LABSTA: LABEL- AND STRUCTURE-AWARE MINI-BATCH SAMPLING FOR NODE CLASSIFICATION WITH GRAPH TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Scalability is a major challenge in graph representation learning, as training on large-scale graphs may require processing millions of nodes that do not fit within limited GPU memory. For Graph Neural Networks (GNNs), mini-batch training provides a practical alternative to full-graph training, typically using connectivity-oriented sampling methods that preserve local connectivity for message passing. However, mini-batch construction for Graph Transformers (GTs) remains under-explored, and many prior works rely on simple random node sampling, which can lead to suboptimal performance. In this work, we motivate two key objectives for mini-batch sampling in GT-based node classification: reducing class-distribution discrepancy and increasing edge retention rate. Based on these objectives, we propose **LabStA**, a Label- and Structure-Aware mini-batch sampling method that augments graph partitions with informative virtual nodes to jointly improve class-distribution alignment and edge preservation. Experiments on diverse node classification benchmarks show that LabStA outperforms existing sampling baselines on most dataset–backbone pairs, especially on heterophilous graphs.

1 INTRODUCTION

Graph representation learning aims to learn node embeddings that capture both node features and graph topology. An early paradigm is Graph Neural Networks (GNNs), which update node representations by aggregating information from local neighborhoods (Kipf & Welling, 2016; Veličković et al., 2017). Despite their success, GNNs suffer from issues such as over-smoothing (Li et al., 2018) and over-squashing (Alon & Yahav, 2020), limiting their ability to capture long-range dependencies. To address these limitations, recent studies have extended the Transformer architecture (Vaswani et al., 2017) to graph-structured data. Specifically, Graph Transformers (GTs) adopt self-attention, a global aggregation mechanism that enables information exchange between distant nodes. However, vanilla self-attention does not explicitly account for graph topology, which may restrict the ability of GTs to exploit relational inductive biases (Ying et al., 2021). To incorporate structural information, prior work injects positional encodings into node features or attention biases (Dwivedi & Bresson, 2020; Ito et al., 2025), or combines local message passing with global self-attention (Rampášek et al., 2022; Deng et al., 2024). By expanding the receptive field beyond local neighborhoods, GTs provide a promising alternative to conventional GNNs.

Nevertheless, scalability remains a major challenge for GTs. A primary bottleneck is the quadratic complexity of self-attention, which makes training on large graphs computationally prohibitive. To mitigate this issue, scalable GTs adopt efficient attention mechanisms, such as linear attention (Wu et al., 2023b) and sparse attention (Shirzad et al., 2023). Another major bottleneck is memory consumption: large graphs can contain millions of nodes and edges (Hu et al., 2020), often exceeding the GPU memory available for full-graph training. Recent studies have explored tokenized formulations, in which training and inference operate on preconstructed token sequences (Chen et al., 2022; Fu et al., 2024). While efficient, these models rely on method-specific tokenization procedures that may limit the graph information accessible to the Transformer. As an alternative, mini-batch training provides a practical and effective way to reduce memory requirements by training on sampled subgraphs that retain relevant information from the full graph.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

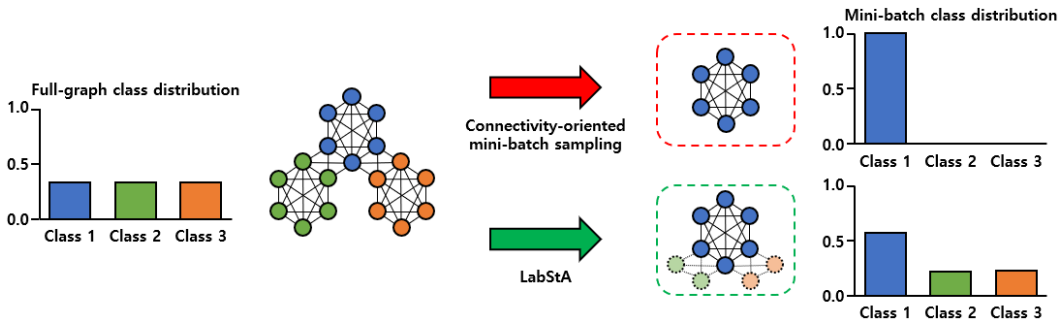

Figure 1: Illustration of the class-distribution bias induced by connectivity-oriented graph sampling. On homophilous graphs, where edges predominantly connect nodes of the same class, connectivity-oriented sampling can produce mini-batches whose class distributions deviate substantially from the full-graph distribution. By contrast, LabStA augments each partition with informative virtual nodes to reduce class-distribution mismatch while preserving edge connectivity.

Prior work on GNNs has developed various graph sampling methods for mini-batch training. Early approaches control neighborhood explosion through node-wise sampling (Hamilton et al., 2017; Ying et al., 2018) or layer-wise sampling (Chen et al., 2018; Zou et al., 2019). Other methods adopt connectivity-oriented sampling. For example, GraphSAINT (Zeng et al., 2019) constructs mini-batches by sampling node-, edge-, or random-walk-based subgraphs, whereas ClusterGCN (Chiang et al., 2019) partitions the graph into clusters and forms mini-batches from one or more clusters. Primarily designed for message-passing GNNs, these methods emphasize edge preservation to maintain consistency between the local neighborhoods seen during mini-batch training and those available at full-graph inference. However, GTs additionally rely on global self-attention, which introduces an additional source of relational information beyond local neighborhoods. This raises a natural question: *do connectivity-oriented sampling methods remain effective for Graph Transformers?*

Figure 1 illustrates a side effect of connectivity-oriented sampling on homophilous graphs, where edges predominantly connect nodes of the same class. By grouping densely connected regions, such sampling can produce mini-batches whose class distributions are concentrated in a few classes and thus deviate from the full-graph distribution. This mismatch can hinder GT training by restricting self-attention's ability to capture meaningful cross-class interactions. At the same time, preserving edge connectivity remains important because modern GTs often incorporate local message passing. These findings motivate two objectives for GT mini-batch sampling: (i) aligning mini-batch class distributions with the full-graph distribution for global self-attention, and (ii) preserving edge connectivity for local message passing. Motivated by these objectives, we propose **LabStA**, a Label- and Structure-Aware mini-batch sampling method for node classification with Graph Transformers, designed to jointly improve class-distribution alignment and edge preservation.

Contributions. Our main contributions are summarized as follows.

- Through controlled experiments, we motivate two key objectives for mini-batch sampling in GT-based node classification: reducing class-distribution discrepancy (CDD) and increasing edge retention rate (ERR). We show that reducing CDD is particularly important for global self-attention, while increasing ERR benefits both message-passing GNNs and GTs with local message passing.
- We propose **LabStA**, a Label- and Structure-Aware mini-batch sampling method that augments graph partitions with informative virtual nodes to jointly improve class-distribution alignment and edge preservation. LabStA is model-agnostic and can be applied to diverse GT backbones without architectural modifications.
- We evaluate LabStA across diverse node classification benchmarks and multiple GT backbones. LabStA outperforms existing sampling baselines on most dataset–backbone pairs, with particularly strong gains on heterophilous graphs, demonstrating the benefit of jointly addressing the two optimization objectives.

2 Related Work

2.1 Graph Transformers

Graph Transformers (GTs) extend the Transformer architecture (Vaswani et al., 2017) to graph data, offering a promising alternative to conventional GNNs by modeling long-range interactions. However, vanilla self-attention does not explicitly account for graph topology, which may restrict the ability of GTs to exploit relational inductive biases (Ying et al., 2021). To incorporate structural information, one line of work injects positional encodings into node features or attention biases, such as Laplacian eigenvectors (Dwivedi & Bresson, 2020; Ito et al., 2025) and random-walk probabilities (Dwivedi et al., 2021; Ma et al., 2023). Another line of work combines local message passing with global self-attention to capture both neighborhood-level and long-range dependencies. GraphGPS (Rampášek et al., 2022) integrates local message passing with global attention within each layer. SGFormer (Wu et al., 2023b) combines a shallow GNN branch with a global-attention module. Polynormer (Deng et al., 2024) adopts a local-to-global design, sequentially applying local and global attention. In this work, we focus on GTs with local message passing because their computations are directly affected by mini-batch construction, whereas positional encodings are typically precomputed from the full graph.

