A Notations

Notations	Descriptions
$\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$	Graph with node set $\mathcal{V}$ and edge set $\mathcal{E}$.
$\mathbf{A},\mathbf{X}$	Adjacency matrix and node features.
$\mathcal{G}^u,\mathcal{G}^v$	Augmented views of graph.
$\mathcal{T},t_u,t_v$	Data augmentation pool and the functions in it.
$H(\cdot,\cdot)$	Cross enptory.
$N(\boldsymbol{\mu},\boldsymbol{\sigma})$	Gaussian Distribution with mean and variance vector.
$f_\mu(\cdot),f_\sigma(\cdot)$	Mean vector projector and variance vector projector.
$\boldsymbol{x}_i$	Feature of i-th nodes.
$\mathrm{Bernoulli}(\phi)$	Bernoulli distribution with the probability vector.
$\mathcal{W}(\cdot\|\|\cdot)$	Wasserstein distance.
$\epsilon\sim\mathrm{Uniform}(0,1)$	Random variable from the uniform distribution.
$\mathbf{P}$	Probabilistic topology learned by OS-GCL.
$p_\theta(\boldsymbol{z}_i\,\|\,\mathbf{X},\mathbf{A})=N(\hat{\boldsymbol{\mu}}_i,\hat{\boldsymbol{\sigma}}_i)$	Probabilistic embedding with mean and variance vectors.
$p_\theta(\boldsymbol{z}_i^{u,v}\,\|\,\mathbf{X},\mathbf{A})$	Augmented embeddings after distribution perturbation.
$p_\theta(\boldsymbol{z}_i^{neg,u,v}\,\|\,\mathbf{X},\mathbf{A})$	Estimated negative distribution.

Table A.1: Notations of OS-GCL.

B Algorithm and Complexity Analysis

B.1 Algorithm

From Figure 4, the proposed OS-GCL consists of the following three main components: (1) *Feature and Topology Probability Estimation* estimates the distribution of feature and topology with Gaussian and Bernoulli distributions, (2) *Probabilistic Message Passing* aggregates the feature distribution on the probabilistic topology, and (3) *ProbNCE* contrasts the distributions of positive and estimated negatives. The training process of OS-GCL is shown in Algorithm 1.

Algorithm 1: Framework of OS-GCL.

Input: Graph $\mathcal{G}=\{\mathbf{A},\mathbf{X}\}$; Hyperparameters $\{\alpha,k\}$.
Output: Optimized model $\mathrm{GNN}^\star_{\boldsymbol{\Theta}}$.

Initialize parameters randomly;
Compute higher-order matrix $\mathcal{A}$;
for *each epoch* **do**
  `// Probability Estimation`
  Estimate the mean and variance vector $p(\boldsymbol{x}_i)=N(\boldsymbol{\mu}_i,\boldsymbol{\sigma}_i)\leftarrow$ Eq.(3);
  `// Probabilistic Message Passing`
  Compute $\mathcal{W}(p(\boldsymbol{x_i})||p(\boldsymbol{x_j}))\leftarrow$ Eq.(5);
  Get the probability parameter $\phi_{ij}\leftarrow$ Eq.(6);
  Learn the probabilistic topology $\mathbf{P}\leftarrow$ Eq.(7);
  Get node embeddings $p_\theta(\boldsymbol{z}_i\,|\,\boldsymbol{x}_i,\mathbf{A})\leftarrow$ Eq.(E.1-E.2);
  Distribution Perturbation $\leftarrow$ Eq, (E.4);
  `// ProbNCE`
  Compute the ProbNCE loss $\mathcal{L}\leftarrow$ Eq.(11);
  Update model parameters by minimizing $\mathcal{L}$;
Return $\boldsymbol{\Theta}$;

B.2 Complexity

OS-GCL consists of three main components: probability estimation, probabilistic message passing, and ProbNCE. Denote $|\mathcal{V}|$ and $|\mathcal{E}|$ as the numbers of nodes and edges in the original graph. The dimensions of the feature and the hidden states are both d. Firstly, in probability estimation, the time complexity is $\mathcal{O}(2|\mathcal{V}|d^2)$. Secondly, in probabilistic message passing, the higher-order transition probability can be computed during the preprocessing of the dataset. Thus, the time complexity of probabilistic message-passing is $\mathcal{O}(|\mathcal{V}|^2(6d+2d^2)+l\rho|\mathcal{V}|d^2+l\rho|\mathcal{E}|d)$ where $\rho\approx 1$ in this paper represents the ratio of the edges to the original one after edge filtering, and l is the number of GNN layers. Thirdly, in ProbNCE, the time complexity is $\mathcal{O}(3|\mathcal{V}|(8d+2d^2))$. Therefore, the time complexity of OS-GCL is $\mathcal{O}(|\mathcal{V}|^2)$ which is of the same magnitude as GCL methods (*e.g.*, GRACE). Compared to them, OS-GCL increases the time cost for calculating the probabilistic topology, while significantly reducing the computational cost of the similarities of the negative samples in InfoNCE loss, which is also $\mathcal{O}(|\mathcal{V}|^2)$. Thus, the time complexity of OS-GCL is consistent with the other GCL methods.

