Pattern Recognition Letters 181 (2024) 43–49

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Graph contrastive learning with consistency regularization

Soohong Lee [a,b], Sangho Lee [a,b], Jaehwan Lee [a,b], Woojin Lee [c,*], Youngdoo Son [a,b,**]

[a] Department of Industrial and Systems Engineering, Dongguk University - Seoul, Seoul 04620, South Korea
[b] Data Science Laboratory (DSLAB), Dongguk University – Seoul, Seoul 04620, South Korea
[c] School of AI Convergence, Dongguk University - Seoul, Seoul 04620, South Korea

ARTICLE INFO

Editor: Jiwen Lu

MSC:
41A05
41A10
65D05
65D17

Keywords:
Contrastive learning
Class collision
Consistency regularization
Graph representation learning
Graph neural network

ABSTRACT

Contrastive learning has actively been used for unsupervised graph representation learning owing to its success in computer vision. Most graph contrastive learning methods use instance discrimination. It treats each instance as a distinct class against a query instance as the pretext task. However, such methods inevitably cause a class collision problem because some instances may belong to the same class as the query. Thus, the similarity shared through instances from the same class cannot be reflected in the pre-training stage. To address this problem, we propose graph contrastive learning with consistency regularization (GCCR), which introduces a consistency regularization term to graph contrastive learning. Unlike existing methods, GCCR can obtain a graph representation that reflects intra-class similarity by introducing a consistency regularization term. To verify the effectiveness of the proposed method, we performed extensive experiments and demonstrated that GCCR improved the quality of graph representations for most datasets. Notably, experimental results in various settings show that the proposed method can learn effective graph representations with better robustness against transformations than other state-of-the-art methods.

1. Introduction

Graphs have widely been used in various fields, such as social networks [1], mechanical properties of individual molecules [2], protein–protein interaction networks [3], and knowledge graphs [4]. Recently, several studies demonstrated that graph neural networks (GNNs) are proficient in handling graph-structured data [5,6]. However, similar to other neural networks, GNNs require a large amount of labeled training data to achieve a decent performance, which might be unrealistic in real-world applications. Thus, GNN-based self-supervised learning has been studied to utilize unlabeled graph-structured data to redeem an insufficient number of labeled instances. Nevertheless, existing graph self-supervised learning methods [7,8] have limitations in that they might overemphasize the proximity information [9] and damage the structural information of a graph [10]. To address these limitations, several studies applied contrastive learning, which has attracted considerable attention in unsupervised learning of the computer vision domain [11], to the graph domain [12–14].

Given a query graph instance in a dataset, positive samples are generated by augmenting the query graph with different augmentation methods, while negative samples are defined as augmented samples of other graphs in the dataset. With the positive and negative samples, the objective of graph contrastive learning is to encourage the positive samples to pull each other while pushing apart the negative samples from the positive ones in the embedding space. However, since contrastive learning considers negative samples randomly from mini-batch, there exist *fake negative samples* that belong to the same class as the positive sample. The fake negative samples encourage the model to ignore semantic features, which are intra-class similarities. This phenomenon referred to as a *class collision* [15] or *sampling bias* [16], can lead to a significant performance drop. Fig. 1(a) illustrates the class collision problem with false negative samples. While several studies have attempted to mitigate the class collision problem in the computer vision domain [16–18], existing graph contrastive learning methods focus on improving model architectures [12] or developing data augmentation methods specialized for each dataset [19]. As a result, there have been limited works that consider the effects of fake negative samples in the graph domain.

In this paper, we propose graph contrastive learning with consistency regularization (GCCR) to address the class collision problem in the previous graph contrastive learning by introducing consistency regularization. As illustrated in Fig. 1(b), by using a novel loss function for consistency regularization, we can encourage fake negative samples

* Corresponding author.
** Corresponding author at: Department of Industrial and Systems Engineering, Dongguk University - Seoul, Seoul 04620, South Korea.
E-mail addresses: lsh0407@dgu.ac.kr (S. Lee), sangho218@dgu.ac.kr (S. Lee), andrew0411@dgu.ac.kr (J. Lee), wj926@dgu.ac.kr (W. Lee), youngdoo@dongguk.edu (Y. Son).

https://doi.org/10.1016/j.patrec.2024.03.014
Received 18 December 2022; Received in revised form 5 May 2023; Accepted 18 March 2024
Available online 21 March 2024

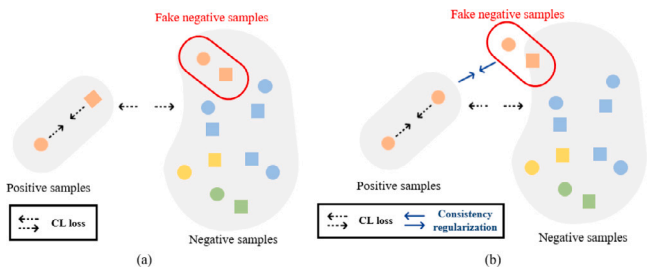

Fig. 1. Illustrations of (a) embedding space with class collision problem and (b) embedding space using consistency regularization.

