theta^{**})$, $V = g(\mathcal{G} \setminus \mathcal{G}')$, $W = Y$, we obtain:

$$\begin{aligned} & I\Big(f(A, X; \Theta^{**}), g(\mathcal{G} \setminus \mathcal{G}'); Y\Big) - I\Big(f(A, X; \Theta^{**}); Y\Big) \\ = & I\Big(g(\mathcal{G} \setminus \mathcal{G}'); Y \big| f(A, X; \Theta^{**})\Big) \,. \end{aligned} \tag{18}$$

We focus on the term $I\Big(f(A, X; \Theta^{**}), g(\mathcal{G} \setminus \mathcal{G}'); Y\Big)$, which quantifies the relationship between two models trained on different partitions of the graph $\mathcal{G}$—namely, $\mathcal{G}'$ and $\mathcal{G} \setminus \mathcal{G}'$—and the labels of the entire graph, $Y$. Specifically, we leverage the data processing inequality [Cover, 1999]:

$$I(U; V) \geq I(U; W), \text{ with Markov chain: } U \to V \to W \,. \tag{19}$$

Notice that $f(; \Theta^*)$, $f(; \Theta^{**})$, and $g$ are all deterministic functions. Hence, by letting $U = Y$, $V = f(A, X; \Theta^*)$, and $W = \Big(f(A, X; \Theta^{**}), g(\mathcal{G} \setminus \mathcal{G}')\Big)$, we can derive the following inequality:

$$I\Big(f(A, X; \Theta^*); Y\Big) \geq I\Big(f(A, X; \Theta^{**}), g(\mathcal{G} \setminus \mathcal{G}'); Y\Big) \,. \tag{20}$$

The inequality encapsulates the intuitive principle that the aggregate contribution from the model trained on the coarsened graph, $f(A, X; \Theta^{**})$, and the additional term representing the information excluded in the coarsened graph, $g(\mathcal{G} \setminus \mathcal{G}')$, is bounded above by the output of the model trained directly on the full graph, $f(A, X; \Theta^*)$.

By combining Equation 18 and Equation 20, and observing that $I\Big(f(A, X; \Theta^*); Y\Big) \geq I\Big(f(A, X; \Theta^{**}); Y\Big)$, the following inequality is derived:

$$\begin{aligned} \Delta &= \Big| I\Big(f(A, X; \Theta^*); Y\Big) - I\Big(f(A, X; \Theta^{**}); Y\Big) \Big| \\ &\geq I\Big(g(\mathcal{G} \setminus \mathcal{G}'); Y | f(A, X; \Theta^{**})\Big) \,. \end{aligned} \tag{21}$$

Finally, applying Lemma A.2 with $A := f(A, X; \Theta^*)$ and $B := f(A, X; \Theta^{**})$, and using the fact that

$$I\Big(f(A, X; \Theta^*); Y\Big) \geq I\Big(f(A, X; \Theta^{**}); Y\Big) \,, \tag{22}$$

we obtain

$$I(f(A, X; \Theta^*); f(A, X; \Theta^{**})) \leq I\Big(f(A, X; \Theta^{**}); Y\Big) \tag{23}$$

$$\leq I\Big(f(A, X; \Theta^*); Y\Big) - I\Big(g(\mathcal{G} \setminus \mathcal{G}'); Y | f(A, X; \Theta^{**})\Big) \tag{24}$$

$$= I\Big(f(A, X; \Theta^*); Y\Big) - \Omega \,, \tag{25}$$

which establishes the desired inequality and completes the proof. □

## B PROOF OF PROPOSITION 3.1

**Proposition 3.1.** *Let $P(\phi^*)$ denote the projector obtained by solving $\min_\phi \mathcal{J}_{ASR}(\phi)$. The resulting $P(\phi^*)$ acts as a conditional spectral filter that interpolates between two distinct regimes based on local homophily:*

- ***Heterophilic Regime:** In regions where $\|X_i - X_j\| \to \infty$, $P(\phi^*)$ approaches the solution of identity reconstruction, prioritizing high-frequency signal fidelity over structural smoothness.*
- ***Homophilic Regime:** In regions where $\|X_i - X_j\| \to 0$, $P(\phi^*)$ approaches the solution of isotropic laplacian smoothing, enforcing local smoothness.*

*Proof.* We analyze the local optimality condition for a specific node $i$ by isolating the terms in $\mathcal{L}(P)$ dependent on $(P(\phi)\boldsymbol{\mu})_i$. The local objective is given by:

$$\mathcal{L}_i((P(\phi)\boldsymbol{\mu})_i) = \|(P(\phi)\boldsymbol{\mu})_i - X_i\|^2 + \lambda \sum_{j \in \mathcal{N}(i)} \exp(-\|X_i - X_j\|^2) \cdot \|(P(\phi)\boldsymbol{\mu})_i - (P(\phi)\boldsymbol{\mu})_j\|^2. \tag{26}$$

The necessary condition for optimality requires the gradient $\nabla_{(P(\phi)\boldsymbol{\mu})_i}\mathcal{L}_i$ to vanish. Differentiating with respect to $(P(\phi)\boldsymbol{\mu})_i$ yields:

$$2((P(\phi)\boldsymbol{\mu})_i - X_i) + 2\lambda \sum_{j\in\mathcal{N}(i)} \exp(-\|X_i - X_j\|^2)((P(\phi)\boldsymbol{\mu})_i - (P(\phi)\boldsymbol{\mu})_j) = 0. \tag{27}$$

Rearranging terms to isolate $(P(\phi)\boldsymbol{\mu})_i$, we obtain the closed-form update rule for the optimal lifted signal:

$$(P(\phi^*)\boldsymbol{\mu})_i \left(1 + \lambda \sum_{j\in\mathcal{N}(i)} \exp(-\|X_i - X_j\|^2)\right) = X_i + \lambda \sum_{j\in\mathcal{N}(i)} \exp(-\|X_i - X_j\|^2)(P(\phi)\boldsymbol{\mu})_j. \tag{28}$$

This leads to the expression:

$$(P(\phi^*)\boldsymbol{\mu})_i = \frac{X_i + \lambda \sum_{j\in\mathcal{N}(i)} \exp(-\|X_i - X_j\|^2)(P(\phi)\boldsymbol{\mu})_j}{1 + \lambda \sum_{j\in\mathcal{N}(i)} \exp(-\|X_i - X_j\|^2)}. \tag{29}$$

We now examine the asymptotic behavior of the weight coefficient $\exp(-\|X_i - X_j\|^2) \in (0, 1]$.

