Improving Graph Contrastive Learning with Community Structure

Xiang Chen[1,2] **Kun Yue**[1,2] **Liang Duan**[1,2] **Lixing Yu**[1,2]

[1]School of Information Science and Engineering, Yunnan University, Kunming, China
[2]Yunnan Key Laboratory of Intelligent Systems and Computing, Yunnan University, Kunming, China

Abstract

Graph contrastive learning (GCL) has demonstrated remarkable success in training graph neural networks (GNNs) by distinguishing positive and negative node pairs without human labeling. However, existing GCL methods often suffer from two limitations: the repetitive message-passing mechanism in GNNs and the quadratic computational complexity of exhaustive node pair sampling in loss function. To address these issues, we propose an efficient and effective GCL framework that leverages community structure rather than relying on the intricate node-to-node adjacency information. Inspired by the concept of sparse low-rank approximation of graph diffusion matrices, our model delivers node messages to the corresponding communities instead of individual neighbors. By exploiting community structures, our method significantly improves GCL efficiency by reducing the number of node pairs needed for contrastive loss calculation. Furthermore, we theoretically prove that our model effectively captures essential structure information for downstream tasks. Extensive experiments conducted on real-world datasets illustrate that our method not only achieves the state-of-the-art performance but also substantially reduces time and memory consumption compared with other GCL methods. Our code is available at https://github.com/chenx-hi/IGCL-CS.

1 INTRODUCTION

Graph neural networks (GNNs) are essential for analyzing complex graph-structured data by learning effective node representations that capture rich structural information through message passing on the graph topology [Veličković et al., 2018]. Most GNNs are trained in a semi-supervised manner, where their performance heavily depends on the availability of labeled nodes [Zheng et al., 2022]. However, obtaining these labels is often expensive and labor-intensive. To address this challenge, graph contrastive learning (GCL) has emerged as a successful method for training GNNs without requiring specific task labels, making it particularly useful in fields such as social network analysis and recommendation systems [Ju et al., 2024].

The core technology of GCL revolves around optimizing a contrastive loss that discriminates positive and negative node pairs to maximize feature consistency across augmented graph views [Zhu et al., 2021, Ko et al., 2023]. While some GCL variants enhance performance by improving the quality and quantity of sampled node pairs and achieve results comparable to or even surpassing those of methods trained with ground truth labels [Shen et al., 2023, Wen et al., 2024], their scalability remains severely constrained by two inherent bottlenecks: (1) the intensive message passing in GNNs, and (2) the quadratic computational complexity of node pairs in contrastive loss. Several methods improve efficiency by simplifying the computation process of GCL, such as eliminating the need for negative node pairs in contrastive loss [Thakoor et al., 2022, Sun et al., 2024], or reducing the number of graphs that require encoding by GNNs [Mo et al., 2022]. Despite these improvements, the high computational cost of the message-passing mechanism remains a bottleneck. Other strategies aim to speedup GNN by decoupling graph convolution and embedding transformation [Wu et al., 2019] or performing message passing on subgraphs [Huang et al., 2021]. However, these techniques are not directly applicable to unlabeled scenarios.

Community structures, characterized by dense internal connections and sparse external connections, are prevalent in many real-world graphs [Li and Pan, 2016]. This inherent property aligns well with the diffusion process on graphs [Girvan and Newman, 2002, Huang et al., 2021, Zhang et al., 2024], and recent methods have demonstrated the effectiveness of utilizing community structures in GCL to improve downstream task performance [Li et al., 2022, Chen et al.,

2023]. However, these methods primarily focus on enhancing task performance and often overlook the scalability challenges associated with GCL. Generally, given a problem, simpler data structures can lead to simpler and more efficient algorithms. The partition matrix indicating node-community memberships offers a straightforward yet essential structure for graph representation [Wu et al., 2022a]. This naturally raises the question: can we leverage community structures to simultaneously address the two scalability issues, while still maintaining high downstream task performance?

For the computation consumption caused by GNNs, we utilize a community partition matrix instead of the original graph structure for message passing. Based on the dense internal connections within the community, we treat each community as a subgraph where internal nodes are interconnected. Thus, all nodes within a community share the same representation, referred to as the community centroid. This approach simplifies the message-passing process by allowing node features in a community to be aggregated at the community centroid instead of individual nodes, avoiding the issue of exponential growth in the number of nodes that need to be computed during message passing.

For the quadratic computational complexity of node pair similarity in contrastive loss, we propose an approach that effectively exploits both intra-community and inter-community information to reconstruct the loss. Specifically, our method encourages the embedding representations of community centroids to be similar to their internal nodes, and leverages the hierarchical structure of communities to implicitly model long-range dependencies between nodes in adjacent communities for capturing both basic (intra-community) and higher-level (inter-community) structural information. Since the number of communities is typically much smaller than the number of individual nodes, this approach significantly reduces the computational load required for calculating node pair similarities in GCL. In addition, our reconstructed loss facilitates the construction of positive and negative samples more effectively, avoiding the mislabeling of closely connected nodes as negative samples to improve the performance of downstream tasks.

The main contributions are summarized as follows:

- We propose a simple and effective method to improve the scalability and performance of GCL by leveraging community structure instead of fine-grained adjacency information between nodes.
- We provide theoretical analysis showing that our community structure-based loss can effectively capture the essential structural information and achieves good generalization performance.
- Extensive experiments on widely used benchmarks across different scales and homophily levels show that our method significantly reduces computational overhead while achieving the best performance.

2 PRELIMINARIES

Graph Neural Network. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph with n nodes, where $\mathcal{V} = \{v_1, \cdots, v_n\}$ is the node set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. The original graph structure can be represented by an adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$, where $\mathbf{A}_{i,j} = 1$ if there is an edge $(v_i, v_j) \in \mathcal{E}$, otherwise $\mathbf{A}_{i,j} = 0$. The node features are represented by a feature matrix $\mathbf{X} \in \mathbb{R}^{n \times h}$, where $\mathbf{x}_i \in \mathbf{X}$ is a h-dimensional feature vector of node v_i. Thus, the complete graph (i.e., graph structure together with node features) can be denoted as $G = (\mathbf{A}, \mathbf{X})$. For scenarios of isolating node features from the graph structure, we define a feature-only graph as $G^0 = (\mathbf{I}_n, \mathbf{X})$, where $\mathbf{I}_n$ is the $n \times n$ identity matrix indicating the absence of edge connections.

In this paper, we focus on training a GNN encoder $f_\theta(G) : \mathcal{G} \to \mathbb{R}^{n \times d}$ parameterized by θ in the absence of labeled data to generate node representations $\mathbf{v}_i = f_\theta(G)[v_i] \in \mathbb{R}^d$ optimized for downstream tasks (e.g., node classification). Specifically, a single-layer GNN [Kipf and Welling, 2017] can be formulated as

$$f_\theta(G) = \sigma(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}) \tag{1}$$

where $\sigma(\cdot)$ denotes a non-linear activation function, $\hat{\mathbf{A}}$ represents the symmetrically normalized adjacency matrix of $\mathbf{A}$, and $\mathbf{W} \in \mathbb{R}^{h \times d}$ is the learnable weight matrix corresponding to θ. It means that the GNN leverages the graph structure by $\hat{\mathbf{A}}$ and the node features by $\mathbf{X}$ to produce effective node embeddings suitable for downstream applications.

Graph Partition. Let $\mathcal{P} = \{P_1, \cdots, P_m\}$ represent a partition of G, where each P_j denotes a community within G that preserves regional structure and clustering properties. For any $j \neq k$, we have $P_j \cap P_k = \emptyset$. A node $v_i \in P_j$ must satisfy the condition that its internal degree within P_j exceeds its external degree. We use $\mathbf{P} \in \{0,1\}^{n \times m}$ to denote the partition matrix corresponding to $\mathcal{P}$, where $\mathbf{P}_{i,j} = 1$ if node $v_i \in P_j$, and $\mathbf{P}_{i,j} = 0$ otherwise. The normalized partition matrix is denoted as $\hat{\mathbf{P}}$. The adjacency matrix of the community-level graph can be constructed by $\mathbf{P}^{\mathrm{T}}\mathbf{A}\mathbf{P}$, where each entry indicates the connections between communities. This formulation enables the analysis of higher-level interactions among communities, instead of focusing on individual nodes.

Graph Contrastive Learning. The typical GCL framework consists of a shared GNN encoder f_θ, a MLP projection head g_φ, and a graph contrastive loss $\mathcal{L}_{gc}$. Initially, GCL generates two augmented views G^1 and G^2 by applying random perturbations to the input graph G, such as DropEdge and feature masking [Hassani and Ahmadi, 2020]. These augmented views are then fed into f_θ to obtain node representations. During the training phase, the projection head g_φ maps the node representations from both views into a common embedding space for contrastive learning. The contrastive loss $\mathcal{L}_{gc}$ is designed to pull together the rep-

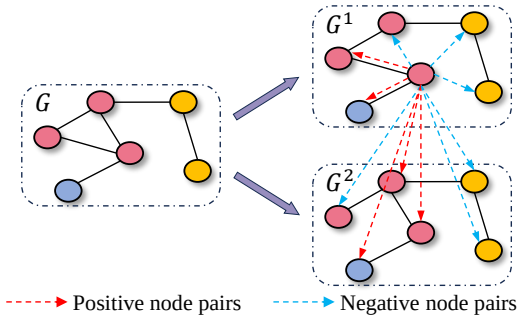

Figure 1: An Example of Contrastive Scheme.

resentations of the same node from different views (i.e., positive node pairs) while pushing apart the representations of different nodes (i.e., negative node pairs). Formally, $\mathcal{L}_{gc}$ can be written as

$$\mathcal{L}_{gc} = -\frac{1}{n}\sum_{v_i \in \mathcal{V}} \log \frac{\exp\left(g_\varphi(\mathbf{v}_i^1)^{\mathrm{T}} g_\varphi(\mathbf{v}_i^2)/\tau\right)}{\sum_{v_j \in \mathcal{V}^-} \exp\left(g_\varphi(\mathbf{v}_i^1)^{\mathrm{T}} g_\varphi(\mathbf{v}_j)/\tau\right)} \tag{2}$$

where $\mathbf{v}_i^1 = f_\theta(G^1)[v_i]$ and $\mathbf{v}_i^2 = f_\theta(G^2)[v_i]$ are the embedding representations of the same node v_i in the two augmented views, $\mathcal{V}^-$ denotes the set of negative node pairs from the two views [Zhu et al., 2021], and $\tau > 0$ is the temperature parameter. In addition, given the graph homogeneity assumption, positive node pairs can be extended from the two augmented views to the neighbors of node v. An illustration of this scheme is presented in Figure 1.

