A Details on GCL Methods, Benchmarks and Experiment Settings

A.1 Brief Introduction of GCL methods

Methods for the node classification task.

- **GRACE** [61]. GRACE generates two graph views by corruption and learns node representations by maximizing the agreement of node representations in these two views. To provide diverse node contexts for the contrastive objective, GRACE proposes a hybrid scheme for generating graph views on both structure and attribute levels.
- **GCA** [63]. GCA proposes adaptive augmentation that incorporates various priors for topological and semantic aspects of the graph. On the topology level, GCA designs augmentation schemes based on node centrality measures, while on the node attribute level, GCA corrupts node features by adding more noise to unimportant node features.
- **ProGCL** [49]. ProGCL observes limited benefits when adopting existing hard negative mining techniques of other domains in graph contrastive learning. ProGCL proposes an effective method to estimate the probability of a negative being true one, and devises two schemes to boost the performance of GCL.
- **DGI** [45]. DGI relies on maximizing mutual information between patch representations and corresponding high-level summaries of graphs—both derived using established graph convolutional network architectures. The learnt patch representations summarize subgraphs centered around nodes of interest, and can thus be reused for downstream node-wise learning tasks.
- **MVGRL** [13]. MVGRL introduces a self-supervised approach for learning node and graph level representations by contrasting structural views of graphs. MVGRL shows that unlike visual representation learning, increasing the number of views to more than two or contrasting multi-scale encodings does not improve performance, and the best performance is achieved by contrasting encodings from first-order neighbors and graph diffusion.

Methods for the graph classification task.

- **GraphCL** [53]. GraphCL designs four types of graph augmentations to incorporate various priors, and learns graph-level representations by maximizing the global representations of two views for a graph.
- **ADGCL** [41]. ADGCL proposes a novel principle, adversarial GCL, which enables GNNs to avoid capturing redundant information during training by optimizing adversarial graph augmentation strategies used in GCL.
- **JOAO** [54]. JOAO proposes a unified bi-level optimization framework to automatically, adaptively and dynamically select data augmentations when performing GraphCL on specific graph data. JOAO is instantiated as min-max optimization.
- **InfoGraph** [38]. InfoGraph maximizes the mutual information between the graph-level representation and the representations of substructures of different scales (*e.g.,* nodes, edges, triangles). By doing so, the graph-level representations encode aspects of the data that are shared across different scales of substructures.

A.2 Introduction of Graph Benchmarks

Node classification benchmarks. 1) Citation Networks [34, 29]. Cora, CiteSeer and PubMed are three popular citation graph datasets. In these graphs, nodes represent papers and edges correspond to the citation relationship between two papers. Nodes are classified according to academic topics. 2) Amazon Co-purchase Networks [35]. Photo and Computers are collected by crawling Amazon websites. Goods are represented as nodes and the co-purchase relationships are denoted as edges. Node features are the bag-of-words representation of product reviews. Each node is labeled with the category of goods. 3) Wikipedia Networks [33]. Squirrel and Chameleon was collected from the English Wikipedia, representing page-page networks on specific topics. Nodes represent articles and edges are mutual links between them.

Graph Classification benchmarks. 1) Molecules. MUTAG [7] is a dataset of nitroaromatic compounds and the goal is to predict their mutagenicity on Salmonella typhimurium. PTC-MR [16] is a collection of 344 chemical compounds represented as graphs that report carcinogenicity for male or female rats. 2) Bioinformatics. PROTEINS [3] is a dataset of proteins that are classified as enzymes or non-enzymes. Nodes represent the amino acids and two nodes are connected by an edge if they are less than 6 Angstroms apart. 3) Social Networks. IMDB-BINARY and IMDB-MULTI [51] are movie collaboration datasets consisting of a network of 1,000 actors/actresses who played roles in movies in IMDB. In each graph, nodes represent actors/actresses, and corresponding nodes are connected if they appear in the same movie. REDDIT-BINARY [51] consists of graphs corresponding to online discussions on Reddit. In each graph, nodes represent users, and there is an edge between them if at least one of them responds to the other's comment.

Statistics of datasets are shown in Table 8.

Table 8: Statistics of classification benchmarks. We report average numbers of nodes, edges, and features across graphs in graph classification datasets. For datasets lacking feature attributes, we use all-one vectors as pseudo attributes in practice.

Task	Category	Dataset	#Graphs	# Nodes	# Edges	# Features	# Classes
Node	Citation	Cora	1	2,708	5,278	1,433	7
		CiteSeer	1	3,327	4,552	3,703	6
		PubMed	1	19,717	44,338	500	3
	Co-purchase	Photo	1	7,650	119,081	745	8
		Computers	1	13,752	245,861	767	10
	Wikipedia	Chameleon	1	2,277	36,101	500	6
		Squirrel	1	5,201	217,073	2,089	4
Graph	Protein	MUTAG	188	17.9	39.6	7	2
		PTC-MR	344	14.3	29.4	18	2
	Bioinformatics	PROTEINS	1113	39.1	145.6	0	2
	Social Networks	IMDB-BINARY	1000	19.8	193.1	0	2
		IMDB-MULTI	1500	13.0	131.9	0	3
		REDDIT-BINARY	2000	429.6	995.5	0	2