**Case 1 (Homophily):** Consider a neighborhood where features are locally smooth, such that $X_i \approx X_j$. Here, $\|X_i - X_j\| \to 0$, implying $\exp(-\|X_i - X_j\|^2) \to 1$. The update rule in Eq. (28) converges to:

$$(P(\phi^*)\boldsymbol{\mu})_i \approx \frac{X_i + \lambda \sum_{j\in\mathcal{N}(i)} (P(\phi)\boldsymbol{\mu})_j}{1 + \lambda d_i}, \tag{30}$$

where $d_i$ is the degree of node $i$. This formulation is equivalent to the first-order approximation of the graph diffusion process (Low-Pass Filter), where the node representation is smoothed by averaging over its neighbors.

**Case 2 (Heterophily):** Consider a neighborhood exhibiting high feature variance (e.g., class boundaries), such that $\|X_i - X_j\|$ is large. Here, $\exp(-\|X_i - X_j\|^2) \to 0$. The regularization term vanishes, yielding:

$$(P(\phi^*)\boldsymbol{\mu})_i \approx \frac{X_i + 0}{1 + 0} = X_i. \tag{31}$$

In this regime, the structural constraint is relaxed. The projector effectively decouples node $i$ from its dissimilar neighbors, allowing the reconstructed signal to retain the high-frequency information residual essential for distinguishing heterophilic boundaries.

Since $\exp(-\|X_i - X_j\|^2)$ varies continuously, $P(\phi^*)$ adaptively interpolates between these two extrema based on the local feature manifold. □

# C TOY EXPERIMENT RESULTS (FULL)

We provide the complete results of the toy experiments introduced in Section 3.1, complementing the summary reported in Table 2. The full results are shown in Table 5. As observed, on heterophilic graphs, adding the simple auxiliary loss yields substantial performance gains, with an average improvement exceeding **6 points** (see the "$\Delta \uparrow$ (Heterophily)" column). These results further strengthen the empirical motivation behind our method.

Interestingly, although the auxiliary loss is designed to address heterophily, we also observe minor improvements on homophilic datasets. This is consistent with the fact that many benchmarks labeled as "homophilic"—such as those from the OGB suite—actually contain mixed patterns with non-negligible local heterophily [Zheng et al., 2024, Gong et al., 2024, Luan et al., 2024]. Such latent heterophily allows the same enhancement to deliver modest benefits even on nominally homophilic graphs.

# D EXPERIMENTAL DETAILS

This section provides all supplementary experimental details, including dataset statistics (Appendix D.1), descriptions of coarsening training baselines (Appendix D.2), backbone GNN introductions (Appendix D.3), and additional experimental results (Appendix D.4).

Table 5: Full toy experiment results. We highlight substantial improvements—observed exclusively on heterophilic datasets—in **bold**. "$\Delta \uparrow$ (Heterophily)" reports the average gain on heterophilic graphs.

| Backbone | Method | Heterophilic Graphs | | | Homophilic Graphs | | $\Delta \uparrow$ (Heterophily) |
|---|---|---|---|---|---|---|---|
| | | Genius | Gamers | Pokec | Arxiv | Products | |
| GCN | Full Graph | 87.42 | 62.18 | 75.45 | 71.74 | 75.64 | / |
| | SCAL | 67.47 | 47.39 | 50.44 | 63.42 | 68.37 | **4.72** |
| | SCAL w/ $\tilde{\mathcal{L}}$ | 71.32 (**+3.85**) | 52.86 (**+5.47**) | 55.29 (**+4.85**) | 64.71 (+1.29) | 69.11 (+0.74) | |
| | FGC | 70.13 | 40.14 | 53.81 | 65.52 | 71.28 | **6.02** |
| | FGC w/ $\tilde{\mathcal{L}}$ | 75.67 (**+5.54**) | 49.26 (**+9.12**) | 57.22 (**+3.41**) | 66.18 (+0.66) | 72.06 (+0.78) | |
| | SGBGC | 71.22 | 44.62 | 55.48 | 64.12 | 71.96 | **6.00** |
| | SGBGC w/ $\tilde{\mathcal{L}}$ | 73.81 (**+2.59**) | 53.09 (**+8.47**) | 62.42 (**+6.94**) | 65.72 (+1.60) | 72.37 (+0.41) | |
| LINKX | Full Graph | 90.77 | 66.06 | 82.04 | 69.54 | 74.59 | / |
| | SCAL | 60.62 | 50.13 | 56.48 | 57.71 | 62.49 | **7.14** |
| | SCAL w/ $\tilde{\mathcal{L}}$ | 70.44 (**+9.82**) | 54.28 (**+4.15**) | 63.93 (**+7.45**) | 59.38 (+1.67) | 63.61 (+1.12) | |
| | FGC | 65.49 | 42.48 | 57.63 | 59.38 | 67.52 | **6.91** |
| | FGC w/ $\tilde{\mathcal{L}}$ | 71.92 (**+6.43**) | 48.62 (**+6.14**) | 65.81 (**+8.18**) | 61.44 (+2.06) | 68.77 (+1.25) | |
| | SGBGC | 62.68 | 47.60 | 54.18 | 61.55 | 64.93 | **6.09** |
| | SGBGC w/ $\tilde{\mathcal{L}}$ | 68.13 (**+5.45**) | 52.83 (**+5.23**) | 61.77 (**+7.59**) | 63.24 (+1.69) | 66.19 (+1.26) | |
| GloGNN | Full Graph | 90.66 | 66.19 | 83.00 | 72.68 | 77.48 | / |
| | SCAL | 59.47 | 46.28 | 61.58 | 57.09 | 66.53 | **6.97** |
| | SCAL w/ $\tilde{\mathcal{L}}$ | 67.41 (**+7.94**) | 53.11 (**+6.83**) | 67.74 (**+6.16**) | 58.32 (+1.23) | 67.44 (+0.91) | |
| | FGC | 63.81 | 45.92 | 66.91 | 61.73 | 70.46 | **6.13** |
| | FGC w/ $\tilde{\mathcal{L}}$ | 72.33 (**+8.52**) | 51.26 (**+5.34**) | 71.44 (**+4.53**) | 63.08 (+1.35) | 71.52 (+1.06) | |
| | SGBGC | 67.33 | 47.15 | 63.77 | 61.24 | 69.11 | **6.23** |
| | SGBGC w/ $\tilde{\mathcal{L}}$ | 73.26 (**+5.93**) | 54.31 (**+7.17**) | 69.36 (**+5.59**) | 62.75 (+1.51) | 70.07 (+0.96) | |