3 METHODOLOGY

In this section, we present the technical details of our method and provide a theoretical analysis to ensure its effective application to downstream tasks.

3.1 COMMUNITY CONTRASTIVE LEARNING

The basic idea behind our method is to encourage the representations of community centroids to be similar to those of their internal nodes, while pushing dissimilar community centroids apart. As illustrated in Figure 2, our method does not utilize a GNN encoder during the training phase, thereby reducing the computational overhead associated with generating negative sample pairs. This simplified GCL framework significantly decreases the computational resources required for model training.

3.1.1 Partition Convolutional Network

Conventional GCL methods suffer from prohibitive complexity due to the exponential growth of the number of nodes that need to be computed in layer-wise message passing and the redundant computations across augmented views. Prior attempts to reduce graph instances processed by GNNs [Mo et al., 2022] have not address the root complexity bottleneck, i.e., dense message passing. We replace the adjacency matrix with a sparse partition matrix, which enables a low-rank approximation of the k-step diffusion matrix: $\hat{\mathbf{A}}^k \approx \mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}$ [Loukas, 2019, Zhang et al., 2024]. Then, the GNN encoder in Eq. 1 simplifies to:

$$f_\theta(G) = \sigma(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}) \approx \sigma(\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}\mathbf{X}\mathbf{W}) = f_\theta(\mathcal{P}) \tag{3}$$

where $\hat{\mathbf{P}}^{\mathrm{T}}\mathbf{X}\mathbf{W}$ denotes the community centroid representations $\mathbf{C} \in \mathbb{R}^{m \times d}$. We refer to Eq. 3 as a single-layer Partition Convolutional Network (PCN), which performs message passing exclusively within each community.

In practice, we use a fast graph partition algorithm METIS [Karypis and Kumar, 1997] to generate $\mathbf{P}$. The sparsity of the partition matrix $\mathbf{P}$ ensures that PCN requires fewer computational resources, making it particularly efficient for large-scale graphs. Moreover, by focusing on message passing within communities, PCN can effectively exchange information among nodes within the same community without unnecessary interactions across communities. Moreover, our experimental results further show that even with a single-layer architecture, PCN achieves high accuracy while significantly reducing computational costs.

3.1.2 Community Contrastive Loss

The success of existing GCL methods lies in emphasizing similarities in the neighborhood representations of the same node across different augmented views [Zhu et al., 2021, Shen et al., 2023]. Inspired by this, we leverage the dense connections within communities to strengthen the learning of neighborhood similarities. Building on the principle of PCN, which focuses on message passing in communities to capture local structural information, we make node representations within the same community more similar, ensuring that intra-community nodes have consistent and closely aligned embeddings. Moreover, to fully extract hierarchical structure information, we also encourage closer proximity between neighboring community centroids.

Intra-community Reconstruction Loss. We utilize the community centroid representations to reconstruct the original features of internal nodes, leveraging the fact that nodes within a community share the same centroid representation. This approach naturally reduces the distance between the internal node representations without needing to compute all pairwise distances within the same community. Formally, the intra-community reconstruction loss is defined as

$$\mathcal{L}_{cr} = \frac{1}{m}\sum_{P_i \in \mathcal{P}} \frac{1}{|P_i|} \sum_{v_j \in P_i} \|g_\varphi(\mathbf{c}_i) - \mathbf{v}_j\|_2^2 \tag{4}$$

where $\mathbf{c}_i = f_\theta(\mathcal{P})[P_i]$ is the centroid representation of P_i and $\mathbf{v}_j = f_\xi(G^0)[v_j]$ is the feature-only representation of

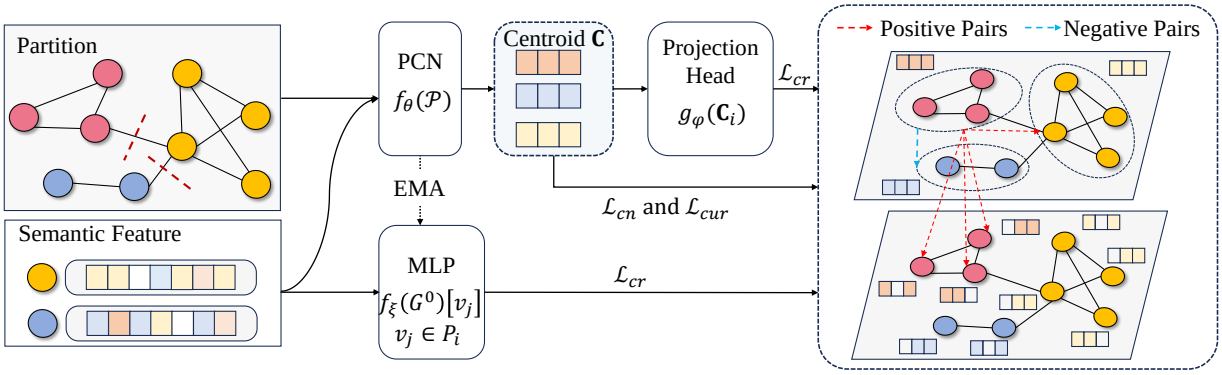

Figure 2: The Framework of Our Proposed GCL Method.

v_j. The term $|P_i|$ is the number of nodes in P_i and $\|\cdot\|_2$ denotes the L2 norm. Minimizing $\mathcal{L}_{cr}$ ensures the embeddings of nodes within the same community are closely aligned with their community centroids.

Inter-community Neighborhood Loss. Note that adjacent communities tend to merge into larger communities, forming hierarchical structures, a phenomenon widely observed in real-world graph data [Li and Pan, 2016]. To incorporate this hierarchical structure information and address the limitations of $\mathcal{L}_{cr}$, where nodes are adjacent but do not belong to the same community, we introduce the inter-community neighborhood loss, defined as

$$\mathcal{L}_{cn} = \frac{1}{m} \sum_{P_i \in \mathcal{P}} \frac{1}{|\mathcal{N}(P_i)|} \sum_{P_k \in \mathcal{N}(P_i)} \|\mathbf{c}_i - \mathbf{c}_k\|_2^2 \quad (5)$$

where $\mathbf{c}_k = f_\theta(\mathcal{P})[P_k]$, and $\mathcal{N}(P)$ represents the neighbors of P, which can be obtained from the community adjacency matrix $\mathbf{P}^{\mathrm{T}}\mathbf{A}\mathbf{P}$. Minimizing $\mathcal{L}_{cn}$ ensures that neighboring communities have similar centroid representations, effectively capturing the hierarchical structure information.

Community Uniformity Regularization Loss. Although $\mathcal{L}_{cr}$ and $\mathcal{L}_{cn}$ provide a framework for combining intra-community similarity and inter-community hierarchy, these losses alone may lead to collapsed representations [Thakoor et al., 2022]. In such a scenario, all node representations degenerate to the same vector on the hyperplane, minimizing the loss but rendering the model ineffective for downstream tasks. To address this issue, we introduce the community uniformity regularization loss to further enhance the representation diversity, defined as

$$\mathcal{L}_{cur} = -\frac{1}{m^2} \sum_{P_i, P_t \in \mathcal{P}} \|\mathbf{c}_i - \mathbf{c}_t\|_2^2 \quad (6)$$

where P_t denotes any community distinct from P_i (i.e., $i \neq t$), and $\mathbf{c}_t$ is the representation of P_t. By maximizing the distances between different community centroids, this loss encourages diversity in the learned representations.

Overall Loss. Directly jointly optimizing the three losses can result in an overall loss $\mathcal{L}_{all} = \mathcal{L}_{cr} + \mathcal{L}_{cn} + \mathcal{L}_{cur}$. However, minimizing $\mathcal{L}_{all}$ is not a suitable optimization objective because $\mathcal{L}_{cur}$ can cause $\mathcal{L}_{all}$ to become negative, leading to abnormal training. Although scaling $\mathcal{L}_{cur}$ via a hyperparameter can mitigate this issue, it increases the complexity and time required to find optimal model parameters. Therefore, we integrate the three losses and derive an upper bound representation as the overall loss:

$$\mathcal{L}_{\mathcal{P}} \leq -\frac{1}{m} \sum_{P_i \in \mathcal{P}} \frac{1}{|\mathcal{N}(P_i)|} \sum_{P_k \in \mathcal{N}(P_i)} \log \ell(P_i) \quad (7)$$

where $\ell(P_i) =$

$$\frac{\exp(\frac{1}{|P_i|} \sum_{v_j \in P_i} g_\varphi(\mathbf{c}_i)^{\mathrm{T}} \mathbf{v}_j + \alpha \mathbf{c}_i^{\mathrm{T}} \mathbf{c}_k)}{\exp(\frac{1}{|P_i|} \sum_{v_j \in P_i} g_\varphi(\mathbf{c}_i)^{\mathrm{T}} \mathbf{v}_j) + \sum_{P_t \in \mathcal{P}} \exp \mathbf{c}_i^{\mathrm{T}} \mathbf{c}_t}$$

, and α controls the influence of the neighboring communities. The larger α is, the more focus on global information of G. For reading simplicity, we omit the temperature parameter τ in Eq. 7 and the specific value can be found in Appendix B.2. The derivation of the overall loss is outlined below, with more details available in Appendix A.1.

