Does GCL Need a Large Number of Negative Samples? Enhancing Graph Contrastive Learning with Effective and Efficient Negative Sampling

Yongqi Huang[1†], Jitao Zhao[1†], Dongxiao He[1*], Di Jin[1], Yuxiao Huang[2], Zhen Wang[3]

[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Department of Data Science, George Washington University, Washington, D.C., U.S.A.
[3]School of Cybersecurity, Northwestern Polytechnical University, Xi'an, China
{yqhuang, zjtao, hedongxiao, jindi}@tju.edu.cn, yuxiaohuang@gwu.edu, w-zhen@nwpu.edu.cn

Abstract

Graph Contrastive Learning (GCL) aims to self-supervised learn low-dimensional graph representations, primarily through instance discrimination, which involves manually mining positive and negative pairs from graphs, increasing the similarity of positive pairs while decreasing negative pairs. Drawing from the success of Contrastive Learning (CL) in other domains, a consensus has been reached that the effectiveness of GCLs depends on a large number of negative pairs. As a result, despite the significant computational overhead, GCLs typically leverage as many negative node pairs as possible to improve model performance. However, given that nodes within a graph are interconnected, we argue that nodes cannot be treated as independent instances. Therefore, we challenge this consensus: *Does employing more negative nodes lead to a more effective GCL model?* To answer this, we explore the role of negative nodes in the commonly used InfoNCE loss for GCL and observe that: (1) Counterintuitively, a large number of negative nodes can actually hinder the model's ability to distinguish nodes with different semantics. (2) A smaller number of high-quality and non-topologically coupled negative nodes are sufficient to enhance the discriminability of representations. Based on these findings, we propose a new method called GCL with Effective and Efficient Negative samples, E2Neg, which learns discriminative representations using only a very small set of representative negative samples. E2Neg significantly reduces computational overhead and speeds up model training. We demonstrate the effectiveness and efficiency of E2Neg across multiple datasets compared to other GCL methods.

Code — https://github.com/hedongxiao-tju/E2Neg

Introduction

Graph Neural Networks (GNNs) (Kipf and Welling 2017; Hamilton, Ying, and Leskovec 2017) have emerged as a powerful strategy for graph mining, aiming to encode structured high-dimensional graphs into low-dimensional embeddings. GNNs are widely applied in various scenarios such as social recommendation (Fan et al. 2019), community detection (Jin et al. 2023, 2021), and sarcasm detection (Wang et al. 2023). Training high-quality GNNs typically requires an extensive amount of labels, which is labor-intensive and costly. The scarcity of labels limits the applicability of GNNs due to label dependency.

Graph Contrastive Learning (GCL) has recently gained significant attention for its ability to self-supervisedly train GNNs, producing expressive representations. Inspired by the success of contrastive learning in other domains, most GCLs are based on instance discrimination (Zhu et al. 2020, 2021b). Specifically, they first mine positive and negative samples, then train the model using the InfoNCE loss (van den Oord, Li, and Vinyals 2018), which maximizes the similarity between positive pairs while minimizing the similarity between negative pairs. For an anchor node, positive samples are typically constructed through augmentation to align the invariant semantics of the perturbed data, while negative samples are often chosen from other nodes to enhance the discriminability of the representations.

The effectiveness of instance-discrimination-based GCLs highly depends on the sampling strategy. ProGCL (Xia et al. 2022) estimates the probability of negatives being true negatives by using a Beta Mixture Model (BMM), and can further leverage this probability to weight the negatives and synthesize new negatives. Local-GCL (Zhang et al. 2022) removes first-order neighbors from negatives to positives based on graph homophily assumption. HomoGCL (Li et al. 2023) estimates the probability of neighbors being true positives using graph homophily and calculates node similarity through soft clustering, using it as a weight for positives. B^2-Sampling (Liu et al. 2023) selects representative negatives by designing a balanced sampling strategy and corrects the labels which can be hard to learn. Although these methods improved the sampling strategy and achieved some success, all these methods share a consensus that incorporating more negative samples improves model performance. This consensus has been discovered and validated by many researchers in Computer Visions (CV) (He et al. 2020; Chen et al. 2020) and other domains.

However, a gap has emerged regarding this consensus between graph and other domains: This consensus is based on the premise that instances are independent of each other. This premise is valid within the CV. In graph data, the presence of topological connections between nodes results in instances that do not satisfy the independence premise. This

[†]These authors contributed equally.
[*]Corresponding Author.
Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

raises an important question: ***Does employing more negative nodes lead to a more effective GCL model?***

To answer this question, we theoretically investigate the role of topologically coupled negative nodes in the InfoNCE loss. We first demonstrate that within the topological receptive field of graph encoders, multiple topologically coupled node representations share the same semantics with only minor differences, thereby violating the assumption of instance independence. Further analysis reveals some interesting and counter-intuitive findings: 1) A large number of negative sample nodes can actually hinder the model's ability to learn semantic distinctions between nodes. 2) For node-level tasks, a small number of representative negative samples are sufficient to produce distinguishable representations.

Based on these findings, we propose a new GCL method called GCL with Effective and Efficient Negative samples (E2Neg), which enables the GNN encoders to learn discriminative representations using only a very small number of representative negative samples (typically fewer than 50). Specifically, E2Neg preprocesses the graph using spectral clustering, selects representative nodes based on centrality, and reconstructs subgraphs around these nodes to identify topologically decoupled and representative samples. We further design a data augmentation for these negative samples, aligning the semantics of nodes that are topologically coupled with them. Additionally, by training with only a small selection of samples, E2Neg significantly reduces computational overhead and boosts training speed.

