Structural and Semantic Contrastive Learning for Unsupervised Node Representation Learning

Kaize Ding, Yancheng Wang, Yingzhen Yang, Huan Liu
Arizona State University, School of Computing and Augmented Intelligence
kaize.ding@asu.edu, yancheng.wang@asu.edu, yingzhen.yang, huan.liu@asu.edu

Abstract

Graph Contrastive Learning (GCL) recently has drawn much research interest for learning generalizable node representations in a self-supervised manner. In general, the contrastive learning process in GCL is performed on top of the representations learned by a graph neural network (GNN) backbone, which transforms and propagates the node contextual information based on its local neighborhoods. However, nodes sharing similar characteristics may not always be geographically close, which poses a great challenge for unsupervised GCL efforts since they have inherent limitations in capturing such global graph knowledge. In this work, we go beyond the existing unsupervised GCL counterparts and address their limitations by proposing a simple yet effective framework – Simple Neural Networks with Structural and Semantic Contrastive Learning (S3-CL). Notably, by virtue of the proposed structural and semantic contrastive learning algorithms, even a simple neural network is able to learn expressive node representations that preserve valuable global structural and semantic patterns. Our experiments demonstrate that the node representations learned by S3-CL achieve superior performance on different downstream tasks compared to the state-of-the-art GCL methods.

1 Introduction

Learning expressive node representations of graph-structured data plays an essential role in a variety of real-world applications, ranging from social network analysis [1], to drug discovery [2], to financial fraud detection [3]. Recently, graph neural networks (GNNs), which generally follow a recursive message-passing scheme, have emerged as powerful architectures in graph machine learning [1, 4, 5, 6, 7, 8]. Though GNNs are empirically effective to handle supervised or semi-supervised graph machine learning tasks, the labor-intensive and resource-expensive data labeling cost is meanwhile unbearable. To relieve the burdensome reliance on human-annotated labels, unsupervised (self-supervised) node representation learning with GNNs has drawn much research attention lately [9, 10, 11, 12].

Inspired by the success in the vision and language domains [13, 14, 15], contrastive learning recently has been contributed to the recent advances of GNNs to improve their performance in graph self-supervised learning [11, 12, 16, 17, 18]. Typically, graph contrastive learning (GCL) methods learn representations by creating two augmented views of each graph element and maximizing the agreement between the encoded representations of the two augmented views. Correspondingly, the relevant view pairs (positive) will be pulled together and the irrelevant view pairs (negative) will be pushed away in the latent space. With only non-semantic labels, unsupervised GCL is able to provide generalizable node representations for various downstream tasks [12, 16, 19], becoming a prevailing paradigm in unsupervised node representation learning.

*Indicates equal contribution.

Preprint. Under review.
Despite the success, the research of unsupervised GCL is still in its infancy – most of the existing GCL methods learn node representations based on the information from the local neighborhoods due to the shallow property of conventional GNNs. While for real-world graphs, \textit{nodes sharing similar characteristics may not always be geographically close}, requiring the learning algorithm to retain such “global” awareness. However, it is a non-trivial task for the existing GCL methods built on top of shallow GNNs since they have inherent limitations in capturing either \textit{structural global knowledge} or \textit{semantic global knowledge}. Specifically: (i) from the structural perspective, long-range node interactions are highly desired for capturing structural global knowledge, especially for many downstream tasks that have large problem radii [20]. To this end, a straightforward way is to employ a deeper GNN encoder to encode the augmented graphs. However, directly stacking multiple GNN layers will not only lead to information distortion caused by the oversmoothing issue [21], but also introduce additional training parameters that hamper the model training efficiency; and (ii) from the semantic perspective, existing unsupervised GCL methods predominately focus on instance-level contrast that leads to a latent space where all nodes are well-separated and each node is locally smooth [22] (i.e., input with different augmentations have similar representations), while the underlying semantic structure (i.e., intra-cluster compactness and inter-cluster separability) of the input graph is largely ignored [22]. The lack of prior knowledge of ground-truth labels (e.g., cluster/class numbers) leaves a significant gap for unsupervised GCL to consolidate the semantic structure from a global view in the latent space. Yet, how to bridge this gap remains unattended.

In this paper, we address the aforementioned limitations by proposing a simple yet effective GCL framework, namely, S$^3$-CL (Simple Neural Networks with Structural and Semantic Contrastive Learning). The proposed two new contrastive learning algorithms enable the framework to outperform other GCL counterparts with a much simpler and parameter-less encoding backbone, such as an MLP or even a one-layer neural network. To capture long-range node interactions without oversmoothing, the \textit{structural contrastive learning} algorithm first generate multiple augmented views of the input graph based on different feature propagation scales (i.e., multi-scale feature propagation). Then by performing contrastive learning on the node representations learned from the local and multiple high-order views, the encoder network is able to improve node-wise discrimination by exploiting the consistency between the local and global structure information of each node. In the meantime, the \textit{semantic contrastive learning} algorithm further enhances intra-cluster compactness and inter-cluster separability to better consolidate the semantic structure from a global view. Specifically, it infers the clusters among nodes and their corresponding prototypes by a new Bayesian non-parametric algorithm, and then performs semantic contrastive learning to enforce those nodes that are semantically similar to cluster around their corresponding cluster prototypes in the latent space. By jointly optimizing the structural and semantic contrastive losses, the pre-trained encoder network can learn highly expressive node representations for various downstream tasks without using any human-annotated labels. We summarize our contributions as follows:

- We develop a new GCL framework S$^3$-CL, which can learn expressive node representations in a self-supervised fashion by using a simple and parameter-less encoding backbone.
- We propose the structural and semantic contrastive learning algorithms, which can be used for explicitly capturing the global structural and semantic patterns of the input graph.
- We conduct extensive experiments to show that our approach significantly outperforms the state-of-the-art GCL counterparts on various downstream tasks.