Proof. Since vectors in contrastive losses are usually normalized, we have

$$\min \mathcal{L}_{cr} \Leftrightarrow \min -\frac{1}{m} \sum_{P_i \in \mathcal{P}} \frac{1}{|P_i|} \sum_{v_j \in P_i} g_\varphi(\mathbf{c}_i)^{\mathrm{T}} \mathbf{v}_j \quad (8)$$

$$\min \mathcal{L}_{cn} \Leftrightarrow \min -\frac{1}{m} \sum_{P_i \in \mathcal{P}} \frac{1}{|\mathcal{N}(P_i)|} \sum_{P_k \in \mathcal{N}(P_i)} \mathbf{c}_i^{\mathrm{T}} \mathbf{c}_k \quad (9)$$

$$\min \mathcal{L}_{cur} \Leftrightarrow \min \frac{1}{m^2} \sum_{P_i, P_t \in \mathcal{P}} \mathbf{c}_i^{\mathrm{T}} \mathbf{c}_t \quad (10)$$

Let $\mathcal{L}_{\mathcal{P}}$ be the sum of Eq. 8, Eq. 9 and Eq. 10, and define $\mathbf{B} = \frac{1}{|P_i|} \sum_{v_j \in P_i} g_\varphi(\mathbf{c}_i)^{\mathrm{T}} \mathbf{v}_j + \mathbf{c}_i^{\mathrm{T}} \mathbf{c}_k$. Then, according to

Jensen's inequality, we have

$$\begin{aligned}
\mathcal{L}_{\mathcal{P}} &\overset{c}{=} -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}(\mathbf{B}-\frac{1}{m}\sum_{P_t\in\mathcal{P}}\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t) \\
&\leq -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}(\mathbf{B}-\log\sum_{P_t\in\mathcal{P}}\frac{\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t}{m}) \\
&\leq -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}[\log\exp\mathbf{B}\ - \\
&\qquad \log(\exp(\frac{1}{|P_i|}\sum_{v_j\in P_i}g_\varphi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j)+\sum_{P_t\in\mathcal{P}}\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t)] \\
&= -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\log\ell(P_i)
\end{aligned}$$

where $\overset{c}{=}$ denotes the equation minimization is equivalent. Since $\min\mathcal{L}_{all}\Leftrightarrow\min\mathcal{L}_{\mathcal{P}}$, we derive the upper bound of the overall loss, as shown in Eq. 7. □

3.1.3 Model Training

Once the training is completed, we retain the model parameters learned in PCN and replace the partition convolutional operator with a graph convolutional operator, i.e., $\sigma(\hat{\mathbf{A}}^k\mathbf{X}\mathbf{W})$ to obtain node representations for downstream tasks. By this way, we avoid complex calculations during the training process and also use the graph convolutional operator to improve model performance, which we have verified in experiments. Note that the weight parameters ξ of the MLP encoder are updated via the exponential moving average (EMA) of the PCN encoder weights θ (i.e., $\xi\leftarrow w\theta+(1-w)\xi$, where $w\in[0,1]$ is the target decay rate). This is a common method in contrastive learning [Thakoor et al., 2022]. The overall procedure of our model is illustrated in Algorithm 1.

Given a graph G with n nodes and m communities, the time complexity of our method primarily arises from PCN and contrastive loss $\mathcal{L}_{\mathcal{P}}$. In the former, aggregating node representations to community centroids takes $O(nd)$, where d is the representation dimension. In the latter, the comparison between centroids and internal nodes is $O(nd)$, and the comparison between communities is $O(m^2d)$. Since $m\ll n$, we can complete the computations in linear time.

3.2 PROPERTIES OF OVERALL LOSS

We provide theoretical evidence to prove that our method can capture essential structural information for downstream tasks, and all detailed proofs can be found in Appendix A. First, we demonstrate that our method can capture structure information of one-hop neighborhood, which is beneficial for heterophilic graphs, as nodes from the same semantic class tend to share similar neighborhood contexts [Xiao et al., 2023].

Algorithm 1 Model Training

Input: a graph $G=(\mathbf{A},\mathbf{X})$
Parameter: number of communities m, hidden dimensions d, training epochs T, PCN encoder f_θ, MLP encoder f_ξ, projection head g_φ
Output: node representations $\mathbf{H}$
Steps:

```
Initiate parameters θ, ξ and φ;
P ← construct a partition of G by METIS;
Construct adjacency matrix based on P;
for t = 1 to T do
    c ← generate community representations via Eq. 3;
    v ← generate node representations via f_ξ(X);
    L_P ← calculate the overall loss via Eq. 7;
    Update model parameters θ and φ via L_P;
    Update model parameters ξ via EMA;
end for
H ← generate node representation via σ(Â^k XW);
return H
```

Theorem 1. *Let $\mathcal{N}(v)$ denote the neighbors of v. Minimizing the community contrastive loss $\mathcal{L}_{\mathcal{P}}$ will try to minimize the alignment loss between one-hop neighbors, which is defined as*

$$\mathcal{L}_{alig}=\frac{1}{n}\sum_{v_j\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_j)|}\sum_{v_i\in\mathcal{N}(v_j)}\|\mathbf{v}_j-g_\varphi(\mathbf{v}_i)\|_2 \quad (11)$$

where $\mathbf{v}_j=f_\theta(G)[v_j]$ and $\mathbf{v}_i=f_\theta(G^0)[v_i]$.

Theorem 1 shows that our method can capture the one-hop neighborhood context from the central node representations, which captures the local structure information of the graph. Next, we prove that our method can capture higher-level structure information, i.e., multi-hop neighborhood dependencies.

Theorem 2. *Suppose the contrastive loss $\mathcal{L}_{\mathcal{P}}$ and $\mathcal{L}_{gc}$ are L-Lipschitz continuous. Then, $\mathcal{L}_{\mathcal{P}}$ can be approximated by $\mathcal{L}_{gc}$ under the graph homophily assumption*

$$\|\mathcal{L}_{gc}-\mathcal{L}_{\mathcal{P}}\|\leq L\|\hat{\mathbf{A}}^k-\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}\|\|\mathbf{X}\|\|\mathbf{W}_{all}\| \quad (12)$$

where L is the Lipschitz constant and $\mathbf{W}_{all}$ is the learnable parameters in GCL model.

Theorem 2 shows that our loss in Eq. 7 can estimate the original contrastive loss $\mathcal{L}_{gc}$ of the diffusion matrix. Minimizing $\|\hat{\mathbf{A}}^k-\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}\|$ is the minimum cut problem in graph theory, which is consistent with the goal of graph partition methods [Hofmeyr, 2016]. Moreover, we also provide formal guarantees on the generalizability for downstream tasks.

Theorem 3. *Let f_θ^* be the optimal model parameters learned by the global minimizer of $\mathcal{L}_\mathcal{P}$, and $y(v_i)$ denote the label of v_i. Then, there exists a linear classification function $\hat{y} : \mathcal{V} \to \mathbb{R}^c$ such that the error upper bound is*

$$\mathbb{E}_{v\in\mathcal{V}}\|y(v_i) - \hat{y}[f_\theta^*(v_i)]\|_2^2 \leq \frac{1-\phi_\mathcal{P}}{\lambda_{d+1}} \tag{13}$$

where λ_{d+1} is the $d+1$ smallest eigenvalue of diffusion matrix $\hat{\mathbf{A}}^k$ and $\phi_\mathcal{P}$ is partition homophily ratio, defined as

$$\phi_\mathcal{P} = \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|^2}\sum_{v_j,v_k\in P_i}\mathbb{1}[y(v_j) = y(v_k)] \tag{14}$$

where $\mathbb{1}[\cdot]$ is the indicator function.

Theorem 3 shows that the classification error of the learned representations is bound by the partition rate $\phi_\mathcal{P}$ and embedding dimension d, which means that we can dynamically adjust the number of communities and d to improve performance. It is worth noting that the nodes within the same community in a homophilic graph tend to be of the same class ($\phi_\mathcal{P} \to 1$), while the opposite may be true in a heterophilic graph ($\phi_\mathcal{P} \to 0$). This theorem indicates that heterophilic graphs might require a larger dimension to ensure model performance.

4 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the effectiveness and scalability of our method.

4.1 EXPERIMENTAL SETTINGS

Datasets. We choose fourteen benchmark datasets for experiments, including: (a) eight homophilic graph datasets Cora, CiteSeer, PubMed, Wiki-CS, Amazon Computer, Amazon Photo, Coauthor CS and Coauthor Physics [Kipf and Welling, 2017, Shchur et al., 2018, Mernyei and Cangea, 2020], (b) four heterophilic graph datasets Texas, Wisconsin, Cornell and Actor [Pei et al., 2020], and (c) two large-scale homophilic datasets Ogbn-Arxiv and Ogbn-Products [Hu et al., 2020].

Splitting Strategies. For Cora, CiteSeer and Pubmed datasets, we adopt the public splits, with each class having 20 nodes for training, another fixed 500 nodes and 1000 nodes for validation and testing, respectively [Kipf and Welling, 2017, Veličković et al., 2019]. For the other five homophilic datasets, we adopt the 10%/10%/80% training/validation/testing splits following previous works [Liu et al., 2023]. For heterophilic and large-scale graphs, we adopt the splits that come with datasets Zheng et al. [2022], Zhang et al. [2024]. The details of the datasets are summarized in Appendix B.1.

Comparison Methods. We compare our method with the following four categories of methods:

- Semi-supervised learning methods: GCN [Kipf and Welling, 2017], GAT [Veličković et al., 2018], SGC [Wu et al., 2019] and NodeFormer [Wu et al., 2022b].
- Classical GCL methods: DGI [Veličković et al., 2019], MVGRL [Hassani and Ahmadi, 2020], gCooL [Li et al., 2022] and CSGCL [Chen et al., 2023].
- Efficiency-oriented GCL methods: BGRL [Thakoor et al., 2022], SUGRL [Mo et al., 2022], GGD [Zheng et al., 2022], SGCL [Sun et al., 2024] and StructComp [Zhang et al., 2024].
- Heterophily-aware GCL methods: HGRL [Chen et al., 2022], DSSL [Xiao et al., 2022], SP-GCL [Wang et al., 2023], GraphACL [Xiao et al., 2023], GREET [Liu et al., 2023] and HEATS [Zhuo et al., 2024].