Table 6: Statistics for the large graph datasets. $\mathcal{H}$ quantifies the homophily level [Zhu et al., 2020], where lower values denote stronger heterophily, and higher values indicate more homophilic graphs.

| Statistics | Heterophilic Graphs | | | | | Homophilic Graphs | |
|---|---|---|---|---|---|---|---|
| | Genius | Gamers | Pokec | Snap | arXiv-year | ogbn-arXiv | ogbn-products |
| # Nodes | 421,961 | 168,114 | 1,632,803 | 2,923,922 | 169,343 | 169,343 | 2,449,029 |
| # Edges | 984,979 | 6,797,557 | 30,622,564 | 13,975,788 | 1,166,243 | 1,157,799 | 61,859,140 |
| # Features | 12 | 7 | 65 | 269 | 128 | 128 | 100 |
| # Classes | 2 | 2 | 2 | 5 | 5 | 40 | 47 |
| # $\mathcal{H}$ | 0.62 | 0.45 | 0.44 | 0.07 | 0.22 | 0.66 | 0.82 |

### D.1 DATASET STATISTICS

All dataset statistics are summarized in Table 6. To characterize homophily levels, we report the *edge homophily* metric [Zhu et al., 2020], where a smaller value of $\mathcal{H}$ indicates stronger heterophily.

### D.2 OVERVIEW OF COARSENING TRAINING BASELINES

We provide an overview of the four advanced coarsening-based GNN training pipelines used in our experiments, summarized as follows:

- **SCAL [Huang et al., 2021].** SCAL introduces a spectral graph coarsening technique [Loukas and Vandergheynst, 2018] that operates in an unsupervised manner, relying solely on the inherent graph structure. The coarsened graph is

constructed during preprocessing, and GNNs are subsequently trained on these reduced graphs. A novel coarsened graph convolution is proposed to maintain consistency between operations on the original and coarsened graphs. Optimization is performed exclusively with respect to the loss computed on the coarsened graph labels. As the first work to explore GNN training on coarsened graphs, SCAL has inspired numerous follow-up studies aimed at improving computational efficiency and scalability.

- **FGC [Kumar et al., 2023b].** FGC enhances graph coarsening by incorporating both structural and feature information. It formulates the coarsening process as the minimization of a Dirichlet energy functional, which unifies graph topology and node attributes. The resulting coarsened graphs are more informative and support superior downstream GNN training.
- **UGC [Kataria et al., 2024].** UGC employs carefully designed hash functions for efficient graph coarsening on attributed graphs. In addition, it explicitly accounts for the heterophily present in such graphs. This combination allows for the generation of coarsened graphs that outperform those produced by FGC in terms of training effectiveness.
- **SGBGC [Xia et al., 2025].** SGBGC constructs coarsened graphs via an iterative granulation process. It partitions the original graph into granular-balls based on a node purity criterion, with each granular-ball treated as a supernode. This approach achieves substantial graph compression while preserving structural integrity, offering notable gains in both efficiency and scalability.

### D.3 BACKBONE GNN INTRODUCTIONS

We briefly summarize the six backbone GNNs used in our experiments:

- **GCN [Kipf and Welling, 2017].** GCN approximates spectral graph convolution via a first-order polynomial, aggregating normalized neighbor features in each layer. Its simplicity and efficiency make it a strong baseline, though its inherent smoothness bias limits performance on heterophilic graphs.
- **LINKX [Lim et al., 2021].** LINKX separately embeds node features and adjacency structure and fuses them using MLPs, avoiding graph convolution altogether. This decoupled design removes the homophily bias and scales well to large, low-homophily graphs where many GNNs struggle.
- **GloGNN [Li et al., 2022].** GloGNN aggregates information using a learned global correlation matrix rather than local neighborhoods, enabling flexible (including signed) information mixing. A closed-form update with Woodbury acceleration allows linear-time scaling, making GloGNN highly effective on heterophilic graphs.
- **GPRGNN [Chien et al., 2021].** GPRGNN learns the coefficients of a generalized PageRank propagation, enabling adaptive multi-hop filtering. By learning propagation weights instead of fixing them, it naturally adjusts to both homophilic and heterophilic patterns and mitigates over-smoothing.
- **SGFormer [Wu et al., 2023].** SGFormer employs a single-layer global attention paired with a lightweight GNN, capturing all-pair interactions in one step with linear complexity. It is extremely efficient but lacks the richer positional and structural encodings common in deeper graph Transformers.
- **Polynormer [Deng et al., 2024].** Polynormer is a kernel-based graph Transformer that computes attention via graph kernels derived from sampled neighborhoods. Its polynomial-expressive architecture learns high-degree equivariant polynomials through a linear local-to-global attention mechanism.