C Proofs and Derivations

C.1 Proof of Proposition 1

Proposition 1. *The InfoNCE-based graph contrastive learning method is a* $(2n-1)$*-way* 1*-shot classifier structured in the form of softmax-based cross-entropy, given a graph* $\mathcal{G}$ *with* n *nodes, i.e., there are* $2n-1$ *classes, each with only one sample.*

Proof. Let $o_i=o_{ii}^v=\mathrm{sim}(\boldsymbol{u}_i,\boldsymbol{v}_i)/\tau$ be the predictive output and reformulate the InfoNCE into a form of *softmax*-based function:

$$\begin{aligned}\mathcal{L}_i&=-\log\frac{\exp(o_{ii}^v)}{\exp(o_{ii}^v)+\sum_{k\neq i}\exp(o_{ik}^v)+\sum_{k\neq i}\exp(o_{ik}^u)}\\&=-\log\underset{\{\mathcal{G}^u\cup\mathcal{G}^v\}\setminus v_i}{\mathrm{softmax}}(o_i).\end{aligned}\tag{C.1}$$

Denote each sample, except the target node $\boldsymbol{u}_i$, as a class and the positive node as the category of the target node, the *softmax* term represents the predicted probability $q_i^u(\boldsymbol{u}_i)$ of it being classified into the correct category:

$$q_i^u(\boldsymbol{u}_i)=\underset{\{\mathcal{G}^u\cup\mathcal{G}^v\}\setminus v_i}{\mathrm{softmax}}(o_i).\tag{C.2}$$

Then, let $p(\boldsymbol{u}_i)=[1,0,0,\cdots]$ be the true probability, then:

$$\begin{aligned}\mathcal{L}_i&=-\log q_i^u(\boldsymbol{u}_i)\\&=-\left(1\cdot\log q_i^u(\boldsymbol{u}_i)+\sum_{k\neq i}0\cdot\log q_k^v(\boldsymbol{u}_i)+\sum_{k\neq i}0\cdot\log q_k^u(\boldsymbol{u}_i)\right)\\&=-\left(p_i^u(\boldsymbol{u}_i)\cdot\log q_i^u(\boldsymbol{u}_i)\right.\\&\quad\left.+\sum_{k\neq i}p_k^v(\boldsymbol{u}_i)\cdot\log q_k^v(\boldsymbol{u}_i)+\sum_{k\neq i}p_k^u(\boldsymbol{u}_i)\cdot\log q_k^u(\boldsymbol{u}_i)\right)\\&=H(p(\boldsymbol{u}_i),q(\boldsymbol{u}_i)),\end{aligned}\tag{C.3}$$

where $H(p(\boldsymbol{u}_i),q(\boldsymbol{u}_i))$ is the *cross-entropy* between the true probability and the predicted probability.

Therefore, the graph contrastive learning method based on InfoNCE is a softmax-based cross-entropy classifier, where each sample, except the target node $\boldsymbol{u}_i$, is considered a class. In total, there are $|\mathcal{V}^u|+|\mathcal{V}^v|-1=n+n-1=2n-1$ classes.

Each node is assigned to a single class, as the augmentation functions do not alter the node from Definition 1, and each class contains only one sample. That is, the InfoNCE-based graph contrastive learning method can be described as a $(2n-1)$-way 1-shot classifier. □

C.2 Derivation of Probabilistic Topology in Equation 7

Consider a discrete distribution $D \sim \text{Discrete}(\alpha)$ where $\alpha_i \in (0, \infty), i = 1, 2, \cdots, n$ represents the probabilities of each class z from a categorical distribution. The Gumbel-softmax yields a continuous and differentiable approximation of the samples:

$$c_i = \frac{\exp((\log(\alpha_i) + G_i)/\tau)}{\sum_{k=1}^{n} \exp((\log(\alpha_k) + G_k)/\tau)} \tag{C.4}$$

$$= \underset{k=1,2,\cdots,n}{\text{softmax}}((\log(\alpha_i) + G_i)/\tau), \tag{C.5}$$

where $G_i \sim \text{Gumbel}(\cdot)$. We then can derive Eq.(7) using the following Lemma.