Metrics. We adopt node classification and node clustering tasks to evaluate the quality of node representations. For node classification, we train a logistic regression classifier on the frozen representations and report the test accuracy. For node clustering, we perform K-Means clustering on the representations, and report the Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) scores. We verify the scalability of our method using GPU memory usage and training time per epoch.

Implementation Details. We randomly initialize all weight parameters of the model and use the Adam optimizer to train the encoder. All experiments are implemented in PyTorch and conducted on a machine equipped with an Intel i9 13900KF CPU, 128GB RAM, and NVIDIA RTX 4090 GPU. Each experiment is repeated 20 times, and the average performance and standard deviation are reported here. The detailed settings of the model and specific hyper-parameters can be found in Appendix B.2.

4.2 EXPERIMENTAL RESULTS

Node Classification. The node classification results on homophilic and heterophilic graphs are reported in Table 1 and Table 2, respectively. These results tell us that: (a) Our method achieves the best performance across all datasets, especially on heterophilic graphs where it outperforms heterophily-aware GCL methods. (b) Unsupervised GCL methods demonstrate significant competitiveness compared to semi-supervised methods that leverage label information during the training process. (c) Community-based GCL methods, such as gCooL and CSGCL, also achieve impressive results, confirming the utility of community information in enhancing contrastive learning. These empirical findings support the theoretical analysis in Section 3.2 and validate the effectiveness of our method.

Table 1: Node Classification Accuracy (%) on Homophilic Graphs (**Bold**: Best, Underline: Second Best).

Methods	Cora	CiteSeer	PubMed	Wiki-CS	Computer	Photo	CS	Physics
GCN	81.5±0.2	70.3±0.4	79.0±0.5	76.9±0.4	86.3±0.5	92.2±0.2	93.1±0.2	95.4±0.2
GAT	83.0±0.7	72.5±0.3	79.0±0.3	77.4±0.2	87.1±0.4	92.7±0.5	92.4±0.3	95.3±0.2
SGC	81.0±0.1	71.9±0.1	78.9±0.0	79.4±0.3	85.9± 0.7	92.3±0.2	92.6±0.2	95.1±0.2
NodeFormer	82.7±0.8	72.4±1.2	79.6±0.9	80.4±0.4	86.6±0.3	92.9±0.1	93.5±0.2	95.9±0.2
DGI	82.3±0.5	71.5±0.7	79.4±0.3	75.7±0.2	84.1±0.4	91.6±0.2	92.0±0.4	94.6±0.4
MVGRL	82.9±0.7	72.8±0.4	80.1±0.4	77.9±0.3	87.1±0.3	92.0±0.1	91.9±0.2	95.5±0.5
gCooL	82.1±0.4	71.4±0.5	82.1±0.3	78.7±0.1	88.9±0.1	92.8±0.2	92.8±0.1	95.1±0.1
CSGCL	81.2±0.2	71.1±0.1	81.2±0.3	78.6±0.1	90.2±0.2	93.2±0.4	93.6±0.1	95.3±0.2
BGRL	82.7±0.6	71.6±0.4	79.9±0.4	80.0±0.1	89.7±0.4	92.9±0.3	93.3±0.4	95.6±0.4
SUGRL	83.4±0.5	73.0±0.4	81.9±0.3	79.8±0.3	88.9±0.2	93.1±0.2	92.7±0.1	94.1±0.4
GGD	83.9±0.4	73.0±0.6	81.3±0.8	78.7±0.6	90.1±0.9	92.5±0.6	92.4±0.2	95.0±0.2
SGCL	83.0±0.2	72.6±0.3	81.3±0.3	79.9±0.5	90.7±0.3	93.4±0.3	93.3±0.2	95.7±0.1
StructComp	82.3±0.8	71.6±1.0	78.3±2.5	80.1±0.1	89.1±1.4	92.7±1.0	93.1±0.4	95.0±0.1
Ours	**84.9±0.5**	**74.2±0.9**	**82.8±0.7**	**81.1±0.3**	**90.9±0.1**	**93.8±0.2**	**94.3±0.3**	**96.3±0.1**

Table 2: Node Classification Accuracy (%) on Heterophilic Graphs (**Bold**: Best, Underline: Second Best).

Methods	Texas	Wisconsin	Cornell	Actor
HGRL	61.8±0.7	63.9±0.6	51.8±1.0	28.0±0.3
DSSL	62.1±1.5	62.3±0.6	53.2±1.3	28.2±0.3
SP-GCL	59.8±1.3	60.1±0.4	52.3±1.2	28.9±0.7
GraphACL	71.1±0.3	69.2±0.4	72.7±3.7	30.0±0.1
GREET	84.6±4.2	80.9±5.2	72.9±1.7	36.1±1.2
HEATS	64.9±4.7	65.9±5.6	67.0±5.9	30.1±1.2
Ours	**85.4±5.6**	**83.7±3.2**	**74.6±5.0**	**37.4±1.3**

Table 3: Node Clustering Results Measured by NMI (%) and ARI (%). K-Means Represents Clustering Directly on the Original Node Features.

Methods	Photo		CS		Physics	
	NMI	ARI	NMI	ARI	NMI	ARI
K-Means	25.8	14.5	60.1	40.4	48.9	27.6
gCooL	56.6	43.1	75.3	62.1	65.2	57.8
CSGCL	58.8	46.3	77.1	63.6	66.1	58.3
SUGRL	63.6	52.8	76.6	62.5	65.7	60.4
GREET	52.3	37.1	75.8	62.1	66.4	63.6
GraphACL	61.1	47.9	74.7	62.8	64.2	62.5
Ours	**64.8**	**54.4**	**77.4**	**63.9**	**69.3**	**66.1**

Node Clustering. We select the methods with clustering effects in their original papers for comparison and report the results in Table 3. The results tell us that: (a) Our method performs better than other methods due to the effective utilization of community structure information. (b) Our method outperforms gCooL and CSGCL which only use the intra-community information. This suggests that leveraging both intra-community and inter-community relationships is crucial for node representation. These results not only validate the effectiveness of our method but also demonstrate its adaptability to other downstream tasks.

Scalability Evaluation. We compare the node classification accuracy and training consumption of our method with efficiency-based GCL methods, and report the results on large-scale datasets in Table 4. For fairness, we omitted the memory usage of methods trained in a mini-batch manner. These results tell us that: (a) Our method achieves the best accuracy while simultaneously reducing time consumption and memory usage. (b) The efficiency improvement of our method becomes more significant as the dataset scale increases. Notably, on Ogbn-Products, where many methods must adopt mini-batch training, our method can be trained in full-batch mode, which is attributed to our method's significant reduction in message passing and contrastive loss calculation consumption. These results validate the scalability of our method.

Impacts of Parameters. We explore the impacts of partition rate β and node embedding dimension d in Figure 3, and report the effect of coefficient α in Appendix B.3. These results tell us that: (a) When β becomes very large, the accuracy tends to flatten or even decrease, indicating that setting β to a smaller value is a practical choice. Moreover, a smaller β requires less GPU memory. (b) A larger d can generally improve node classification accuracy, especially on heterophilic graphs. This observation supports Theorem 3, which posits that a larger dimension can effectively reduce the upper bound of the classification error. (c) On homophilic graphs, a smaller α is required to emphasize

Table 4: Scalability Evaluation on Node Classification. 'Acc': Accuracy (%), 'Time': Training Time per Epoch, 'Mem': GPU Memory (GB), '-': Training in Mini-batch.

Methods	Ogbn-Arxiv			Ogbn-Products		
	Acc	Time(s)	Mem	Acc	Time(m)	Mem
BGRL	71.6	0.29	10.7	64.0	53.3	-
SUGRL	67.8	**0.05**	**2.6**	72.9	1.5	23.5
GGD	71.6	0.95	14.3	75.7	12.7	-
SGCL	71.0	0.09	5.1	76.0	1.9	-
S.Comp	71.7	**0.05**	3.4	75.7	0.06	12.0
Ours	**71.9**	0.06	5.4	**76.8**	**0.001**	**6.3**

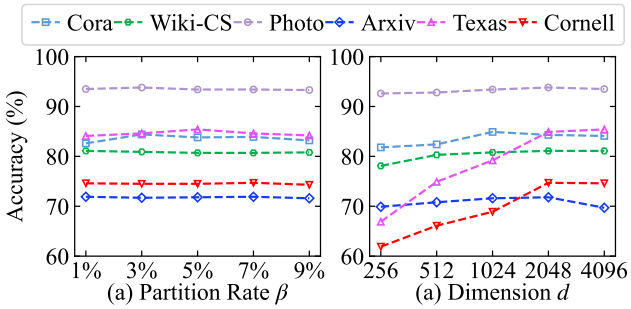

Figure 3: Impacts of Hyperparameters β and d ('$\beta = 1\%$ ': The Number of Communities is Fixed to $0.01 \times n$).

the local information of the graph, while on heterophilic graphs, increasing the value of α is needed to focus on global information.

Impacts of Partition Algorithms. We evaluate the impacts of different graph partition methods, including Graph Cut (GC) [Loukas, 2019], Louvain [Blondel et al., 2008], and Structural Entropy (SE) [Li and Pan, 2016], on node classification in Figure 4. The results tell us that: (a) SE performs better across all datasets because it does not require manual specification of the number of communities, which avoids overfitting to a some extent. This indicates that more powerful partition method leads to higher accuracy improvement. (b) Despite its simplicity, METIS also demonstrates good performance on node classification. Thus, we adopt METIS in our method. The graph partition time consumption for all datasets is summarized in Appendix B.4, showing that it requires only a few seconds on large-scale datasets.