A.3 Experimental Details

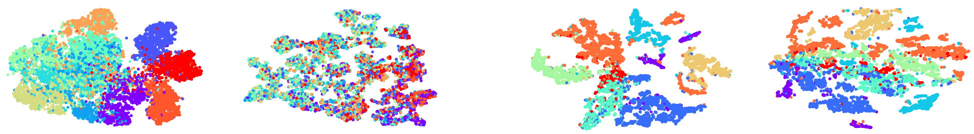
For the node classification task, following Zhu et al. [61], Velickovic et al. [45], Hassani and Khasahmadi [13], we use linear evaluation protocol, where the model is trained in an unsupervised manner and feeds the learned representation into a linear logistic regression classifier. In the training procedure, a 2-layer Graph Convolutional Network (GCN) [22] is adopted as the encoder. We adopt the default settings of Zhu et al. [61]. Specifically, we use removing edges and masking node features as data augmentations. We grid search augmentation ratios in $\{0.0, 0.1, 0.2, 0.3, 0.4\}$. All experiments are trained with Adam SGD optimizer [21] with the learning rate selected from $\{0.01, 0.001, 0.0005\}$. The epoch number is selected from $\{200, 1000, 2000\}$. The other parameters are fixed for all datasets. In the evaluation procedure, we randomly split each dataset with a training ratio of 0.8 and a test ratio of 0.1, and hyperparameters are fixed as the same for all the experiments. Each experiment is repeated ten times with mean and standard derivation of accuracy score.

For the graph classification task, in the training procedure, a Graph Isomorphism Network (GIN) [50] is adopted as the encoder whose layer number is chosen from $\{4, 8, 12\}$ and hidden dimension chosen from $\{32, 512\}$. We use Adam SGD optimizer with the learning rate selected in $\{10^{-3}, 10^{-4}, 10^{-5}\}$ and the number of epochs in $\{20, 100\}$. Following Sun et al. [38], You et al. [53], we feed the generated graph embeddings into a linear Support Vector Machine (SVM) classifier, and the parameters of the downstream classifier are independently tuned by cross-validation. The C parameter is tuned in $\{10^{-3}, 10^{-2}, \cdots, 10^{2}, 10^{3}\}$. We report the mean 10-fold cross-validation accuracy with standard deviation. All experiments are conducted on a single 24GB NVIDIA GeForce RTX 3090.

B Visualization of VCL and GCL via T-SNE

To further illustrate the difference between VCL and GCL, we visualize the representations learned with contrastive loss and uniformity loss using T-SNE [43]. The results are shown in Figure 2. For

VCL, the representations learned by uniformity loss distribute more randomly without clear decision boundaries, compared to those learned by InfoNCE loss. However, for GCL, the representations learned by the two losses both achieve good clustering effects.



(a) VCL-InfoNCE (b) VCL-Uniformity (c) GCL-InfoNCE (d) GCL-Uniformity

Figure 2: T-SNE visualization of representations learned by VCL and GCL, with InfoNCE loss and uniformity loss. Figure 2(a) and 2(b) are conducted with SimCLR on CIFAR10. Figure 2(c) and 2(d) are conducted with GRACE on Amazon-Photo dataset.

C Results of Extensive benchmarks

In our paper, we have chosen commonly estimated benchmarks (Cora, CiteSeer, PubMed, Amazon-Computers, and Amazon Photo) following the original papers (GRACE [61], GCA [63], and so on). Here, we also provide results and discussions about extensive benchmarks including heteophily benchmarks and large benchmarks.

Heteophily benchmarks. We conduct experiments on two heterophilic datasets Wikipedia-Chameleon and Wikipedia-Squirrel [33] with the GRACE method. As observed in Table 9, training with only negative samples (NO Pos) also gains benefits compared with randomly initialized models (NO Training). However, the gap between using uniformity loss (NO Pos) and using contrastive loss (Contrast) is larger than that of homophilic datasets. In conclusion, the positive-free property of GCL is more applicable to homophilic graphs. It agrees with our theoretical analysis in Section 4.2 which assumes neighbors as positive samples.

Table 9: Test accuracy (%) on the homophily and heteophily datasets with the GRACE methods. We compare the performances of models trained the InfoNCE loss (Contrast), uniformity loss (NO Pos), alignment loss (NO Neg), and no optimization objective (NO Training). Mean accuracy with standard derivation is reported after 10 runs. Average accuracy across datasets is reported. We conduct significance testing using Wilcoxon Signed Rank Test [48], comparing the contrastive loss with other loss types. The p-value is averaged across datasets. A value below 0.05 denotes significant accuracy difference (red), while a value above 0.05 indicates insignificance (green).