to be placed nearby positive samples in the embedding space. Our contributions can be summarized as follows:

- We propose a novel graph contrastive learning method, GCCR, which can solve the class collision problem with a simple and efficient approach.
- The proposed method achieves state-of-the-art performance through extensive experiments on several datasets under various settings.
- By measuring the cohesion of the instances with the same class, we demonstrate that GCCR effectively derives the representations for the instances according to each class.

2. Related work

In the computer vision domain, contrastive learning has actively been studied in various ways to learn effective representations in a self-supervised manner. For example, Wu et al. [20] and He et al. [21] used previously obtained representations to calculate the contrastive loss for improving the representation quality. Chen et al. [22] introduced a normalized temperature-scare cross-entropy (NT-Xent) loss for memory-efficient training, and Zbontar et al. [23] presented a variant of contrastive loss that can reduce the redundancy of representations. Grill et al. [24] and Chen and He [25] introduced novel architectures allowing efficient and stable training with contrastive loss. Moreover, Caron et al. [26] used cluster assignments as the basis for contrastive learning to generate robust representations.

With the promising performance of contrastive learning in the computer vision domain, many studies have expanded contrastive learning to the graph domain. In general, graph contrastive learning learns one of two types of representations: *node-level* and *graph-level*.

On the one hand, graph contrastive learning for learning the node-level representations of graphs aims to capture the local structure and properties of each node suitable for several tasks, such as node classification and link prediction [14,27–30]. For example, Chu et al. [31] and Jin et al. [32] introduced curriculum contrastive learning and multi-scale contrastive siamese networks, respectively. Velickovic et al. [9] maximized mutual information between global graph representations and local patch representations to generate effective node-level representations. In addition, Xia et al. [33] presented a contrastive learning approach that leverages the progressiveness of graph data to address the high complexity and potential bias of the graph data. Thakoor et al. [34] combined multiple graph representations learned from different subgraphs of the original graph.

On the other hand, the graph-level representations, the goal of this paper, are useful for graph classification, where the objective is to assign a pre-defined label or category to an entire graph by capturing its global structure and properties [35]. Tong et al. [13] introduced a directed graph contrastive learning method to obtain useful graph-level representations, and Li et al. [36] combined contrastive learning and community detection to handle community structure in graphs. Recently, You et al. [12] proposed GraphCL, a graph contrastive learning method that achieved state-of-the-art performance in generating graph-level representations. GraphCL pre-trains a GNN encoder using NT-Xent loss [22] with graph augmentation methods and fine-tuned the encoder for a downstream task. However, most of the existing contrastive learning methods, including GraphCL, suffer from class collision problems because they usually utilize randomly selected negative samples and consider them as the same negative samples regardless of whether they are *real* or *fake*.

In the computer vision domain, to solve this problem, Chuang et al. [16] presented a debiased version of contrastive loss by indirectly approximating the distribution of negative samples, and Huynh et al. [18] proposed a strategy for finding a candidate of fake negative samples and eliminated them during the pre-training stage. Moreover, Tejankar et al. [17] updated the teacher–student network by minimizing the Kullback–Leibler (KL) divergence between two similarity distributions, which are similarities between two augmented query images and the anchor images rather than using the conventional contrastive loss function.

However, in the graph domain, there has been little effort in attempting to identify fake negatives. To the best of our knowledge, there have been two papers that deal with the class collision problem in graph contrastive learning. Zhao et al. [37] proposed a graph-debiased contrastive learning method where the clusters are constructed by predicting their class information with an auxiliary target distribution and alleviated the class collision problem by randomly selecting negative samples from those clusters. In addition, Lin et al. [38] mitigate class collision problems by constructing a set of negative examples for each class based on a clustering algorithm. However, these two approaches require the clustering process that causes an additional computational burden, and a performance drop can be induced when the clusters contain fake negative samples. In contrast, the proposed method introduces a consistency regularization to address the class collision problem; hence, the additional computation and performance drop are minimized.

3. Preliminaries

We briefly review the graph neural network and the graph contrastive learning that background of our proposed method.