2.2 Graph Sampling Methods

Graph sampling has been widely studied in the context of GNNs to enable mini-batch training on large-scale graphs. Early approaches control neighborhood explosion through node-wise sampling (Hamilton et al., 2017; Ying et al., 2018) or layer-wise sampling (Chen et al., 2018; Zou et al., 2019). Other methods adopt connectivity-oriented sampling. GraphSAINT (Zeng et al., 2019) constructs mini-batches by sampling node-, edge-, or random-walk-based subgraphs, while ClusterGCN (Chiang et al., 2019) partitions the graph into clusters and forms mini-batches from one or more clusters. However, these approaches were primarily designed for message-passing GNNs and thus may be less suitable for attention-based GTs. By contrast, LabStA jointly promotes class-distribution alignment and edge preservation, supporting both global self-attention and local message passing in modern GTs.

Meanwhile, systematic studies of GT mini-batch sampling remain limited, and many prior works rely on random node sampling in large-scale experiments (Wu et al., 2022; 2023b; Deng et al., 2024). Although simple and broadly applicable, we later show that random sampling can lead to inferior node classification performance. A separate line of work develops tokenized GTs that perform training and inference on preconstructed token sequences (Chen et al., 2022; Fu et al., 2024). These methods improve scalability by applying attention over compact token sequences, but depend on method-specific tokenization procedures that may limit the structural information exposed to the Transformer. In contrast, LabStA constructs mini-batches that preserve edge connectivity, retaining more of the local neighborhoods used by GT message-passing modules. Other scalable GTs jointly design sampling methods and model architectures to reduce memory and computational costs (Zhang et al., 2022; Kong et al., 2023). While effective, their sampling procedures are coupled with model-specific attention or hierarchical components, limiting their use as general-purpose samplers. By contrast, LabStA provides a model-agnostic mini-batch sampling framework that can be applied to diverse GT backbones without architectural modifications.

3 Preliminaries

3.1 Notation

Let $G = (V, E)$ be an undirected graph with $n = |V|$ nodes. Let $X \in \mathbb{R}^{n \times d}$ denote the node feature matrix, where $\mathbf{x}_v \in \mathbb{R}^d$ is the feature vector of node v. Let $A \in \{0, 1\}^{n \times n}$ denote the adjacency matrix, where $A_{uv} = 1$ if and only if nodes u and v are connected. Each node v is associated with a class label $y_v \in C$, where C denotes the set of classes. We consider transductive node classification, in which X and A are available for all nodes during training, while labels are observed only for the training nodes. We also adopt a standard mini-batch training setting, where the model is trained on sampled mini-batches and evaluated on the full graph.

3.2 TRANSFORMER ARCHITECTURE

Vaswani et al. (2017) introduce the Transformer architecture, in which each token aggregates information from all tokens through global self-attention. Given an input representation H, self-attention first projects H into query, key, and value matrices:

$$Q = HW_Q, \quad K = HW_K, \quad V = HW_V, \tag{1}$$

where W_Q, W_K, W_V are learnable projection matrices. The attention output is then computed as

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{2}$$

where d_k is the key dimension. This operation computes a weighted aggregation of value vectors, with attention weights determined by pairwise query-key similarities.

3.3 MINI-BATCH STATISTICS

Class-distribution discrepancy (CDD) quantifies the discrepancy between each mini-batch class distribution and the full-graph class distribution. We define

$$\mathrm{CDD} = \frac{\sum_b n_b \cdot \frac{1}{2}\sum_c \left|\frac{n_{b,c}}{n_b} - \frac{n_c}{n}\right|}{\left(\sum_b n_b\right)\left(1 - \min_c \frac{n_c}{n}\right)}, \tag{3}$$

where n_b denotes the number of nodes in mini-batch b, n_c the total number of nodes in class c, and $n_{b,c}$ the number of class-c nodes in mini-batch b. Therefore, CDD is the node-weighted average of the total variation distance between the mini-batch and full-graph distributions, normalized to $[0, 1]$. A larger CDD indicates greater class-distribution mismatch.

Edge retention rate (ERR) measures the fraction of unique graph edges preserved across mini-batches. We define

$$\mathrm{ERR} = \frac{\left|\bigcup_b E_b\right|}{|E|}, \tag{4}$$

where E_b denotes the set of edges whose endpoints are both included in mini-batch b. Note that we take the union over mini-batches rather than summing $|E_b|$ to avoid repeated counting of edges that appear in multiple mini-batches. A larger ERR indicates greater preservation of full-graph edge connectivity across the sampled mini-batches.

4 MOTIVATING EXAMPLES

4.1 CLASS-DISTRIBUTION DISCREPANCY VS. ACCURACY

Prior studies have shown that skewed class distributions across mini-batches can hinder gradient-based optimization, partly by increasing gradient variance (Kunstner et al., 2024; Li et al., 2025). However, the effect of class-distribution mismatch on GTs, particularly on global self-attention, remains relatively underexplored. Here, we examine the effect of CDD on message-passing GNNs and attention-based GTs, hypothesizing that reducing CDD is more important for global self-attention than for local message passing.

Experiment. To evaluate the effect of CDD on node classification, we design a controlled mini-batch sampling strategy. Specifically, we construct $|C|$ mini-batches and assign each mini-batch a representative class. For each mini-batch, a fraction ρ of nodes is sampled from its representative class, while the remaining $1 - \rho$ fraction is sampled uniformly across all classes. Varying ρ over $\{0.0, 0.1, \ldots, 1.0\}$ progressively increases CDD. To ensure equal mini-batch sizes, we undersample majority classes to obtain a balanced class distribution. We compare GAT (message-passing-only), Transformer (self-attention-only), and GraphGPS (local-and-global aggregation).

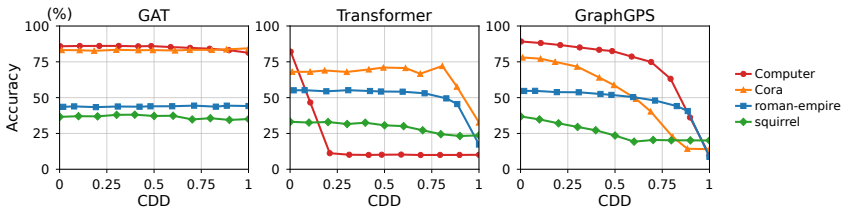

Figure 2: Effect of CDD on node classification accuracy. As CDD increases, GAT exhibits only minor accuracy degradation, whereas Transformer and GraphGPS show substantial accuracy drops, with performance approaching random-guess levels at high CDD.

Results. Figure 2 plots accuracy as CDD varies. GAT, which relies solely on message passing, exhibits only minor accuracy degradation as CDD increases, suggesting that the **optimization effect associated with increased gradient variance is modest** in this setting. In contrast, Transformer and GraphGPS, both of which employ global self-attention, show substantial accuracy drops, with performance approaching random-guess levels at high CDD. Notably, GraphGPS degrades more gradually than Transformer, potentially due to its additional local message-passing component, although both reach similarly low accuracy at $\mathrm{CDD} = 1$. Overall, these results indicate that class-distribution alignment is more critical for attention-based GTs than for message-passing GNNs, likely due to the attention-candidate mismatch between mini-batch training and full-graph inference.

4.2 Edge Retention Rate vs. Accuracy

Edge preservation is less essential for GTs that incorporate topology information through positional encodings, as these encodings are typically derived from the full graph before training. In contrast, GTs with local message passing depend on the sampled graph structure, making edge preservation important for mini-batch training. Here, we examine the effect of ERR on message-passing GNNs and attention-based GTs, showing that increasing ERR similarly benefits message-passing GNNs and GTs with local message passing.