Lemma 1. *In the binary case, a softmax-based categorical distribution with $\alpha_1, \alpha_2 (\alpha_1 + \alpha_2 = 1)$ is equivalent to the* $\text{sigmoid}(\alpha_1\text{-}\alpha_2)$.

Proof. Let c_i be the output probability of the class i. Since the distribution is degenerate, we thus consider c_1:

$$c_1 = \underset{k=1,2}{\text{softmax}}(\alpha_1) = \frac{\exp(\alpha_1)}{\exp(\alpha_1) + \exp(\alpha_2)} = \frac{1}{1 + \frac{\exp(\alpha_2)}{\exp(\alpha_1)}}$$
$$= \frac{1}{1 + \exp(-(\alpha_1 - \alpha_2))} = \text{sigmoid}(\alpha_1\text{-}\alpha_2)\,. \tag{C.6}$$

That is, the binary case of a softmax-based categorical distribution is equivalent to the sigmoid output of the probability difference between the two classes. □

Lemma 1 gives a connection between the softmax and sigmoid functions, based on which we have the equivalent approximation as the Gumbel-softmax in the binary case:

$$\begin{aligned} c_i &= \underset{k=1,2,\cdots,n}{\text{softmax}}((\log(\alpha_i) + G_i)/\tau) \\ &= \text{sigmoid}((\log(\alpha_1) + G_1)/\tau - (\log(\alpha_2) + G_2)/\tau) \\ &= \text{sigmoid}((\log(\alpha_1) - \log(\alpha_2) + G_1 - G_2)/\tau) \\ &= \text{sigmoid}((\log \frac{\alpha_1}{\alpha_2} + G_1 - G_2)/\tau) \\ &= \text{sigmoid}((\log \frac{\alpha_1}{1 - \alpha_1} + G_1 - G_2)/\tau). \end{aligned} \tag{C.7}$$

The difference between two Gumbel distribution is a logistic distribution, *i.e.*, $G_1 - G_2 \sim \text{Logistic}$ which can be sampled by $\log \epsilon - \log(1 - \epsilon)$, where $\epsilon \sim \text{Uniform}(0, 1)$. Therefore, the binary version of Gumbel-softmax is:

$$\begin{aligned} c_i &= \text{sigmoid}((\log \frac{\alpha_1}{1 - \alpha_1} + \log \epsilon - \log(1 - \epsilon))/\tau) \\ &= \text{sigmoid}((\log \frac{\alpha_1}{1 - \alpha_1} + \log \frac{\epsilon}{1 - \epsilon}))/\tau). \end{aligned} \tag{C.8}$$

D Related Works

D.1 Graph Contrastive learning

Contrastive learning (CL) involves the identification of individual samples in a latent space without the need for supervision by distinguishing each sample from all others [Liu *et al.*, 2021]. This approach has gained traction in the fields of computer vision (CV) [Logeswaran and Lee, 2018; He *et al.*, 2020; Chen *et al.*, 2020; Chuang *et al.*, 2020; Yeh *et al.*, 2022] and natural language processing (NLP) [Oord *et al.*, 2018], leading to increased exploration of CL in graph representation learning.

Drawing inspiration from the effective self-supervised learning techniques, several studies have delved into graph contrastive learning (GCL) [Wu *et al.*, 2021; Liu *et al.*, 2022; Liang *et al.*, 2023]. Notably, various GCL methods, such as DGI [Velickovic *et al.*, 2019], MVGRL [Hassani and Khasahmadi, 2020], and GRACE [Zhu *et al.*, 2020].

The existing GCL methods have leveraged graph neural networks (GNNs) [Kipf and Welling, 2017; Veličković *et al.*, 2018] as encoders to maximize the mutual information between local and global representations, utilize graph diffusion for generating distinct graph views, and apply the InfoNCE loss function on graphs, respectively.

Additionally, studies are focusing on node-level GCL, including BGRL [Thakoor *et al.*, 2021], CCA-SSG [Zhang *et al.*, 2021], and GRADE [Wang *et al.*, 2022], while GraphCL [You *et al.*, 2020] concentrates on graph-level tasks. Recent efforts have also been directed towards addressing issues in GCL through augmentation mechanisms and negative sampling techniques [Yu *et al.*, 2022; Xia *et al.*, 2022; Zhang *et al.*, 2023].