Our contributions can be summarized as follows:

- We explore the function of negative samples in InfoNCE and find that increasing the number of negatives does not improve the performance of the model. Instead, it may hinder the model's ability to distinguish nodes with different semantics. Moreover, using only a very small subset of nodes as negatives is sufficient to enhance the discriminability of representations.
- Based on these findings, we propose a novel GCL method called GCL with Effective and Efficient Negative samples (E2Neg), where training only requires a very small set of representative negatives. E2Neg reduces computational overhead and significantly accelerates model training.
- We conduct extensive experiments on multiple datasets. Experimental results demonstrate the effectiveness and efficiency of E2Neg.

Preliminaries

Notations. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_N\}$ is the set of nodes, $|\mathcal{V}| = N$, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. $\boldsymbol{X} \in \mathbb{R}^{N \times F}$ is the feature matrix, $\boldsymbol{X}_i \in \mathbb{R}^F$ is the feature of v_i. $\boldsymbol{A} \in \{0,1\}^{N \times N}$ is the adjacency matrix, and $\boldsymbol{A}_{ij} = 1$ iff $(v_i, v_j) \in \mathcal{E}$. The normalized Laplacian matrix is defined as $\boldsymbol{L} = \boldsymbol{I}_N - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{I}_N$ is an identity matrix and $\boldsymbol{D}$ is diagonal degree matrix with $\boldsymbol{D}_{ii} = \sum_{j=1}^{N} \boldsymbol{A}_{ij}$ for $i \in \mathcal{V}$.

Eigenvector and Eigenvalue. The normalized graph Laplacian can be decomposed as $\boldsymbol{L} = \boldsymbol{U}\Lambda\boldsymbol{U}^\top$, where $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_N] \in \mathbb{R}^{N \times N}$ is the eigenbasis, which consists of a set of eigenvectors. $\Lambda = diag(\{\lambda_i\}_{i=1}^N)$ are the eigenvalues. As for each eigenvector $\boldsymbol{u}_i \in \mathbb{R}^N$ has a corresponding eigenvalue λ_i. Without loss of generality, assume $0 \leq \lambda_1 \cdots \leq \lambda_N \leq 2$.

Graph Contrastive Learning. Our objective is to train an encoder f to get representation $\boldsymbol{H} = f(\boldsymbol{A}, \boldsymbol{X}) \in \mathbb{R}^{N \times D}$ without label information. Augmented graphs are typically generated through augmentation techniques, such as edge dropping and feature masking. These augmented graphs are fed into f, which generates the representations $\boldsymbol{H}^\alpha$ and $\boldsymbol{H}^\beta$ for the two views. The objective function is defined by InfoNCE (van den Oord, Li, and Vinyals 2018) loss as:

$$\ell\left(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_i^\beta\right) = \log \frac{e^{\theta(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_i^\beta)/\tau}}{e^{\theta(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_i^\beta)/\tau} + \sum_{k \neq i} e^{\theta(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_k^\beta)/\tau} + \sum_{k \neq i} e^{\theta(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_k^\alpha)/\tau}}, \tag{1}$$

where $\theta(\cdot)$ denotes the cosine similarity function, with $\theta(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_i^\beta)$ representing the similarity between the same node in different views, which constitutes positive samples. Conversely, $\theta(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_k^\beta)$ represents the similarity between different nodes in different views, and $\theta(\boldsymbol{h}_i^\alpha, \boldsymbol{h}_k^\alpha)$ represents the similarity between different nodes in the same view, both of these cases correspond to negative samples.

Negative Sampling Analysis

In this section, we theoretically analyze negative samples in the InfoNCE loss. We observe that the current definition of negatives tends to diminish the distinctiveness of samples during train. To address this issue, we propose a new sampling strategy that effectively mitigates this limitation.

Definition 1 (Receptive Field $\mathcal{R}$) *The receptive field of v_i is defined as the set of nodes whose aggregated information influences the representation of node v_i during the propagation process. It can be formulated as:*

$$\mathcal{R}(v_i) = \bigcup_{t=0}^{\hat{k}} \{v_j \in \mathcal{V} \mid \textit{dist}(v_i, v_j) = t \textit{ and } w_t(v_i, v_j) > \xi\}, \tag{2}$$

where $\hat{k}$ is the number of propagation layers, dist(v_i, v_j) represents the shortest path distance between nodes v_i and v_j, $w_t(v_i, v_j)$ is the weight of the information aggregated from v_j to v_i at layer t, and ξ is a threshold for significant contributions. The receptive field expands as the number of layers increases, incorporating a broader range of nodes that contribute to the representation of v_i.

Specifically, if the number of propagation layers is $\hat{k}$, the receptive field of v_i includes all nodes within a distance of $\hat{k}$ that significantly contribute to $\boldsymbol{h}_i$. $w_t(v_i, v_j)$ quantify this contribution, with higher values indicating a greater impact on the representation $\boldsymbol{h}_i$.

Definition 2 (Semantic Block $\mathcal{S}$) *A semantic block $\mathcal{S}$ is defined as a set of nodes within a graph that form a cohesive*

group centered around a core semantic s_c. This block $\mathcal{S}$ can be formulated as:

$$\mathcal{S} = \{v_j \in \mathcal{V} \mid \|x_j - s_c\| \leq \delta\}, \tag{3}$$

where s_c represents the core semantic of $\mathcal{S}$, and δ is a small value determining the allowable difference between the nodes and s_c.

A corresponding $\mathcal{S}$ exists for each graph node. These semantic blocks do not overlap, meaning that different semantic blocks are non-coupled.

Theorem 1 (Semantic Block-Based Decomposition) *Assume that a graph contains k semantic blocks $\mathcal{S} = \{\mathcal{S}_1, \cdots, \mathcal{S}_k\}$, corresponding to a set of core semantics $s = \{s_1, \ldots, s_k\}$, where each s_j is the core semantic associated with the block $\mathcal{S}_j$. If an anchor node v_i belongs to the semantic block $\mathcal{S}_j$, x_i can be decomposed as:*

$$x_i = s_j + \epsilon_i, \tag{4}$$

where s_j is the core semantic of the block $\mathcal{S}_j$, and ϵ_i represents the individual deviation of v_i from this core semantic.