Ablation Study. We conduct an ablation study, as shown in table 5. The results tell us that: (a) All components contribute to the performance improvement of our method. (b) The message passing mechanism is critical for improving the accuracy of node classification on homophilic graphs. (c) For node classification, the impact of $\mathcal{L}_{cr}$ is greater than that of $\mathcal{L}_{cn}$, which indicates that intra-community information is more useful than inter-community information. (d) Even without the graph convolutional operator, our method still

Table 5: Ablation Study on Node Classification. **A1**: Removing Graph Convolutional Operator during Testing Phase, **A2**: Removing Intra-community Reconstruction Loss $\mathcal{L}_{cr}$, **A3**: Removing Inter-community Neighborhood Loss $\mathcal{L}_{cn}$, '-': Without using Graph Convolutional Operator, that is, $k = 0$.

Baselines	CiteSeer	PubMed	Photo	Actor
MLP	56.1±0.4	71.4±0.1	78.5±0.1	35.6±0.9
GCN	70.3±0.4	79.0±0.5	92.4±0.2	30.8±0.7
A1 (w/o GC)	69.5±0.2	76.2±0.8	88.3±0.2	-
A2 (w/o $\mathcal{L}_{cr}$)	70.8±0.8	45.7±1.7	90.2±0.5	34.5±1.2
A3 (w/o $\mathcal{L}_{cn}$)	72.7±0.2	80.9±0.3	93.3±0.1	36.1±1.4
A1 & A2	52.6±0.2	35.9±0.4	75.1±0.6	-
A1 & A3	68.9±0.2	74.6±0.4	88.4±0.1	-
Ours	**74.2±0.9**	**82.8±0.7**	**93.8±0.2**	**37.4 ±1.3**

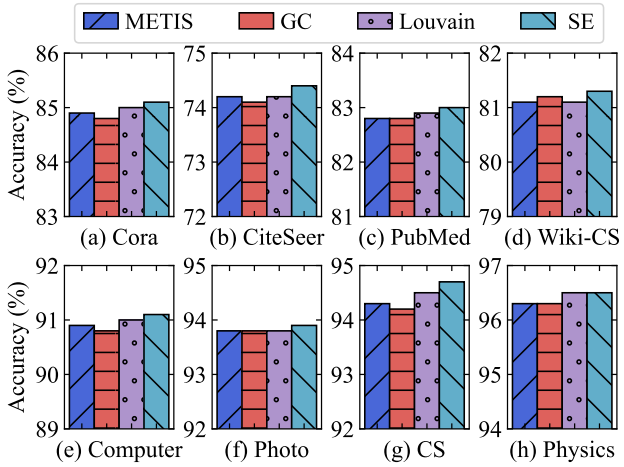

Figure 4: Comparison of Different Partition Algorithms.

outperforms semi-supervised MLP and remains competitive with GCN, which verifies the rationality of our method. These explore the contributions of different components of our method.

Visualization. We evaluate the effectiveness of our loss in capturing graph structural information by removing the GNN during the test phase, and report the pairwise cosine similarity of node representations for randomly sampled nodes, one-hop neighbors, and multi-hop neighbors in Figure 5. The results tell us that: (a) Our method increases the similarity of node representations for one-hop neighbors compared to random pairs, indicating effective preservation of local structural information. (b) Node representations of multi-hop neighbors maintain high similarity, demonstrating our method's ability to capture higher-level structural patterns without stacking multiple GNN layers. These results confirm that our loss function effectively captures essential graph structural information.

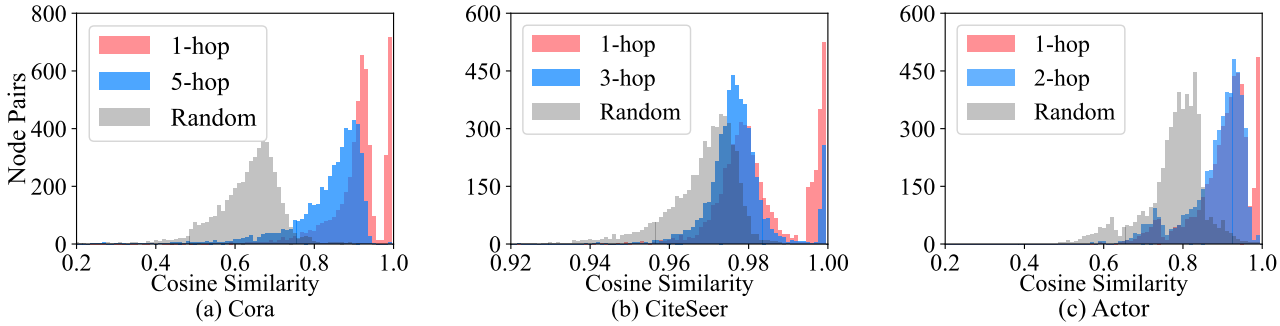

Figure 5: The Pair-wise Similarity Distribution of Randomly Sampled Node, One-hop and Multi-hop Neighbors.

5 RELATED WORKS

Scalable Graph Neural Networks. GNNs facilitate feature propagation between nodes through the message passing mechanism Kipf and Welling [2017]. To improve the scalability of GNNs, GraphSage [Hamilton et al., 2017] and Cluster-GCN [Chiang et al., 2019] employ subgraph sampling techniques or mini-batch processing mode to train models. SGC [Wu et al., 2019] simplifies GNNs by removing the non-linear function of graph convolutional layers. Coarse-GNN [Huang et al., 2021] proposes to use a compreseed graph for scalable training of GNNs. Other methods use global attention mechanisms with linear complexity to process large-scale graphs, such as NodeFormer [Wu et al., 2022b]. However, these methods are not suitable for graph representation learning without task-specific labels.

Graph Contrastive Learning. GCL has demonstrated excellent performance in graph representation learning tasks without labels [Shen et al., 2023, Ju et al., 2024]. DGI [Veličković et al., 2019] and MVGRL [Hassani and Ahmadi, 2020] maximize the mutual information between local and global embeddings to learn node representations. gCool [Li et al., 2022] and CSGCL [Chen et al., 2023] improve node representations by introducing community structure to construct positive and negative samples. Recent works focus on the scalability of GCL [Ju et al., 2024], such as BGRL [Thakoor et al., 2022] and SGCL [Wu et al., 2019] compute the contrastive loss without negative node pairs. SUGRL [Mo et al., 2022] reduces the number of graphs that need to be processed by GCL. GGD [Zheng et al., 2022] directly uses binary cross entropy loss to distinguish between positive and negative samples. StructComp [Zhang et al., 2024] conducts contrastive learning on the constructed compressed graph, significantly improving the scalability of GCL. However, it relies on a fixed graph coarsening process, leading to overly homogeneous center representations and loss of node-level information [Huang et al., 2024].

There are also some methods that explore the potential of GCL on heterophilic graphs [Yang and Mirzasoleiman, 2024]. HGRL captures distant neighbors to learn node representations [Chen et al., 2022]. DSSL decouples different neighborhood contexts of nodes [Xiao et al., 2022]. SP-GCL studies the concentration property of features on heterophilic graphs [Wang et al., 2023]. GraphACL captures one-hop neighborhood information and two-hop monophily similarity [Xiao et al., 2023]. GREET learns node representations by distinguishing homophilic and heterophilic edges [Liu et al., 2023]. HEATS optimizes positive sampling techniques for heterophilic graphs [Zhuo et al., 2024]. However, most of these methods still need to use mini-batch mode when processing large-scale graphs.

6 CONCLUSION

In this paper, we propose a simple and efficient method to improve the scalability of GCL by leveraging community structures. The core idea is to replace finer-grained node adjacency information with community-level structures. Specifically, we use a sparse partition matrix for message passing with linear time complexity, and design an efficient contrastive loss function that considers both intra-community and inter-community structural information. Theoretical analysis shows that our loss can effectively capture the basic and high-level structural information of the graph and has good generalization performance guarantees. Experimental results demonstrate that our method achieves the best performance while significantly reducing the time and memory overhead. We plan to explore a simple adaptive graph partition technique to improve the robustness of our method, addressing potential issues where community structures become unreliable due to noise in the graph.

Acknowledgements

This work was supported by the Key Program of National Natural Science Joint Foundation of China (U23A20298); Program of Yunnan Key Laboratory of Intelligent Systems and Computing (202405AV340009); Yunnan Fundamental Research Projects (202501AS070102, 202401AS070138); Scientific Research Fund Project of Yunnan Education Department (2025Y0061). For any correspondence, please refer to Liang Duan.

References

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

Han Chen, Ziwen Zhao, Yuhua Li, Yixiong Zou, Ruixuan Li, and Rui Zhang. Csgcl: community-strength-enhanced graph contrastive learning. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence* (IJCAI), pages 2059–2067, 2023.

Jingfan Chen, Guanghui Zhu, Yifan Qi, Chunfeng Yuan, and Yihua Huang. Towards self-supervised learning on graphs with heterophily. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management* (CIKM), pages 201–211, 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 Conference on Knowledge Discovery and Data Mining* (SIGKDD), pages 257–266, 2019.

Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), pages 1024–1034, 2017.

Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. In *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), volume 34, pages 5000–5011, 2021.

Kaveh Hassani and Amir Hosein Khas Ahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning* (ICML), pages 4116–4126, 2020.

David P Hofmeyr. Clustering by minimum cut hyperplanes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(8):1547–1560, 2016.

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 *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), pages 22118–22133, 2020.

Siyuan Huang, Yunchong Song, Jiayue Zhou, and Zhouhan Lin. Cluster-wise graph transformer with dual-granularity kernelized attention. In *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), pages 33376–33401, 2024.

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 Conference on Knowledge Discovery and Data Mining* (SIGKDD), pages 675–684, 2021.

Wei Ju, Yifan Wang, Yifang Qin, Zhengyang Mao, Zhiping Xiao, Junyu Luo, Junwei Yang, Yiyang Gu, Dongjie Wang, Qingqing Long, et al. Towards graph contrastive learning: A survey and beyond. *arXiv preprint arXiv:2405.11868*, 2024.