### D.4 SUPPLEMENTARY EXPERIMENTAL RESULTS

This section provides additional experimental results that were moved from the main text due to space constraints. Specifically, we include:

- **Table 7:** Standard deviations corresponding to the mean results in Table 3.
- **Table 8:** ACE-integrated results for the remaining backbone GNNs: GPRGNN, SGFormer, and Polynormer.
- **Table 9:** Results on **task-optimized** coarsening training pipelines—ConvMatch and GCond—using GCN and GPRGNN across all benchmarks.
- **Table 10:** Results under a 1% coarsening ratio, using GCN and GPRGNN as backbones.
- **Table 11:** Comparison between ACE-enhanced coarsening methods and state-of-the-art approaches under GPRGNN backbone.

Table 7: Standard deviations corresponding to the mean results reported in Table 3.

| Backbone | Method | Heterophilic Graphs | | | | | Homophilic Graphs | |
|---|---|---|---|---|---|---|---|---|
| | | Genius | Gamers | Pokec | Snap | arXiv-year | ogbn-arXiv | ogbn-products |
| GCN | - | 0.34 | 0.29 | 0.21 | 0.22 | 0.38 | 0.23 | 0.26 |
| | SCAL | 0.18 | 0.35 | 0.23 | 0.26 | 0.26 | 0.23 | 0.23 |
| | + ACE | 0.21 | 0.29 | 0.21 | 0.27 | 0.22 | 0.22 | 0.38 |
| | FGC | 0.18 | 0.23 | 0.25 | 0.23 | 0.25 | 0.21 | 0.18 |
| | + ACE | 0.24 | 0.18 | 0.31 | 0.21 | 0.27 | 0.38 | 0.22 |
| | UGC | 0.27 | 0.26 | 0.24 | 0.24 | 0.23 | 0.22 | 0.20 |
| | + ACE | 0.31 | 0.24 | 0.21 | 0.26 | 0.22 | 0.21 | 0.23 |
| | SGBGC | 0.18 | 0.20 | 0.25 | 0.19 | 0.23 | 0.20 | 0.23 |
| | + ACE | 0.22 | 0.26 | 0.18 | 0.23 | 0.25 | 0.23 | 0.29 |
| LINKX | - | 0.33 | 0.19 | 0.26 | 0.29 | 0.32 | 0.21 | 0.19 |
| | SCAL | 0.32 | 0.21 | 0.17 | 0.33 | 0.29 | 0.20 | 0.23 |
| | + ACE | 0.29 | 0.19 | 0.27 | 0.32 | 0.21 | 0.29 | 0.27 |
| | FGC | 0.24 | 0.21 | 0.25 | 0.20 | 0.25 | 0.23 | 0.31 |
| | + ACE | 0.32 | 0.24 | 0.32 | 0.26 | 0.32 | 0.20 | 0.23 |
| | UGC | 0.19 | 0.26 | 0.31 | 0.17 | 0.38 | 0.17 | 0.38 |
| | + ACE | 0.25 | 0.31 | 0.17 | 0.23 | 0.23 | 0.23 | 0.25 |
| | SGBGC | 0.17 | 0.22 | 0.26 | 0.20 | 0.27 | 0.20 | 0.19 |
| | + ACE | 0.33 | 0.17 | 0.25 | 0.32 | 0.23 | 0.33 | 0.23 |
| GloGNN | - | 0.32 | 0.23 | 0.27 | 0.17 | 0.23 | 0.33 | 0.26 |
| | SCAL | 0.33 | 0.33 | 0.19 | 0.23 | 0.32 | 0.29 | 0.33 |
| | + ACE | 0.24 | 0.26 | 0.32 | 0.20 | 0.26 | 0.27 | 0.38 |
| | FGC | 0.19 | 0.22 | 0.27 | 0.32 | 0.31 | 0.33 | 0.20 |
| | + ACE | 0.27 | 0.32 | 0.33 | 0.25 | 0.27 | 0.31 | 0.26 |
| | UGC | 0.19 | 0.26 | 0.17 | 0.26 | 0.20 | 0.29 | 0.29 |
| | + ACE | 0.27 | 0.17 | 0.38 | 0.19 | 0.27 | 0.31 | 0.33 |
| | SGBGC | 0.22 | 0.22 | 0.17 | 0.24 | 0.20 | 0.19 | 0.26 |
| | + ACE | 0.24 | 0.29 | 0.29 | 0.19 | 0.26 | 0.29 | 0.27 |

- **Table 12 and Table 13:** Comprehensive ablation study of ACE components under the GCN and GPRGNN backbones.
- **Figure 5:** Complete ablation results for the tradeoff coefficient $\beta$.
- **Figure 6:** Full preprocessing-time comparison between vanilla and ACE-enhanced coarsening training pipelines.
- **Figure 7:** Full per-epoch training-time comparison across coarsening training pipelines.
- **Figure 8:** Full training memory consumption comparison across coarsening training pipelines.
- **Figure 9:** Assessment of feature noise robustness for ACE and state-of-the-art methods.

Table 8: Results of integrating ACE into GPRGNN, SGFormer, and Polynormer. Improvements on heterophilic datasets appear in **bold**.