The feature difference between nodes varies depending on whether they belong to the same semantic block. For nodes within the same semantic block, the difference is determined by the individual deviations from the core semantic. For nodes belonging to different semantic blocks, the difference is influenced by the core semantic associated with each block. Given two different semantic blocks $\mathcal{S}_p$ and $\mathcal{S}_q$, it can be expressed as:

$$\begin{aligned} \triangle\text{diff}_{\text{intra}} &= \|\epsilon_i - \epsilon_j\|, \quad \text{if } v_i, v_j \in \mathcal{S}_p, \\ \triangle\text{diff}_{\text{inter}} &= \|s_p - s_q\|, \quad \text{if } v_i \in \mathcal{S}_p \text{ and } v_j \in \mathcal{S}_q, \end{aligned} \tag{5}$$

where $\triangle\text{diff}_{\text{intra}}$ represents the difference between negative samples within $\mathcal{S}_p$. An increase in this difference corresponds to a decrease in the similarity of negative samples within the same semantic block. Conversely, $\triangle\text{diff}_{\text{inter}}$ represents the difference between negative samples across $\mathcal{S}_p$ and $\mathcal{S}_q$. An increase in this difference is equivalent to reducing the similarity of negative samples between $\mathcal{S}_p$ and $\mathcal{S}_q$, precisely the behavior we desire from the model. This approach ensures that while distinguishing between semantic blocks, the model maintains a high similarity between node pairs within the same semantic block.

Theorem 2 (Sample Threshold Gradient Boundary) *Assume that during the optimization process, we consider two types of negative samples: inter- and intra-semantic block negatives, denoted as $\mathbf{N}_{inter}$ and $\mathbf{N}_{intra}$, respectively. During optimization, the gradients associated with these negatives are accumulated and compared. For the average similarity $\hat{\theta}(h_i, h_{j'})$ between samples from different semantic blocks, P represents the number of sample pairs in the same semantic block, τ is the temperature coefficient. When $Pe^{1/\tau} = \sum_{j'=1, j' \neq i}^{N-P} e^{\theta(h_i, h_{j'})/\tau}$, the model achieves a balance in distinguishing between $\mathbf{N}_{inter}$ and $\mathbf{N}_{intra}$.*

The model's discriminative focus is determined by the ratio of these gradients, which can be expressed as:

$$\frac{\sum_{(u,v) \in \mathbf{N}_{\text{inter}}} \nabla \mathcal{L}(u, v)}{\sum_{(u,v) \in \mathbf{N}_{\text{intra}}} \nabla \mathcal{L}(u, v)} \geq \gamma, \tag{6}$$

where γ act as a boundary to determine whether the model focuses more on inter- or intra-semantic block distinctions. Define $\Delta\text{diff}_{\text{inter}} = \|s_p - s_q\|$ representing the semantic difference between inter-semantic block pairs and $\Delta\text{diff}_{\text{intra}} = \|\epsilon_u - \epsilon_v\|$ representing the variation within the same semantic block. Their differences quantify the distinctions within respective semantic blocks. When $\Delta\text{diff}_{\text{intra}}$ increases, the model reduces the similarity within the same semantic block, encouraging node separation. When $\Delta\text{diff}_{\text{inter}}$ increases, the model decreases the similarity between different blocks, improving distinction across semantic blocks. τ helps balance these effects for effective block distinction while maintaining similarity within the same block.

We analyze the issues with existing sampling methods from the perspective of semantic blocks. To do so, we take the derivative of the InfoNCE loss for the similarity of negative samples, which yields the following expressions:

$$\frac{\partial \mathcal{L}_{\text{InfoNCE}}}{\partial \theta(h_i, h_j)} = \frac{1}{\tau} \frac{\phi(i, j)}{\phi(i, i') + \sum_{k=1, k \neq i}^{n} (\phi(i, k) + \phi(i, k'))}, \tag{7}$$

where $\phi(i, j) = e^{\theta(h_i, h_j)/\tau}$, and $\theta(h_i, h_j)$ represents the gradient of any negative sample pair in the model, where i and j may belong to the same semantic block or different semantic blocks. By summing the gradients of all negative samples for an anchor node v_i, we can break down the total loss into contributions from pairs within the same semantic block and pairs across different semantic blocks, as follows:

$$\begin{aligned} \text{SG}(i)_{\text{intra}} &= \sum_{j=1, j \neq i}^{P} \frac{\partial \mathcal{L}_{\text{InfoNCE}}}{\partial \theta(h_i, h_j)}, \\ \text{SG}(i)_{\text{inter}} &= \sum_{j'=1, j' \neq i}^{N-P} \frac{\partial \mathcal{L}_{\text{InfoNCE}}}{\partial \theta(h_i, h_{j'})}, \end{aligned} \tag{8}$$

where $\text{SG}(i)_{\text{intra}}$ represents the sum of the gradients for negatives within the same semantic block, $\text{SG}(i)_{\text{inter}}$ represents the gradients for negatives across different semantic blocks. P represents the number of sample pairs in the same semantic block as v_i. When the sum of gradients for negatives within the same semantic block exceeds that for negatives across different semantic blocks, the model tends to differentiate negatives within the same semantic block. Conversely, if the sum of gradients across different semantic blocks is greater, the model will focus on distinguishing negatives between different semantic blocks. To explore how the model switches between these targeting modes, we first assume that $\text{SG}(i)_{\text{intra}} = \text{SG}(i)_{\text{inter}}$. Through Eq. (7) and (8), we obtain the following expression:

$$\sum_{j=1, j \neq i}^{P} e^{\theta(h_i, h_j)/\tau} = \sum_{j'=1, j' \neq i}^{N-P} e^{\theta(h_i, h_{j'})/\tau}, \tag{9}$$

since within the same semantic block, the difference between samples, as expressed by $\triangle\text{diff}_{\text{intra}}$ in Equation (5), is much smaller than $\triangle\text{diff}_{\text{inter}}$, we assume $\theta(h_i, h_j) = 1$ for simplicity, which can be simplified as follows:

$$Pe^{1/\tau} = \sum_{j'=1, j' \neq i}^{N-P} e^{\theta(h_i, h_{j'})/\tau}. \tag{10}$$