3.1. Graph neural network

Let $G = (\mathcal{V}, \mathcal{E})$ denotes an undirected graph with node feature vector $\boldsymbol{X}_v$ for $v \in \mathcal{V}$, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of nodes and edges, respectively. The GNN can be represented as a mapping function $\boldsymbol{f} : G \rightarrow \mathbb{R}^D$ that transforms a graph G into a D-dimensional vector. GNNs broadly follow a recursive neighborhood aggregation scheme, where each node aggregates the feature vectors of its neighbors to compute a new feature vector [39]. Formally, the node representation $\boldsymbol{h}_v^l$ obtained through the propagation of the lth layer is

$$\boldsymbol{h}_v^l = Combine(\boldsymbol{h}_v^{l-1}, \boldsymbol{f}(\boldsymbol{h}_u^{l-1}, u \in Neighbor(v))), \tag{1}$$

where $Combine(\cdot)$ denotes a concatenation function such as a linear mapping in GraphSAGE [8] and an element-wise mean pooling in graph convolutional network [5], and $Neighbor(v)$ is the set of nodes connected to the node v in the graph. Next, a readout function aggregates the node features at the final kth layer to obtain the representation of the entire graph $\boldsymbol{h}_G$:

$$\boldsymbol{h}_G = Readout(\boldsymbol{h}_v^k, v \in G), \tag{2}$$

where $Readout(\cdot)$ can be any permutation invariant function, such as average or summation.

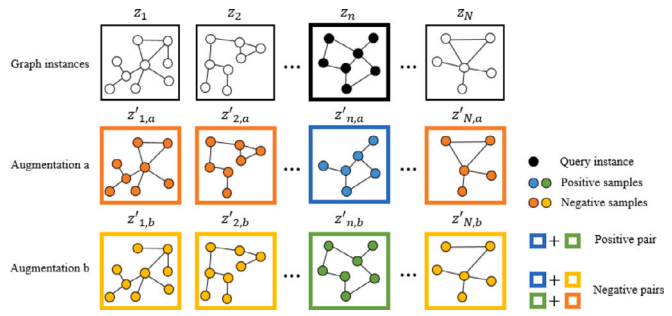

Fig. 2. Illustration of positive and negative pairs derived from positive and negative samples. Given a query graph colored black, the graphs colored blue and green indicate positive samples, while the graphs colored orange and yellow denote negative samples. A positive pair consists of blue and green graphs (positive samples). On the contrary, a negative pair consists of blue and yellow graphs or green and orange graphs (positive and negative samples).

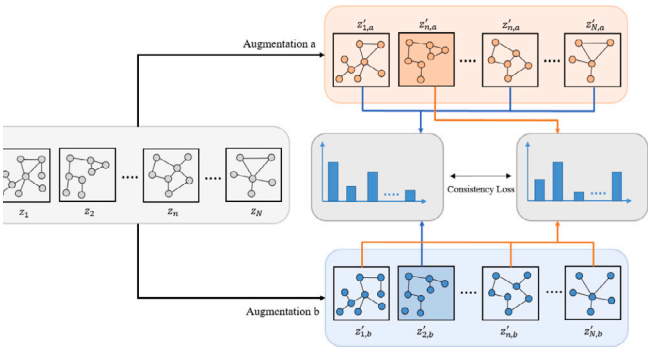

Fig. 3. Overview of GCCR. Graphs colored dark blue and dark orange refer to positive and negative samples generated from z_2. The distribution connected by a blue line represents the similarities between $z'_{2,a}$ and its negative samples, and that by an orange line describes the similarities between $z'_{2,b}$ and its negative samples. The consistency regularization forces two distributions to be similar.

3.2. Graph contrastive learning

Graph contrastive learning [12] is a GNN pre-training method that aims to learn perturbation-invariant representations for diverse graph-structured data.

In graph contrastive learning, a graph encoder is trained to maximize the agreement between the samples of a positive pair while minimizing it between the samples in negative pairs. In the pre-training stage, it samples a mini-batch with N graphs and generates $2N$ augmented graphs with two different graph augmentation methods. As shown in Fig. 2, $G'_{n,a}$ and $G'_{n,b}$ denote the augmented graphs with two different augmentation methods, a and b, for the nth graph G_n, respectively. When $G'_{n,a}$ passes through the graph encoder $\boldsymbol{f}(\cdot)$, which has a GNN architecture, the graph representation $\boldsymbol{h}_{G'_{n,a}}$ is obtained from Eq. (2). Then, a non-linear projection head $\boldsymbol{g}(\cdot)$ maps the representation $\boldsymbol{h}_{G'_{n,a}}$ to the embedding space where the contrastive loss is applied. Let $\boldsymbol{z}_{n,a} = \boldsymbol{g}(\boldsymbol{h}_{G'_{n,a}})$ be the representation of an augmented graph $G'_{n,a}$ in the embedding space. The representations of positive samples, which are augmented from the query graph G_n, are expressed as $\boldsymbol{z}_{n,a}$ and $\boldsymbol{z}_{n,b}$, and those of negative samples as $\boldsymbol{z}_{n',a}$ and $\boldsymbol{z}_{n',b}$ for all $n' \neq n$. Therefore, for all $n' \neq n$, we can represent the positive and negative pairs of the graph representations for an augmented graph $G'_{n,a}$ as $(\boldsymbol{z}_{n,a}, \boldsymbol{z}_{n,b})$ and $(\boldsymbol{z}_{n,a}, \boldsymbol{z}_{n',b})$, respectively. Finally, the contrastive loss function is applied to an augmented graph $G'_{n,a}$. For example, You et al. [12] employed NT-Xent loss, one of the most popular loss functions in contrastive learning, for an augmented graph $G'_{n,a}$ as follows:

$$\mathcal{L}_{CL} = -\frac{1}{N}\sum_{n=1}^{N} log \frac{exp(sim(\boldsymbol{z}_{n,a}, \boldsymbol{z}_{n,b})/\tau)}{\sum_{n'=1,n'\neq n}^{N} exp(sim(\boldsymbol{z}_{n,a}, \boldsymbol{z}_{n',b})/\tau)}, \tag{3}$$

where τ denotes the temperature hyperparameter and $sim(\cdot)$ is a similarity function, such as cosine similarity. The contrastive loss is computed across all positive pairs in the mini-batch.

4. Proposed method

Although the NT-Xent loss has shown promising results in representation learning, the one-hot similarity label used in the NT-Xent loss may not provide sufficient information, so it may not derive appropriate representations. As shown in Fig. 2, given a query instance G_n, positive samples, $G'_{n,a}$ and $G'_{n,b}$, are defined as the samples augmented from G_n, while negative samples, $G'_{n'\neq n,a}$ and $G'_{n'\neq n,b}$, are the samples augmented from other instances except G_n. For each graph instance, one positive pair consisting of two positive samples and several negative pairs containing positive and negative samples are derived. Among the negative samples, there exist fake negative samples, which have the same class as the query; however, NT-Xent loss treats them equally as *real negative samples* because of considering the similarity between any negative pairs as zero. Therefore, it is difficult to reflect the *intra-class* similarity between the samples in a negative pair with the same class labels. In other words, NT-Xent loss may occur the *class collision* problem.

To solve this problem, we propose a novel graph contrastive learning method, GCCR, which can reflect intra-class similarities between the samples in negative pairs, which have been ignored in the existing graph contrastive learning methods based on NT-Xent loss. Fig. 3 illustrates an overview of GCCR. Let $G'_{n,a}$ be an augmented positive sample and $G'_{n',b}$ for $n' \in \{1, 2, 3, \ldots, N\}$ and $n' \neq n$ be the negative samples, where $G'_{n,a}$ and $G'_{n',b}$ are obtained by using an augmentation method $\mathcal{A}_a$ and another augmentation method $\mathcal{A}_b$, respectively. We define a similarity distribution between $G'_{n,a}$ and $G'_{n',b}$ as follows:

$$P(n'|n) = \frac{exp(sim(\boldsymbol{z}_{n,a}, \boldsymbol{z}_{n',b})/\tau)}{\sum_{k=1}^{N} exp(sim(\boldsymbol{z}_{n,a}, \boldsymbol{z}_{k,b})/\tau)}, \tag{4}$$

where $z_{i,j}$ denotes the representation of the augmented graph $G'_{i,j}$. Also, the similarity distribution between $G'_{n,b}$ and its negative samples $G'_{n',a}$ is defined as

$$Q(n'|n) = \frac{exp(sim(\boldsymbol{z}_{n,b}, \boldsymbol{z}_{n',a})/\tau)}{\sum_{k=1}^{N} exp(sim(\boldsymbol{z}_{n,b}, \boldsymbol{z}_{k,a})/\tau)}. \tag{5}$$

Then, we define the consistency regularization term for graph contrastive learning as the symmetric KL-divergence of two similarity distributions P and Q as

$$\mathcal{L}_{CR,n} = \frac{1}{2}D_{KL}(P \parallel Q) + \frac{1}{2}D_{KL}(Q \parallel P), \tag{6}$$

following the concept of consistency regularization in [40], which has shown significant performance improvements in many previous studies for semi-supervised learning [41]. Finally, our loss function derived from (3) and (6) is as follows:

$$\mathcal{L} = \mathcal{L}_{CL} + \alpha\mathcal{L}_{CR}, \tag{7}$$

where α denotes a coefficient that balances the two loss functions, and $\mathcal{L}_{CR} = \frac{1}{N}\sum_{n=1}^{N}\mathcal{L}_{CR,n}$.