2 Preliminaries

We start by introducing the notations used throughout the paper. An attributed graph with $N$ nodes can be formally represented by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denote the set of nodes and edges respectively. Let $A \in \{0, 1\}^{N \times N}$ be the adjacency matrix of graph $\mathcal{G}$. $A_{ij} = 1$ if and only if $(v_i, v_j) \in \mathcal{E}$. $\tilde{A}$ stands for the adjacency matrix for a graph with added self-loops $I$. We let $D$ and $\tilde{D}$ denote the diagonal degree matrix of $A$ and $\tilde{A}$ respectively. $x_i$ is the $i$-th row of $X$ the attribute matrix $X \in \mathbb{R}^{N \times D}$, which denotes the feature of node $v_i$. Hence, an attributed graph can be described as $\mathcal{G} = (X, A)$ for simplicity.

\textbf{Graph Contrastive Learning.} In general, graph contrastive learning aims to pre-train a graph encoder that can maximize the node-wise agreement between two augmented views of the same graph element in the latent space via a contrastive loss. Generally, given an attributed graph $\mathcal{G} =$
(X, A), two different augmented views of the graph, denoted as $G^{(1)} = (X^{(1)}, A^{(1)})$ and $G^{(2)} = (X^{(2)}, A^{(2)})$, are generated through the data augmentation function(s). The node representations on $G^1$ and $G^2$ are denoted as $H^{(1)} = f_\theta(X^{(1)}, A^{(1)})$ and $H^{(2)} = f_\theta(X^{(2)}, A^{(2)})$, where $f_\theta(\cdot)$ is an encoder network. The agreement between the node representations is commonly measured through Mutual Information (MI). Thus, the contrastive objective can be generally formulated as:

$$
\max_\theta \sum_{i=1}^{N} \mathcal{M}(h^{(1)}_i, h^{(2)}_i).
$$

Following this formulation, Deep Graph Infomax (DGI) [11] is the first method that contrasts the patch representations with high-level graph representations by maximizing their mutual information. InfoGraph [23] obtains representations for whole graphs by contrasting the graph-level representations with the substructure-level representations. MVGRL [16] adopts graph diffusion to generate an augmented view, and contrast representations of first order neighbors with a graph diffusion. GCC [17] and GRACE [24] create the augmented views by sampling subgraphs. MERIT [25] adopts a siamese self-distillation network, and performs contrastive learning across views and networks at the same time. Nonetheless, existing unsupervised GCL methods only focus on short-range node interactions and are also ineffective to capture semantic structure of graphs.

3 Methodology

In this paper, we propose a novel graph contrastive learning framework S$^3$-CL for unsupervised/self-supervised node representation learning. The overall framework is illustrated in Figure 1. Our proposed framework consists of three main components: (i) a simple (e.g., 1-layer) encoder network; (ii) a structural contrastive learning algorithm; and (iii) a semantic contrastive learning algorithm.

3.1 Structural Contrastive Learning

Existing GCL methods for unsupervised node representation learning aim to achieve node-wise discrimination by maximizing the agreement between the representations of the same graph element in different augmented views. Despite their success, they commonly ignore the global structure knowledge due to the limitations of either the adopted data augmentation function or encoding GNN backbone. In this work, we propose the structural contrastive learning algorithm, which enables a simple neural network to capture both local and global structural knowledge by performing contrastive learning on multi-scale augmented views of the input graph.

Multi-scale Feature Propagation. In order to capture long-range node interactions without suffering the oversmoothing issue, in our structural contrastive learning algorithm, we propose to adopt multi-scale feature propagation to augment the input graph from the structural perspective. Compared to arbitrarily modifying the graph structure such as perturbing edges or nodes, feature propagation not only allows incorporating long-range node interactions, but also mitigates the noises in the original graph. Different from existing GCL algorithms that perform only two augmentations for each instance, we perform feature propagation with different scales to generate $L$ augmented feature matrices $\{X^{(i)}_L\}_{i=1}^L$, each of which encodes the $i$-hop node interactions in the graph. Then each augmented feature matrix $X^{(i)}$ can be encoded by an encoder network $f_\theta(\cdot)$ and the corresponding node representations can be computed by:

$$
H^{(i)} = f_\theta(X^{(i)}) = \text{ReLU}(\tilde{X}^{(i)} \Theta), \quad \tilde{X}^{(i)} = T^i X,
$$

where $T \in \mathbb{R}^{N \times N}$ is a generalized transition matrix and we take $T = \tilde{A}_{\text{sym}} = \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2}$ in this work. Specifically, $H^{(1)}$ is learned from a local view as the message-passing is only enabled between direct neighbors, while $\{H^{(i)}\}_{i=2}^L$ are learned from a set of high-order views that encode the long-range node interactions at different scales.

It is noteworthy that our model inherently separates the feature propagation step, i.e., $\tilde{X}^{(i)} = T^i X$, and transformation step, i.e., $f_\theta(X^{(i)})$ into the data augmentation and representation learning modules, respectively. Compared to standard GNNs that couples the two steps together in each layer, this decoupling strategy allows the model to perform the high-order feature propagation without conducting non-linear transformations, reducing the risk of over-smoothing [26, 27]. In the
meanwhile, we can use a much simpler encoding backbone to transform the augmented features to node representations without stacking multiple GNN layers.