However, despite these advancements, they ignore the fundamental nature of GCL that graph contrastive learning is a one-shot learner and thus faces the limited self-supervised signal issue.

E Model Details

E.1 Encoders

We use two encoders (*i.e.*, mean encoder and variance encoder) to obtain the distribution of embedding $p_\theta(\boldsymbol{z}_i \mid \mathbf{X}, \mathbf{A})$, which is based on the learned probabilistic topology:

$$p_\theta(\boldsymbol{z}_i \mid \mathbf{X}, \mathbf{A}) = N(\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\sigma}}_i), \tag{E.1}$$

$$\hat{\boldsymbol{\mu}}_i = \boldsymbol{\Theta}_\mu^\top \sum_{j \in \mathcal{N}_i \cup \{i\}} \frac{\mathbf{P}_{ji}}{\sqrt{\hat{d}_j \hat{d}_i}} \boldsymbol{\mu}_j, \tag{E.2}$$

$$\hat{\boldsymbol{\sigma}}_i = \boldsymbol{\Theta}_\sigma^\top \sum_{j \in \mathcal{N}_i \cup \{i\}} \frac{\mathbf{P}_{ji}}{\sqrt{\hat{d}_j \hat{d}_i}} \boldsymbol{\sigma}_j, \tag{E.3}$$

where $p_\theta(\boldsymbol{z}_i \mid \boldsymbol{x}_i, \mathbf{A})$ represents the distribution of embedding, $\hat{\boldsymbol{\mu}}_i$ and $\hat{\boldsymbol{\sigma}}_i$ denote the mean vector and variance vector, and $\boldsymbol{\Theta}_\mu^\top$ and $\boldsymbol{\Theta}_\sigma^\top$ are the parameters of the two encoders. $\mathcal{N}_i$ represents the higher-order neighbors in the adjacency matrix $\mathbf{A}$ and $\hat{d}_i = 1 + \sum_{j \in [1,n]} \mathbf{A}_{ji}$.

E.2 Distribution Perturbation as Augmentation

In contrast to traditional graph contrastive learning, we design a distribution perturbation mechanism to create augmented views of the embedding $p_\theta(\boldsymbol{z}_i \mid \mathbf{X}, \mathbf{A})$, rather than randomly augmenting the input data. Although GCL obtains more samples through data augmentation, it still suffers from the one-shot learning setting and cannot estimate all potential samples in the corresponding CI-classes. In contrast, OS-GCL learns the underlying data distribution. One can perturb the learned distribution to obtain more discriminative sample distributions and use them for augmentation. We thus design an distribution perturbation mechanism as follows:

$$p_\theta(\boldsymbol{z}_i^{u/v} \mid \mathbf{X}, \mathbf{A}) = N(\hat{\boldsymbol{\mu}}_i + \boldsymbol{\epsilon}_\mu^{u/v}, \hat{\boldsymbol{\sigma}}_i + \boldsymbol{\epsilon}_\sigma^{u/v}), \tag{E.4}$$

where $\boldsymbol{\epsilon} \sim \text{Uniform}(0, I)$ are random perturbations and $p_\theta(\boldsymbol{z}_i^{u/v} \mid \mathbf{X}, \mathbf{A})$ are two augmented distributions.

Remark 2. *OS-GCL with distribution perturbation has no need to input data into the encoder twice to generate the positive/negative embeddings, which maintains high efficiency.*

F Datasets and Baselines

F.1 Datasets

We evaluate OS-GCL on 7 node classification benchmarks: (1) Citation networks [Kipf and Welling, 2017] including Cora, CiteSeer, and PubMed are widely used benchmarks in graph representation learning. (2) Co-purchase networks [Shchur *et al.*, 2018] including Amazon Photo. (3) Co-author networks [Shchur *et al.*, 2018] including Coauthor CS and Coauthor Physics. (4) Large dataset [Hu *et al.*, 2020]: ogbn-arXiv. The statistics of the datasets are listed in the Table F.1.

Dataset	#Nodes	#Edges	#Features	#Classes
Cora	2,708	10,556	1,433	7
CiteSeer	3,327	9,104	3,703	6
PubMed	19,717	88,648	500	3
Photo	7,650	238,162	745	8
CS	18,333	163,788	6,805	15
Physics	34,493	495,924	8,415	5
Ogbn-arXiv	169,343	1,166,243	128	40

Table F.1: Statistics of Dataset.