| Backbone | Method | Heterophilic Graphs | | | | | Homophilic Graphs | |
|---|---|---|---|---|---|---|---|---|
| | | Genius | Gamers | Pokec | Snap | arXiv-year | ogbn-arXiv | ogbn-products |
| GPRGNN | - | 90.05 | 61.89 | 78.83 | 40.19 | 45.07 | 71.78 | 78.29 |
| | SCAL | 66.19 | 43.48 | 60.92 | 28.22 | 26.53 | 63.28 | 65.92 |
| | + ACE | 78.57 (**+12.38**) | 51.74 (**+8.26**) | 70.25 (**+9.33**) | 32.42 (**+4.20**) | 35.59 (**+9.06**) | 64.76 (+1.48) | 68.25 (+2.33) |
| | FGC | 69.28 | 45.92 | 57.14 | 26.44 | 28.60 | 62.19 | 63.28 |
| | + ACE | 79.02 (**+9.74**) | 52.38 (**+6.46**) | 64.87 (**+7.73**) | 31.22 (**+4.78**) | 36.73 (**+8.13**) | 64.29 (+2.10) | 65.83 (+2.55) |
| | UGC | 68.06 | 41.20 | 61.33 | 25.38 | 29.12 | 61.44 | 65.26 |
| | + ACE | 78.17 (**+10.11**) | 51.06 (**+9.86**) | 69.37 (**+8.04**) | 31.46 (**+6.08**) | 34.59 (**+5.47**) | 63.19 (+1.75) | 67.84 (+2.58) |
| | SGBGC | 69.93 | 44.26 | 62.41 | 26.93 | 31.26 | 63.14 | 63.02 |
| | + ACE | 77.39 (**+7.46**) | 50.61 (**+6.35**) | 67.38 (**+4.97**) | 30.24 (**+3.31**) | 38.42 (**+7.16**) | 65.02 (+1.88) | 68.47 (+5.45) |
| SGFormer | - | 85.01 | 65.93 | 80.84 | 63.84 | 51.23 | 72.63 | 81.54 |
| | SCAL | 63.91 | 51.74 | 63.47 | 42.73 | 30.81 | 61.38 | 71.74 |
| | + ACE | 75.81 (**+11.90**) | 57.38 (**+5.64**) | 71.88 (**+8.41**) | 53.62 (**+10.89**) | 43.76 (**+12.95**) | 63.02 (+1.64) | 73.84 (+2.10) |
| | FGC | 67.26 | 52.09 | 61.37 | 40.29 | 32.48 | 63.81 | 73.62 |
| | + ACE | 76.31 (**+9.05**) | 56.38 (**+4.29**) | 70.33 (**+8.96**) | 50.13 (**+9.84**) | 43.02 (**+10.54**) | 65.11 (+1.30) | 74.84 (+1.22) |
| | UGC | 65.38 | 49.07 | 64.35 | 44.16 | 28.60 | 60.77 | 70.04 |
| | + ACE | 75.48 (**+10.10**) | 58.34 (**+9.27**) | 71.48 (**+7.13**) | 49.78 (**+5.62**) | 38.26 (**+9.66**) | 63.29 (+2.52) | 73.16 (+3.12) |
| | SGBGC | 68.73 | 52.66 | 60.73 | 46.38 | 34.01 | 61.62 | 68.30 |
| | + ACE | 76.12 (**+7.39**) | 57.02 (**+4.36**) | 69.66 (**+8.93**) | 52.09 (**+5.71**) | 39.74 (**+5.73**) | 64.21 (+2.59) | 72.73 (+4.43) |
| Polynormer | - | 85.64 | 64.72 | 86.06 | 61.92 | 53.48 | 73.40 | 83.82 |
| | SCAL | 64.93 | 48.42 | 64.38 | 40.63 | 32.74 | 63.93 | 71.38 |
| | + ACE | 75.73 (**+10.80**) | 55.62 (**+7.20**) | 73.16 (**+8.78**) | 47.63 (**+7.00**) | 40.61 (**+7.87**) | 65.28 (+1.35) | 73.47 (+2.09) |
| | FGC | 62.03 | 44.22 | 66.63 | 38.66 | 35.26 | 64.22 | 72.48 |
| | + ACE | 74.79 (**+12.76**) | 57.48 (**+13.26**) | 77.81 (**+11.18**) | 50.13 (**+11.47**) | 46.72 (**+11.46**) | 65.79 (+1.57) | 74.61 (+2.13) |
| | UGC | 66.02 | 45.49 | 68.49 | 43.84 | 30.34 | 61.28 | 71.79 |
| | + ACE | 74.20 (**+8.18**) | 57.02 (**+11.53**) | 76.24 (**+7.75**) | 51.26 (**+7.42**) | 43.28 (**+12.94**) | 64.10 (+2.82) | 73.45 (+1.66) |
| | SGBGC | 67.98 | 41.37 | 62.13 | 45.01 | 32.63 | 63.33 | 70.16 |
| | + ACE | 77.83 (**+9.85**) | 53.81 (**+12.44**) | 74.86 (**+12.73**) | 49.38 (**+4.37**) | 38.97 (**+6.34**) | 65.03 (+1.70) | 72.88 (+2.72) |

Table 9: Additional results for **task-optimized** coarsening pipelines—GCond [Jin et al., 2022] and ConvMatch [Dickens et al., 2024]—evaluated using GCN (homophily-oriented) and GPRGNN (heterophily-oriented) backbones across diverse benchmarks.

| Backbone | Method | Heterophilic Graphs | | | | | Homophilic Graphs | |
|---|---|---|---|---|---|---|---|---|
| | | Genius | Gamers | Pokec | Snap | arXiv-year | ogbn-arXiv | ogbn-products |
| GCN | - | 87.42 | 62.18 | 75.45 | 45.65 | 46.02 | 71.74 | 75.64 |
| | GCond | 70.22 | 51.27 | 53.77 | 29.78 | 30.22 | 66.38 | 71.19 |
| | + ACE | 78.33 (**+8.11**) | 56.26 (**+4.99**) | 58.49 (**+4.72**) | 36.37 (**+6.59**) | 37.83 (**+7.61**) | 67.15 (+0.77) | 72.01 (+0.82) |
| | ConvMatch | 68.13 | 43.74 | 50.66 | 24.37 | 25.94 | 66.12 | 70.93 |
| | + ACE | 76.44 (**+8.31**) | 52.07 (**+8.33**) | 55.39 (**+4.73**) | 32.29 (**+7.92**) | 30.85 (**+4.91**) | 67.95 (+1.83) | 72.83 (+1.90) |
| GPRGNN | - | 90.05 | 61.89 | 78.83 | 40.19 | 45.07 | 71.78 | 78.29 |
| | GCond | 71.39 | 49.47 | 58.65 | 30.47 | 31.38 | 65.93 | 70.27 |
| | + ACE | 79.63 (**+8.24**) | 55.65 (**+6.18**) | 71.33 (**+12.68**) | 34.28 (**+3.81**) | 37.19 (**+5.81**) | 66.49 (+0.56) | 72.01 (+1.74) |
| | ConvMatch | 66.31 | 44.67 | 51.40 | 22.33 | 24.60 | 66.19 | 67.28 |
| | + ACE | 77.24 (**+10.93**) | 50.42 (**+5.75**) | 66.67 (**+15.27**) | 30.05 (**+7.72**) | 31.43 (**+6.83**) | 67.43 (+1.24) | 68.44 (+1.19) |