**Structural Contrastive Objective.** In our structural contrastive learning, we aim to maximize the agreement between the representations of each node learned from the local view and its different high-order views by maximizing their mutual information. Instead of directly contrasting the output of the encoder network, we follow previous research in contrastive learning [14] and apply a projection head \( g_\psi(\cdot) \) to the node representations computed by the encoder network. As such, the representations we contrast in our structural contrastive learning can be denoted by \( \{U^{(i)}\}_{i=2}^{L} \), where \( U^{(i)} = g_\psi(H^{(i)}) \), and \( g_\psi(\cdot) \) is a two-layer MLP in our implementation.

In our work, we adopt InfoNCE [28] to estimate the lower bound of the mutual information between the node representations learned from a local view \( U^{(1)} \) and different high-order views \( \{U^{(i)}\}_{i=2}^{L} \) of the input graph. The loss function of structural contrastive loss can be defined as:

\[
\mathcal{L}_{str} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{l=2}^{L} \log \frac{\exp(\langle u_i^{(1)} \cdot u_i^{(l)} \rangle / \tau_1)}{\sum_{j=1}^{M+L-1} \exp(\langle u_i^{(1)} \cdot u_j^{(l)} \rangle / \tau_1)},
\]

where \( \tau_1 \) is the temperature parameter. Note that \( \{u_i^{(j)}\}_{j=1}^{M+L-1} \) contains \( L - 1 \) positive examples and \( M \) negative examples sampled from the augmented views of other nodes.

By performing the proposed structural contrastive learning based on multi-scale augmentations of the input graph, the encoder network \( f_\theta(\cdot) \) not only encourages accurate node-wise discrimination but also captures multi-scale global structural knowledge during the learning process. The final node representations \( H \) can be computed by feeding the mixed-order propagated features \( \bar{X} = \frac{1}{L} \sum_{l=1}^{L} T^l X \) to the encoder network. This enables the learned node representations to preserve both local and global structure information compared with directly using \( S^{(L)} X [29, 26] \).

### 3.2 Semantic Contrastive Learning

Despite the structural contrastive learning algorithm can provide better node-wise discrimination by exploiting the global structural knowledge based on the multi-scale propagated features, it has the same limitation with existing GCL efforts that cannot explicitly encode the semantic structure of the input graph. To further capture the semantic global knowledge, we propose a semantic contrastive learning algorithm that implicitly encodes the semantic structure of data into the embedding space.

Since the prior knowledge of node clusters is unknown, we propose to iteratively infers the clusters among nodes and the corresponding prototypes in based on the node representations, and performs semantic contrastive learning to promote those nodes that are semantically similar clustering around their corresponding cluster prototypes.
Let denote the cluster prototype representation by a matrix $C \in \mathbb{R}^{K \times D'}$, where $K$ is the number of prototypes inferred from the data. We use $c_k$ to denote the $k$-th row of $C$, which is the representation of the $k$-th prototype in the latent space. The prototype assignments or pseudo labels$^2$ of nodes are denoted by $Z = \{z_i\}_{i=1}^N$, where $z_i \in \{1, \ldots, K\}$ is the pseudo label of node $v_i$.

**Bayesian Non-parametric Prototype Inference.** A key function of our semantic contrastive learning algorithm is to infer highly representative cluster prototypes. However, the optimal number of cluster is unknown under the setting of unsupervised node representation learning. To bridge the gap, we propose a Bayesian non-parametric prototype inference algorithm to approximate the optimal number of clusters and simultaneously compute the cluster prototypes. Specifically, we build a Dirichlet Process Mixture Model (DPMM) and assume the distribution of node representations is a mixture of Gaussians, in which each component is used to model the prototype of a cluster. Note that the components share the same fixed covariance matrix $\sigma I$. The DPMM model is defined as:

$$ G \sim \text{DP}(G_0, \alpha), \phi_i \sim \mathcal{N}(\phi_i, \sigma I), \quad i = 1, \ldots, N, $$

where $G$ is a Gaussian distribution drawn from the Dirichlet process DP$(G_0, \alpha)$, and $\alpha$ is the concentration parameter for DP$(G_0, \alpha)$. $\phi_i$ is the mean of the Gaussian sampled for node representation $h_i$. $G_0$ is the prior over means of the Gaussians. We take $G_0$ to be a zero-mean Gaussian $\mathcal{N}(0, \rho I)$, where $\rho I$ is the covariance matrix.

Next, we use a collapsed Gibbs sampler [30] to infer the Gaussian components. The Gibbs sampler iteratively samples pseudo labels for the nodes given the means of the Gaussian components, and samples the means of the Gaussian components given the pseudo labels of the nodes. Following [31], such process is almost equivalent to K-Means when the variance of the Gaussian components $\sigma \to 0$. The almost zero variance eliminates the needs of estimating the variance $\sigma$, thus makes the inference efficient. Let $\hat{K}$ denote the number of inferred prototypes at current iteration step, the prototype assignment update can be formulated as:

$$ z_i = \arg \min_k \{d_{ik}\}, \quad \text{for } i = 1, \ldots, N, \quad d_{ik} = \left\{ \begin{array}{ll} ||h_i - c_k||^2 & \text{for } k = 1, \ldots, \hat{K} \\ \xi & \text{for } k = \hat{K} + 1, \end{array} \right. $$

where $d_{ik}$ is the distance to determine the pseudo labels of node representation $h_i$. $\xi$ is the margin to initialize a new prototype. In practice, we choose the value of $\xi$ by performing cross-validation on each dataset. With the formulation in Equation (5), a node will be assigned to the prototype modeled by the Gaussian component corresponding to the closest mean of Gaussian, unless the squared Euclidean distance to the closest mean is greater than $\xi$. After obtaining the pseudo labels, the cluster prototype representations can be computed by: $c_k = \sum_{z_i = k} h_i / \sum_{z_i = k} 1$, for $k = 1, \ldots, \hat{K}$.