When the sum of $\phi(i, j')$ exceeds this value, the model tends to reduce the similarity of negatives within the same semantic block. Conversely, if the sum of $\phi(i, j')$ is below this value, the model focuses on reducing the similarity of negatives across different semantic blocks. This indicates that the current sampling method in InfoNCE has an efficiency threshold. When the gradient ratio fluctuates around this threshold, the model's ability to distinguish between negatives within and across semantic blocks decreases, impacting its performance.

Finding 1: A large number of negative nodes can hinder the model's ability to distinguish nodes with different semantics. A large number of negatives does not improve the model's ability to differentiate between core semantics. Theorem 2 establishes the existence of a threshold for negatives. When the ratio exceeds this threshold, the model starts to distinguish the ϵ between negatives, hindering performance improvement.

Finding 2: A smaller number of high-quality and non-topologically coupled negative nodes are sufficient to enhance the discriminability of representations. To enhance model performance, it is crucial to differentiate the semantics of nodes belonging to distinct semantic blocks. This differentiation primarily relies on separating the core semantics. Thus, it is adequate to sample solely the nodes with core semantics as negatives, instead of sampling all nodes.

Methodology

In the previous section, we obtain two key findings by taking the derivative of the InfoNCE loss about negative samples, demonstrating that negatives exhibit certain limitations during training, highlighting flaws in the existing sampling strategy. In this section, we primarily introduce the mechanism of E2Neg.

Centrality Sampling. We calculate the normalized Laplacian matrix $\boldsymbol{L}$ and then apply EigenValue Decomposition (EVD). For the resulting eigenvectors $\boldsymbol{U}$, we select the low-frequency ones, as these vectors effectively capture the critical topological structures of the graph. Specifically, we select the first k eigenvectors corresponding to the smallest eigenvalues $\boldsymbol{U}_k = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_k]$. We apply the K-means (MacQueen 1967) algorithm to the rows of $\boldsymbol{U}_k$ to partition the nodes into k clusters:

$$\mathbf{y} = \text{K-means}\left(\{\boldsymbol{u}_i \mid \boldsymbol{u}_i \in \boldsymbol{U}_k\}\right), \tag{11}$$

where λ_k is used to select the low-frequency eigenvectors. The set $\{\boldsymbol{u}_i \mid \boldsymbol{u}_i \in \boldsymbol{U}_k\}$ consists of eigenvectors with eigenvalues $\lambda_i \leq \lambda_k$, which are then clustered using K-means. $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ is the vector of cluster assignments for each node, with y_i representing the cluster label assigned to v_i. For each cluster C_j, where $C_j = \{v_i \mid y_i = j\}$, we further analyze the intra- and inter-cluster properties. By evaluating the node's influence on the cluster's structural integrity, we identify a representative node, the cluster center, which is chosen to capture the cluster's spectral properties. These features are derived from the eigenvectors corresponding to the smallest eigenvalues of the graph's Laplacian matrix, which encapsulate the essential connectivity and structure of the cluster. The cluster center c_i for cluster C_i is determined by:

$$c_i = \arg\max_{v \in C_i} \|\mathbf{x}_v\|, \tag{12}$$

where $\mathbf{x}_v$ represents the spectral feature of v. The norm $\|\mathbf{x}_v\|$ quantifies the node's centrality within the spectral domain, ensuring that the selected cluster center represents the cluster's overall structure.

Topology Reconstruction. Following the sampling process, $\mathcal{G}$ is preprocessed into multiple clusters. For each cluster C_i, we select a cluster center c_i based on cluster centrality. To facilitate graph reconstruction, we identify the $\hat{k}$-hop neighbors of each c_i (denoted as $\hat{k}$) and establish direct connections between these neighbors and the center. This procedure results in the construction of subgraphs, each centered around its respective cluster center, as described below:

$$V_i^{(\hat{k})} = \{v_j \in \mathcal{V} \mid \text{dist}(c_i, v_j) \leq \hat{k}\} \cup \{c_i\}, \tag{13}$$

where the edge set of the subgraph $E_i^{(\hat{k})}$ is given by:

$$E_i^{(\hat{k})} = \{(v_j \to c_i) \mid v_j \in V_i^{(\hat{k})} \setminus \{c_i\}\}, \tag{14}$$

where $v_j \to c_i$ indicates a directed edge from v_j to c_i. For each c_i, a corresponding subgraph $\hat{\mathcal{G}}_i$ is generated. When constructing these subgraphs, we consider the potential issue of topological coupling, which may lead to redundant information propagation, causing a node's features to be transmitted to multiple cluster centers. To address this, we ensure that each neighboring node is connected to at most one cluster center. Additionally, all subgraphs are constructed with directed edges to avoid creating redundant propagation structures, ensuring that the model focuses on training centers rather than the neighboring nodes. Thus, for the selected k cluster centers, each will generate a corresponding subgraph, forming a new reconstructed graph $\hat{\mathcal{G}} = (\hat{\boldsymbol{A}}, \boldsymbol{X}) = \{\hat{\mathcal{G}}_c \mid c = 1, 2, \cdots, k\}$, where $\hat{\boldsymbol{A}}_i = (V_i^{(\hat{k})}, E_i^{(\hat{k})}), \forall i \in \{1, 2, \cdots, k\}$, and $\hat{\boldsymbol{A}}_i \cap \hat{\boldsymbol{A}}_j = \varnothing, \forall i \neq j$. In the following components of the model, we replace $\mathcal{G}$ with $\hat{\mathcal{G}}$ for training.