		Homophily					Heteophily			
		Cora	CiteSeer	PubMed	**Avg**	**Avg p-value**	Chameleon	Squirrel	**Avg**	**Avg p-value**
GRACE	Contrast	84.67 ± 1.39	73.47 ± 2.32	85.80 ± 0.16	81.31	-	48.12 ± 2.35	33.63 ± 1.86	40.88	-
	NO Training	69.12 ± 4.18	60.60 ± 2.59	80.65 ± 0.80	70.12	0.0020	32.23 ± 1.82	25.34 ± 1.22	28.79	0.0020
	NO Pos	82.65 ± 1.18	73.50 ± 2.41	85.28 ± 0.79	80.48	0.1934	42.97 ± 2.11	30.48 ± 2.25	36.73	0.0254
	NO Neg	29.85 ± 1.45	20.42 ± 2.26	39.63 ± 0.81	29.97	0.0020	20.61 ± 2.38	19.58 ± 1.36	20.10	0.0020
GCA	Contrast	84.04 ± 1.55	72.63 ± 2.68	85.92 ± 0.69	80.86	-	46.64 ± 2.85	35.24 ± 1.57	40.94	-
	NO Training	71.25 ± 2.32	58.50 ± 1.32	80.07 ± 0.47	69.94	0.0020	33.36 ± 2.04	25.76 ± 2.39	29.56	0.0020
	NO Pos	83.09 ± 2.03	70.42 ± 3.07	84.68 ± 0.63	79.40	0.1322	40.17 ± 3.93	28.60 ± 1.05	34.39	0.0107
	NO Neg	31.40 ± 3.61	22.16 ± 3.01	39.58 ± 0.83	31.05	0.0020	21.92 ± 4.15	20.19 ± 0.55	21.10	0.0020
ProGCL	Contrast	85.42 ± 3.41	72.85 ± 2.99	OOM	79.14	-	48.38 ± 3.65	33.47 ± 1.93	40.93	-
	NO Training	79.41 ± 0.90	58.08 ± 1.27	83.54 ± 0.83	73.68	0.0026	34.21 ± 1.15	25.26 ± 2.24	29.74	0.0020
	NO Pos	86.76 ± 0.52	70.76 ± 1.63	OOM	78.76	0.2266	46.44 ± 4.14	30.98 ± 4.32	38.71	0.1064
	NO Neg	30.15 ± 2.70	21.08 ± 1.45	21.13 ± 1.20	24.12	0.0020	20.09 ± 1.63	20.46 ± 1.57	20.28	0.0020

Large benchmarks. Here, we further consider a larger node classification benchmark OGB-arxiv [18] with 169,343 nodes and 1,166,243 edges, using the GRACE method. A node-wise similarity matrix is needed when computing the contrastive loss, but its time complexity and space usage are intolerable for large datasets. The scalability problem is one of the reasons why larger datasets are not reported in many original papers. To solve this problem, we randomly sample N=5000 nodes when computing the similarity matrix, and send the resulting matrix to the objective function. For each iteration, we repeat such sampling 5 times and use the mean loss. The random sampling strategy is simple and straightforward, and more complicated strategies will be considered in the future.

As shown in Table 10, the performance only using negative samples is on par with that using contrastive objectives. And only using positive samples on the node classification task also results in collapse. These observations are consistent with our findings.

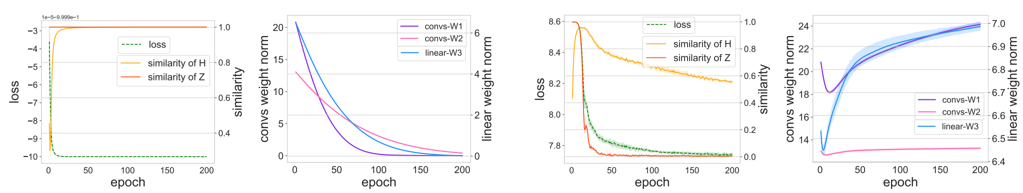
Table 10: Test accuracy (%) on the OGB-arxiv benchmark using GRACE method with the sampled InfoNCE loss (Contrast), uniformity loss (NO Pos), and alignment loss (NO Neg).

	Contrast	NO Pos	NO Neg
OGB-arxiv	65.97 ± 0.23	65.49 ± 0.32	23.88 ± 0.46

D Feature Collapse in Negative-free GCL for Node Classification

In Table 2, we find that the absence of negative samples in GCL leads to a significant performance drop for the node classification task. Numerous factors may be responsible for the suboptimal performance. Here we visualize the training process with alignment loss and InfoNCE loss to show that feature collapse is the underlying cause.

Specifically, we show the tendency of loss, average similarities of node representations $\mathbf{H} = f(\mathbf{X})$ and $\mathbf{Z} = g(\mathbf{H})$, and L_2 norms of weight matrices in Figure 3. From Figure 3(a), we can find that when trained with the alignment loss, the training loss steeply converges to -1 (optimal for the alignment loss) after the start of training. However, the similarities among node representations $\mathbf{H}$ and $\mathbf{Z}$ both unite towards one. It indicates that once the training starts, the model quickly learns the short-cut where most node representations are identical to meet the alignment loss. We also delineate L_2 norms of the weight matrices, which consistently converge to zero during training. As a comparison, we show the training process with InfoNCE loss in Figure 3(b). When trained with InfoNCE loss, the average similarities of node representations are relatively low and norms of weights are non-zero, showing that the collapse issue does not occur in the training process.



(a) Alignment Loss (b) InfoNCE Loss

Figure 3: Tendency of loss, average similarities of node representations $\mathbf{H}$ and $\mathbf{Z}$, and L_2 norms of weight matrices. We choose weight matrices of the first and the second convolutional layer (Convs-W1 and Convs-W2), and the first linear layer of the projection head (Linear-W3). Experiments are conducted on Cora with GRACE.