Experiment. To assess the effect of ERR on node classification, we control the number of edges retained during mini-batch sampling. Specifically, we randomly remove a fraction p of edges from ClusterGCN mini-batches. Varying $p \in \{0.0, 0.1, \ldots, 1.0\}$ progressively decreases ERR. Note that we report accuracy normalized by its value at $\mathrm{ERR} = 1$, which corresponds to standard ClusterGCN mini-batch training. As in Section 4.1, we compare GAT, Transformer, and GraphGPS.

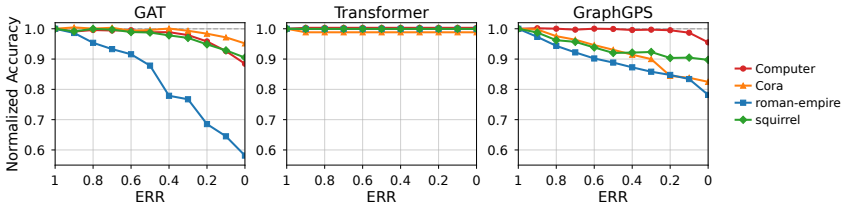

Figure 3: Effect of ERR on node classification accuracy. As ERR decreases, Transformer remains nearly unchanged, whereas GAT and GraphGPS exhibit declines in normalized accuracy, highlighting the sensitivity of local message passing to edge preservation.

Results. Figure 3 plots normalized accuracy as ERR varies. Transformer, which relies solely on global self-attention, remains nearly unchanged across ERR levels because graph connectivity is not incorporated during training. This helps explain why **random node sampling could have been effective for GTs without local message passing**, as adopted in prior large-scale experiments (Wu et al., 2022; 2023a). Nevertheless, GraphGPS exhibits clear performance degradation as ERR decreases, with trends broadly comparable to those of GAT on Computer, Cora, and squirrel. Overall, these results show that architectures with local message passing are similarly sensitive to edge preservation, motivating connectivity-aware mini-batch sampling for node classification with GTs. Theoretical analyses of both objectives are provided in Appendix A.

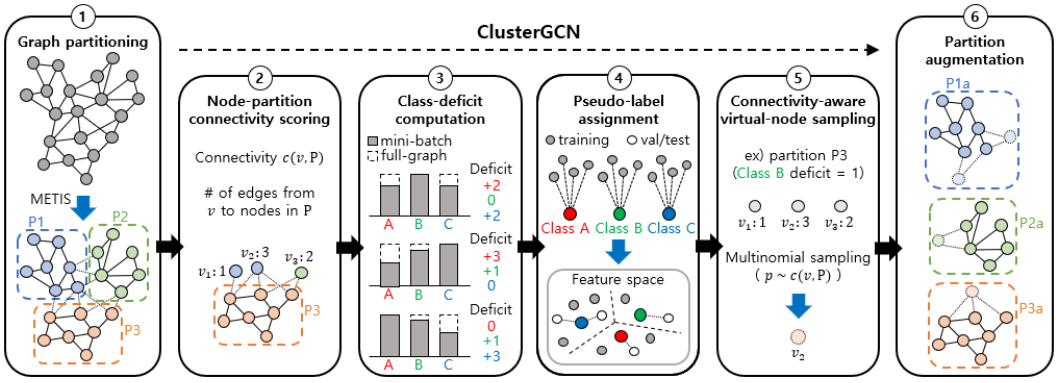

Figure 4: Overview of the proposed LabStA framework. LabStA builds on ClusterGCN by augmenting initial graph partitions with informative virtual nodes. Steps 3 and 4 reduce CDD by restricting candidate nodes to classes that are underrepresented relative to the full-graph distribution. Steps 2 and 5 increase ERR by prioritizing nodes that are strongly connected to the target partition.

5 Proposed Method: LabStA

We propose **LabStA**, a Label- and Structure-Aware mini-batch sampling method for node classification with GTs. Figure 4 illustrates the overall framework. LabStA builds on ClusterGCN (Chiang et al., 2019) by augmenting initial graph partitions with informative nodes to jointly address the two objectives discussed in Section 4. (i) To reduce CDD, LabStA restricts candidate nodes to classes that are underrepresented relative to the full-graph class distribution. (ii) To increase ERR, LabStA prioritizes nodes that are strongly connected to the target partition. Note that the selected nodes are treated as virtual nodes: they participate in global self-attention and local message passing, but are excluded from loss computation.

5.1 Reducing Class-Distribution Discrepancy

As depicted in Figure 1, connectivity-oriented sampling can induce high CDD, resulting in increased gradient variance and attention-candidate mismatch. To mitigate this issue, LabStA augments graph partitions with virtual nodes that improve class-distribution alignment. For each partition, Step 3 computes the class deficit, i.e., the number of virtual nodes required from each class to match the full-graph class distribution, and restricts virtual-node candidates to deficient classes. In this case, val/test nodes cannot be selected as virtual nodes because their ground-truth labels are unavailable during training. Step 4 therefore assigns pseudo-labels to val/test nodes according to their distances from class-wise feature means computed over training nodes. This enables high-connectivity val/test nodes to serve as virtual-node candidates, preserving useful structural information associated with them. Since virtual nodes are excluded from loss computation, their labels do not directly contribute to the optimization objective; instead, they influence GT mini-batch training through their features and corresponding self-attention weights.

5.2 Increasing Edge Retention Rate

LabStA further supports local message passing in modern GTs by restoring edges lost during graph partitioning. For each partition, Step 2 computes node–partition connectivity, defined as the number of edges between each candidate node and the target partition. Step 5 then selects high-connectivity nodes within each deficient class among labeled training nodes and pseudo-labeled val/test nodes. Here, a straightforward approach is to select top-k nodes by connectivity, but repeated sorting can be computationally expensive. Instead, LabStA samples virtual nodes from a multinomial distribution parameterized by node–partition connectivity. As a result, nodes that recover more edges are selected with higher probability, providing an efficient approximation to top-k node selection. If the virtual-node budget is not fully utilized, the remaining virtual nodes are sampled uniformly from zero-connectivity candidates. Complexity analysis is provided in Appendix D.

Table 1: Node classification accuracy (%, mean ± std over 10 runs) across sampling methods and model architectures on small- and medium-scale datasets. For each dataset–backbone pair, the best result is shown in **bold**.

Heterophilous datasets							
Model	Sampler	r.-empire	a.-ratings	squirrel	chameleon	Actor	Average
GraphGPS	Random	67.58±0.68	45.25±0.39	33.35±1.59	51.21±1.95	35.97±1.31	46.67
	GraphSAINT	71.01±0.83	46.42±0.39	33.89±2.07	52.54±1.78	35.61±1.35	47.89
	ClusterGCN	**82.05±0.75**	45.68±0.79	37.35±1.45	51.64±2.28	35.97±1.22	50.54
	LabStA (ours)	81.88±0.48	**47.16±0.74**	**39.07±3.20**	**53.51±1.53**	**36.16±1.27**	**51.56**
SGFormer	Random	66.24±0.62	43.81±0.60	31.62±2.14	49.39±2.11	36.48±0.93	45.51
	GraphSAINT	67.91±0.57	43.61±0.51	31.20±1.70	48.14±2.11	35.68±1.25	45.31
	ClusterGCN	75.72±0.76	45.12±0.76	31.75±1.54	52.04±2.25	36.38±0.98	48.20
	LabStA (ours)	**75.98±0.40**	**45.28±0.76**	**32.27±1.81**	**52.70±2.34**	**36.68±0.76**	**48.58**
Polynormer	Random	71.01±0.59	45.07±0.74	33.02±1.30	49.14±2.47	35.73±1.28	46.79
	GraphSAINT	78.48±0.59	48.32±0.51	35.38±1.79	52.21±1.19	35.59±1.22	50.00
	ClusterGCN	90.98±0.39	51.94±0.53	40.90±1.24	56.84±2.69	35.72±1.17	55.28
	LabStA (ours)	**91.13±0.48**	**52.12±0.70**	**42.80±1.25**	**59.01±1.94**	**36.12±1.08**	**56.24**