Consistency regularization generates pseudo labels for unlabeled data with the assumption that a good model should derive similar results on augmented views of the same data. Thus, for graph contrastive learning, we assume that similarities between each of the positive samples generated by two different augmentation methods from the same query graph and its negative samples should be similar. In GCCR, we add the symmetric KL-divergence as a consistency regularization term to the original graph contrastive learning loss function; thereby, those two similarities can serve as pseudo labels of each other. In other words, $Q(n'|n)$ serves as pseudo-labels of the similarities between $G'_{n,a}$

Algorithm 1 Graph contrastive learning with consistency regularization (GCCR)

Input: Mini-batch of graph instances $\mathcal{G} = \{G_1, \cdots, G_N\}$, two augmentation methods $\mathcal{A}_a$ and $\mathcal{A}_b$, initial graph encoder $f(\cdot)$ and non-linear projection head $g(\cdot)$

Output: Updated $f(\cdot)$ and $g(\cdot)$

for $n = 1, 2, \cdots, N$ **do**
  Get $G'_{n,a}$ and $G'_{n,b}$ by applying $\mathcal{A}_a$ and $\mathcal{A}_b$ to G_n
  $z_{n,a} \leftarrow g(f(G'_{n,a})), z_{n,b} \leftarrow g(f(G'_{n,b}))$
  Compute P and Q by Eq. (4) and Eq. (5)
  Compute $\mathcal{L}_{CR,n}$ by Eq. (6)
end for
Compute $\mathcal{L}_{CL}$ by Eq. (3)
$\mathcal{L}_{CR} = \frac{1}{N}\sum_{n=1}^{N} \mathcal{L}_{CR,n}$
$\mathcal{L} = \mathcal{L}_{CL} + \alpha\mathcal{L}_{CR}$
Update $f(\cdot)$ and $g(\cdot)$ with $\mathcal{L}$

and its negative samples; likewise, $P(n'|n)$ serves as pseudo-labels of the similarities between $G'_{n,b}$ and its negatives.

In general, for pseudo labeling in semi-supervised learning, an unlabeled instance is assigned a label of the class with the highest probability among the predictions for it using the model trained with the labeled data. By contrast, pseudo labels of the proposed method are the similarities between unlabeled instances and their augmented views. Although this approach assigns a label to an unlabeled instance in a different way from the pseudo labeling in semi-supervised learning, they serve the same role in allowing unlabeled data to train the model in a supervised manner; in addition, we can improve the model's generalization ability by pseudo-labeling with augmentation. Therefore, pseudo-labeling with the similarity distributions used in GCCR can address the class collision problem and derive superior representations by utilizing the intra-class similarity overlooked in the previous contrastive learning methods. In addition, our method achieves promising results with a simple and inexpensive approach, which only requires one additional regularization term in the process of calculating loss.

The algorithm of GCCR is summarized in Algorithm 1.

5. Experiments

To verify the effectiveness of GCCR, we performed extensive experiments under various settings with several datasets from TU dataset [42]. First, we evaluated the effectiveness of the generated representations for classification under semi-supervised and unsupervised settings. Then, we compared our method to GraphCL to measure the robustness against transformations that are augmentations not used in the pre-training. Finally, we quantitatively checked whether the representations obtained from GCCR well-reflect the intra-class similarity.

We performed Wilcoxon signed rank test [43] between GCCR and the second-best method in terms of average performance to validate the experiment results statistically. By the statistical significance test with a significance level of 0.1, we confirmed that the performance improvement of our method was statistically significant in all experiments.

5.1. Graph augmentation methods

Graph augmentation should create a novel graph instance that is different from the original graph without damaging its semantic properties. Unlike images, graph data is extremely sensitive to the selection of the augmentation method owing to its rich structured information and various contexts. In this paper, we adopted the four graph augmentation methods introduced from [12] for a fair comparison. We also assume that all the augmentation methods should not harm the semantic properties of the graph as in [12].

Table 1
Basic statistics of datasets.

Datasets	Category	# of graphs	Avg. nodes	Avg. edges
NCI1	Molecules	4110	29.87	32.30
PROTEINS	Molecules	1113	39.60	72.87
MUTAG	Molecules	188	17.93	19.79
DD	Bioinformatics	1178	84.32	715.56
IMDB-B	Social networks	1000	19.77	96.53
COLLAB	Social networks	5000	74.49	2457.78
RDT-B	Social networks	2000	508.52	497.75
RDT-M-5K	Social networks	4999	508.52	594.87
GITHUB	Social networks	12 725	113.79	234.64
MNIST	Superpixel graphs	70 000	70.57	8
CIFAR10	Superpixel graphs	60 000	117.63	234.64

- *Node dropping* is a graph structure transformation method that discards a certain percentage of nodes from a graph. When a node is determined to be dropped, its neighboring edges are also discarded.
- *Edge perturbation* is a graph structure transformation method that removes or adds a certain percentage of edges by inverting elements in the adjacency matrix, which represents the connectivity in the graph.
- *Subgraph* augmentation is a sampling transformation method that samples a subgraph of the graph. Among several subgraph-generating algorithms, the random walk method is used in this paper.
- *Attribute masking* is a graph feature transformation method that masks a certain percentage of node attributes.