F.2 Baselines

We evaluate OS-GCL in comparison with the following three types of baselines, including the semi-supervised graph neural networks, unsupervised network embedding methods, and the most related self-supervised graph contrastive learning methods:

- **(Semi-) Supervised GNNs:** GCN [Kipf and Welling, 2017] and GAT [Veličković *et al.*, 2018], representing the capability of GNNs given the label information. UaGGP [Liu *et al.*, 2020] is a semi-supervised distribution-based graph learning method.
- **Unsupervised network embedding methods:** DeepWalk [Perozzi *et al.*, 2014] and VGAE [Kipf and Welling, 2016], which are traditional graph learning that do not rely on label information. VGAE also incorporates the feature probability distribution through Gaussian distribution.
- **Self-supervised graph contrastive learning models** including DGI [Velickovic *et al.*, 2019], GRACE [Zhu *et al.*, 2020], InfoGCL [Xu *et al.*, 2021], MVGRL [Hassani and Khasahmadi, 2020], SpCo [Zhang *et al.*, 2023], CCA-SSG [Zhang *et al.*, 2021], GRADE [Wang *et al.*, 2022], and GraphACL [Xiao *et al.*, 2023], which are the SOTA baselines in graph contrastive learning. BGCL [Hasanzadeh *et al.*, 2021] is a probability estimation method in GCL.

G Experimental Details

G.1 Evaluation Protocol

We adhere to the commonly employed evaluation procedure in graph contrastive learning [Zhu *et al.*, 2020; Zhang *et al.*, 2021], in which the model is initially trained using all nodes and edges without label information. Subsequently, an additional linear classifier is trained using the fixed node embeddings from the model. The final results of OS-GCL are presented as the mean value with the standard deviation of 10 random splits.

G.2 Implantation Details

We employ one single linear layer as the probability estimator and a two-layer GCN as the graph encoder. The dimension of latent space is 256. The learning rate is tuned in {1e-3, 1e-4, 1e-5}, the temperatures $\tau \in (0, 1)$. For OS-GCL, the maximum high order is set to $k = 10$, the feature-topology trade-off parameter $\alpha \in [0, 1]$, and the threshold range for probabilistic topology is searched in $[0, 1]$. All experiments were performed using a single Nvidia V100 32G GPU. The hyperparameters are listed in Table G.1.

Hypermeters	Cora	CiteSeer	PubMed	Photo	CS	Physics
learning rate lr	1e-4	1e-4	1e-5	1e-5	1e-4	1e-4
trade-off α	0.8	0.8	1.0	0.8	0.9	0.9
higher order k	5	3	5	1	2	9
threshold δ	0.6	0.5	0.7	0.5	0.5	0.7
loss weight λ	0.1	0.1	0.1	0	0	0.0
temperature τ_1	0.5	0.7	0.5	0.5	0.1	0.9
temperature τ_2	0.4	0.9	0.6	0.1	0.5	0.7

Table G.1: Hyperparameters of OS-GCL.

G.3 OS-GCL Variants

We compare OS-GCL with the following variants in Section 4.3.

1. OS-GCL *w/o* Feature Probability Estimation: we discard the Gaussian distribution, while still maintaining structure learning and using InfoNCE loss.
2. OS-GCL *w/o* Topology Probability Estimation: graph structure remains unchanged during the training process.
3. OS-GCL *w/o* Probability Estimation (Both): the variant is identical to GRACE, as the OS-GCL is non-operational without the provision of probability distributions.

4. OS-GCL *w/o* Feature in Probabilistic Message-Passing: the variant exclusively employs higher-order proximities for probabilistic message-passing (*i.e.*, $\alpha = 1$).
5. OS-GCL *w/o* Topology in Probabilistic Message-Passing: the probabilistic topology is learned using only feature distance (*i.e.*, when $\alpha = 0$).
6. OS-GCL *w/o* Higher Order in Probabilistic Message-Passing: we use the first order proximity to learn the probabilistic topology for probabilistic message-passing (*i.e.*, when $k = 1$).
7. OS-GCL *w/o* Distribution Perturbation: we perturb the feature estimation to create two views before the encoder.
8. OS-GCL *repl.* KL Divergence: we use the KL divergence instead of the WS distance.
9. OS-GCL *repl.* JS Divergence: we use the JS divergence instead of the WS distance.
10. OS-GCL *w/o* ProbNCE-C: the discrete ProbNCE introduced before, which utilizes parameterization to obtain deterministic embedding.
11. OS-GCL *w/o Wassertein* Distance in ProbNCE-C Loss: we use the mean vector as the final embedding.
12. OS-GCL *w/o* Negative Estimation in ProbNCE-C Loss: the initial version of continuous ProbNCE does not involve the estimation of the negative distribution.