Homophilous datasets							
Model	Sampler	Computer	Photo	Cora	CiteSeer	PubMed	Average
GraphGPS	Random	89.67±0.33	95.31±0.26	63.17±2.14	56.18±2.35	71.05±0.90	75.08
	GraphSAINT	90.92±0.29	95.56±0.36	**70.21±1.67**	**61.48±2.61**	**73.47±1.44**	**78.33**
	ClusterGCN	90.43±0.39	95.83±0.21	62.59±1.31	53.22±2.57	72.01±1.68	74.82
	LabStA (ours)	**91.54±0.32**	**95.90±0.23**	63.34±1.46	55.46±2.79	72.57±1.68	75.76
SGFormer	Random	85.67±0.46	93.13±0.31	73.39±1.56	63.17±1.47	76.15±0.78	78.30
	GraphSAINT	86.09±0.26	**94.33±0.46**	79.02±1.54	65.39±1.63	77.12±0.90	80.39
	ClusterGCN	**87.11±0.50**	94.08±0.47	78.83±1.10	64.76±1.78	77.43±0.63	80.44
	LabStA (ours)	86.95±0.51	94.10±0.60	**79.97±0.75**	**66.01±1.37**	**78.24±0.51**	**81.05**
Polynormer	Random	91.32±0.36	95.66±0.28	73.17±1.53	64.75±1.50	76.09±0.94	80.20
	GraphSAINT	92.11±0.33	95.86±0.26	76.25±1.77	65.91±1.38	76.47±1.79	81.32
	ClusterGCN	92.11±0.35	95.93±0.24	76.83±1.44	66.07±1.35	76.22±1.30	81.43
	LabStA (ours)	**92.36±0.34**	**95.95±0.16**	**77.33±0.85**	**67.61±1.22**	**76.86±0.88**	**82.02**

6 EXPERIMENT

6.1 EXPERIMENTAL SETUP

Datasets. We evaluate LabStA on diverse node classification benchmarks. The heterophilous datasets include roman-empire, amazon-ratings (Platonov et al., 2023), squirrel, chameleon (Rozemberczki et al., 2021), and Actor (Pei et al., 2020). The homophilous datasets include Computer, Photo (Shchur et al., 2018), Cora, CiteSeer, and PubMed (Yang et al., 2016). To assess scalability, we further consider three large-scale datasets: ogbn-arxiv, ogbn-products (Hu et al., 2020), and pokec (Lim et al., 2021). Additional dataset details are provided in Appendix B.

Baselines. We compare LabStA with representative graph sampling methods across multiple GT backbones. The sampling baselines include random sampling, random-walk-based GraphSAINT (Zeng et al., 2019), and ClusterGCN (Chiang et al., 2019). As GT backbones, we adopt GraphGPS (Rampášek et al., 2022), which performs both local and global aggregation; SGFormer (Wu et al., 2023b), which couples linear attention with graph convolution; and Polynormer (Deng et al., 2024), which adopts a local-to-global architecture. Implementation details are provided in Appendix C.

6.2 MAIN RESULTS

Small- and medium-scale datasets. Table 1 reports node classification accuracy across sampling methods and model architectures on small- and medium-scale datasets. On heterophilous datasets, LabStA achieves the highest average accuracy for all three backbones, outperforming the strongest competing baseline, ClusterGCN, by $1.02\%p$ (GraphGPS), $0.38\%p$ (SGFormer), and $0.96\%p$ (Polynormer). Meanwhile, random node sampling achieves the lowest accuracy in many cases, emphasizing the importance of structure-aware mini-batch construction for GTs with local message passing. These results suggest that jointly improving class-distribution alignment and edge preservation

enhances node classification, particularly under heterophily. On homophilous datasets, LabStA remains competitive and achieves the highest average accuracy for SGFormer (+0.61%p) and Polynormer (+0.59%p). GraphGPS exhibits an exception: GraphSAINT performs particularly well on Cora and CiteSeer, suggesting a favorable interaction between its random-walk-based sampling and the GraphGPS backbone on these datasets (see Appendix E.1 for further analysis). Nevertheless, GraphGPS still underperforms SGFormer and Polynormer on both datasets, indicating that this advantage is specific to GraphGPS rather than a general benefit of GraphSAINT across backbones. Overall, LabStA provides an effective mini-batch sampling strategy broadly applicable across GT architectures, delivering clear gains on heterophilous graphs while maintaining competitive performance on homophilous graphs.

Table 2: GraphGPS node classification accuracy (%, mean ± std over 3 runs) across sampling methods on large-scale datasets. For each dataset, the best result is shown in **bold**.

Model	Sampler	ogbn-arxiv	ogbn-products	pokec	Average
GraphGPS	Random	63.91±0.36	66.09±0.20	62.45±0.09	64.15
	GraphSAINT	69.06±0.22	74.76±0.44	69.39±0.70	71.07
	ClusterGCN	68.52±0.25	75.30±0.59	71.59±0.20	71.80
	LabStA (ours)	**69.22±0.41**	**76.04±0.62**	**72.52±0.43**	**72.59**

Large-scale datasets. Table 2 reports node classification accuracy across sampling methods on large-scale datasets using GraphGPS. Note that we repeat each experiment 3 times due to the cost of full-graph inference, while mini-batch training itself remains affordable. Overall, LabStA consistently outperforms all sampling baselines, achieving an average gain of 0.79%p over the strongest competing method, ClusterGCN. The improvement is particularly pronounced on pokec, which happens to be a heterophilous graph with an edge homophily ratio of 0.45. These results suggest that LabStA remains effective in large-scale settings, while maintaining its advantage under heterophily. Again, random sampling performs substantially worse, trailing LabStA by up to 10.07%p on pokec. Hence, despite its simplicity, random node sampling is strongly discouraged for mini-batch training of GTs due to its inferior performance.

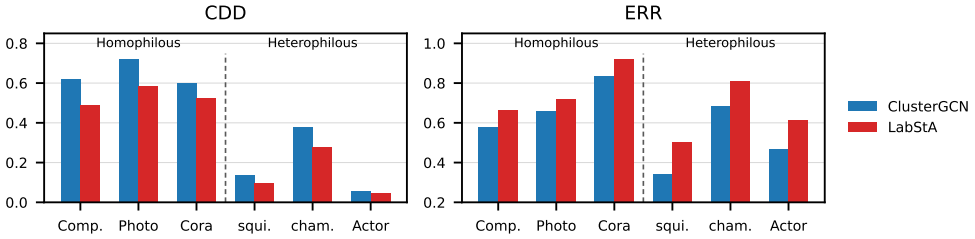

Figure 5: CDD and ERR improvements of LabStA compared to ClusterGCN. Across all datasets, LabStA consistently reduces CDD while increasing ERR, indicating joint improvements in class-distribution alignment and edge preservation. ClusterGCN exhibits particularly high CDD on homophilous graphs, whereas LabStA achieves larger ERR gains on heterophilous graphs.

6.3 Metric Improvements

We further examine whether LabStA achieves its two design goals: reducing class-distribution discrepancy (CDD) and increasing edge retention rate (ERR). Figure 5 compares the CDD and ERR of LabStA with those of ClusterGCN. Across all datasets, LabStA consistently reduces CDD while increasing ERR, indicating joint improvements in class-distribution alignment and edge preservation. In particular, ClusterGCN exhibits high CDD on homophilous graphs, consistent with the tendency of connectivity-oriented sampling to induce greater class-distribution mismatch under homophily (Fig. 1). On the other hand, LabStA achieves larger ERR gains on heterophilous graphs, which aligns with its larger performance improvements under heterophily (Tab. 1). Overall, these results show that LabStA achieves both design goals across homophilous and heterophilous settings, supporting its effectiveness for mini-batch training of GTs.