5.2. Datasets and settings

We employ eight datasets, including NCI 1, PROTEINS, MUTAG, DD, IMDB-B, COLLAB, RDT-B, RDT-M-5K, GITHUB, MNIST, and CIFAR10, from the TU dataset [42] to evaluate the performances of GCCR under the various settings. Detailed information about the dataset is provided in Table 1.

For all experiments in the pre-training stage, we set the batch size to 128, and the hyperparameters, including learning rate, maximum epoch, augmentation strength, and α, were determined by grid search.

5.3. Semi-supervised learning

We first evaluated our method using a graph classification task in the semi-supervised learning setting. After pre-training without label information, supervised learning was performed by assigning labels to 10% of the dataset. We evaluated the model using 10-fold cross-validation; we repeated the experiments five times to reduce the randomness; the averages of the results are reported.

Baseline and *Aug* denote the performance of training from scratch and that with only augmentation without contrastive learning, respectively. In addition, we applied two conventional graph representation learning methods, *GAE* and *Infograph*, for comparison. GAE [7] reconstructed the adjacency information based on a variational autoencoder. Infograph [35] maximized the mutual information between the substructure and entire graph representations. We adopted graph convolutional network (GCN) as the encoder, $f(\cdot)$, with the default setting in [12,44]. Then, we applied a two layers perceptron as a projection head, $g(\cdot)$, to obtain the representation.

As shown in Table 2, the proposed GCCR showed the best performance in five of the nine datasets; also, our method achieved the second-best performance for the remaining four datasets even without additional computational load. Specifically, compared with GraphCL, GCCR showed better performance in eight out of nine datasets. In the case of CIFAR10, our method achieved about 3%p performance improvement over GraphCL. Furthermore, to verify the strength of the

Table 2
Semi-supervised learning results of the proposed method and comparison methods. GraphCL, GCCR (NN), and GCCR are evaluated in the same experiment setting, and the other results are referred to [12]. GCCR (NN) is GCCR using non-augmented negative samples when calculating the similarity distribution. The best and second-best performances are boldfaced and underlined, respectively.

Datasets	NCI1	PROTEINS	DD	COLLAB	RDT-B	RDT-M-5K	GITHUB	MNIST	CIFAR10
Baseline	73.72	70.40	73.56	73.71	86.63	51.33	60.87	70.71	35.78
Aug	73.59	70.29	74.30	74.19	87.74	52.01	60.91	83.99	34.24
GAE	74.36	70.51	74.54	75.09	87.69	53.58	63.89	86.67	36.35
Infograph	**74.86**	72.27	75.78	73.76	**88.66**	**53.61**	65.21	83.34	41.07
GraphCL	74.23	**74.22**	76.51	74.53	88.12	52.95	65.27	91.91	41.20
GCCR (NN)	73.60	72.50	77.05	74.82	88.10	53.03	65.21	92.15	42.70
GCCR	74.51	73.32	**77.21**	**75.14**	88.43	53.41	**65.42**	**92.95**	**44.78**

Table 3
Unsupervised learning results of the proposed method and the comparison methods. GraphCL, GCCR (NN), and GCCR are evaluated in the same experiment setting, and the other results are referred to [12]; thus, results that were not reported in [12] were blanked. GCCR (NN) is GCCR using non-augmented negative samples when calculating the similarity distribution. The best and second-best performances are boldfaced and underlined, respectively.

Datasets	NCI1	DD	MUTAG	COLLAB	RDT-B	RDT-M-5K	IMDB-B
GL	–	–	81.66	–	77.34	41.01	65.87
WL	80.01	72.92	80.72	–	62.82	46.06	72.30
DGK	**80.31**	73.30	87.44	–	78.04	41.27	66.96
node2vec	54.89	–	72.63	–	–	–	–
sub2vec	52.84	–	61.05	–	71.48	36.68	55.26
graph2vec	73.22	–	83.15	–	75.78	47.86	71.10
InfoGraph	76.20	72.85	89.01	70.65	82.50	53.46	**73.03**
PGCL	77.96	75.32	88.58	**74.53**	**91.15**	55.91	72.31
GraphCL	79.55	78.48	89.24	71.33	90.70	55.79	71.38
GCCR (NN)	79.50	78.52	88.90	71.29	90.66	55.34	71.10
GCCR	79.72	**78.68**	**89.67**	72.23	90.93	**56.05**	71.14

Table 4
Robustness of semi-supervised learning on transformed data for GCCR and GraphCL. The best results are boldfaced. Transformation is an augmentation method that has not been used in the pre-training stage. The transformations were applied to the test data. *maskN* and *permE* denote attribute masking and edge perturbation, respectively.