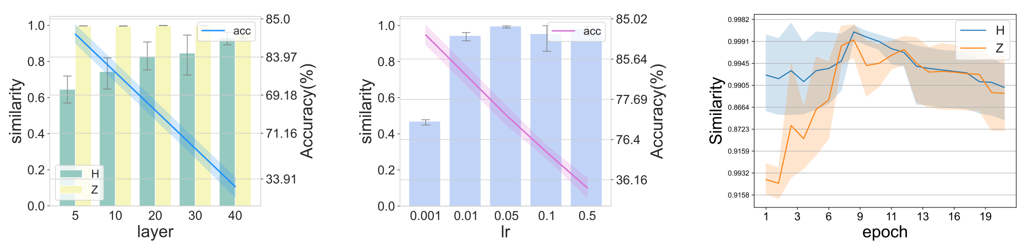
E Why No-negative GCL Not Collapse in the Graph Classification

In Section 5, we observe different phenomena in the graph classification and node classification. Specifically, in the graph classification task, GCL methods achieve decent performance in the no-negative setting, while the representations collapse in the node classification task. From the architecture perspective, we find in the graph classification task, the representations learned by the projector tend to be identity, while the representations learned by the encoder escape from collapse. We suspect that learning a collapsed solution is relatively easier for the global graph representation, which can be achieved solely by the projection head.

Here, we provide some empirical insights into these conjectures. Instead of researching how to make representations not collapse in the node classification, we choose to explore *when no-negative GCL collapses in the graph classification*. A straightforward method is stacking more layers within the encoder. The well-known over-smoothing issue in GNNs states that when the layer number increases, the representations will become identical and lose expressiveness [25]. This is exactly what the alignment loss needs. Taking the MUTAG dataset as an illustrated example, we indeed find an increase in the similarities of representation $\mathbf{H}$ and $\mathbf{Z}$, and a drop in the performance (Figure 4(a)). Another choice is removing the projection head and exposing the encoder. Additionally, we increase the learning rate, whose motivation is enforcing the encoder to iterate to the collapsed solution more quickly. In Figure 4(b), we find that after removing the projection head, the encoder also collapses when the learning rate is raised to 0.01. Besides the above two extreme cases, here we propose a more convincing method. Imitating the node-wise loss in the node classification, we transform the loss in GraphCL to an L-L version. Formally, the L-L align loss for the graph classification is:

$$\hat{\mathcal{L}}_{align} = -\frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_i}\sum_{\mathbf{u}\in\mathcal{G}_i} s(\mathbf{u},\mathbf{v}), \tag{11}$$

where M denotes the number of graphs, N_i denotes the number of nodes in the graph $\mathcal{G}_i$, and the positive sample $\mathbf{v}$ is the corresponding node of $\mathbf{u}$ in the augmented graph. Using this alignment loss, we train the modified GraphCL method and get a terrible test accuracy of 68.18% compared to the original performance of 86.36%. Figure 4(c) shows that the similarities of $\mathbf{H}$ and $\mathbf{Z}$ both converge close to one during training under this loss. These observations further validate our conjecture.



(a) Increase Layer Number (b) Remove Projection Head (c) Use L-L Loss

Figure 4: Experiments for the collapse of no-negative GCL in the graph classification. As the layer number of encoder increases, the similarity of representations $\mathbf{H}$ converges close to one and the performance degrades greatly (Figure 4(a)). A similar phenomenon is observed when removing the projection head and training the encoder with a relatively high learning rate (Figure 4(b)). Additionally, by modifying the graph-level alignment loss to a local node-wise version, we also observe a collapse in the encoder (Figure 4(c)). Experiments are conducted on MUTAG with GraphCL.

F Proof of Theorems

F.1 Derivation of Theorem 4.1

Proof. It is easy to see that under the definition of the positive samples, the alignment loss can be written equivalently as

$$\begin{aligned}\tilde{\mathcal{L}}_{\text{align}}(\mathbf{H}) &= -\mathbb{E}_{x,x^+\sim\mathcal{P}_{\mathcal{G}}(x,x^+)}[\mathbf{h}_x^\top\mathbf{h}_{x^+}] &(12)\\ &= -\sum_{x,x^+}\mathcal{P}_{\mathcal{G}}(x,x^+)[\mathbf{h}_x^\top\mathbf{h}_{x^+}] &(13)\\ &= -\sum_{x,x^+}[\hat{\mathbf{A}}_{x,x^+}\mathbf{h}_x^\top\mathbf{h}_{x^+}]/\sum_{x,x^+}[\hat{\mathbf{A}}_{x,x^+}] &(14)\\ &= -\text{tr}\left(\mathbf{H}\hat{\mathbf{A}}\mathbf{H}^\top\right)/c, &(15)\end{aligned}$$

where $c = \sum_{x,x^+}[\hat{\mathbf{A}}_{x,x^+}]$ is a constant.