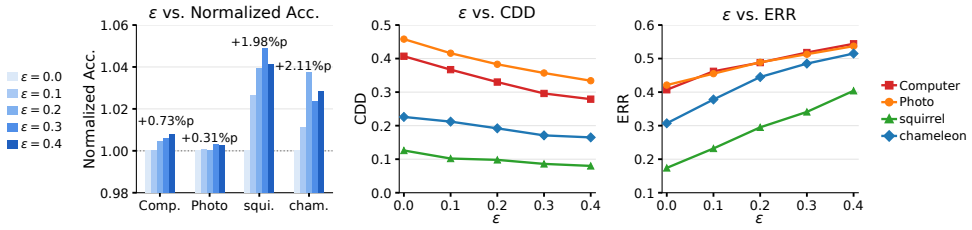

Figure 6: Accuracy, CDD, and ERR as virtual-node ratio ε varies. As ε increases, LabStA generally improves accuracy while reducing CDD and increasing ERR. Accuracy gains are more pronounced on squirrel and chameleon, which also exhibit larger improvements in ERR.

6.4 Hyperparameter Study

The main hyperparameter of LabStA is ε, which controls the extent of virtual-node augmentation applied to each graph partition. Figure 6 plots accuracy, CDD, and ERR as ε varies. Note that accuracy is normalized by its value at $\varepsilon = 0$, which corresponds to ClusterGCN. As ε increases, accuracy generally improves, while CDD decreases and ERR increases. In particular, accuracy gains are pronounced on squirrel and chameleon, which also exhibit larger improvements in ERR. These results indicate that virtual-node augmentation effectively improves both target metrics and that higher ERR is associated with improved node classification accuracy on heterophilous graphs. In the main experiments, we fix $\varepsilon = 0.2$ to balance the benefits of virtual-node augmentation against its computational overhead.

6.5 Ablation Study

Table 3: Ablation study of LabStA virtual-node selection.

Dataset	random	CDD-only	ERR-only	LabStA
squirrel	41.82	41.61	42.66	**43.45**
chameleon	57.63	57.54	57.96	**59.01**
Computer	91.94	92.00	91.62	**92.24**
Cora	76.87	76.96	77.32	**77.47**
CiteSeer	66.74	66.14	66.85	**67.61**

Finally, we conduct an ablation study of LabStA's virtual-node selection step. Specifically, we compare the accuracy of LabStA, which considers both CDD and ERR, with three different variants: random selection, CDD-only selection, and ERR-only selection. As shown in Table 3, LabStA consistently outperforms all three variants, demonstrating the advantage of jointly improving class-distribution alignment and edge preservation. In particular, LabStA consistently improves over random selection, indicating that its gains are not simply due to larger mini-batches. Moreover, the CDD-only and ERR-only variants do not consistently improve over random selection, suggesting that optimizing either objective in isolation can be insufficient. This may indicate a trade-off between the two objectives, which we further analyze in Appendix E.2. Overall, these results show that jointly reducing CDD and increasing ERR yields the best performance, supporting the effectiveness of LabStA for node classification with Graph Transformers.

7 Conclusion

In this work, we identified two key objectives for mini-batch sampling in GT-based node classification: reducing class-distribution discrepancy (CDD) and increasing edge retention rate (ERR). Controlled experiments showed that reducing CDD is particularly important for global self-attention, whereas increasing ERR benefits both message-passing GNNs and GTs with local message passing. Based on these observations, we proposed LabStA, a mini-batch sampling method that augments graph partitions with informative virtual nodes to jointly improve class-distribution alignment and edge preservation. Experiments across diverse node classification benchmarks show that LabStA outperforms existing sampling baselines on most dataset–backbone pairs, with particularly strong gains on heterophilous graphs. Overall, our findings emphasize the importance of label- and structure-aware mini-batch construction for GTs and motivate the development of more advanced sampling strategies beyond conventional random node sampling.

AI DISCLOSURE

In this work, we used generative AI tools to assist with formulating mathematical claims, identifying key components of their proofs, and drafting proof text (Appendix A). We did not use generative AI tools to propose or refine research hypotheses, design the research methodology or experiments, provide feedback on experimental design, implement the proposed method, clean or reformat datasets, conduct qualitative or thematic analysis, or interpret experimental results. The generation of synthetic datasets and the development of theoretical models or conceptual frameworks are not applicable to this work. We additionally used generative AI tools to suggest experimental parameters, create or edit software code, improve the clarity and readability of the manuscript, and identify potentially relevant literature. All AI-assisted content was independently reviewed and verified by the authors. In particular, AI-generated code was tested for correctness by the primary author, and AI-assisted revisions to the manuscript were reviewed to ensure that they preserved the authors' intended meaning. The authors take full responsibility for the final content of this work, including all text, claims, analyses, and artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies mini-batch training for Graph Transformers and does not involve human subjects, sensitive personal data, or deployment in high-risk settings. We follow the ICLR Code of Ethics by reporting our methodology and results transparently and appropriately acknowledging prior work. We are not aware of any ethical risks specific to this work beyond those generally associated with the development and use of machine learning systems.

REPRODUCIBILITY STATEMENT

We provide the information required to reproduce our results in the appendix. Appendix B summarizes the datasets, including their statistics and preprocessing procedures. Appendix C provides the training protocol, sampler configurations, model architectures, and dataset-specific hyperparameters used in our experiments. For further details, refer to the code implementation provided in the supplementary materials.

REFERENCES

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018.

Jinsong Chen, Kaiyuan Gao, Gaichao Li, and Kun He. Nagphormer: A tokenized graph transformer for node classification in large graphs. *arXiv preprint arXiv:2206.04910*, 2022.

Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 257–266, 2019.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Chenhui Deng, Zichao Yue, and Zhiru Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. *arXiv preprint arXiv:2403.01232*, 2024.

Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*, 2021.

Dongqi Fu, Zhigang Hua, Yan Xie, Jin Fang, Si Zhang, Kaan Sancak, Hao Wu, Andrey Malevich, Jingrui He, and Bo Long. Vcr-graphormer: A mini-batch graph transformer via virtual connections. In *International Conference on Learning Representations*, volume 2024, pp. 46501–46520, 2024.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

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. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Michael Ito, Jiong Zhu, Dexiong Chen, Danai Koutra, and Jenna Wiens. Learning laplacian positional encodings for heterophilous graphs. *arXiv preprint arXiv:2504.20430*, 2025.

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

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Kezhi Kong, Jiuhai Chen, John Kirchenbauer, Renkun Ni, C Bayan Bruss, and Tom Goldstein. Goat: A global transformer on large-scale graphs. In *International Conference on Machine Learning*, pp. 17375–17390. PMLR, 2023.

Frederik Kunstner, Alan Milligan, Robin Yadav, Mark Schmidt, and Alberto Bietti. Heavy-tailed class imbalance and why adam outperforms gradient descent on language models. *Advances in Neural Information Processing Systems*, 37:30106–30148, 2024.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Tianjian Li, Haoran Xu, Weiting Tan, Kenton Murray, and Daniel Khashabi. Upsample or upweight? balanced training on heavily imbalanced datasets. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3325–3343, 2025.

Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. *Advances in neural information processing systems*, 34:20887–20902, 2021.

Liheng Ma, Chen Lin, Derek Lim, Adriana Romero-Soriano, Puneet K Dokania, Mark Coates, Philip Torr, and Ser-Nam Lim. Graph inductive biases in transformers without message passing. In *International Conference on Machine Learning*, pp. 23321–23337. PMLR, 2023.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

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? *arXiv preprint arXiv:2302.11640*, 2023.

Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

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

Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J Sutherland, and Ali Kemal Sinop. Exphormer: Sparse transformers for graphs. In *International Conference on Machine Learning*, pp. 31613–31632. PMLR, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in neural information processing systems*, 35:27387–27401, 2022.

Qitian Wu, Chenxiao Yang, Wentao Zhao, Yixuan He, David Wipf, and Junchi Yan. Difformer: Scalable (graph) transformers induced by energy constrained diffusion. *arXiv preprint arXiv:2301.09474*, 2023a.

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. *Advances in neural information processing systems*, 36:64753–64773, 2023b.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pp. 40–48. PMLR, 2016.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 974–983, 2018.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019.