George Karypis and Vipin Kumar. Metis: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. 1997.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of International Conference on Learning Representations* (ICLR), 2017.

Taewook Ko, Yoonhyuk Choi, and Chong-Kwon Kim. Universal graph contrastive learning with a novel Laplacian perturbation. In *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence* (UAI), pages 1098–1108, 2023.

Angsheng Li and Yicheng Pan. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory*, 62(6):3290–3339, 2016.

Bolian Li, Baoyu Jing, and Hanghang Tong. Graph communal contrastive learning. In *Proceedings of the ACM Web Conference* (WWW), pages 1203–1213, 2022.

Yixin Liu, Yizhen Zheng, Daokun Zhang, Vincent CS Lee, and Shirui Pan. Beyond smoothing: Unsupervised graph representation learning with edge heterophily discriminating. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence* (AAAI), pages 4516–4524, 2023.

Andreas Loukas. Graph reduction with spectral and cut guarantees. *Journal of Machine Learning Research*, 20 (116):1–42, 2019.

Péter Mernyei and Cătălina Cangea. Wiki-cs: A wikipedia-based benchmark for graph neural networks. *arXiv preprint arXiv:2007.02901*, 2020.

Yujie Mo, Liang Peng, Jie Xu, Xiaoshuang Shi, and Xiaofeng Zhu. Simple unsupervised graph representation learning. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence* (AAAI), pages 7797–7805, 2022.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *Proceedings of International Conference on Learning Representations* (ICLR), 2020.

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

Xiao Shen, Dewang Sun, Shirui Pan, Xi Zhou, and Laurence T Yang. Neighbor contrastive learning on learnable graph augmentation. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence* (AAAI), pages 9782–9791, 2023.

Wangbin Sun, Jintang Li, Liang Chen, Bingzhe Wu, Yatao Bian, and Zibin Zheng. Rethinking and simplifying bootstrapped graph latents. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining* (WSDM), page 665–673, 2024.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L Dyer, Remi Munos, Petar Veličković, and Michal Valko. Large-scale representation learning on graphs via bootstrapping. In *Proceedings of International Conference on Learning Representations* (ICLR), 2022.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *Proceedings of International Conference on Learning Representations* (ICLR), 2018.

Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *Proceedings of International Conference on Learning Representations* (ICLR), 2019.

Haonan Wang, Jieyu Zhang, Qi Zhu, Wei Huang, Kenji Kawaguchi, and Xiaokui Xiao. Single-pass contrastive learning can work for both homophilic and heterophilic graph. *Transactions on Machine Learning Research*, 2023.

Qianlong Wen, Zhongyu Ouyang, Chunhui Zhang, Yiyue Qian, Chuxu Zhang, and Yanfang Ye. Gcvr: Reconstruction from cross-view enable sufficient and robust graph contrastive learning. In *Proceedings of the 40th Conference on Uncertainty in Artificial Intelligence* (UAI), volume 244, pages 3747–3764, 2024.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning* (ICML), pages 6861–6871, 2019.

Junran Wu, Shangzhe Li, Jianhao Li, Yicheng Pan, and Ke Xu. A simple yet effective method for graph classification. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence* (IJCAI), pages 3580–3586, 2022a.

Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), pages 27387–27401, 2022b.

Teng Xiao, Zhengyu Chen, Zhimeng Guo, Zeyang Zhuang, and Suhang Wang. Decoupled self-supervised learning for graphs. In *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), pages 620–634, 2022.

Teng Xiao, Huaisheng Zhu, Zhengyu Chen, and Suhang Wang. Simple and asymmetric graph contrastive learning without augmentations. In *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), pages 16129–16152, 2023.

Wenhan Yang and Baharan Mirzasoleiman. Graph contrastive learning under heterophily via graph filters. In *Proceedings of the 40th Conference on Uncertainty in Artificial Intelligence* (UAI), pages 3936–3955, 2024.

Shengzhong Zhang, Wenjie Yang, Xinyuan Cao, Hongwei Zhang, and Zengfeng Huang. Structcomp: Substituting propagation with structural compression in training graph contrastive learning. In *Proceedings of International Conference on Learning Representations* (ICLR), 2024.

Yizhen Zheng, Shirui Pan, Vincent Lee, Yu Zheng, and Philip S Yu. Rethinking and scaling up graph contrastive learning: An extremely efficient approach with group discrimination. In *Proceedings of Advances in Neural Information Processing Systems* (NeurIPS), pages 10809–10820, 2022.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph contrastive learning with adaptive augmentation. In *Proceedings of the ACM Web Conference* (WWW), pages 2069–2080, 2021.

Jiaming Zhuo, Feiyang Qin, Can Cui, Kun Fu, Bingxin Niu, Mengzhu Wang, Yuanfang Guo, Chuan Wang, Zhen Wang, Xiaochun Cao, et al. Improving graph contrastive learning via adaptive positive sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (CVPR), pages 23179–23187, 2024.

Improving Graph Contrastive Learning with Community Structure (Appendix)

Xiang Chen[1,2] **Kun Yue**[1,2] **Liang Duan**[1,2] **Lixing Yu**[1,2]

[1]School of Information Science and Engineering, Yunnan University, Kunming, China
[2]Yunnan Key Laboratory of Intelligent Systems and Computing, Yunnan University, Kunming, China

A PROOF DETAILS

A.1 THE OVERALL LOSS

We provide a detailed derivation of the overall loss in Eq. 7.

Proof. For the loss $\mathcal{L}_{cr}$ in Eq. 4, we have

$$\begin{aligned}\mathcal{L}_{cr} &= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|g_\varphi(\mathbf{c}_i)-\mathbf{v}_j\|_2^2 \\ &= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}(g_\phi(\mathbf{c}_i)-\mathbf{v}_j)^{\mathrm{T}}(g_\varphi(\mathbf{c}_i)-\mathbf{v}_j) \\ &= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}(g_\phi(\mathbf{c}_i)^{\mathrm{T}}g_\varphi(\mathbf{c}_i)+\mathbf{v}_j^{\mathrm{T}}\mathbf{v}_j-2g_\varphi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j)\end{aligned} \tag{15}$$

Since vectors in contrastive losses are usually normalized. Thus, we have

$$\min\mathcal{L}_{cr}\Leftrightarrow\min-\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j \tag{16}$$

Similarly, minimizing $\mathcal{L}_{cn}$ in Eq. 5 can be written as

$$\min\mathcal{L}_{cn}\Leftrightarrow\min-\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k \tag{17}$$

Minimizing $\mathcal{L}_{cur}$ in Eq. 6 can be written as

$$\min\mathcal{L}_{cur}\Leftrightarrow\min\frac{1}{m^2}\sum_{P_i\in\mathcal{P}}\sum_{P_t\in\mathcal{P}}\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t \tag{18}$$

Combining Eq. 16, Eq. 17 and Eq. 18. Then, we have

$$\begin{aligned}\mathcal{L}_{\mathcal{P}} &= -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\frac{1}{|P_i|}\sum_{v_j\in P_i}g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j+\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k-\frac{1}{m}\sum_{P_t\in\mathcal{P}}\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t \\ &= -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\frac{1}{|P_i|}\sum_{v_j\in P_i}g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j+\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k-\sum_{P_t\in\mathcal{P}}\frac{1}{m}\log\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t\end{aligned} \tag{19}$$

According to Jensen's inequality, we have

$$
\begin{aligned}
\mathcal{L}_{\mathcal{P}} &= -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\frac{1}{|P_i|}\sum_{v_j\in P_i} g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j + \mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k - \sum_{P_t\in\mathcal{P}}\frac{1}{m}\log\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t \\
&\le -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\frac{1}{|P_i|}\sum_{v_j\in P_i} g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j + \mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k - \log\sum_{P_t\in\mathcal{P}}\frac{\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t}{m} \\
&\le -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\frac{1}{|P_i|}\sum_{v_j\in P_i} g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j + \mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k - \log\sum_{P_t\in\mathcal{P}}\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t \\
&\le -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\frac{1}{|P_i|}\sum_{v_j\in P_i} g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j + \mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k - \log(\exp(\frac{1}{|P_i|}\sum_{v_j\in P_i} g_\varphi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j) + \sum_{P_t\in\mathcal{P}}\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t) \\
&= -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\log\frac{\exp(\frac{1}{|P_i|}\sum_{v_j\in P_i} g_\phi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j + \mathbf{c}_i^{\mathrm{T}}\mathbf{c}_k)}{\exp(\frac{1}{|P_i|}\sum_{v_j\in P_i} g_\varphi(\mathbf{c}_i)^{\mathrm{T}}\mathbf{v}_j) + \sum_{P_t\in\mathcal{P}}\exp\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_t}
\end{aligned}
\tag{20}
$$

To this end, we derive the upper bound of the combination loss, as shown in Eq. 7. □

A.2 PROOF OF THEOREM 1

Theorem 1. *Let $\mathcal{N}(v)$ denote the neighbors of v. Minimizing the community contrastive loss $\mathcal{L}_{\mathcal{P}}$ will try to minimize the alignment loss between one-hop neighbors, which is defined as*

$$
\mathcal{L}_{alig} = \frac{1}{n}\sum_{v_j\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_j)|}\sum_{v_i\in\mathcal{N}(v_j)}\|\mathbf{v}_j - g_\varphi(\mathbf{v}_i)\|_2 \tag{21}
$$

where $\mathbf{v}_j = f_\theta(G)[v_j]$ and $\mathbf{v}_i = f_\theta(G^0)[v_i]$.