Here, to maintain the feature scale, we further consider a regularization term on the norm of node features:

$$\hat{\tilde{\mathcal{L}}}_{\text{align}}(\mathbf{H}) = \tilde{\mathcal{L}}_{\text{align}}(\mathbf{H}) + \|\mathbf{H}\|^2/c. \tag{16}$$

Therefore, the gradient update of the alignment objective (Eq 6) gives the following update rule of node features $\mathbf{H}$:

$$\begin{aligned}\mathbf{H}_{\text{new}} &= \mathbf{H} - \alpha\nabla_{\mathbf{H}}\hat{\tilde{\mathcal{L}}}_{\text{align}}(\mathbf{H}) &(17)\\ &= \mathbf{H} - \alpha/c(-2\mathbf{A}\mathbf{H} + 2\mathbf{H}) &(18)\\ &= (1-2\alpha/c)\mathbf{H} + 2\alpha/c\cdot\mathbf{A}\mathbf{H}, &(19)\end{aligned}$$

where α is the step size. When we choose a specific learning rate $\alpha = c/2$, we recover the graph convolution operation in GCN [22]:

$$\mathbf{H}_{\text{new}} = \mathbf{A}\mathbf{H}, \tag{20}$$

which completes the proof. □

F.2 Derivation of Theorem 5.1

Proof. Denote $c = \sum_{x,x^+}[\hat{\mathbf{A}}_{x,x^+}]$ as a constant. Calculating the gradient of the uniformity loss w.r.t. each node feature $\mathbf{h}_x$ gives the following rule

$$\nabla_{\mathbf{h}_x}\tilde{\mathcal{L}}_{\text{uniform}} = 2/cP_{\mathcal{G}}(x)\sum_{x'}\mathbf{A}_{x,x'}\mathbf{h}_{x'}. \tag{21}$$

In a matrix form, we have

$$\nabla_{\mathbf{H}}\tilde{\mathcal{L}}_{\text{uniform}} = 2/c\mathbf{D}\mathbf{A}\mathbf{H}, \tag{22}$$

where $\mathbf{D}$ is the diagonal matrix containing $\mathcal{P}(x) = \sum_{x'}\mathbf{A}_{x,x'}, \forall x \in \mathcal{V}$.

Therefore, the gradient descent update of the defined uniformity loss gives

$$\mathbf{H}_{\text{new}} = \mathbf{H} - \alpha\nabla_{\mathbf{H}}\tilde{\mathcal{L}}_{\text{uniform}} = \mathbf{H} - 2/c\mathbf{D}\mathbf{A}\mathbf{H}, \tag{23}$$

where α is the step size. It is easy to see its equivalence to the ContraNorm update. □

F.3 Derivation of Theorem 5.2

Proof. Combining Theorem 4.1 and Theorem 5.1, we can directly obtain Theorem 5.2 as a corollary. □

G Discussion on More GCL Methods

The contrastive mode has three mainstreams: local-to-local (L-L), global-to-global (G-G), and global-to-local (G-L) [62]. For the local-to-local perspective, the corresponding nodes in the two augmented views of a graph are seen as positive pairs while all the other node pairs are negative ones. Global-to-global mode is often used when there are multiple graphs, and contrastive objects are the global representations of augmented views. In this mode, augmented views of the same graph are positives and all the other graph pairs are negatives. For the global-to-local perspective, positive pairs are taken as the global representation and nodes of augmented views for the corresponding graph, and negative pairs are the global representation and nodes of augmented views for other graphs.

In previous sections, we investigate the GCL methods with L-L or G-G modes, and the G-L mode on the graph classification (like InfoGraph). In this section, we discuss two methods of the G-L mode on node classification task: DGI [45] and MVGRL [13]. For experiments, we use the same settings as in Section 4. As seen from Table 11, there is an obvious degeneration in accuracy when no positive samples or negative samples are used, which is close to the no training setting. Recall that we find the positive samples are not needed in Section 4, and the observations on DGI and MVGRL seem to contradict our arguments. Here we attribute the inconsistency to the flaw in the methods themselves.

We start with an intriguing finding on DGI. Here we disorder the contrastive correspondence with a wrong view as global representations. Specifically, we take the local representation of the graph and its global representation as *negatives*, while local representations and global representations of the corrupted view are seen as *positives*. Note that the corruption operation in DGI is used to generate negative samples by shuffling rows of node attributes. See Figure 5 for illustration. We compare the disordered version with the original DGI in Table 12, and find using a wrong view as global representations does not affect performance. It implies that global representations lose efficacy in this framework. Inspired by Zheng et al. [60], we compare the two global representations and find they are nearly identical with every dimension being about 0.5. Extensive experiments also show the global representation is a constant vector for inappropriate usage of the Sigmoid function in both DGI and MVGRL [60].

This finding explains why the loss without positive samples does not work. Trained with such loss, node representations are only enforced to be far away from a constant vector, which gives no semantic guarantee. However, after adding positive samples to loss, the model learns to pull positive samples near a constant vector, while pushing negative samples away from such vector. It intrinsically achieves the goal of contrastive learning by gathering positives and repulsing negatives simultaneously. Thus the model trained with both positive and negative samples can obtain satisfying performance, explaining why DGI works with constant global representations.

Table 11: Test accuracy (%) of node classification benchmarks using DGI and MVGRL methods. We compare the performances of models trained with JSD loss (Contrast), loss part only involving negative pairs (NO Pos), loss only involving positive pairs (NO Neg), and no optimization objective (NO Training). Mean accuracy with standard derivation is reported after 10 runs. We conduct significance testing using Wilcoxon Signed Rank Test [48], comparing the contrastive loss with other loss types. The p-value is averaged across datasets. A value below 0.05 denotes significant accuracy difference (red), while a value above 0.05 indicates insignificance (green).