Augmentation Strategy. Traditional augmentation strategies in GCLs usually employ feature masking and edge dropping to generate the augmented graphs. Nevertheless, due to the substantial reduction in the number of nodes and edges in $\hat{\mathcal{G}}$ compared to $\mathcal{G}$, conventional augmentation strategies can only introduce a restricted amount of disruption. Therefore, we employ a specialized augmentation for $\hat{\mathcal{G}}$, which entails randomly exchanging the cluster centers in $\hat{\mathcal{G}}$ to create the augmented graph, denoted as $\tilde{\mathcal{G}} = (\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{X}}) = \{\tilde{\mathcal{G}}_c \mid c = 1, 2, \cdots, k\}$. The difference between $\hat{\mathcal{G}}$ and $\tilde{\mathcal{G}}$ can be expressed as $c_i^{\hat{\mathcal{G}}} \neq c_i^{\tilde{\mathcal{G}}}, \forall i \in \{1, 2, \cdots, k\}$, where $c_i^{\hat{\mathcal{G}}}$ and $c_i^{\tilde{\mathcal{G}}}$ represent the centers of $\hat{\mathcal{G}}_i$ and $\tilde{\mathcal{G}}_i$ respectively. Both $\hat{\mathcal{G}}$ and $\tilde{\mathcal{G}}$ are utilized to train the encoder f.

Encoder. We use GNN to acquire the node representations by the node attributes and topology structure of the graph,

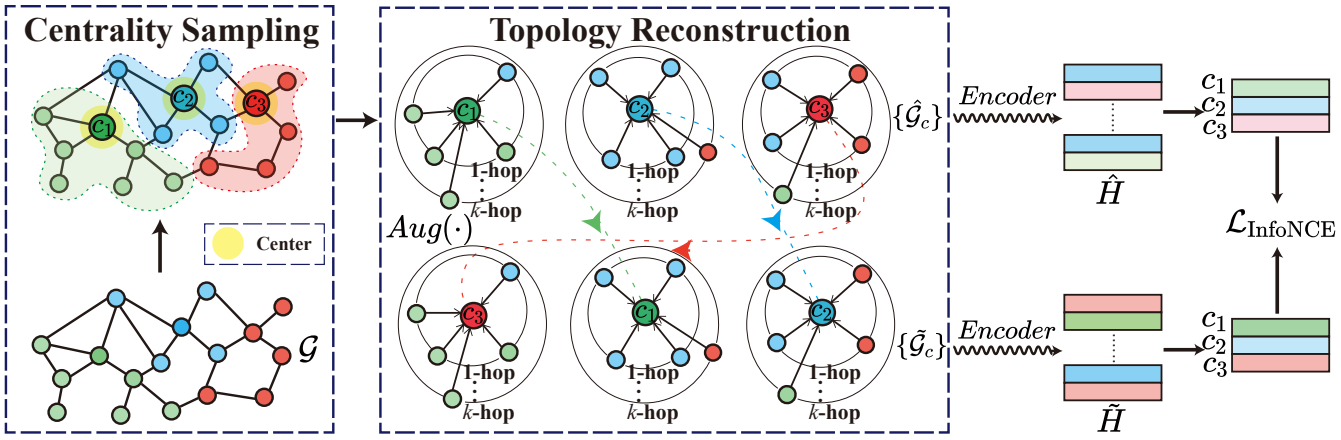

Figure 1: Given a graph $\mathcal{G}$, we use a centrality sampling strategy to select a representative set of nodes from $\mathcal{G}$. Next, we perform topological reconstruction based on the neighbors of these nodes to generate a reconstructed graph $\hat{\mathcal{G}}$. The function $Aug(\cdot)$ is a custom augmentation used to create augmented graph $\tilde{\mathcal{G}}$. $\hat{\mathcal{G}}$ and $\tilde{\mathcal{G}}$ are then input into the encoder f to generate representations $\hat{\boldsymbol{H}}$ and $\tilde{\boldsymbol{H}}$. Finally, we select the representative node set to compute the loss.

which consists of an Encoder and a Projector. The backbone GNN Encoder may use different models (e.g., GCN (Kipf and Welling 2017), GAT (Veličković et al. 2018)). In E2Neg, we set GCN as our backbone and Multi-Layer Perception (MLP) as the Projector. The Encoder uses $\hat{\mathcal{G}}$ and $\tilde{\mathcal{G}}$ as inputs and separately obtains the reconstructed representation $\hat{\boldsymbol{H}}$ and the augmented representation $\tilde{\boldsymbol{H}}$.

Loss Function. We adopt the InfoNCE (van den Oord, Li, and Vinyals 2018) as the loss function. Unlike the settings of other GCLs, we do not feed the entire representations of $\hat{\mathcal{G}}$ and $\tilde{\mathcal{G}}$ into the loss calculation. To correspond with our findings, we optimize the model using only the representations of $c_i^{\hat{\mathcal{G}}}$ and $c_i^{\tilde{\mathcal{G}}}$, as shown in the following equation:

$$\ell\left(\hat{\boldsymbol{h}}_i, \tilde{\boldsymbol{h}}_i\right) = \log \frac{e^{\theta(\hat{\boldsymbol{h}}_i, \tilde{\boldsymbol{h}}_i)/\tau}}{e^{\theta(\hat{\boldsymbol{h}}_i, \tilde{\boldsymbol{h}}_i)/\tau} + \sum_{j \neq i}^{k-1} \left(e^{\theta(\hat{\boldsymbol{h}}_i, \tilde{\boldsymbol{h}}_j)/\tau} + e^{\theta(\hat{\boldsymbol{h}}_i, \tilde{\boldsymbol{h}}_j)/\tau}\right)}, \tag{15}$$

where $i \in \{1, 2, \cdots, k\}$. $\hat{\boldsymbol{h}}_i$ and $\tilde{\boldsymbol{h}}_i$ respectively represent the embeddings of $c_i^{\hat{\mathcal{G}}}$ and $c_i^{\tilde{\mathcal{G}}}$ in $\hat{\boldsymbol{H}}$ and $\tilde{\boldsymbol{H}}$.