Table 10: Additional results under an extreme 1% coarsening ratio.

| Backbone | Method | Heterophilic Graphs | | | | | Homophilic Graphs | |
|---|---|---|---|---|---|---|---|---|
| | | Genius | Gamers | Pokec | Snap | arXiv-year | ogbn-arXiv | ogbn-products |
| GCN | - | 87.42 | 62.18 | 75.45 | 45.65 | 46.02 | 71.74 | 75.64 |
| | SCAL | 55.42 | 39.21 | 46.26 | 20.41 | 22.43 | 60.39 | 64.69 |
| | + ACE | 73.76 (**+18.34**) | 52.43 (**+13.22**) | 53.80 (**+7.54**) | 28.73 (**+8.32**) | 29.38 (**+6.95**) | 64.47 (+4.08) | 68.73 (+4.04) |
| | FGC | 63.84 | 36.08 | 45.37 | 21.23 | 21.40 | 62.86 | 66.14 |
| | + ACE | 72.74 (**+8.90**) | 50.18 (**+14.10**) | 54.55 (**+9.18**) | 30.25 (**+9.02**) | 29.01 (**+7.61**) | 63.71 (+0.85) | 70.22 (+4.08) |
| | SGBGC | 64.17 | 40.25 | 48.66 | 23.24 | 22.42 | 62.56 | 64.81 |
| | + ACE | 71.02 (**+6.85**) | 52.39 (**+12.14**) | 53.29 (**+4.63**) | 28.63 (**+5.39**) | 30.22 (**+7.80**) | 64.66 (+2.10) | 68.37 (+3.56) |
| GPRGNN | - | 90.05 | 61.89 | 78.83 | 40.19 | 45.07 | 71.78 | 78.29 |
| | SCAL | 59.39 | 37.82 | 50.68 | 26.28 | 22.15 | 61.47 | 63.06 |
| | + ACE | 73.54 (**+14.15**) | 47.46 (**+9.64**) | 61.62 (**+10.94**) | 30.33 (**+4.05**) | 30.42 (**+8.27**) | 63.22 (+1.75) | 65.81 (+2.75) |
| | FGC | 57.47 | 36.81 | 51.42 | 23.37 | 26.55 | 61.05 | 60.39 |
| | + ACE | 71.23 (**+13.76**) | 45.23 (**+8.42**) | 59.83 (**+8.41**) | 27.83 (**+4.46**) | 31.17 (**+4.62**) | 63.14 (+2.09) | 63.77 (+3.38) |
| | SGBGC | 61.63 | 38.41 | 54.48 | 23.41 | 25.07 | 61.39 | 60.97 |
| | + ACE | 70.78 (**+9.15**) | 47.81 (**+9.40**) | 64.13 (**+9.65**) | 28.35 (**+4.94**) | 30.93 (**+5.86**) | 63.77 (+2.38) | 65.48 (+4.51) |

Table 11: Comparison with state-of-the-art scalable GNN training methods based on graph data reduction. Results on a **GPRGNN** backbone with a 10% reduction ratio are reported in terms of test accuracy and total training time.

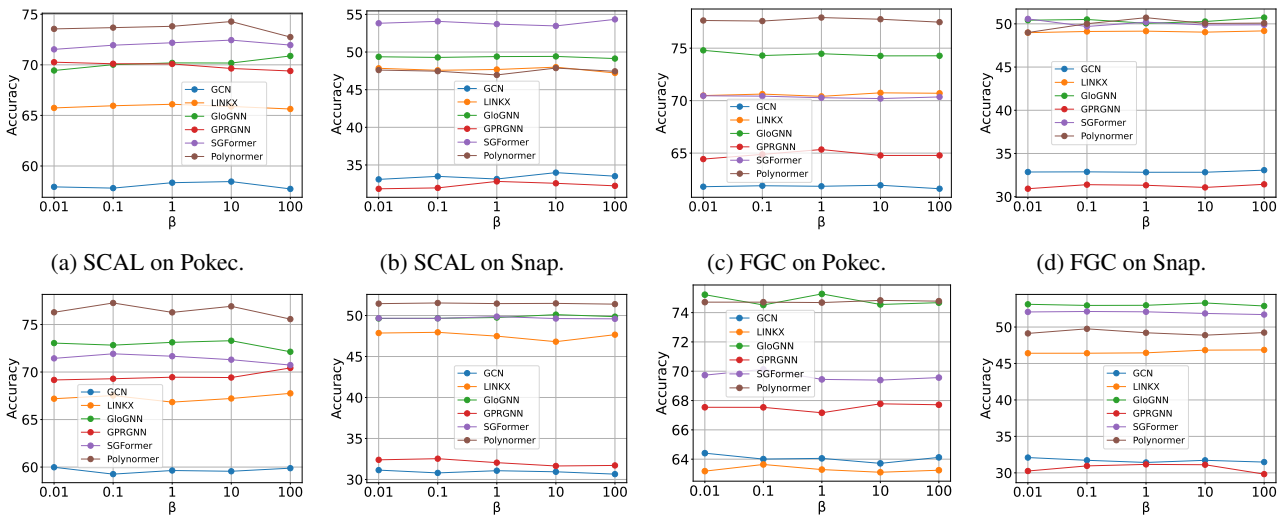
| Category | Method | Pokec | | Snap | | Gamers | | Ogbn-product | |
|---|---|---|---|---|---|---|---|---|---|
| | | ACC | Time (min) | ACC | Time (min) | ACC | Time (min) | ACC | Time (min) |
| Coarsening | FGC | 57.14 | 22.48 | 26.44 | 30.84 | 45.92 | 10.28 | 63.28 | 32.17 |
| | UGC | 61.33 | 21.55 | 25.38 | 28.26 | 41.20 | 10.56 | 65.26 | 30.46 |
| Data-Adaptive | GECC | 64.19 | 32.16 | 30.81 | 36.43 | 49.62 | 16.81 | 66.77 | 43.49 |
| Subgraph-Sampling | AGS | 62.77 | 19.92 | 28.34 | 28.38 | 48.13 | 9.44 | 66.11 | 29.54 |
| **Ours** | FGC+ACE | **64.87** | 22.96 | **31.22** | 31.37 | **52.38** | 10.86 | 65.83 | 33.09 |
| | UGC+ACE | **69.37** | 21.88 | **31.46** | 29.84 | **51.06** | 11.16 | **67.84** | 31.22 |