Method	**Loss**	Cora	CiteSeer	PubMed	Photo	Computers	Chameleon	Squirrel	**Avg**	**Avg p-value**
DGI	Contrast	83.38 ± 2.67	72.07 ± 2.37	84.77 ± 0.71	88.10 ± 1.81	83.35 ± 0.71	39.56 ± 2.86	34.55 ± 0.88	69.40	-
	NO Training	69.78 ± 3.39	55.15 ± 2.09	79.56 ± 1.35	69.08 ± 3.30	56.03 ± 1.97	31.44 ± 1.70	24.57 ± 1.22	55.09	0.0020
	NO Pos	66.84 ± 3.54	54.79 ± 3.33	78.25 ± 0.99	58.34 ± 3.92	71.98 ± 1.38	35.81 ± 2.34	26.99 ± 0.20	56.14	0.0020
	NO Neg	67.35 ± 4.61	58.17 ± 2.57	77.23 ± 1.05	62.75 ± 3.75	72.66 ± 1.48	31.62 ± 4.06	27.75 ± 1.85	56.79	0.0022
MVGRL	Contrast	84.41 ± 1.44	75.27 ± 0.79	85.62 ± 0.63	89.23 ± 1.52	79.58 ± 0.15	42.45 ± 2.43	33.97 ± 2.54	70.08	-
	NO Training	77.94 ± 2.23	58.92 ± 2.88	82.13 ± 0.63	81.15 ± 3.25	69.07 ± 0.40	32.23 ± 1.94	24.41 ± 1.10	60.84	0.0022
	NO Pos	75.44 ± 1.42	61.08 ± 2.48	81.26 ± 1.30	36.03 ± 1.57	38.36 ± 0.55	36.86 ± 2.56	29.98 ± 1.52	51.29	0.0020
	NO Neg	54.93 ± 4.67	35.03 ± 5.20	56.26 ± 1.91	36.47 ± 2.37	38.36 ± 0.56	29.34 ± 2.04	28.06 ± 1.66	39.78	0.0020

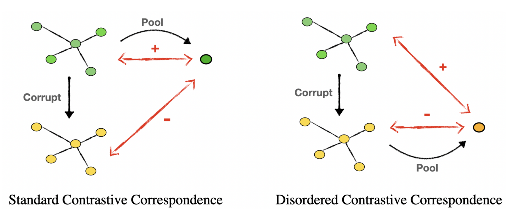


Figure 5: Illustration for disordering contrastive correspondence of views on DGI.

Table 12: Test accuracy (%) of DGI in standard contrastive correspondence (Std) and disordered correspondence (Dis).

Method	Contrast	Cora	CiteSeer	PubMed
DGI	Std.	83.38 ± 2.68	72.07 ± 2.37	84.77 ± 0.71
	Dis.	83.35 ± 2.68	72.04 ± 2.17	84.70 ± 0.68

H Results of the Fine-tuning Protocol

In this section, we provide the fine-tuning protocol results of main experiments of our paper. Specifically, we add a linear classification head after the encoder. In the fine-tuning phase, we fine-tune the whole networks according to downstream tasks, with the learning rate selected from $[0.01, 0.001]$ and the number of epochs selected from $[100, 200, 500]$. In Table 13 and Table 14, we report the fine-tuning results for the node classification task with GRACE and DGI methods, and for the graph classification tasks with GraphCL method, respectively. Sharing the same conclusion as the linear probing protocol, only using negative samples achieves comparable performance as that using contrastive objectives. On the other hand, for the node classification task, only using positive samples escapes severe collapse. We think the guidance of true labels in the fine-tuning helps the networks relearn parameters and thus prevents collapse.

Furthermore, we report the fine-tuning results about augmentations in Table 15. For the default augmentations (FM+PE), we set the ratio of each augmentation to 0.2 to save engineering effort. For a fair comparison, the standard deviation σ of the random Gaussian noise is fixed to 1e-4. Other hyperparameters are the same across the three augmentation settings (FM+PE, Gaussian, and NO Aug). As seen from the table, in the fine-tuning evaluation setting, random noise augmentation is on average the best for each loss type. It further justifies our analysis that domain-agnostic augmentations are enough for GCL.

Table 13: Fine-tuning accuracy (%) of node classification benchmarks using GCL methods. We compare the performances of models trained with InfoNCE loss (Contrast), uniformity loss (NO Pos), alignment loss (NO Neg), and no optimization objective (NO Training). Mean accuracy with standard derivation is reported after 10 runs. Average accuracy across datasets is reported. We conduct significance testing using Wilcoxon Signed Rank Test [48], comparing the contrastive loss and other loss types. The p-value is averaged across datasets. A value below 0.05 denotes significant accuracy difference (red), while a value above 0.05 denotes insignificance (green).

Method	**Loss**	Cora	CiteSeer	PubMed	Photo	Computers	Chameleon	Squirrel	**Avg**	**Avg p-value**
GRACE	Contrast	85.15 ± 3.07	74.19 ± 3.66	84.64 ± 2.47	92.89 ± 0.56	88.92 ± 1.27	39.56 ± 3.76	33.13 ± 4.33	71.21	-
	NO Pos	84.49 ± 3.50	74.07 ± 3.92	82.38 ± 2.36	92.84 ± 0.51	89.45 ± 1.14	38.60 ± 3.99	31.40 ± 3.76	70.46	0.3139
	NO Neg	81.62 ± 4.05	69.52 ± 4.46	83.87 ± 2.65	92.05 ± 0.89	89.07 ± 1.03	35.98 ± 5.13	30.48 ± 2.54	68.94	0.1398
DGI	Contrast	85.66 ± 2.39	74.55 ± 1.68	85.69 ± 0.26	92.94 ± 0.88	90.03 ± 0.79	42.01 ± 6.07	32.74 ± 5.49	71.95	-
	NO Pos	86.91 ± 2.16	74.79 ± 0.92	85.49 ± 0.25	92.73 ± 0.69	89.45 ± 0.65	42.01 ± 6.05	31.98 ± 5.73	71.91	0.4395
	NO Neg	85.00 ± 2.39	74.97 ± 1.08	85.44 ± 0.38	92.73 ± 0.67	90.13 ± 0.66	42.97 ± 6.52	32.13 ± 5.24	71.91	0.3672