E2Neg Effectiveness Analysis. We use spectral clustering to the graph and select central nodes within each cluster to serve as negative samples. This approach allows us to maximize the distinction between negatives across different semantic blocks, rather than within the same block. Even if the cluster does not perfectly align with the semantic blocks of the graph, the difference between non-aligned negatives is amplified from ϵ to the core semantics s of the respective negative samples, achieving the same training effect. To align with the homophily assumption, we reconstruct the topology by subgraph connections for nodes likely belonging to the same semantic block. By linking these nodes to their neighbors, the encoder enables direct aggregation of neighborhood information, facilitating the generation of discriminative representations.

Experiments

Experimental Setup

Datasets. In our experiments, we adopt six widely-used datasets, including *PubMed* (Yang, Cohen, and Salakhutdinov 2016), *Amazon-Photo*, *Amazon-Computers*, *Coauthor-CS* (Shchur et al. 2018), *Coauthor-Physics* and *Wiki-CS* (Mernyei and Cangea 2020). The statistics of the datasets are shown in the Table 2. A detailed introduction to these datasets is in the Appendix.

Baselines. We compare E2Neg with three types of baseline methods, including **(1)** Classical unsupervised algorithms: DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) and Node2Vec (Grover and Leskovec 2016). **(2)** Semi-supervised baselines GCN (Kipf and Welling 2017). **(3)** GCL baselines: BGRL (Thakoor et al. 2021), MVGRL (Hassani and Ahmadi 2020), DGI (Velickovic et al. 2019), GBT (Bielak, Kajdanowicz, and Chawla 2022), GRACE (Zhu et al. 2020), GCA (Zhu et al. 2021b), ProGCL (Xia et al. 2022). A detailed introduction to these methods can be found in the Appendix.

Implementation Details. We test E2Neg on the node classification task. We use a single-layer GCN as the encoder and a two-layer MLP as the projector. The settings for all of the hyperparameters can be found in the Appendix. During testing, we use the encoder to generate representations for downstream tasks. We use 10% of the data for training the downstream classifier and the remaining 90% for testing, and we follow the testing methods from the PyGCL (Zhu et al. 2021a). All methods are implemented using PyTorch Geometric framework (Fey and Lenssen 2019), and all experiments are conducted on an RTX 3090 GPU with 24GB

Method	Data	PubMed	CS	Photo	Computers	Wiki-CS	Physics
Raw Features	X	84.80	90.37	78.53	73.81	71.98	93.58
Node2Vec	A	80.30	85.08	89.67	84.39	71.79	91.19
DeepWalk	A	80.50	84.61	89.44	85.68	74.35	91.77
DeepWalk + Features	X, A	83.70	87.70	90.05	86.28	77.21	94.90
BGRL	X, A	85.88 ± 0.83	92.16 ± 0.35	92.26 ± 0.92	87.51 ± 0.74	79.49 ± 0.80	95.04 ± 0.26
MVGRL	X, A	85.61 ± 1.10	92.10 ± 0.44	92.68 ± 0.75	86.66 ± 0.91	80.34 ± 1.01	95.22 ± 0.35
DGI	X, A	85.92 ± 0.53	93.12 ± 0.50	92.75 ± 0.82	88.35 ± 0.68	80.16 ± 0.91	95.77 ± 0.43
GBT	X, A	86.37 ± 1.03	93.18 ± 0.50	92.39 ± 0.86	**88.91** ± 0.98	80.31 ± 1.30	95.77 ± 0.24
GRACE	X, A	85.80 ± 0.58	92.70 ± 0.74	92.01 ± 0.85	88.17 ± 0.91	79.94 ± 0.68	OOM
GCA	X, A	86.44 ± 0.19	92.41 ± 0.08	91.15 ± 0.18	86.58 ± 0.32	79.87 ± 0.44	OOM
ProGCL	X, A	86.46 ± 0.54	92.30 ± 0.55	92.72 ± 1.03	87.65 ± 0.87	77.85 ± 1.06	OOM
E2Neg	X, A	**86.72** ± 1.09	**93.48** ± 0.59	**93.36** ± 0.76	88.72 ± 0.96	**80.89** ± 1.21	**95.86** ± 0.30
Supervised-GCN	X, A, Y	84.80	93.03	92.42	86.51	77.19	95.65

Table 1: Performance on node classification. All results highlight the best with **bold** and the runner-up with an underline. X, A, Y denote the node attributes, adjacency matrix, and labels in the datasets, OOM signifies out-of-memory on RTX 3090 GPU with 24GB of memory. Data without variance are drawn from (Zhu et al. 2020, 2021b).

Dataset	*#Nodes*	*#Edges*	*#Features*	*#Classes*
PubMed	19,717	88,651	500	3
CS	18,333	163,788	6,805	15
Photo	7,650	238,163	745	8
Computers	13,752	491,722	767	10
Wiki-CS	11,701	431,726	300	10
Physics	34,493	991,848	8,451	5

Table 2: Statistics of datasets used in experiments.

of memory and a Core i5-12400 CPU. More implementation details can be found in the Appendix.

Experimental Results

Node Classification. In Table 1, we present the accuracy of E2Neg and other baseline models for the node classification task. Our proposed E2Neg regularly outperforms the baseline GRACE and other sampling methods such as GCA and ProGCL across all datasets, indicating our sampling strategy, in contrast to complete sampling, successfully mitigates the adverse effects of sample number on model performance and facilitates downstream classification more efficiently. Although BGRL does not rely on negative sampling, topological coupling still persists and impacts the performance. This is because the model operates on the original graph, where nodes inherently exhibit topological coupling. E2Neg does not directly resolve topological coupling, mitigating its effects by sampling within subgraphs. This further confirms our findings that choosing a small, representative subset of nodes as samples is enough to train the model.