Zaixi Zhang, Qi Liu, Qingyong Hu, and Chee-Kong Lee. Hierarchical graph transformer with adaptive node sampling. *Advances in Neural Information Processing Systems*, 35:21171–21183, 2022.

Difan Zou, Ziniu Hu, Yewen Wang, Song Jiang, Yizhou Sun, and Quanquan Gu. Layer-dependent importance sampling for training deep and large graph convolutional networks. *Advances in neural information processing systems*, 32, 2019.

A Theoretical Analysis

A.1 Class-Distribution Discrepancy and Global Self-Attention

We analyze how class-distribution mismatch affects the population-level self-attention output.

Let the full graph have class distribution

$$\pi = (\pi_1, \ldots, \pi_C), \tag{5}$$

and let a mini-batch have class distribution

$$\hat{\pi} = (\hat{\pi}_1, \ldots, \hat{\pi}_C). \tag{6}$$

We define the class-distribution discrepancy as

$$\mathrm{CDD}(\hat{\pi}, \pi) = \frac{1}{2} \left\| \hat{\pi} - \pi \right\|_1 . \tag{7}$$

For a fixed query representation q_i, let (K_c, V_c) denote the random key–value pair of a candidate node drawn from class c. Define

$$A_{ic} = \mathbb{E}\left[\exp\left(\frac{q_i^\top K_c}{\sqrt{d_k}} \right) \right], \tag{8}$$

and

$$B_{ic} = \mathbb{E}\left[\exp\left(\frac{q_i^\top K_c}{\sqrt{d_k}} \right) V_c \right]. \tag{9}$$

For any class distribution $p = (p_1, \ldots, p_C)$, define the corresponding population-level self-attention output as

$$H_i(p) = \frac{\sum_{c=1}^{C} p_c B_{ic}}{\sum_{c=1}^{C} p_c A_{ic}}. \tag{10}$$

Thus, $H_i(\pi)$ and $H_i(\hat{\pi})$ represent the population-level self-attention outputs induced by the full-graph and mini-batch class distributions, respectively, under fixed class-conditional key–value distributions.

Proposition 1. *Suppose that the class-conditional key–value distributions are the same for the full graph and the mini-batch, and that*

$$\|V_c\|_2 \le V \tag{11}$$

and

$$\left| \frac{q_i^\top K_c}{\sqrt{d_k}} \right| \le L_i \tag{12}$$

almost surely for every class c. Then

$$\|H_i(\hat{\pi}) - H_i(\pi)\|_2 \le 4V e^{2L_i} \mathrm{CDD}(\hat{\pi}, \pi). \tag{13}$$

Proof. For simplicity, write

$$A_c = A_{ic}, \qquad B_c = B_{ic}. \tag{14}$$

Define

$$N = \sum_c \pi_c B_c, \qquad Z = \sum_c \pi_c A_c, \tag{15}$$

and

$$\hat{N} = \sum_c \hat{\pi}_c B_c, \qquad \hat{Z} = \sum_c \hat{\pi}_c A_c. \tag{16}$$

Then

$$H_i(\pi) = \frac{N}{Z}, \qquad H_i(\hat{\pi}) = \frac{\hat{N}}{\hat{Z}}. \tag{17}$$

Since the attention logits are bounded by L_i almost surely,

$$e^{-L_i} \le A_c \le e^{L_i}. \tag{18}$$

Because π and $\hat{\pi}$ are probability distributions, it follows that

$$Z \geq e^{-L_i}, \qquad \hat{Z} \geq e^{-L_i}. \tag{19}$$

Moreover, since $\|V_c\|_2 \leq V$ almost surely,

$$\begin{aligned} \|B_c\|_2 &= \left\| \mathbb{E}\left[e^{q_i^\top K_c/\sqrt{d_k}} V_c \right] \right\|_2 \\ &\leq \mathbb{E}\left[e^{q_i^\top K_c/\sqrt{d_k}} \|V_c\|_2 \right] \\ &\leq V A_c \\ &\leq V e^{L_i}. \end{aligned} \tag{20}$$

Let

$$\delta_c = \hat{\pi}_c - \pi_c. \tag{21}$$

Then

$$\begin{aligned} \|\hat{N} - N\|_2 &= \left\| \sum_c \delta_c B_c \right\|_2 \\ &\leq \sum_c |\delta_c| \|B_c\|_2 \\ &\leq V e^{L_i} \sum_c |\delta_c| \\ &= 2V e^{L_i} \mathrm{CDD}(\hat{\pi}, \pi). \end{aligned} \tag{22}$$

Similarly,

$$\begin{aligned} |\hat{Z} - Z| &= \left| \sum_c \delta_c A_c \right| \\ &\leq \sum_c |\delta_c| A_c \\ &\leq e^{L_i} \sum_c |\delta_c| \\ &= 2 e^{L_i} \mathrm{CDD}(\hat{\pi}, \pi). \end{aligned} \tag{23}$$

Next,

$$\left\| \frac{\hat{N}}{\hat{Z}} - \frac{N}{Z} \right\|_2 \leq \frac{\|\hat{N} - N\|_2}{\hat{Z}} + \|N\|_2 \frac{|\hat{Z} - Z|}{Z\hat{Z}}. \tag{24}$$

Using $\hat{Z} \geq e^{-L_i}$,

$$\frac{\|\hat{N} - N\|_2}{\hat{Z}} \leq 2V e^{2L_i} \mathrm{CDD}(\hat{\pi}, \pi). \tag{25}$$

Furthermore,

$$\begin{aligned} \|N\|_2 &= \left\| \sum_c \pi_c B_c \right\|_2 \\ &\leq \sum_c \pi_c \|B_c\|_2 \\ &\leq V \sum_c \pi_c A_c \\ &= V Z. \end{aligned} \tag{26}$$

Therefore,

$$\begin{aligned} \|N\|_2 \frac{|\hat{Z} - Z|}{Z\hat{Z}} &\leq V \frac{|\hat{Z} - Z|}{\hat{Z}} \\ &\leq 2V e^{2L_i} \mathrm{CDD}(\hat{\pi}, \pi). \end{aligned} \tag{27}$$

Combining the two terms yields

$$\|H_i(\hat{\pi}) - H_i(\pi)\|_2 \leq 4Ve^{2L_i}\mathrm{CDD}(\hat{\pi}, \pi). \tag{28}$$

This completes the proof. □

Interpretation. Proposition 1 shows that the discrepancy between the population-level self-attention outputs induced by the mini-batch and full-graph class distributions is bounded linearly by their CDD. Under fixed class-conditional key–value distributions, $\hat{\pi} = \pi$ implies

$$H_i(\hat{\pi}) = H_i(\pi). \tag{29}$$

For finite mini-batches, however, matching class proportions does not guarantee identical self-attention outputs because different nodes may be sampled within each class. Thus, the proposition isolates the discrepancy due specifically to class-distribution mismatch from other finite-sample variation.

A.2 Edge Retention Rate and Local Message Passing

We analyze how edge removal during mini-batch sampling affects local message passing.