Proof. Let $S(v_i)$ represents the set of one-hop neighbor nodes of v_i in the same community and $M(v_i) = \mathcal{N}(v_i) - S(v_i)$ represents the set of one-hop neighbor nodes of v_j not in the same community. Thus, we have

$$
\begin{aligned}
\mathcal{L}_{cr} &= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|\mathbf{c}_i - \mathbf{v}_j\|_2^2 \\
&\overset{\mathrm{c}}{=} \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|\sum_{v\in P_i}\mathbf{v} - \mathbf{v}_j\|_2^2 \\
&\Rightarrow -\frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}(\mathbf{v}_j^T\sum_{v\in S(v_j)}\mathbf{v}) \\
&\overset{\mathrm{c}}{=} \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|\mathbf{v}_j - \sum_{v\in S(v_j)}\mathbf{v}\|_2
\end{aligned}
\tag{22}
$$

For the loss $\mathcal{L}_{cn}$ in Eq. 5, we have

$$
\begin{aligned}
\mathcal{L}_{cn} &= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\|\mathbf{c}_i - \mathbf{c}_k\|_2^2 \\
&\overset{\mathrm{c}}{=} \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{P_k\in\mathcal{N}(P)}\|\sum_{v_j\in P_i}\mathbf{v}_j - \sum_{v\in P_k}\mathbf{v}\|_2^2 \\
&\Rightarrow \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|\mathbf{v}_j - \sum_{u\in M(v_j)}\mathbf{u}\|_2
\end{aligned}
\tag{23}
$$

Combining the above two Equations, we have

$$\begin{aligned}\mathcal{L}_{cr}+\mathcal{L}_{cn} &\geq \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|2\mathbf{v}_j-(\sum_{u\in M(v_j)}\mathbf{u}+\sum_{v\in S(v_j)}\mathbf{v})\|_2 \\ &\overset{\text{c}}{=} \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|2\mathbf{v}_j-\sum_{v\in\mathcal{N}(v_j)}\mathbf{v}\|_2 \\ &\overset{\text{c}}{=} \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\sum_{v_j\in P_i}\|\mathbf{v}_j-\sum_{v\in\mathcal{N}(v_j)}\mathbf{v}\|_2 \\ &\overset{\text{c}}{=} \frac{1}{n}\sum_{v_j\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_j)|}\sum_{v_i\in\mathcal{N}(v_j)}\|\mathbf{v}_j-g_\varphi(\mathbf{v}_i)\|_2\end{aligned} \tag{24}$$

The last equation holds since it is consistent with the goal of $\mathcal{L}_G$, which is to minimize the distance between neighbor representations.

□

A.3 PROOF OF THEOREM 2

Theorem 2. *Suppose the contrastive loss $\mathcal{L}_{\mathcal{P}}$ and $\mathcal{L}_{gc}$ are L-Lipschitz continuous. Then, $\mathcal{L}_{\mathcal{P}}$ can be approximated by $\mathcal{L}_{gc}$ under the graph homophily assumption*

$$\|\mathcal{L}_{gc}-\mathcal{L}_{\mathcal{P}}\|\leq L\|\hat{\mathbf{A}}^k-\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}\|\|\mathbf{X}\|\|\mathbf{W}_m\| \tag{25}$$

where L is the Lipschitz constant and $\mathbf{W}_m$ is the model parameters in GCL framework.

Proof. Let the computational process of contrastive learning represents as a function $l(\cdot)$. Then, we have

$$\begin{aligned}\|\mathcal{L}_G-\mathcal{L}_{\mathcal{P}}\| &= |l(\hat{\mathbf{A}}^k)-l(\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}})| \\ &\leq L\|\hat{\mathbf{A}}^k\mathbf{X}\mathbf{W}_m-\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}\mathbf{X}\mathbf{W}_m\| \\ &= L\|(\hat{\mathbf{A}}^k-\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}})\mathbf{X}\mathbf{W}_m\| \\ &\leq L\|\hat{\mathbf{A}}^k-\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}\|\|\mathbf{X}\|\|\mathbf{W}_{\mathrm{m}}\|\end{aligned} \tag{26}$$

Intuitively, our loss can be viewed as standard contrastive loss performed on $\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}$. □

A.4 PROOF OF THEOREM 3

To prove Theorem 3, we first introduce a lemma, which provides the following theoretical guarantees for the model learned using spectral contrastive loss [HaoChen et al., 2021].

Lemma 1. *Let f^* be the minimizer of the spectral contrastive loss: $\mathcal{L}_{scl}=-2\sum_{x,x'}w_{xx'}\cdot f(x)^{\mathrm{T}}f(x')+\sum_{x,x'}w_{xx'}\cdot(f(x)^{\mathrm{T}}f(x'))^2$, where $w_{xx'}$ is the probability of a random positive pair being (x,x') while w_x the probability of a random selected data point being x. Then, we have*

$$\mathbb{E}_{v_i\in\mathcal{V}}\|y(v_i)-\hat{y}[f^*(v_i)]\|_2^2\leq\frac{1-\phi_G}{\lambda_{d+1}} \tag{27}$$

where λ_{d+1} is the $d+1$ smallest eigenvalue of normalized matrix $\hat{\mathbf{A}}$ and ϕ_G is the graph homophily ratio, defined as

$$\phi_G=\frac{1}{n}\sum_{v_i\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j\in\mathcal{N}(v_i)}\mathbb{1}[y(v_i)=y(v_j)] \tag{28}$$

According to Lemma 1, we only need to prove that our loss in Eq. 7 can be expressed as a spectral contrastive loss to complete the proof.

Theorem 3. *Let f_θ^* be the optimal model parameters obtained by the global minimizer of $\mathcal{L}_\mathcal{P}$ and $y(v)$ denote the label of v. Then, there exists a linear classification function $\hat{y} : \mathcal{V} \to \mathbb{R}^c$ such that the error upper bound is*

$$\mathbb{E}_{v\in\mathcal{V}}\|y(v) - \hat{y}[f_\theta^*(v)]\|_2^2 \leq \frac{1-\phi_\mathcal{P}}{\lambda_{d+1}} \tag{29}$$

where λ_{d+1} is the $d+1$ smallest eigenvalue of diffusion matrix $\hat{\mathbf{A}}^k$ and $\phi_\mathcal{P}$ is the partition homophily ratio, defined as

$$\phi_\mathcal{P} = \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|P_i|}\frac{1}{|P_i|}\sum_{v_j\in P_i}\sum_{v_k\in P_i}\mathbb{1}[y(v_j) = y(v_k)] \tag{30}$$

where $\mathbb{1}[\cdot]$ is the indicator function.

Proof. Let $\mathcal{N}(v_i)$ denotes the neighbors of node v_i in the matrix $\mathbf{P}\hat{\mathbf{P}}^\mathrm{T}$. According to Theorem 1, the positive node pairs of our loss in Equ. 7 can be expressed as the one-hop neighbors of node. Thus, we have

$$\begin{aligned}
\mathcal{L}_P^+ &= \frac{1}{n}\sum_{v_i\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j\in\mathcal{N}(v_i)}\|\mathbf{v}_i - \mathbf{v}_j\|_2^2 \\
&\overset{c}{=} -\frac{1}{n}\sum_{v_i\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j\in\mathcal{N}(v_i)}\mathbf{v}_i^\mathrm{T}\mathbf{v}_j \\
&\overset{c}{=} -\frac{1}{n}\sum_{v_i\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j\in\mathcal{N}(v_i)}2\cdot\mathbf{v}_i^\mathrm{T}\mathbf{v}_j
\end{aligned} \tag{31}$$

For the negative node pairs in Equ. 7, we have

$$\begin{aligned}
\mathcal{L}_\mathcal{P}^- &= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\log\sum_{P_t\in\mathcal{P}}\exp(\mathbf{c}_i^\mathrm{T}\mathbf{c}_t) \\
&= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\log(\sum_{P_t\in\mathcal{P}}\frac{\exp(\mathbf{c}_i^\mathrm{T}\mathbf{c}_t)}{m}m) \\
&= \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\log\sum_{P_t\in\mathcal{P}}\frac{\exp(\mathbf{c}_i^\mathrm{T}\mathbf{c}_t)}{m} + \log m \\
&\geq \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\log\sum_{P_t\in\mathcal{P}}\frac{\exp(\mathbf{c}_i^\mathrm{T}\mathbf{c}_t)}{m}
\end{aligned} \tag{32}$$

According to Jensen's inequality, we have

$$\begin{aligned}
\mathcal{L}_\mathcal{P}^- &\geq \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\log\sum_{P_t\in\mathcal{P}}\frac{\exp(\mathbf{c}_i^\mathrm{T}\mathbf{c}_t)}{m} \\
&\geq \frac{1}{m}\sum_{P_i\in\mathcal{P}}\frac{1}{|\mathcal{N}(P_i)|}\sum_{P_k\in\mathcal{N}(P_i)}\frac{1}{m}\sum_{P_t\in\mathcal{P}}\mathbf{c}_i^\mathrm{T}\mathbf{c}_t \\
&\Rightarrow \frac{1}{m^2}\sum_{P_i\in\mathcal{P}}\sum_{v_j\in P_i}\sum_{v_t\in\mathcal{V}}\mathbf{v}_j^\mathrm{T}\mathbf{v}_t \\
&\overset{c}{=} \frac{1}{n^2}\sum_{v_i\in\mathcal{V}}\sum_{v_j\in\mathcal{V}}\mathbf{v}_i^\mathrm{T}\mathbf{v}_j \\
&\overset{c}{=} \frac{1}{n^2}\sum_{v_i\in\mathcal{V}}\sum_{v_j\in\mathcal{V}}(\mathbf{v}_i^\mathrm{T}\mathbf{v}_j)^2
\end{aligned} \tag{33}$$

□

Combining the Eq. 31 and Eq. 33, we have

$$\mathcal{L}_{\mathcal{P}} \geq -\frac{1}{n}\sum_{v_i\in\mathcal{V}}\frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j\in\mathcal{N}(v_i)} 2\cdot \mathbf{v}_i^{\mathrm{T}}\mathbf{v}_j + \frac{1}{n}\frac{1}{n}\sum_{v_i\in\mathcal{V}}\sum_{v_j\in\mathcal{V}}(\mathbf{v}_i^{\mathrm{T}}\mathbf{v}_j)^2 \tag{34}$$

Since $\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}$ can be used as a low-rank approximation of the k-step diffusion matrix $\hat{\mathbf{A}}^k$, and the $\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}$ connects nodes within the same community to each other and removes the connections between nodes in different communities. According to Lemma 1, our loss can be expressed as a spectral loss on $\mathbf{P}\hat{\mathbf{P}}^{\mathrm{T}}$. Thus, we complete the proof of Theorem 3.