Note that we iteratively update prototype assignments and prototype representations till convergence, and we set the number of prototypes $K$ to be the number of inferred prototypes $\hat{K}$ afterwards. The algorithm for prototype inference is summarized in Supplementary B.1. Afterwards, we refine the cluster prototypes using label propagation and the we attach the details in Supplementary B.2 due to the space limit.

**Semantic Contrastive Objective.** After obtaining the prototype assignments $Z$ and prototype representations $C$, our semantic contrastive objective aims to consolidate the semantic structure (i.e., intra-cluster compactness and intra-cluster separability) of the learned node representation $H$ by updating the encoder parameter $\theta$. To this end, we maximize the likelihood of the nodes in the graph given $C$ and $\theta$:

$$ Q(\theta) = \sum_{i=1}^N \log p(x_i | \theta, C) = \sum_{i=1}^N \sum_{k=1}^K \log p(x_i, z_i = k | \theta, C), $$

where $p$ is the probability density function. Directly optimizing log-likelihood $Q(\theta)$ is intractable as the labels of nodes are unknown. Instead, we optimize the variational lower bound of $Q(\theta)$ given by:

$$ Q(\theta) \geq \sum_{i=1}^N \sum_{k=1}^K q(k|x_i) \log \frac{p(x_i, z_i = k | \theta, C)}{q(k|x_i)} $$

$$ = \sum_{i=1}^N \sum_{k=1}^K q(k|x_i) \log p(x_i, z_i = k | \theta, C) - \sum_{i=1}^N \sum_{k=1}^K q(k|x_i) \log q(k|x_i). $$

$^2$We use those two terms in this paper interchangeably.
Algorithm 1: The learning algorithm of $S^3$-CL.

**Input:** Attribute matrix $X$; adjacency matrix $A$; propagation step $L$  
**Output:** Pretrained encoder network $f_\theta(\cdot)$

1. Initialize encoder parameter $\theta$ and $\theta'$
2. **while not converge**
   1. Compute node representations of different augmented views $\{H^{(i)}\}_{i=1}^L$ and $\{U^{(i)}\}_{i=1}^L$
   2. Compute the prototype representations $C$ and prototype assignments $Z$ $\triangleright$ E-Update
   3. Calculate loss $L_{str}$ and $L_{sem}$ by Equation (3) and Equation (10), respectively
   4. $L = \gamma L_{str} + (1 - \gamma) L_{sem}$
   5. Update $\theta$ by minimizing $L$ $\triangleright$ M-Step Update
   6. Update momentum encoder $\theta'$
3. **return** the encoder network $f_\theta(\cdot)$

where $q(k|x_i) = p(z_i = k | x_i, \theta, C)$ denotes the posterior of $z_i$. Note that $q(k|x_i) = \mathbb{1}_{z_i=k}$, and the second term above is a constant. Thus, we can simplify the above inequation as:

$$Q(\theta) \geq \sum_{i=1}^{N} \log p(x_j, z_i | \theta, C).$$  \hspace{1cm} (8)

Under the assumption of a uniform prior distribution of $x_i$ over different prototypes, we have $p(x_i, z_i | \theta, C) \propto p(x_i | z_i, \theta, C)$. Since the distribution of node representation around each prototype generated by DPMM is an isotropic Gaussian, after applying $\ell_2$ normalization on the representation of nodes and prototypes, we can estimate $p(x_i | z_i, \theta, C)$ by:

$$p(x_i | z_i, \theta, C) \propto \frac{\exp(h_i \cdot c_z / \tau_2)}{\sum_{k=1}^{K} \exp(h_i \cdot c_k / \tau_2)},$$  \hspace{1cm} (9)

where $c_z$ is the representations of $z_{i}$-th prototype. The temperature parameter $\tau_2 \propto \sigma^2$ is related to the concentration of node representation around each prototype, and $\sigma$ is the variance of the Gaussians in the DPMM model defined by Equation (4). For the simplicity of training, we directly take $\tau_2$ as a hyperparameter. Taking Equation (9) into Equation (8), we can maximize $Q(\theta)$ by minimizing the following loss:

$$L_{sem} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(h_i \cdot c_z / \tau_2)}{\sum_{k=1}^{K} \exp(h_i \cdot c_k / \tau_2)}.$$  \hspace{1cm} (10)

### 3.3 Model Learning

Given the proposed $S^3$-CL learning framework, our goal is to learn expressive node representations that preserve both valuable structural and semantic knowledge without any semantic labels. In this section, we will introduce the overall loss function, and also the optimization of the proposed framework with regard to the network parameters, prototype assignments and prototype representations.

**Overall Loss.** To train our model in an end-to-end fashion and learn the encoder $f_\theta(\cdot)$, we jointly optimize both the structural and semantic contrastive learning losses. The overall objective function is defined as:

$$L = \gamma L_{str} + (1 - \gamma) L_{sem},$$  \hspace{1cm} (11)

where we aim to minimize $L$ during training, and $\gamma$ is a balancing parameter to control the contribution of each contrastive learning loss. For the sake of the stability of training of the encoder network, we apply our Bayesian non-parametric prototype inference algorithm on the node representations computed by a momentum encoder [13]. The parameters of the momentum encoder $\theta'$ is updated by the moving average of $\theta$ by $\theta' = (1 - m) \cdot \theta + m \cdot \theta'$, where $m \in [0, 1)$ is the momentum coefficient.

Notably, in the semantic contrastive learning, the computed pseudo labels $Z$ can be utilized in the negative example sampling process in our structural contrastive learning to avoid sampling bias issues [32]. We select negative samples in Equation (3) for each node from nodes assigned to different prototypes. Detailed analysis on negative example sampling can be found in Supplementary C.2.