Let

$$G = (V, E) \tag{30}$$

denote the original graph. To isolate the effect of edge removal, we keep the node set and node representations fixed and consider the retained-edge graph

$$G_B = (V, E_B), \qquad E_B \subseteq E. \tag{31}$$

For this analysis, each undirected edge is represented as two directed message-passing edges, and $|E|$ and $|E_B|$ count these directed edges. This convention does not change their ratio. We define the edge retention rate as

$$\mathrm{ERR} = \frac{|E_B|}{|E|}. \tag{32}$$

Consider the message-passing layer

$$h_i^G = \sigma\left(W_0 h_i + \sum_{j:(j,i)\in E} a_{ij} W_1 h_j\right). \tag{33}$$

On the retained-edge graph, the corresponding representation is

$$h_i^B = \sigma\left(W_0 h_i + \sum_{j:(j,i)\in E_B} a_{ij} W_1 h_j\right). \tag{34}$$

Proposition 2. *Suppose that σ is L_σ-Lipschitz, the aggregation coefficients a_{ij} are computed on the full graph and held fixed after edge removal, and*

$$\|a_{ij} W_1 h_j\|_2 \leq M \tag{35}$$

for every directed edge $(j, i) \in E$. Then

$$\frac{1}{|V|}\sum_{i\in V} \|h_i^B - h_i^G\|_2 \leq \frac{L_\sigma M |E|}{|V|}\left(1 - \mathrm{ERR}\right). \tag{36}$$

Proof. Let

$$R = E \setminus E_B \tag{37}$$

denote the set of removed directed edges, and define

$$R_i = \{j : (j, i) \in R\}. \tag{38}$$

The difference between the full-graph and retained-edge pre-activations is

$$\begin{aligned}&\left(W_0 h_i + \sum_{j:(j,i)\in E} a_{ij} W_1 h_j\right) - \left(W_0 h_i + \sum_{j:(j,i)\in E_B} a_{ij} W_1 h_j\right) \\ &= \sum_{j\in R_i} a_{ij} W_1 h_j.\end{aligned} \tag{39}$$

By the L_σ-Lipschitz continuity of σ,

$$\begin{aligned}\|h_i^G - h_i^B\|_2 &\le L_\sigma \left\| \sum_{j\in R_i} a_{ij} W_1 h_j \right\|_2 \\ &\le L_\sigma \sum_{j\in R_i} \|a_{ij} W_1 h_j\|_2 \\ &\le L_\sigma M |R_i|.\end{aligned} \tag{40}$$

Summing over all nodes gives

$$\sum_{i\in V} \|h_i^G - h_i^B\|_2 \le L_\sigma M \sum_{i\in V} |R_i|. \tag{41}$$

Because each removed directed edge contributes to exactly one set R_i,

$$\sum_{i\in V} |R_i| = |R| = |E| - |E_B|. \tag{42}$$

Therefore,

$$\sum_{i\in V} \|h_i^G - h_i^B\|_2 \le L_\sigma M \left(|E| - |E_B|\right). \tag{43}$$

Dividing both sides by $|V|$, we obtain

$$\begin{aligned}\frac{1}{|V|}\sum_{i\in V} \|h_i^G - h_i^B\|_2 &\le \frac{L_\sigma M |E|}{|V|}\left(1 - \frac{|E_B|}{|E|}\right) \\ &= \frac{L_\sigma M |E|}{|V|}\left(1 - \mathrm{ERR}\right).\end{aligned} \tag{44}$$

This completes the proof. □

Interpretation. Proposition 2 shows that the average discrepancy between the full-graph and retained-edge representations is bounded linearly by the fraction of removed edges, $1 - \mathrm{ERR}$. Thus, for a fixed graph, a larger ERR yields a smaller upper bound on the perturbation induced by edge removal in local message passing. As with Proposition 1, this bound does not imply that the realized representation discrepancy increases monotonically as ERR decreases.

The proposition assumes that the aggregation coefficients a_{ij} are computed on the full graph and remain fixed after edge removal. If they are recomputed on the retained-edge graph, as in degree-normalized GCN or attention-based GAT, edge removal can also alter the aggregation weights of retained neighbors. The resulting discrepancy then includes an additional term due to changes in the aggregation coefficients, and ERR alone is insufficient to characterize the exact perturbation without further assumptions.

For a Transformer without topology-dependent local message passing, changing E_B while keeping the node set, node representations, and structural encodings fixed does not affect self-attention. Hence, ERR has no direct effect on the forward computation under these conditions.

Overall, Propositions 1 and 2 characterize two complementary sources of mini-batch discrepancy: CDD bounds a population-level discrepancy in global self-attention, whereas ERR bounds the representation discrepancy induced by edge removal in local message passing. These results provide theoretical mechanisms consistent with the empirical observations in Section 4, without implying that CDD or ERR alone determines downstream classification accuracy.

B Dataset Details

Table 4: Summary of dataset statistics.

Type	Dataset	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert C\rvert$	d
Heterophilous	roman-empire	22,662	65,854	18	300
	amazon-ratings	24,492	186,100	5	300
	squirrel	5,201	217,073	5	2,089
	chameleon	2,277	36,101	5	2,325
	Actor	7,600	30,019	5	932
Homophilous	Computer	13,752	491,722	10	767
	Photo	7,650	238,162	8	745
	Cora	2,708	10,556	7	1,433
	CiteSeer	3,327	9,104	6	3,703
	PubMed	19,717	88,648	3	500
Large-scale	ogbn-arxiv	169,343	1,166,243	40	128
	ogbn-products	2,449,029	123,718,280	47	100
	pokec	1,632,803	30,622,564	2	65

B.1 Dataset Statistics

Table 4 summarizes the node classification datasets used in our experiments, reporting the number of nodes ($|V|$), edges ($|E|$), classes ($|C|$), and feature dimensions (d). The benchmark suite includes both heterophilous and homophilous graphs and covers a broad range of graph sizes, class counts, and feature dimensionality. We additionally consider three large-scale datasets—ogbn-arxiv, ogbn-products, and pokec—to assess the scalability of LabStA. Notably, ogbn-products and pokec contain millions of nodes and tens of millions of edges, representing settings in which full-batch training is impractical and mini-batch training is essential.

B.2 Preprocessing Details

For all datasets, we use the official or publicly available fixed data splits. We convert each graph to an undirected graph, remove existing self-loops, and add exactly one self-loop to each node. We apply feature normalization to datasets squirrel, chameleon, Actor, Computer, Photo, Cora, CiteSeer, and PubMed. For all other datasets, we retain the original features without additional normalization, following standard preprocessing practices for these benchmarks.

C Implementation Details

C.1 Training Protocol

For small- and medium-scale datasets, we set the hidden dimension to 64 and use 8 attention heads for all architectures. We train all models with Adam (Kingma & Ba, 2014) using a learning rate of 0.001, a weight decay of 5×10^{-4}, a hidden-layer dropout of 0.5, and the negative log-likelihood loss. Unless otherwise specified, models are trained for up to $2{,}000$ epochs with early stopping based on validation accuracy and a patience of 150 epochs. We repeat each experiment over 10 runs and use all sampled mini-batches in each epoch.

For large-scale datasets, we repeat each experiment over 3 runs and randomly sample 100 mini-batches per epoch. We perform full-graph inference every 50 epochs to reduce CPU inference overhead. Early stopping is not used for the large-scale experiments.

C.2 Graph Sampling Methods

We summarize the hyperparameter settings for each sampling method below. Note that all hyperparameters are selected based on validation accuracy. For a fair comparison, we use a fixed mini-batch size for each dataset across all sampling methods, as reported in Table 5.

Table 5: Dataset-specific hyperparameters.

Type	Dataset	ClusterGCN		Polynormer			
		B	q	e_{wu}	e_{lg}	n_l	n_g
Heterophilous	roman-empire	1024	4	100	2500	10	2
	amazon-ratings	1024	8	200	2500	10	1
	squirrel	256	2	100	2000	10	2
	chameleon	128	2	100	1500	10	2
	Actor	256	1	100	2000	10	3
Homophilous	Computer	512	8	200	1000	5	1
	Photo	256	4	200	1000	7	2
	Cora	128	4	200	1000	7	2
	CiteSeer	128	2	200	1000	7	2
	PubMed	1024	4	100	1500	5	2
Large-scale	ogbn-arxiv	10000	4	—	—	—	—
	ogbn-products	10000	8	—	—	—	—
	pokec	10000	8	—	—	—	—

- **Random**: Random sampling does not introduce additional hyperparameters.
- **GraphSAINT**: We use the random-walk-based sampler with a walk length of 2. To match the mini-batch count and size, we halve the number of random-walk starting nodes.
- **ClusterGCN**: We select the number of partitions per mini-batch from $q \in \{1, 2, 4, 8\}$. The values selected for Polynormer are reported in Table 5.
- **LabStA**: For a fair comparison, we use the same number of partitions per mini-batch q as ClusterGCN. The virtual-node ratio is fixed at $\varepsilon = 0.2$.