Computational Complexity Analysis. E2Neg demonstrates significantly higher efficiency in both time and memory consumption compared to other self-supervised baselines, as shown in Table 3. For a fair comparison, we set the number of hidden dimension to 256 for all methods, with other parameters consistent with those in Table 1. Its strategy of selecting representative nodes for training reduces memory utilization by over 90% on average compared to sampling-based baselines like GCA and ProGCL, which have high memory overhead. In terms of training time, E2Neg achieve remarkable improvement, with average speedups ranging from tens to hundreds of times. On the larger dataset, *Physics*, E2Neg is over 20 times faster than all other methods, and it consistently achieves speedup factors exceeding 100 times across all datasets. Moreover, it is evident that traditional sampling methods, such as GRACE, GCA, and ProGCL, consume a significant amount of memory and time across these six datasets. Additionally, for other types of methods like BGRL and DGI, the traditional sampling approaches do not offer competitive efficiency, whereas our E2Neg significantly outperforms these methods in terms of efficiency. The significant improvement in time and memory efficiency of E2Neg can be attributed to its substantial reduction in the sample, which prevents the model from generating a large number of negatives for similarity calculations during training.

Model Analysis

Ablation Study. To further validate the correctness of our theory and the effectiveness of E2Neg. In this section, we substitute several components of E2Neg to analyze the impact of each component. In our study, we modify the sampling strategy by using random and full sampling. Additionally, we remove the augmentation from E2Neg. The performance is presented in Table 4. E2Neg outperforms all the different experimental variations. For the sampling strategy, E2Neg selects representative nodes that are more effective in capturing the core semantics of the graph. Compared to random sampling, these representative nodes significantly enhance the model's ability to distinguish negative samples across different semantic blocks, leading to improved discriminative performance. E2Neg provides a more representative sample than selecting all nodes, which aligns with Finding 1 from our theoretical analysis. The augmentation

Dataset		BGRL	MVGRL	DGI	GBT	GRACE	GCA	ProGCL	**E2Neg**	Improvement
PubMed	Mem	7,166	22,466	970	1,432	11,610	11,180	23,054	404	**58.4-98.2%**
	Time	0.0427	0.5412	0.0130	0.0261	0.2177	0.1598	0.6421	0.0028	**4.6-229.3**×
CS	Mem	7,770	OOM	2,064	3,324	11,960	12,660	22,896	900	**56.4-96.1%**
	Time	0.0637	/	0.0281	0.0597	0.2189	0.1603	0.5727	0.0030	**9.4-190.9**×
Photo	Mem	2,322	14,690	1,154	2,156	3,694	2,478	5,304	388	**66.4-97.4%**
	Time	0.0284	0.4125	0.0185	0.0273	0.0548	0.0361	0.1141	0.0028	**6.6-147.3**×
Computers	Mem	5,722	OOM	1,914	2,570	8,952	6,548	15,506	406	**78.8-97.4%**
	Time	0.0566	/	0.0354	0.0496	0.1382	0.0964	0.3321	0.0028	**12.6-118.6**×
Wiki-CS	Mem	3,928	17,964	1,696	3,074	5,698	4,898	11,402	380	**77.6-97.9%**
	Time	0.0465	0.5139	0.0300	0.0418	0.1073	0.0731	0.2470	0.0028	**10.7-183.5**×
Physics	Mem	23,688	OOM	4,218	8,002	OOM	OOM	OOM	1,560	**63.0-93.4%**
	Time	0.1635	/	0.0675	0.1319	/	/	/	0.0031	**21.8-52.7**×

Table 3: Comparison of training time per epoch in seconds and memory usage in MBs on six datasets. Improvement means how many times E2Neg is faster than baselines or the percentage of memory savings that E2Neg provides compared to baselines. OOM signifies out-of-memory on 24GB RTX 3090. '-' means the improvement range.

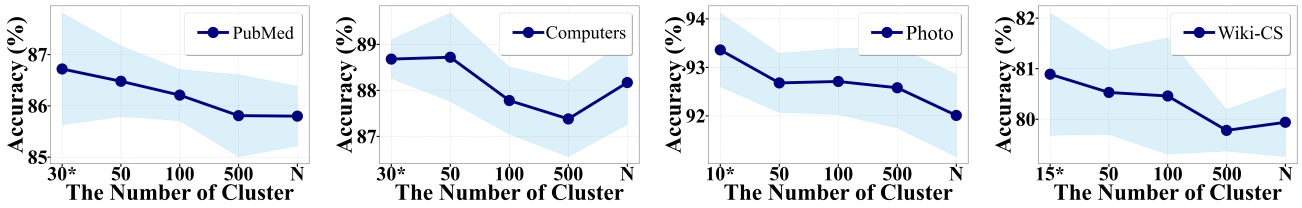

Figure 2: Hyperparameter Analysis, where * denotes the parameter used by E2Neg, and N represents the number of nodes in each dataset.

Method	Random	Full	No Aug	E2Neg
CS	92.87 ± 0.47	92.70 ± 0.74	93.12 ± 0.59	**93.48** ± 0.59
Photo	92.42 ± 1.31	92.01 ± 0.85	92.54 ± 0.96	**93.36** ± 0.76
Computers	87.83 ± 0.70	88.17 ± 0.91	88.18 ± 0.74	**88.72** ± 0.96
Wiki-CS	79.80 ± 1.22	79.94 ± 0.68	79.45 ± 1.12	**80.89** ± 1.21
Physics	94.97 ± 0.32	OOM	94.53 ± 0.45	**95.86** ± 0.30

Table 4: Ablation study with several components of E2Neg. All results highlight the best with bold.

enables individual samples to acquire the semantic information available in various subgraphs, improving the discriminative capability of the representations.