**Model Optimization via EM.** We adopt EM algorithm to alternately estimate the posterior distribution $p(z_i | x_i, \theta, C)$ and optimize the network parameters $\theta$. We describe the details for the E-step and M-step applied in our methods as follows:
Algorithm 1 outlines the learning process of the proposed framework. After the self-supervised pre-training on the unlabeled input graph, the pre-trained encoder can be directly used to generate node representations for various downstream tasks.

4 Experiments

4.1 Experimental Settings

**Evaluation Datasets.** We evaluate $S^3$-CL on six public benchmark datasets that are widely used for node representation learning, including Cora [33], Citeseer [33], Pubmed [34], Amazon-P [35], Coauthor CS [35] and ogbn-arxiv [36]. Cora, Citeseer and Pubmed are three most widely used citation networks. Amazon-P is a co-purchase graph. Coauthor CS is co-authorship graph. The ogbn-arxiv is a directed citation graph. We summarize the statistics of the datasets in the Section A of the Supplementary. Among the six benchmarks we use, ogbn-arxiv is a large-scale open graph benchmark dataset, which is more challenging to deal with for unsupervised GCL.

**Compared Methods.** To demonstrate the effectiveness of our proposed method, six state-of-the-arts graph self-supervised learning methods are compared in our experiments, including DGI [11], MVGRL [16], GMI [37], GRACE [38], MERIT [25], and SUGRL [39]. As we consider node classification as our downstream task, we also include five representative supervised node classification methods, namely MLP [11], LP [40], GCN [1], GAT [4], and SGC [7], as baselines for the evaluation on node classification.

To evaluate the model performance for node clustering, we compare $S^3$-CL against methods including different K-Means [41], GAE [9], adversarially regularized GAE (ARGA) and VGAE (ARVGA) [10], GALA [42], DBGAN[43], MVGRL [16], MERIT [25], and SUGRL [39].

4.2 Evaluation Results

**Node Classification.** To evaluate the trained encoder network, we adopt a linear evaluation protocol by training a separate logistic regression classifier on top of the learned node representations. We follow the evaluation protocols in previous works [11, 35, 36] for node classification on different datasets. The mean classification accuracy with standard deviation on the test nodes after 10 runs of training is reported. To avoid out-of-memory issue when evaluating MVGRL, GRACE, and MERIT

| Methods  | Cora       | Citeseer  | Pubmed    | Amazon-P  | Coauthor CS | ogbn-arxiv       |
|----------|------------|-----------|-----------|-----------|-------------|-----------------|
| MLP      | 55.2 ± 0.4 | 46.5 ± 0.5| 71.4 ± 0.3| 78.5 ± 0.2| 76.5 ± 0.3  | 55.5 ± 0.2      |
| LP [40]  | 68.0 ± 0.5 | 45.3 ± 0.6| 73.0 ± 0.3| 75.4 ± 0.0| 74.3 ± 0.0  | 68.3 ± 0.0      |
| GCN [1]  | 81.7 ± 0.4 | 70.5 ± 0.3| 79.4 ± 0.4| 87.3 ± 1.0| 91.8 ± 0.1  | 71.7 ± 0.3      |
| GAT [4]  | 83.0 ± 0.7 | 72.5 ± 0.7| 79.0 ± 0.3| 86.2 ± 1.5| 90.5 ± 0.7  | **73.2 ± 0.2**  |
| SGC [7]  | 81.5 ± 0.2 | 73.1 ± 0.1| 79.7 ± 0.4| 88.3 ± 1.1| 91.5 ± 0.3  | 69.8 ± 0.2      |

**SUPERVISED**

| Methods  | Cora       | Citeseer  | Pubmed    | Amazon-P  | Coauthor CS | ogbn-arxiv       |
|----------|------------|-----------|-----------|-----------|-------------|-----------------|
| DGI [11] | 81.7 ± 0.6 | 71.5 ± 0.7| 77.3 ± 0.6| 83.1 ± 0.3| 90.0 ± 0.3  | 67.1 ± 0.4      |
| GMI [37] | 82.7 ± 0.2 | 73.0 ± 0.3| 80.1 ± 0.2| 85.1 ± 0.0| 91.0 ± 0.0  | 69.6 ± 0.3      |
| MVGRL [16]| 82.9 ± 0.7 | 72.6 ± 0.7| 79.4 ± 0.3| 87.3 ± 0.1| 91.3 ± 0.1  | 71.3 ± 0.2      |
| GRACE [38]| 80.0 ± 0.4 | 71.7 ± 0.6| 79.5 ± 1.1| 81.8 ± 0.8| 90.1 ± 0.8  | 71.1 ± 0.2      |
| MERIT [25] | 83.1 ± 0.6 | 74.0 ± 0.7| 80.1 ± 0.4| 88.8 ± 0.4| 92.4 ± 0.4  | 71.7 ± 0.1      |
| SUGRL [39]| 83.4 ± 0.5 | 73.0 ± 0.5| **81.9 ± 0.5** | 88.5 ± 0.2| 92.2 ± 0.5  | 69.3 ± 0.2      |
| $S^3$-CL (ours) | **84.5 ± 0.4** | **74.6 ± 0.4** | 80.8 ± 0.3 | **89.0 ± 0.5** | **93.1 ± 0.4** | **72.8 ± 0.3** |

**SELF-SUPERVISED + FINE-TUNING**

Table 1: Node classification performance comparison on benchmark datasets.
The node classification results of different methods are reported in Table 1. We can clearly see that $S^3$-CL outperforms the state-of-the-art self-supervised node representation learning methods across the five public benchmarks. Such superiority mainly stems from two factors: (i) our approach $S^3$-CL grants each node the access to information of nodes in a larger neighborhood; (ii) $S^3$-CL infers the semantic information of nodes, and enforces intra-cluster compactness and inter-cluster separability on the node representation. With the help of these extra information, node representation generated by $S^3$-CL are more informative and distinctive. Without the access to label, $S^3$-CL even outperforms supervised methods like SGC and GAT.