Table 14: Fine-tuning accuracy (%) of graph classification benchmarks using GCL methods. We compare the performances of models trained with InfoNCE loss (Contrast), uniformity loss (NO Pos), alignment loss (NO Neg), and no optimization objective (NO Training). Mean accuracy with standard derivation is reported after 10 runs. Average accuracy across datasets is reported. We conduct significance testing using Wilcoxon Signed Rank Test [48], comparing the contrastive loss and other loss types. The p-value is averaged across datasets. A value below 0.05 denotes significant accuracy difference (red), while a value above 0.05 denotes insignificance (green).

Method	**Loss**	MUTAG	PTC-MR	PROTEINS	IMDB-BINARY	IMDB-MULTI	REDDIT-BINARY	**Avg**	**Avg p-value**
GraphCL	Contrast	93.48 ± 2.52	80.64 ± 4.09	79.88 ± 0.43	64.53 ± 1.32	43.64 ± 0.63	79.67 ± 1.82	73.64	-
	NO Pos	93.12 ± 0.74	80.45 ± 4.85	79.34 ± 3.10	63.13 ± 1.55	42.24 ± 0.31	76.73 ± 4.23	72.50	0.1966
	NO Neg	93.11 ± 1.14	80.36 ± 3.64	79.01 ± 3.69	62.37 ± 2.81	41.60 ± 0.47	76.75 ± 2.90	72.20	0.1400

Table 15: Fine-tuning accuracy (%) of node classification benchmarks using GRACE method with different augmentations under three loss settings. We compare no augmentations (NO Aug), domain-agnostic augmentations (Gaussian), and default domain-specific augmentations (FM+EP). Average accuracy and p-value are reported. We conduct significance testing using Wilcoxon Signed Rank Test [48], comparing the default augmentation with other settings. The p-value is averaged across datasets. A value below 0.05 denotes significant accuracy difference (red), while a value above 0.05 indicates insignificance (green).

Loss	**Aug**	Cora	CiteSeer	PubMed	Photo	Computers	Chameleon	Squirrel	**Avg**	**Avg p-value**
Contrast	FM+EP	85.15 ± 3.07	74.19 ± 3.66	84.64 ± 2.47	92.89 ± 0.56	88.92 ± 1.27	39.56 ± 3.76	33.13 ± 4.33	71.21	-
	Gaussian	86.69 ± 2.39	74.91 ± 2.98	84.52 ± 2.05	92.94 ± 1.02	88.94 ± 1.14	42.62 ± 6.55	31.55 ± 4.64	71.74	0.2829
	NO Aug	85.00 ± 3.20	74.07 ± 3.92	82.64 ± 2.69	93.10 ± 0.42	89.58 ± 1.20	37.03 ± 4.28	31.59 ± 4.49	70.43	0.2531
NO Pos	FM+EP	84.49 ± 3.50	74.07 ± 3.92	82.38 ± 2.36	92.84 ± 0.51	89.45 ± 1.14	38.60 ± 3.99	31.40 ± 3.76	70.46	-
	Gaussian	86.40 ± 2.84	74.67 ± 3.90	83.95 ± 1.72	92.73 ± 1.52	88.72 ± 1.18	40.79 ± 6.03	30.17 ± 4.73	71.06	0.2609
	NO Aug	85.00 ± 3.20	74.07 ± 3.92	82.42 ± 2.57	92.97 ± 0.58	89.49 ± 1.10	38.43 ± 3.88	31.59 ± 4.48	70.57	0.3859
NO Neg	FM+EP	81.62 ± 4.05	69.52 ± 4.46	83.87 ± 2.65	92.05 ± 0.89	89.07 ± 1.03	35.98 ± 5.13	30.48 ± 2.54	68.94	-
	Gaussian	84.26 ± 2.80	72.46 ± 4.75	84.49 ± 1.97	91.56 ± 1.75	88.37 ± 1.73	38.25 ± 2.54	28.25 ± 2.81	69.66	0.3273
	NO Aug	80.96 ± 5.24	71.38 ± 5.59	82.45 ± 2.69	92.03 ± 2.12	86.16 ± 5.95	33.97 ± 4.41	26.76 ± 2.49	67.67	0.2854

I Extensive Experiments of ContraNorm in GCL methods

In Table 5, we show that by simply incorporating the normalization layer into the encoder, the collapse issue can be rooted out for the GRACE method. In this section, we incorporate ContraNorm into multiple GCL methods under the no-negative setting. The results are shown in Table 16. It is obvious that for these GCL methods, applying ContraNorm when there are no negative samples achieves comparable performance with models trained with the contrastive loss (both positive and negative samples). The extensive experiments validate the effectiveness of ContraNorm across different GCL methods.