C.3 Model Architectures

GraphGPS (Rampášek et al., 2022) follows the GPS architecture, where each layer combines a local message-passing module with a global attention module; in our implementation, we use GAT (Veličković et al., 2017) for local propagation and Performer (Choromanski et al., 2020) for global attention. For SGFormer (Wu et al., 2023b), we follow the official architecture, which combines a single-layer global attention module with a shallow GNN branch and aggregates their representations through a weighted combination. For Polynormer (Deng et al., 2024), we follow its local-to-global architecture, which first applies GAT-based local attention to capture graph structure and then performs linear global attention over the resulting node representations. Table 5 reports the Polynormer-specific hyperparameters, including the number of warm-up epochs e_{wu}, local-to-global epochs e_{lg}, local layers n_l, and global layers n_g.

D Time Complexity Analysis

We analyze the time complexity of each step of LabStA and compare its preprocessing cost with that of a single efficient GT layer, which combines linear attention ($\mathcal{O}(|V|)$) with local message passing ($\mathcal{O}(|V| + |E|)$). (i) Node–partition connectivity scoring (Step 2) scans the edge set to compute the connectivity between nodes and partitions, requiring $\mathcal{O}(|E|)$ time. (ii) Class-deficit computation (Step 3) requires $\mathcal{O}(|V| + P|C|)$ time, where P is the number of partitions. With a fixed number of classes $|C|$ and a fixed mini-batch size $B = |V|/P$, this reduces to $\mathcal{O}(|V|)$. (iii) Pseudo-label assignment (Step 4) computes class-wise feature means and assigns pseudo-labels in $\mathcal{O}(|V||C|d)$ time, which reduces to $\mathcal{O}(|V|)$ when $|C|$ and the feature dimension d are treated as constants. (iv) Connectivity-aware virtual-node sampling (Step 5) processes positive-connectivity candidates in $\mathcal{O}(|E|)$ time and handles any remaining zero-connectivity candidates in $\mathcal{O}(|V|)$ time, yielding an overall complexity of $\mathcal{O}(|V| + |E|)$. Overall, LabStA introduces an additional $\mathcal{O}(|V| + |E|)$ preprocessing overhead relative to ClusterGCN, matching the asymptotic complexity of a single efficient GT layer. Moreover, this preprocessing is performed only once and is therefore amortized over all training epochs.

Table 6: Average number of connected components per mini-batch. Higher values indicate greater subgraph fragmentation.

Dataset	GraphSAINT	ClusterGCN
squirrel	7.44	2.43
chameleon	6.64	4.00
Actor	18.83	3.27
Photo	16.44	5.90
Cora	31.45	5.00
CiteSeer	43.77	17.92

Table 7: Node classification accuracy (%, mean ± std over 5 runs) as the number of METIS regions per mini-batch R varies. For each backbone, the best result is shown in **bold**.

Dataset	R	GraphGPS	SGFormer	Polynormer
Cora	1	68.80±1.34	76.17±1.04	**77.33±0.81**
	2	69.62±1.54	76.07±1.70	75.70±1.11
	4	**70.28±1.91**	**78.83±1.97**	75.67±0.21
CiteSeer	1	63.78±1.65	64.30±1.18	**67.17±0.21**
	2	63.44±0.37	**64.63±0.64**	66.13±0.99
	4	**64.72±1.32**	64.13±2.80	64.37±0.80

E Additional Experiments

E.1 Exception Analysis

Table 1 exhibits an exception on the homophilous benchmarks: GraphSAINT performs particularly well on Cora and CiteSeer under the GraphGPS backbone, whereas this advantage does not extend to SGFormer and Polynormer. We analyze this phenomenon from three different perspectives and formulate a hypothesis to explain the observed sampler–backbone interaction.

- **Sampling mechanism**: GraphSAINT differs from ClusterGCN and LabStA in how it constructs mini-batches. Its random-walk sampler collects nodes visited by walks initiated from multiple seed nodes, whereas ClusterGCN and LabStA form mini-batches from graph partitions that preserve densely connected regions. As a result, GraphSAINT may cover a broader set of local regions within each mini-batch. Table 6 supports this distinction: on datasets including Cora and CiteSeer, GraphSAINT produces mini-batches with more connected components than ClusterGCN, indicating greater subgraph fragmentation.
- **Backbone interaction**: Although GraphGPS, SGFormer, and Polynormer all incorporate local propagation, they differ in how local and global information are integrated. GraphGPS combines local message passing and global attention within each layer, whereas SGFormer combines a shallow GNN branch with a global-attention module, and Polynormer follows a sequential local-to-global design. We therefore hypothesize that GraphGPS may be more sensitive to sampler-induced differences in subgraph structure due to its repeated within-layer integration of local message passing and global attention.
- **Dataset interaction**: Cora and CiteSeer are homophilous citation networks in which neighboring nodes frequently share class labels. On such graphs, partition-based sampling may concentrate mini-batches within a small number of structurally coherent regions, whereas random-walk sampling may cover a broader range of local neighborhoods. Taken together, we hypothesize that GraphSAINT may provide greater local structural diversity, which interacts favorably with GraphGPS's repeated integration of local and global propagation and contributes to its unusually strong performance on Cora and CiteSeer.

To test this hypothesis, we construct mini-batches by combining $R \in \{1, 2, 4\}$ METIS partitions while keeping the average batch size and edge retention approximately constant. As shown in Table 7, GraphGPS achieves its highest mean accuracy at $R = 4$ on both datasets, improving from

Table 8: CDD of LabStA variants. Best result is in **bold**, while worst result is underlined.

Dataset	LabStA	CDD-only	ERR-only
r.-empire	**0.040**	**0.040**	0.047
a.-ratings	0.070	**0.061**	0.083
squirrel	0.098	**0.092**	0.114
chameleon	0.192	**0.180**	0.200
Actor	**0.042**	**0.042**	0.049
Computer	0.330	**0.321**	0.385
Photo	0.383	**0.366**	0.448
Cora	0.295	**0.271**	0.324
CiteSeer	0.237	**0.221**	0.257
PubMed	0.204	**0.195**	0.219

Table 9: ERR of LabStA variants. Best result is in **bold**, while worst result is underlined.

Dataset	LabStA	CDD-only	ERR-only
r.-empire	**1.000**	0.996	**1.000**
a.-ratings	0.989	0.958	**0.998**
squirrel	**0.295**	0.194	0.293
chameleon	0.445	0.321	**0.458**
Actor	0.518	0.392	**0.519**
Computer	0.488	0.416	**0.535**
Photo	0.489	0.427	**0.543**
Cora	0.698	0.590	**0.750**
CiteSeer	0.820	0.737	**0.880**
PubMed	0.831	0.719	**0.868**

68.80% to 70.28% on Cora and from 63.78% to 64.72% on CiteSeer. In contrast, SGFormer shows no consistent trend as R increases, whereas Polynormer performs best at $R = 1$ on both datasets and decreases as R increases. These results provide partial support for our hypothesis that broader local structural coverage can benefit GraphGPS more than other architectures. Overall, they suggest that GraphSAINT's unusual advantage on Cora and CiteSeer may partly arise from a dataset- and backbone-specific interaction between broader structural coverage induced by sampling and the GraphGPS architecture.

E.2 Trade-Off Analysis

Section 6.5 suggests a potential trade-off between the two objectives of LabStA: reducing CDD and increasing ERR. We examine this trade-off by comparing the mini-batch statistics of LabStA with its CDD-only and ERR-only variants. Tables 8 and 9 report the corresponding CDD and ERR values. CDD-only yields the strongest CDD reduction, while ERR-only generally yields the strongest ERR improvement. Importantly, each variant performs worst on the other metric: CDD-only yields the lowest ERR, whereas ERR-only yields the highest CDD. These results confirm a trade-off between class-distribution alignment and edge preservation, with LabStA balancing the two objectives to jointly improve both metrics.