Node Clustering. To evaluate the quality of the node representations learned by different methods, we conduct experiments on node clustering. We follow the same evaluation protocol as in [16]. K-Means is applied on the learned node representation to get clustering results. We use accuracy (ACC), normalized mutual information (NMI), and adjusted rand index (ARI) to measure the performance of clustering. We report the averaged clustering results over 10 times of execution.

The clustering results are displayed in Table 2. Our approach achieves remarkable performance gain over compared methods. For example, the NMI on Cora is improved by 2.2% against previous SOTA method SUGRL. Such improvement greatly attributes to the fact that $S^3$-CL explores the semantic information of nodes instead of enforcing node-wise discrimination alone as previous GCL methods. Thus, the node representation learned by $S^3$-CL works well for simple clustering algorithms.

Node Classification with Few Labels. We further evaluate the impact of label rate on the downstream node classification task. Specifically, we evaluate all self-supervised learning methods from Table 1 under different low-resource settings. The results in Figure 2 show that our proposed framework $S^3$-CL can still outperform existing methods given a lower label rate. It validates that the node representations learned by our approach $S^3$-CL can encode valuable structural and semantic knowledge from the input graph. As a result, the node representations can be effectively used for the node classification task even with a extremely small label ratio.

![Figure 2: Node classification results with limited training labels.](image-url)
**Effect of Feature Propagation.** Next we investigate the effect of multi-scale feature propagation in the structural contrastive learning by altering the propagation steps $L$. A larger $L$ allows message-passing within a larger neighborhood for learning the node representations. To demonstrate the power of our approach in utilizing structural global knowledge, we compare $S^3$-CL against GCN, DGI, MVGRL, MERIT, and SUGRL with different number of layers $L$. The node classification results are shown in Figure 3. By increasing the propagation steps (number of layers), we can clearly observe that existing unsupervised GCL methods severely degrade as vanilla GCN due to the oversmoothing issue. In contrast, $S^3$-CL consistently achieves improved performance by making use of information in a larger neighborhood for node representation learning.

![Figure 3: Parameter analysis on the propagation steps ($L$) in multi-scale feature propagation.](image)

**Ablation Study.** To validate the effectiveness of the structural contrastive learning and semantic contrastive learning in $S^3$-CL, we conduct ablation study on Citesser, Cora, and Pubmed with two variants of $S^3$-CL, each of which has one of the contrastive learning components removed. The node classification results are shown in Table 3. We can observe that the performance of $S^3$-CL degrades when any of the components is removed. Our $S^3$-CL using all components achieves the best performance as the structural and semantic contrastive components are complementary to each other. Hence, the effectiveness of each component is verified.

![Table 3: Ablation study on contrastive components.](image)

**Representation Visualization.** To visually show the quality of the node representation learned by $S^3$-CL, we use t-SNE to visualize the node representations. For comparison, we also visualize the node representations generated by the best performing baseline on Citesser, i.e., MERIT. The visualization results are shown in Figure 4. Each dot represents the representation of a node, and the color of the dot denotes its ground-truth label. From the figure, we can observe that though some classes can be identified by MERIT, the boundaries between different classes are unclear.

![Figure 4: Node representation visualization comparison on the Citesser dataset.](image)

5 Conclusion

In this paper, we propose a new GCL framework named $S^3$-CL, which is able to effectively capture the global knowledge from both structural and semantic perspectives for unsupervised node representation learning. By jointly optimizing the structural and semantic contrastive learning losses, we are able to build the encoder network with simple neural networks to learn expressive node representations for different downstream tasks without using any human-annotated labels. We conduct extensive experiments and demonstrate that $S^3$-CL can outperform the state-of-the-art unsupervised GCL counterparts on multiple benchmark graph datasets.
References

[1] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In ICLR, 2017.
[2] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. In Advances in Neural Information Processing Systems, 2017.
[3] Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. Deep anomaly detection on attributed networks. In SDM, 2019.
[4] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In ICLR, 2018.
[5] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In NeurIPS, 2017.
[6] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In ICLR, 2019.
[7] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In ICML, 2019.
[8] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In ICML, 2020.
[9] Thomas N Kipf and Max Welling. Variational graph auto-encoders. arXiv preprint arXiv:1611.07308, 2016.
[10] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularizer graph autoencoder for graph embedding. In IJCAI, 2018.
[11] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In ICLR, 2019.
[12] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In NeurIPS, 2020.
[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In CVPR, 2020.
[14] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In ICML, 2020.
[15] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In EMNLP, 2021.
[16] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In ICML, 2020.
[17] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In KDD, 2020.
[18] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph contrastive learning with adaptive augmentation. arXiv preprint arXiv:2010.14945, 2020.
[19] Yuanqi Du, Shiyu Wang, Xiaojie Guo, Hengning Cao, Shujie Hu, Junji Jiang, Aishwarya Varala, Abhinav Angirekula, and Liang Zhao. Graphgt: Machine learning datasets for deep graph generation and transformation. In NeurIPS, 2021.
[20] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In ICLR, 2021.
[21] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In AAAI, 2020.
[22] Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. In ICLR, 2021.
[23] Fan-Yun Sun, Jordan Hoffman, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In ICLR, 2019.
[24] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep Graph Contrastive Representation Learning. In ICML Workshop, 2020.