Table 16: Test accuracy (%) of node classification benchmarks using GCL methods. We compare the performances of models trained with InfoNCE loss (Contrast), alignment loss (NO Neg), and alignment loss with ContraNorm in encoders (GCN+CN). Mean accuracy with standard derivation is reported after 10 runs. Average accuracy across datasets is reported. We conduct significance testing using Wilcoxon Signed Rank Test [48], comparing the default setting (first line) with others. The p-value is averaged across datasets. A value below 0.05 denotes significant accuracy difference (red), while a value above 0.05 indicates insignificance (green). OOM denotes out of memory.

Method	**Loss**	**Encoder**	Cora	CiteSeer	PubMed	Photo	Computers	**Avg**	**Avg p-value**
GRACE	Contrast	GCN	84.67 ± 1.39	73.47 ± 2.32	85.80 ± 0.16	91.42 ± 1.27	89.01 ± 0.60	84.87	-
	NO Neg	GCN	29.85 ± 1.45	20.42 ± 2.26	39.63 ± 0.81	25.10 ± 1.74	36.84 ± 1.30	30.37	0.0020
	NO Neg	GCN + CN	82.35 ± 2.28	72.25 ± 1.86	83.30 ± 0.63	92.43 ± 0.82	84.48 ± 1.01	82.96	0.1520
GCA	Contrast	GCN	84.04 ± 1.55	72.63 ± 2.68	85.92 ± 0.69	93.07 ± 0.66	86.58 ± 0.75	84.45	-
	NO Neg	GCN	31.40 ± 3.61	22.16 ± 3.01	39.58 ± 0.83	28.13 ± 1.14	37.34 ± 0.95	31.72	0.0020
	NO Neg	GCN + CN	82.21 ± 1.29	72.87 ± 0.98	82.40 ± 0.78	92.47 ± 0.96	86.15 ± 0.58	83.22	0.2125
ProGCL	Contrast	GCN	85.42 ± 3.41	72.85 ± 2.99	OOM	93.81 ± 0.48	86.35 ± 1.28	84.61	-
	NO Neg	GCN	30.15 ± 2.70	21.08 ± 1.45	21.13 ± 1.20	4.88 ± 0.33	3.11 ± 0.65	16.07	0.0020
	NO Neg	GCN + CN	80.00 ± 1.75	73.35 ± 1.17	84.02 ± 0.91	93.59 ± 0.38	85.67 ± 0.43	83.33	0.2336

J Gaussian Augmentations under Different Loss Settings

In Section 6, we perform experiments using the GRACE method with different augmentations under the InfoNCE loss. Here, we further report results under different losses in Table 17. For loss without negative samples, the average performance gap between domain-specific augmentations and noise augmentations is only 0.74%. When no augmentations, the performance drops 4.88%. We conjecture that when no negative samples exist, the application of augmentations brings diversity in representations, thus making collapse more difficult. For contrastive loss and loss without positive samples, the gap between domain-specific augmentations and noise augmentations is also narrow.

Table 17: Test accuracy (%) of node classification benchmarks using GRACE method with different augmentations under three loss settings. We compare no augmentations (NO Aug), domain-agnostic augmentations (Gaussian), and default domain-specific augmentations (FM+EP). Average accuracy and p-value are reported. We conduct significance testing using Wilcoxon Signed Rank Test [48], comparing the default augmentation with other settings. The p-value is averaged across datasets. A value below 0.05 denotes significant accuracy difference (red), while a value above 0.05 indicates insignificance (green).

Loss	**Encoder**	**Aug**	Cora	CiteSeer	PubMed	Photo	Computers	**Avg**	**Avg p-value**
Contrast	GCN	FM+EP	84.67 ± 1.39	73.47 ± 2.32	85.80 ± 0.16	91.42 ± 1.27	89.01 ± 0.60	84.87	-
		Gaussian	82.72 ± 2.38	72.60 ± 1.21	85.24 ± 0.61	91.32 ± 1.37	82.77 ± 1.09	82.93	0.1816
		NO Aug	79.56 ± 2.18	71.83 ± 1.83	84.68 ± 0.58	90.99 ± 1.26	82.83 ± 0.86	81.98	0.1008
NO Pos	GCN	FM+EP	82.65 ± 1.18	73.50 ± 2.41	85.28 ± 0.79	91.32 ± 0.10	84.40 ± 0.43	83.43	-
		Gaussian	80.04 ± 1.93	70.84 ± 1.85	84.88 ± 0.89	91.33 ± 1.18	83.26 ± 1.24	82.07	0.1840
		NO Aug	79.37 ± 2.30	71.80 ± 1.84	84.69 ± 0.63	90.92 ± 1.21	82.49 ± 0.87	81.85	0.1176
NO Neg	GCN + CN	FM+EP	82.35 ± 2.28	72.25 ± 1.86	83.30 ± 0.63	92.43 ± 0.82	84.48 ± 1.01	82.96	-
		Gaussian	79.08 ± 2.47	72.43 ± 1.32	83.55 ± 0.22	91.59 ± 1.19	84.48 ± 1.07	82.23	0.2750
		NO Aug	75.59 ± 3.45	66.98 ± 3.40	82.14 ± 1.28	81.91 ± 1.42	83.79 ± 1.14	78.08	0.0688