Table 6: Datasets Statistics.

Dataset	Nodes	Edges	Classes	Features	Train / Val / Test
Cora	2708	10556	7	1433	140 /500 / 1000
CiteSeer	3327	9104	6	3703	120 / 500 / 1000
Pubmed	19717	88648	3	500	60 / 500 / 1000
Wiki-CS	11701	431206	10	300	1170 / 1171 / 9360
Amazon-Computer	13752	491722	10	767	1375 / 1376 / 11001
Amazon-Photo	7650	238162	8	745	765 / 765 / 6120
Coauthor-CS	18333	163788	15	6805	1833 / 1834 / 14666
Coauthor-Physics	34493	495924	5	841	3449 / 3450 / 27594
Texas	183	309	5	1703	87 /59 / 37
Wisconsin	251	499	5	1703	120 /80 / 51
Cornell	183	295	5	1703	87 /59 /37
Actor	7600	29926	5	932	3634 /2432 / 1520
Ogbn-Arxiv	169343	1166243	40	128	90941 /29799 / 48603
Ogbn-Products	2449029	61859140	47	100	196615 /39323 / 2213091

B EXPERIMENTAL STUDY

B.1 DETAILS OF DATASETS

The statistics of all datasets are summarized in Table 6.

- **Cora, CiteSeer and Pubmed.** They are three citation network datasets, where nodes represent articles, edges represent citation relationships, features consist of bag-of-words representations of articles, labels correspond to the academic domains or metadata of the articles.
- **Wiki-CS.** It is a ciation network extracted from Wikipedia dataset, where nodes represent articles about computer science, edges represent the hyperlinks between two articles, features consist of bag-of-words representations of articles, and labels are different fields of each article.
- **Amazon-Computer and Amazon-Photo.** They are two co-purchase networks from Amazon dataset, where nodes represent products, edges represent pairs of products often bought together, features consist of bag-of-words representations of product reviews, and labels are the category of products.
- **Coauthor-CS and Coauthor-Physics.** They are two co-authorship networks extracted from Microsoft Academic Graph in KDD Cup 2016 challenge, where nodes represent authors, edges represent co-authorship relationships, features consist of bag-of-words representations of article keywords, and labels are the research fields of authors.
- **Texas, Wisconsin and Cornell.** They are three subsets of WebKB dataset, where nodes represent web pages, edges represent hyperlinks, features are described by a word vector comprising keywords extracted from page content, labels represent the categories of the web pages.
- **Actor.** It is a an actor co-occurrence network, where nodes represent actors, edges represent two actors have co-occurrence in the same movie, features represent the key word in the Wikipedia pages, and labels are the words of corresponding actors.

Table 7: Details of the Hyper-parameters in Our Method.

Dataset	lr	T	k	β	d	α	τ
Cora	0.005	100	3	0.02	1024	0.1	0.1
CiteSeer	0.0005	100	2	0.04	2048	0.2	0.8
Pubmed	0.001	75	2	0.05	1024	0.4	0.45
Wiki-CS	0.005	50	2	0.01	2048	0.9	0.35
Amazon-Computer	0.0005	150	2	0.1	2048	0.6	0.2
Amazon-Photo	0.001	150	1	0.03	2048	0.4	0.6
Coauthor-CS	0.005	100	1	0.05	1024	0.2	0.6
Coauthor-Physics	0.1	25	1	0.09	2048	0.5	0.55
Texas	0.0005	100	0	0.05	4096	1.0	0.8
Wisconsin	0.001	25	0	0.07	4096	0.9	1.0
Cornell	0.0005	150	0	0.06	2048	0.9	0.3
Actor	0.001	25	0	0.01	2048	0.9	0.75
Ogbn-Arxiv	0.001	25	3	0.03	1500	0.6	0.4
Ogbn-Products	0.002	25	10	0.001	128	0.8	0.7

- **Ogbn-Arxiv and Ogbn-Products.** They are two datasets in the Open Graph Benchmark. Ogbn-Arxiv is a cation network, where nodes represent articles, edges represent citation relationships, features through averaging the embeddings of words in its title and abstract, labels represent the categories of the articles. Ogbn-Products is a co-purchase network, where nodes represent products, edges represent pairs of products often bought together, features consist of bag-of-words representations of product reviews, and labels are the category of the products.

B.2 PARAMETERS SETTINGS

For fair comparison, we use the results provided by the authors in their original papers. For baselines not reported on specific datasets or those not utilizing standard public data splits, we carefully tune the hyper-parameters based on the authors' official code. We implement our method in PyTorch with Adam optimizer. All experiments are conducted on a machine with Intel 13900KF CPU, 128GB RAM and RTX4090 GPU, running Windows 11. Each experiment is repeated for 20 times.

We use one layer of PCN and one layer of MLP as the mapping head to implement our method. The learning rate lr is chosen from $0.0001, 0.0005, 0.001, 0.002, 0.005, 0.1$. The training epochs T are chosen from $25, 50, 75, 100, 150$. The order of the diffusion matrix k is chosen from 0, 1, 2, 3, and Ogbn-Products is set to 10 since it is more complex. The partition rate β, temperature parameter τ and coefficient α are selected from 0 to 1. The embedding dimension d is chosen from 1024, 1500, 2048, 4096, and Ogbn-Products is set to 128. The hyper-parameters for each dataset are listed in Table 7. More detailed hyper-parameters can be found in the provided code.

B.3 IMPACTS OF PARAMETER α

We explore the impacts of coefficient α for controlling community neighbor loss in Figure 6. According to Table 8, we found across 13 datasets that the optimal α generally correlates with the homophily level: higher homophily ratio requires a smaller α (emphasizing local information), while lower homophily ratio requires a larger α (emphasizing high-order structural information). This trend was observed in most of the datasets.

B.4 PRETEXT TIME

We report the time consumption for community segmentation using METIS (the fastest algorithm) and Structure Entropy (the algorithm achieving the highest accuracy) for all datasets in Table 8. It can be seen that METS is very efficient. Given that METIS offers a considerable speed advantage in constructing partitions, we propose a practical recommendation based on the performance-efficiency trade-off: structural entropy is suitable for medium-scale graphs to potentially achieve better performance, whereas METIS is recommended for large-scale graphs to ensure scalability.

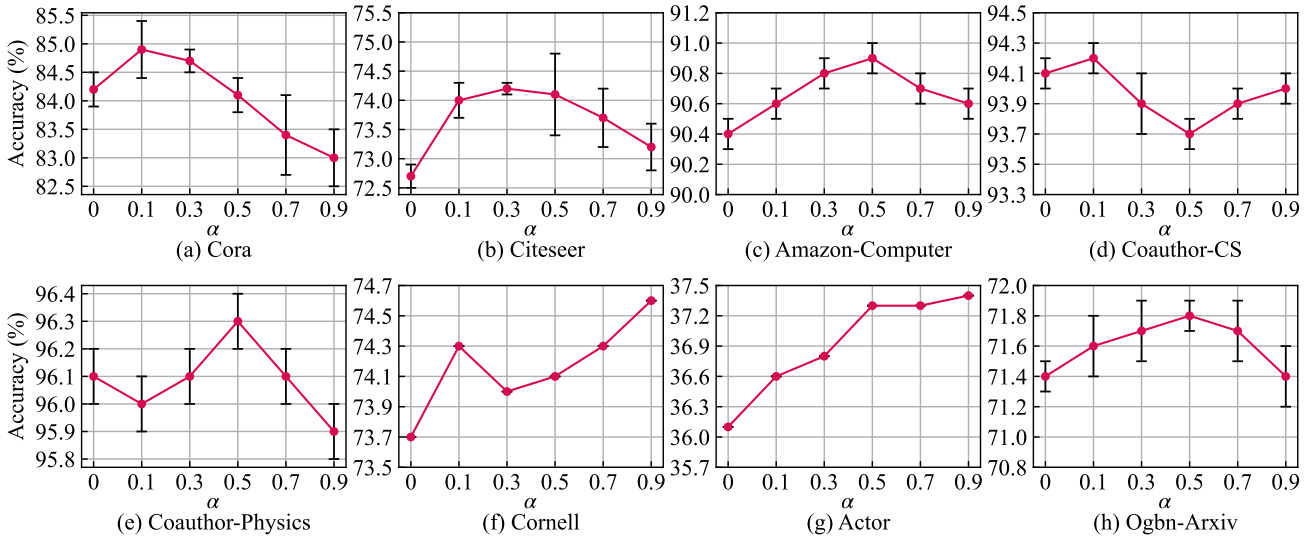

Figure 6: Sensitivity Analysis of the Hyper-parameters α. We Omitted the Variance Lines in Cornell and Actor.

Table 8: Partition Time on All Dataset.

Dataset	METIS Time	SE Time	Homophily Ratio	α
Cora	0.021s	1.43s	0.77	0.1
CiteSeer	0.027s	1.51s	0.63	0.2
Pubmed	0.385s	12.42s	0.66	0.4
Wiki-CS	0.247s	178.14s	0.57	0.9
Amazon-Computer	0.773s	7.04m	0.70	0.6
Amazon-Photo	0.275s	62.58s	0.77	0.4
Coauthor-CS	0.437s	33.22s	0.76	0.2
Coauthor-Physics	1.226s	11.72m	0.85	0.5
Texas	0.002s	1.18s	0.0013	1.0
Wisconsin	0.003s	1.18s	0.0941	0.9
Cornell	0.002s	1.17s	0.0311	0.9
Actor	0.095s	3.72s	0.0110	0.9
Ogbn-Arxiv	6.995s	3h	0.42	0.6
Ogbn-Products	79.73s	>12h	0.46	0.8