[25] Ming Jin, Yizhen Zheng, Yuan-Fang Li, Chen Gong, Chuan Zhou, and Shirui Pan. Multi-scale contrastive siamese networks for self-supervised graph representation learning. arXiv preprint arXiv:2105.05682, 2021.

[26] Wenzheng Feng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang, Evgeny Kharlamov, and Jie Tang. Graph random neural networks for semi-supervised learning on graphs. In NeurIPS, 2020.

[27] Kaize Ding, Jianling Wang, James Caverlee, and Huan Liu. Meta propagation networks for graph few-shot semi-supervised learning. In AAAI, 2022.

[28] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.

[29] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In ICML, 2018.

[30] Philip Resnik and Eric Hardisty. Gibbs sampling for the uninitiated. Technical report, Maryland Univ College Park Inst for Advanced Computer Studies, 2010.

[31] Brian Kulis and Michael I Jordan. Revisiting k-means: New algorithms via bayesian nonparametrics. arXiv preprint arXiv:1111.0332, 2011.

[32] Ching-Yao Chuang, Joshua Robinson, Lin Yen-Chen, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. arXiv preprint arXiv:2007.00224, 2020.

[33] Galileo Namata, Ben London, Lise Getoor, Bert Huang, and UMD EDU. Query-driven active surveying for collective classification. In Workshop on MLG, 2012.

[34] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. In NeurIPS Relational Representation Learning Workshop, 2018.

[35] 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 NeurIPS, 2020.

[36] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. Graph representation learning via graphical mutual information maximization. In WWW, 2020.

[37] Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In ICLR, 2014.

[38] Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In NeurIPS, 2004.
[46] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In ICLR, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]
A Dataset Details

For the experiments on Cora, Citeseer, and Pubmed datasets, we use the same train/validation/test data splits adopted by [1, 11]. For the Amazon-P and Coauthor CS datasets, we follow the same train/validation/test data splits in [35]. For the experiments with ogbn-arxiv, we follow the OGB benchmarking protocol [36]. Specifically, for the node classification with few labels task, we sample partial labeled nodes from the original training set and use the same validation and test splits as the standard node classification task. Note that for each of the datasets, we run the experiment 10 times and report the average performance.

| Dataset   | Nodes  | Edges  | Features | Classes |
|-----------|--------|--------|----------|---------|
| Cora      | 2,708  | 5,429  | 1,433    | 7       |
| Citeseer  | 3,327  | 4,732  | 3,703    | 6       |
| Pubmed    | 19,717 | 44,338 | 500      | 3       |
| Amazon-P  | 7,650  | 238,162| 745      | 8       |
| Coauthor CS | 18,333 | 81,894 | 6,805    | 15      |
| ogbn-arxiv |169,343 |1,166,243|128      |40      |

Table 4: The statistics of the datasets.

B Implementation Details

We implement our proposed framework in PyTorch and optimize it with Adam [44]. All experiments are conducted on a Nvidia Tesla v100 16GB GPU. We set $L$ to 10 in our multi-scale feature propagation. The output dimension of our encoder network is fixed to 512. The number of hidden units for the MLP projector is set to 2048. We set the number of negative samples $M$ in $L_{str}$ to 512. For the training of $S^3$-CL, we first pre-train the encoder network by minimizing the structural contrastive loss $L_{str}$ and initialize the model parameters with the pre-trained weights. After that, we optimize the model as illustrated in Algorithm 1. We tune the balancing parameter $\gamma$ within the search space $\{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$. The value of $\xi$ in our Bayesian non-parametric prototype inference algorithm is selected from $\{0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$. For all the baseline methods, we also use Adam as the optimizer. We do grid search for the learning rate in $\{1 \times 10^{-5}, 5 \times 10^{-5}, 1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}, 5 \times 10^{-3}, 1 \times 10^{-2}, 5 \times 10^{-2}, 1 \times 10^{-1}, 5 \times 10^{-1}\}$ on different datasets. Our implementation of $S^3$-CL can be found at https://github.com/S3CL-Code/S-3-CL

B.1 Bayesian Non-parametric Prototype Inference Algorithm

To better illustrate the process of Bayesian non-parametric prototype inference, we summarize the steps in Algorithm 2.

Algorithm 2: Bayesian Non-parametric Prototype Inference.

Input: Node representation $h_1, ..., h_N$; threshold to generate new prototype $\beta$
Output: Prototype representation $C$, pseudo labels $Z$, and number of clusters $K$
1. Initialize $\tilde{K} = 1$, $z_i = 1$ and $c_1 = \frac{1}{N} \sum_{i=1}^{N} h_i$.
2. while not converge do
3.   for $i = 1$ to $N$ do
4.     Update the pseudo label $z_i$ according to Eq.(5)
5.     if $\min_k d_{ik} > \beta$, set $\tilde{K} = \tilde{K} + 1$
6.   for $k = 1$ to $\tilde{K}$ do
7.     Compute the prototype representation $c_k = \sum_{z_i = k} h_i$
8. set $K = \tilde{K}$
9. return $C$, $Z$, and $K$
B.2 Prototype Refinement via Label Propagation

Considering the fact that the pseudo labels inferred by the Bayesian non-parametric algorithm could be inaccurate, we further refine the pseudo labels by label propagation [45] on the graph. This way we can smooth the inaccurate pseudo labels and refine the cluster prototype representations by leveraging graph structure knowledge. Firstly, we convert the prototype assignments \( Z \) to a one-hot pseudo label matrix \( Z \in \mathbb{R}^{N \times K} \), where \( Z_{ij} = 1 \) if and only if \( z_i = k \). Following the idea of Personalized PageRank (PPR) [46], the pseudo labels after \( T \) aggregation steps \( Z^{(T)} \) are updated by:

\[
Z^{(t+1)} = (1 - \beta) \tilde{A}_{sym} Z^{(t)} + \beta Z^{(0)},
\]

where \( Z^{(0)} = Z \) and \( \beta \) can be considered as the teleport probability in PPR. Next, we convert the propagated results \( Z^{(T)} \) back to hard pseudo labels by setting \( z_i = \arg \max_k Z^{(T)}_{ik} \) for \( i \in \{1, ..., N\} \).