Hyperparameter Analysis. To further explore the correctness and effectiveness of the E2Neg sampling and to substantiate the analysis in our theoretical analysis, we test the impact of the number of clusters on the model's performance, as shown in Figure 2. It can be observed that when the cluster size is large, the performance of E2Neg decreases across various datasets. This is because the number of clusters far exceeds the actual number of semantic blocks in the graph, which declines the model's ability to distinguish between negatives from different semantic blocks. Conversely, keeping the cluster size small allows the number of clusters to align with the semantic blocks, leading to an improvement in model performance. This is consistent with our findings in the theoretical analysis section, where we observe that selecting a representative subset of nodes, significantly enhances the model's performance. The experimental results further validate the correctness of our theoretical analysis and demonstrate the effectiveness of E2Neg.

Conclusion

In this paper, we explore the role of negative samples in GCL. Specifically, we challenge the consensus of selecting all instances as negative samples and theoretically discover that a large number of negatives can hinder the model's ability to distinguish nodes with different semantics. Furthermore, using only a very small subset of nodes as negatives is sufficient to enhance the discriminative power of the representations. Based on these findings, we propose E2Neg, which trains using only a very small set of representative negatives. E2Neg reduces computational overhead and significantly accelerates model training. Finally, we conduct extensive experiments on multiple datasets. The experimental results demonstrate the effectiveness of E2Neg.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. U22B2036, No. 62422210 , No. 62276187, No. 92370111 and No. 62272340), the National Science Fund for Distinguished Young Scholarship (No. 62025602), and the XPLORER PRIZE.

References

Bielak, P.; Kajdanowicz, T.; and Chawla, N. V. 2022. Graph Barlow Twins: A self-supervised representation learning framework for graphs. *Knowl. Based Syst.*, 256: 109631.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. E. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of the 37th International Conference on Machine Learning*, 1597–1607.

Fan, W.; Ma, Y.; Li, Q.; He, Y.; Zhao, Y. E.; Tang, J.; and Yin, D. 2019. Graph Neural Networks for Social Recommendation. In *The World Wide Web Conference*, 417–426.

Fey, M.; and Lenssen, J. E. 2019. Fast Graph Representation Learning with PyTorch Geometric. *CoRR*, abs/1903.02428.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855–864.

Hamilton, W. L.; Ying, Z.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30*, 1024–1034.

Hassani, K.; and Ahmadi, A. H. K. 2020. Contrastive Multi-View Representation Learning on Graphs. In *Proceedings of the 37th International Conference on Machine Learning*, 4116–4126.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. B. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9726–9735.

Jin, D.; Wang, X.; He, D.; Dang, J.; and Zhang, W. 2021. Robust Detection of Link Communities With Summary Description in Social Networks. *IEEE Trans. Knowl. Data Eng.*, 33(6): 2737–2749.

Jin, D.; Yu, Z.; Jiao, P.; Pan, S.; He, D.; Wu, J.; Yu, P. S.; and Zhang, W. 2023. A Survey of Community Detection Approaches: From Statistical Modeling to Deep Learning. *IEEE Trans. Knowl. Data Eng.*, 35(2): 1149–1170.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations*.

Li, W.; Wang, C.; Xiong, H.; and Lai, J. 2023. HomoGCL: Rethinking Homophily in Graph Contrastive Learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1341–1352.

Liu, M.; Lin, Y.; Liu, J.; Liu, B.; Zheng, Q.; and Dong, J. S. 2023. B^2-Sampling: Fusing Balanced and Biased Sampling for Graph Contrastive Learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1489–1500.

MacQueen, J. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*.

Mernyei, P.; and Cangea, C. 2020. Wiki-CS: A Wikipedia-Based Benchmark for Graph Neural Networks. *CoRR*, abs/2007.02901.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. DeepWalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 701–710.

Shchur, O.; Mumme, M.; Bojchevski, A.; and Günnemann, S. 2018. Pitfalls of Graph Neural Network Evaluation. *CoRR*, abs/1811.05868.

Thakoor, S.; Tallec, C.; Azar, M. G.; Munos, R.; Veličković, P.; and Valko, M. 2021. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*.

van den Oord, A.; Li, Y.; and Vinyals, O. 2018. Representation Learning with Contrastive Predictive Coding. *CoRR*, abs/1807.03748.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. *International Conference on Learning Representations*. Accepted as poster.

Velickovic, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. In *7th International Conference on Learning Representations*.

Wang, X.; Dong, Y.; Jin, D.; Li, Y.; Wang, L.; and Dang, J. 2023. Augmenting Affective Dependency Graph via Iterative Incongruity Graph Learning for Sarcasm Detection. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, February 7-14, 2023*, 4702–4710. AAAI Press.

Xia, J.; Wu, L.; Wang, G.; Chen, J.; and Li, S. Z. 2022. ProGCL: Rethinking Hard Negative Mining in Graph Contrastive Learning. In *International Conference on Machine Learning*, 24332–24346.

Yang, Z.; Cohen, W. W.; and Salakhutdinov, R. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *Proceedings of the 33nd International Conference on Machine Learning*.

Zhang, H.; Wu, Q.; Wang, Y.; Zhang, S.; Yan, J.; and Yu, P. S. 2022. Localized Contrastive Learning on Graphs. *CoRR*, abs/2212.04604.

Zhu, Y.; Xu, Y.; Liu, Q.; and Wu, S. 2021a. An Empirical Study of Graph Contrastive Learning. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, December 2021, virtual*.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2020. Deep Graph Contrastive Representation Learning. *CoRR*, abs/2006.04131.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2021b. Graph Contrastive Learning with Adaptive Augmentation. In *WWW '21: The Web Conference*, 2069–2080.