Table 12: Detailed ablation study of individual components in ACE. Results are reported on a **GCN** backbone with a 10% reduction ratio, including test accuracy for the full method (ACE base) and relative performance drop for each variant.

| Method | Variant | Genius | Gamers | Pokec | Snap | Ogbn-Product | Average Drop |
|---|---|---|---|---|---|---|---|
| FGC | ACE (base) | 77.32 | 52.52 | 61.67 | 32.73 | 72.47 | - |
| | ACE (fixed weighting) | -0.35 | -0.59 | -0.41 | -0.62 | -0.06 | -0.41 |
| | ACE (no anisotropic reg) | -0.70 | -1.83 | -1.90 | -1.30 | -0.18 | -1.18 |
| | ACE (fixed projector) | -1.34 | -2.60 | -3.51 | -1.66 | -0.36 | **-1.90** |
| | naive auxiliary loss | -1.65 | -3.26 | -4.45 | -2.07 | -0.41 | **-2.37** |
| UGC | ACE (base) | 75.28 | 53.83 | 59.37 | 30.82 | 70.19 | - |
| | ACE (fixed weighting) | -0.26 | -0.71 | -0.63 | -0.50 | -0.11 | -0.44 |
| | ACE (no anisotropic reg) | -0.81 | -2.06 | -1.75 | -1.22 | -0.19 | -1.20 |
| | ACE (fixed projector) | -1.72 | -3.61 | -3.83 | -1.93 | -0.35 | **-2.29** |
| | naive auxiliary loss | -2.44 | -4.23 | -4.16 | -2.65 | -0.56 | **-2.70** |

Table 13: Detailed ablation study of individual components in ACE. Results are reported on a **GPRGNN** backbone with a 10% reduction ratio, including test accuracy for the full method (ACE base) and relative performance drop for each variant.

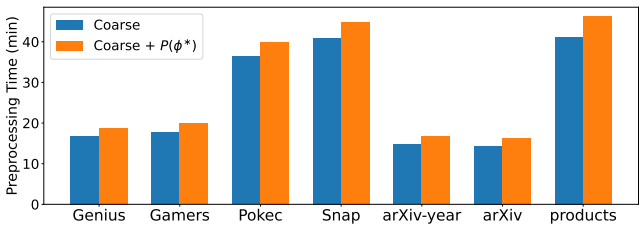
| Method | Variant | Genius | Gamers | Pokec | Snap | Ogbn-Product | Average Drop |
|---|---|---|---|---|---|---|---|
| FGC | ACE (base) | 79.02 | 52.38 | 64.87 | 31.22 | 65.83 | - |
| | ACE (fixed weighting) | -0.27 | -0.41 | -0.46 | -0.44 | -0.10 | -0.34 |
| | ACE (no anisotropic reg) | -0.79 | -1.23 | -1.64 | -1.14 | -0.15 | -0.99 |
| | ACE (fixed projector) | -1.65 | -2.84 | -3.47 | -1.92 | -0.28 | **-2.03** |
| | naive auxiliary loss | -2.12 | -3.33 | -3.93 | -2.47 | -0.43 | **-2.46** |
| UGC | ACE (base) | 78.17 | 51.06 | 69.37 | 31.46 | 67.84 | - |
| | ACE (fixed weighting) | -0.38 | -0.56 | -0.42 | -0.53 | -0.08 | -0.39 |
| | ACE (no anisotropic reg) | -0.75 | -1.78 | -1.93 | -1.42 | -0.16 | -1.21 |
| | ACE (fixed projector) | -2.14 | -3.59 | -3.45 | -2.46 | -0.30 | **-2.39** |
| | naive auxiliary loss | -2.81 | -4.02 | -4.09 | -3.16 | -0.45 | **-2.91** |



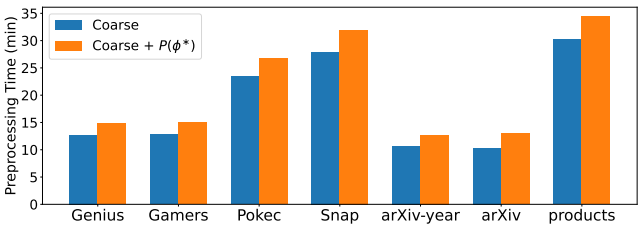

(a) SCAL on Pokec. (b) SCAL on Snap. (c) FGC on Pokec. (d) FGC on Snap.

(e) UGC on Pokec. (f) UGC on Snap. (g) SGBGC on Pokec. (h) SGBGC on Snap.

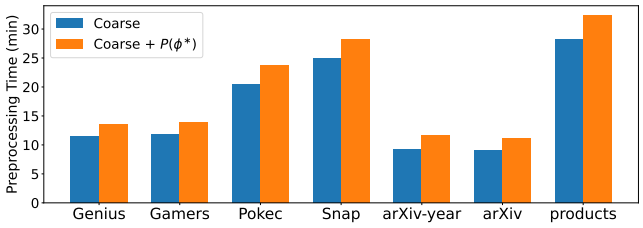
Figure 5: Effect of $\beta$ across coarsening pipelines, GNN backbones, and datasets.



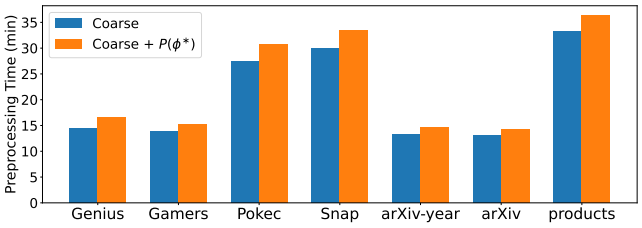

(a) Preprocessing time: SCAL vs. SCAL+ACE.