C Additional Experimental Results

C.1 Study on Prototype Inference.

In the proposed semantic contrastive learning, we propose a Bayesian non-parametric algorithm to infer prototypes of node representations. During the training, a new prototype will be instantiated if the distance between all existing prototype representations and the representation of a node is larger than a threshold \( \beta \). To verify the effectiveness of the prototype inference algorithm, we design a baseline model that adopts K-Means clustering algorithm to infer prototypes as in [22]. For a fair comparison, label propagation is also applied to refine the prototypes obtained by the baseline model. The results shown in Table 5 demonstrate that the proposed Bayesian non-parametric prototype inference algorithm is good enough to capture semantic information of the nodes, even without the knowledge on the number of classes.

| Datasets | Cora | Citeseer | Pubmed |
|----------|------|----------|--------|
| \( \beta \) | 0.20 | 0.15 | 0.35 |
| Estimated \( K \) | 8 | 6 | 3 |
| Classes | 7 | 6 | 3 |
| \( S^3 \)-CL | 84.5±0.4 | 74.6±0.4 | 80.8±0.3 |
| Baseline | 84.8±0.6 | 74.6±0.3 | 80.7±0.2 |

Table 5: Ablation study on prototype inference.

C.2 Effect of Negative Example Sampling

We study the influence of the size of sampled negative examples in the structural contrastive loss of \( S^3 \)-CL. The results are shown in Table 6. We can see that as the number of sampled negative examples increase from a small value, the performance of \( S^3 \)-CL can be improved. However, further improving the number of sampled negative examples to \( M > 512 \) does not lead to better performance.

| \( M \) | Cora | Citeseer | Pubmed |
|--------|------|----------|--------|
| 64 | 84.0±0.2 | 74.1±0.3 | 80.3±0.3 |
| 256 | 84.2±0.4 | 74.4±0.4 | 80.3±0.2 |
| 512 | 84.5±0.4 | 74.6±0.4 | 80.8±0.3 |
| 1024 | 84.6±0.7 | 74.6±0.6 | 80.7±0.6 |
| 2048 | 84.6±0.5 | 74.5±0.2 | 80.8±0.5 |

Table 6: Effect analysis on the value of balancing parameter (\( \gamma \)) in the overall loss function.

C.3 Effect of Balancing Parameter.

In this subsection, we study the effect of the balancing parameter \( \gamma \) in the final loss function on three datasets, including Cora, Citeseer and Pubmed. Note that we also have similar observations on
the other datasets. The results from Figure 5 show that $S^3$-CL usually achieve the best results with balancing parameter $\gamma$ set between 0.4 to 0.6.

![Graphs showing effect analysis on the value of balancing parameter ($\gamma$).](a) Cora (b) Citeseer (c) Pubmed)

Figure 5: Effect analysis on the value of balancing parameter ($\gamma$).

C.4 Efficiency Analysis.

In this subsection, we evaluate the model efficiency of different methods in terms of training time, memory consumption, and parameter size. We first compare $S^3$-CL with existing unsupervised GCL methods and show the results in Table 7. Though the baseline methods only use shallow GNNs that cannot capture global structural knowledge, $S^3$-CL still has less memory consumption and smaller parameter size, and also is competitive with the most efficient baselines DGI and SUGRL in terms of training time per epoch. The main reason is that our approach uses multi-scale feature propagation to capture the global structural knowledge, which saves the augmentation time during the training and allows the network encoder to be built with one-layer neural network.

Moreover, for each of the existing unsupervised GCL methods, we increase the encoder network depth to 10 and show their efficiency evaluation results in Table 8. It is noteworthy that by increasing the depth of the GNN encoder, existing methods have to introduce more model parameters to capture long-range node interactions, leading to slower training speed and larger memory consumption. In this case, our proposed framework $S^3$-CL has the best efficiency for capturing the same scale of global structural knowledge.

| Methods | Cora | Citeseer | Pubmed |
|---------|------|----------|--------|
| DGI     | 4.2  | 3967     | 3.1 x 10^6 |
| GMII    | 7.3  | 4275     | 3.4 x 10^6 |
| MVGRL   | 9.2  | 2105     | 3.1 x 10^6 |
| MERIT   | 11.0 | 2507     | 3.4 x 10^6 |
| S^3-CL  | 5.2  | 1405     | 7.3 x 10^5 |

Table 7: Efficiency comparisons between $S^3$-CL and baseline methods w.r.t. training time (seconds/epoch), memory occupation (MB) and number of parameters.

| Methods | Cora | Citeseer | Pubmed |
|---------|------|----------|--------|
| DGI     | 5.9  | 3967     | 3.1 x 10^6 |
| GMII    | 7.3  | 4275     | 3.4 x 10^6 |
| MVGRL   | 9.2  | 2105     | 3.1 x 10^6 |
| MERIT   | 11.0 | 2507     | 3.4 x 10^6 |
| S^3-CL  | 5.2  | 1405     | 7.3 x 10^5 |

Table 8: Efficiency comparisons between $S^3$-CL and baseline methods (10-layer encoder network) w.r.t. training time (seconds/epoch), memory occupation (MB) and number of parameters.