H More Experimental Results

H.1 Large-Scale Dataset

We also evaluate OS-GCL on one commonly used large-scale graph dataset, ogbn-arXiv [Hu *et al.*, 2020]. We report the results on both validation and test sets in Table H.1 as suggested in previous works [Thakoor *et al.*, 2021]. As most NCE-based methods are rarely evaluated on large datasets, limited by the computational complexity of InfoNCE. To achieve that, we use sub-sampling to train mini-batches for the evaluation of the ogbn-arXiv dataset. We adhere to the commonly employed evaluation procedure of self-supervised graph learning on ogbn-arXiv [Thakoor *et al.*, 2021; Zhang *et al.*, 2023]. As shown in Table H.1, the proposed OS-GCL still achieves competitive performance on both the validation and test sets on large-scale graph dataset, showing the scalability of the proposed method.

H.2 Computational Cost of $\mathcal{A}$

As stated previously, the higher-order transition probability $\mathcal{A}$ can be calculated as in the preprocessing of datasets. We provide the calculation time of $\mathcal{A}$ in contrast with the training time of our models in Table H.2. It is obvious that the computational cost of $\mathcal{A}$ is acceptable even in the large-scale dataset.

H.3 Embedding Similarity Analysis

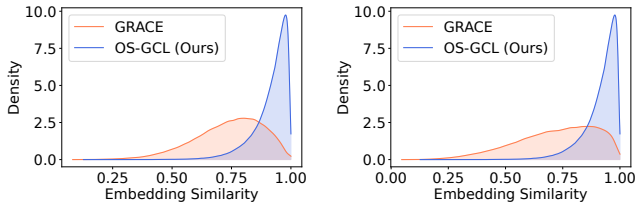
We compare the density of embedding similarity with the same label and different labels. It is obvious that OS-GCL can effectively make nodes within the same label closer and push nodes within distinct labels far away, which indicates the importance of the limited self-supervised signal issue in GCL.

Methods	Validation	Test
MLP	57.6±0.2	55.5±0.3
node2vec	71.3±0.1	70.1±0.1
Random-Init	69.9±0.1	68.9±0.2
DGI	71.2±0.1	70.3±0.1
GRACE full-graph	OOM	OOM
GRACE-Subsampling (k=2)	60.5±3.7	60.2±4.1
GRACE-Subsampling (k=8)	71.3±0.2	70.3±0.2
GRACE-Subsampling (k=32)	72.2±0.2	71.2±0.2
GRACE-Subsampling (k=2048)	72.6±0.2	71.5±0.1
MVGRL	69.3±0.3	68.2±0.2
SUGRL	70.2±0.1	69.3±0.2
Graph Barlow Twin	71.1±0.3	70.0±0.2
SFA	72.3±0.1	71.6±0.4
OS-GCL (Ours)	**72.6±0.3**	**71.8±0.2**

Table H.1: Accuracy (%±standard deviation) of on ogbn-arXiv.

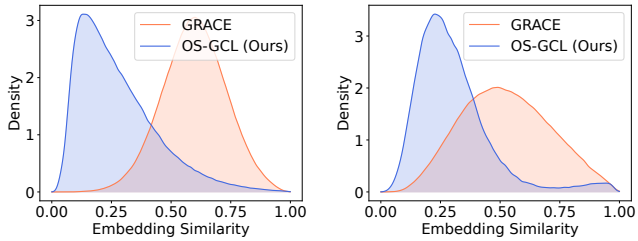
Time	Cora	CiteSeer	PubMed	Photo	CS	Physics	ogbn-arXiv
Training	46.57	41.25	63.8	66.65	124.68	464.85	32571.3
$\mathbf{A}^5$	0.63	0.99	10.98	5.18	29.91	64.76	1354.6
Ratio	1.3%	2.4%	17.2%	7.8%	24.0%	14.0%	4.2%

Table H.2: Computational time (seconds) of $\mathcal{A}$.



(a) Cora Dataset. (b) Citeseer Dataset.

Figure H.1: Density of embedding similarity with the same label.



(a) Cora Dataset. (b) Citeseer Dataset.

Figure H.2: Density of embedding similarity with the distinct labels.