(b) Preprocessing time: FGC vs. FGC+ACE.

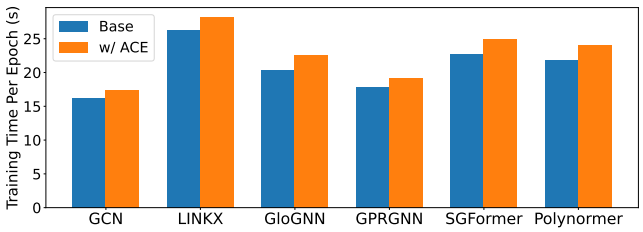



(c) Preprocessing time: UGC vs. UGC+ACE.

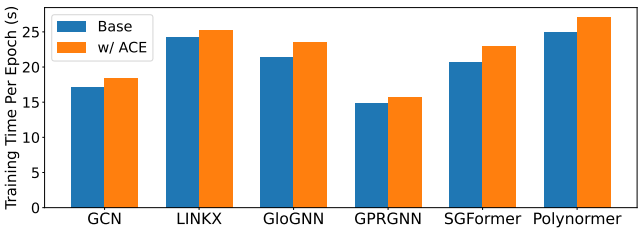



(d) Preprocessing time: SGBGC vs. SGBGC+ACE.

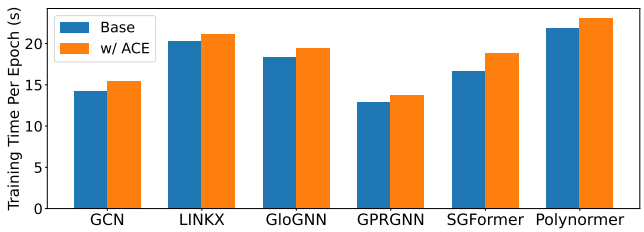
Figure 6: Preprocessing time of each coarsening training pipelines and those with our ACE across diverse benchmarks.




(a) SCAL training time per epoch.

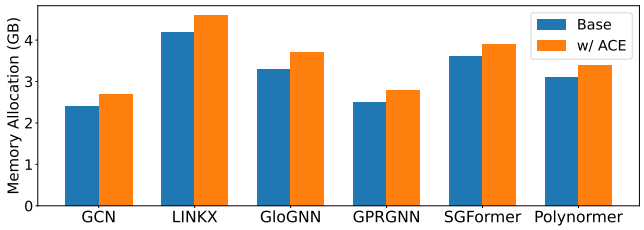



(b) FGC training time per epoch.

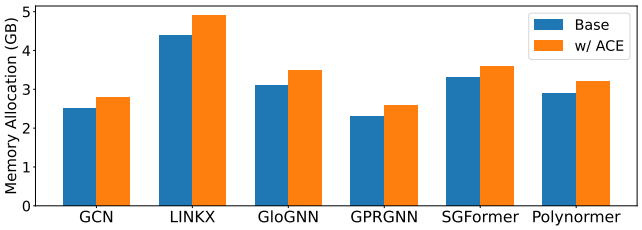



(c) SGBGC training time per epoch.

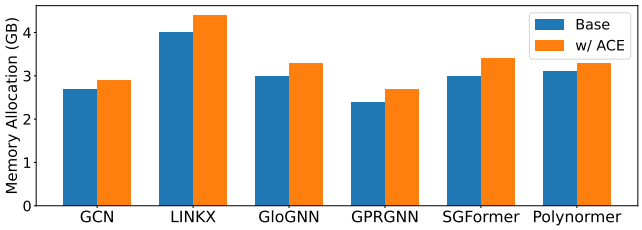
Figure 7: Per-epoch training time of SCAL, FGC, SGBGC, and their ACE-improved versions across diverse backbone GNNs.




(a) SCAL training memory usage.

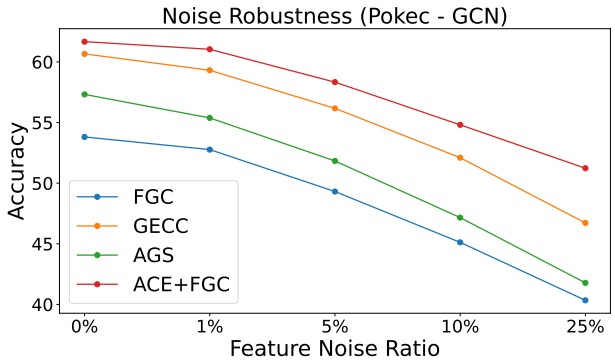



(b) FGC training memory usage.

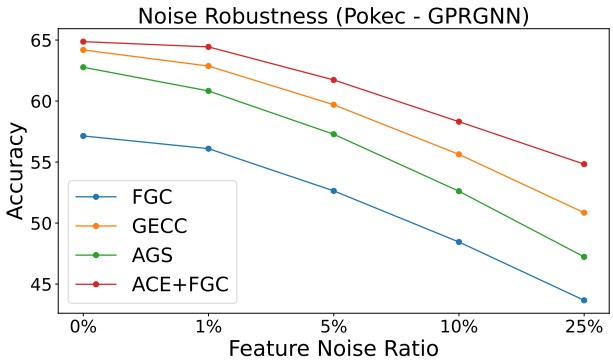



(c) SGBGC training memory usage.

Figure 8: Training GPU memory allocation of SCAL, FGC, SGBGC, and their ACE-improved versions across diverse backbone GNNs.




(a) GCN as backbone.




(b) GPRGNN as backbone.

Figure 9: Noise robustness evaluation on the Pokec dataset. Experiments are conducted using GCN and GPRGNN backbones with a 10% graph reduction ratio. We report performance under feature perturbations, where Gaussian noise is added to randomly sampled node features in the original graph, with sampling ratios of {0%, 1%, 5%, 10%, 25%}.