Datasets	COLLAB		NCI1		DD		PROTEINS	
Transformation	*maskN*	*permE*	*maskN*	*permE*	*maskN*	*permE*	*subgraph*	*permE*
GraphCL	54.70	54.32	66.67	66.69	78.36	78.35	71.04	71.24
GCCR	**56.01**	**55.04**	**67.07**	**67.02**	**78.82**	**79.01**	**72.01**	**72.66**

regularization term of the proposed method, we derived the results of GCCR (NN), which is all the same as GCCR except that it uses non-augmented negative samples to calculate the similarity distribution as in [17]. Since GCCR outperforms GCCR (NN) for all datasets, we demonstrated that calculating the similarities with the representations of augmented negative samples effectively reflects the intra-class similarities and leads to performance improvement.

5.4. Unsupervised learning

We evaluated our method on graph classification in an unsupervised learning setting, where the graph representations generated by unsupervised methods were fed to a downstream support vector machine classifier without any additional refinement. In addition, we additionally compared our method with Graphlet kernel *(GL)* [45], Weisfeiler–Lehman sub-tree kernel *(WL)* [46], and deep graph kernel *(DGK)*, which are the kernel methods that perform well in graph unsupervised learning; *node2vec* [47], *sub2vec* [48], and *graph2vec* [49], which are the graph-level unsupervised learning methods to train neural networks for vectorizing nodes, subgraphs, and graphs, respectively; PGCL [38], which is recently outperformed GraphCL by mitigating class collision problem in graph contrastive learning, were also selected. In this experiment, a GIN [6] architecture was adopted as a GNN encoder with the default setting in [35] to generate a graph representation.

Table 3 shows that GCCR was superior to GraphCL in all datasets except IMDB-B, and achieved the best performance in five out of seven datasets. Specifically, for a large dataset, such as COLLAB, GCCR improved the performance by 1% over GraphCL. Moreover, GCCR outperforms GCCR (NN) for all datasets, as the same in the semi-supervised results.

5.5. Relative in-class cohesion

GCCR solves the class collision problem by reflecting the intra-class similarities between the negative samples during the training phase. Therefore, GCCR should well-aggregate the representations of the instances belonging to the same class while detaching those from the different classes.

To verify this argument, we compute the average distance of the instances belonging to the same class c, $d_{intra}(c)$, as follows:

$$d_{intra}(c) = \frac{1}{N_c^2} \sum_{n,m=1}^{N_c} d(\boldsymbol{z}_n^{(c)}, \boldsymbol{z}_m^{(c)}), \tag{8}$$

where $d(\cdot)$ is a distance function, N_c denotes the number of instances in class c, $\boldsymbol{z}_n^{(c)}$ and $\boldsymbol{z}_m^{(c)}$ denote the representations of two different graphs, G_n and G_m, belonging to the class c, obtained from the graph encoder. Then, we define the inter-class distance for the class c, $d_{inter}(c)$, as the average of the distances between every pair of an instance in class c and that in other classes.

$$d_{inter}(c) = \frac{1}{N_c(N - N_c)} \sum_{k \neq c} \sum_{m=1}^{N_k} \sum_{n=1}^{N_c} d(\boldsymbol{z}_n^{(c)}, \boldsymbol{z}_m^{(k)}), \tag{9}$$

where N denotes the number of instances in a dataset, N_c and N_k are the numbers of instances in the classes c and k, and $\boldsymbol{z}_n^{(c)}$ and $\boldsymbol{z}_m^{(c)}$ denote the representations of two graphs, G_n and G_m, belonging to the classes c and k, respectively. Finally, we introduce a novel metric, relative in-class cohesion (RIC), quantifying how closely the graph representations belonging to the same class are located, as follows:

$$RIC(c) = \frac{d_{intra}(c)}{d_{inter}(c)}. \tag{10}$$

The smaller the RIC value, the higher the cohesion of representations in the same class.

As shown in Fig. 4, GCCR showed lower RIC values compared to Infograph in all classes, as well as compared to GraphCL in all classes except C3 in the COLLAB dataset. Thus, we demonstrate that the representations of each class generated by GCCR are better aggregated in embedding space than those of GraphCL and Infograph.

5.6. Robustness to transformations

Finally, we considered the robustness to transformations, which are augmentation methods not used in the pre-training stage, of GCCR. In real-world problems, an abundance of transformed data causes deep learning networks to misclassify the data. Therefore, it is important to find robust representations against perturbated data [50].

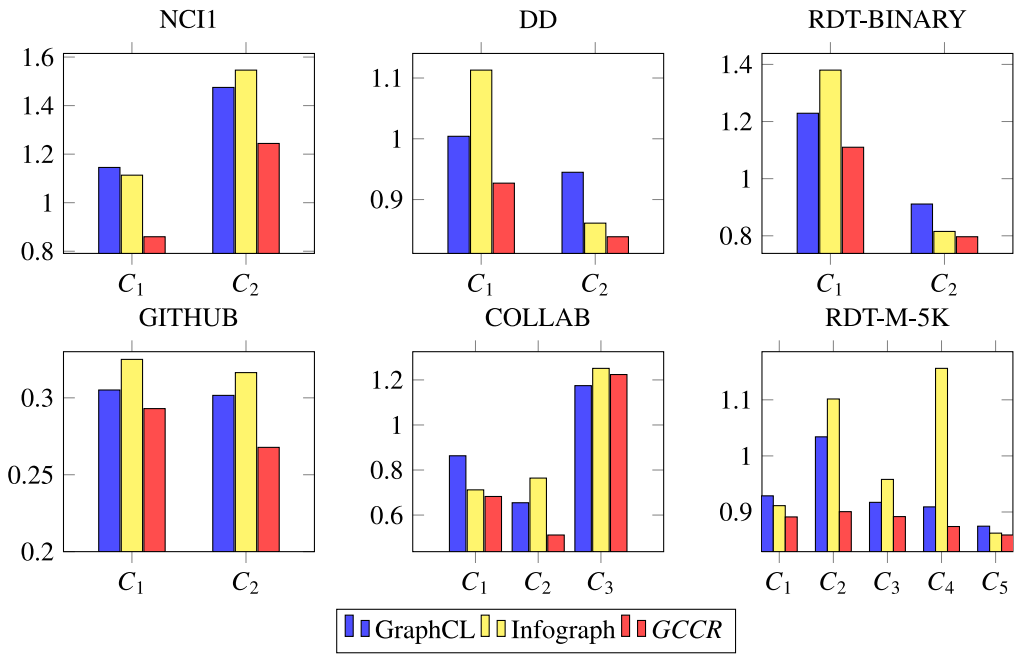

Fig. 4. Relative in-class cohesion (RIC) of GCCR, GraphCL, and Infograph. The blue, yellow, and red bars show RIC values for each class of the pre-trained model using GraphCL, Infograph, and GCCR, respectively.

Since the data augmentation method transforms data in a way that does not impair semantic properties, we considered the graph augmentation methods not used in the pre-training stage as transformations to examine the robustness of the proposed GCCR and GraphCL. Specifically, for the pre-trained and fine-tuned models with the augmentation methods $\mathcal{A}_a$ and $\mathcal{A}_b$, we derived the classification accuracy after applying different augmentation methods not used in the training phase to test data. For example, if the model is trained with *node dropping* and *attribute masking*, we applied *subgraph* and *edge perturbation* to the test data. As shown in Table 4, GCCR was more robust to transformations than GraphCL.

6. Conclusion

We propose a novel graph contrastive learning method, GCCR, to address the class collision problem with a simple and efficient approach. We introduce the consistency regularization to the loss function of the existing graph contrastive learning to obtain a representation by reflecting the intra-class similarities among negative samples. Through extensive experiments on several graph datasets, GCCR achieved significant performance improvement compared to comparative methods in various tasks. By measuring the intra-class similarities for the representations obtained from each method, GCCR alleviated the class collision problem.

Nevertheless, GCCR has a limitation in that it cannot work satisfactorily for node-level tasks. For node-level tasks, such as node classification, applying the proposed method may not be suitable because GCCR obtains graph-level representations by contrasting representations of graphs, not considering edge information capturing relationships between nodes of a graph, which is important to learn node representations effectively. Thus, it is necessary to improve the loss function by considering the edge information of a graph reflecting relationships of nodes helpful even in node-level tasks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Science and ICT (MSIT) of Korea (Nos. RS-2023-00208412 and 2022R1F1A1074393) and by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Education of Korea (No. RS-2023-00271054.

References

[1] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. Theory Exp. 2008 (10) (2008) P10008.

[2] C. Chen, W. Ye, Y. Zuo, C. Zheng, S.P. Ong, Graph networks as a universal machine learning framework for molecules and crystals, Chem. Mater. 31 (9) (2019) 3564–3572.

[3] A. Fout, J. Byrd, B. Shariat, A. Ben-Hur, Protein interface prediction using graph convolutional networks, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 6533–6542.

[4] X. Ge, Y.-C. Wang, B. Wang, C.J. Kuo, CORE: A knowledge graph entity type prediction method via complex space regression and embedding, Pattern Recognit. Lett. 157 (2022) 97–103.

[5] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.

[6] K. Xu, W. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks? 2018, arXiv preprint arXiv:1810.00826.

[7] T.N. Kipf, M. Welling, Variational graph auto-encoders, 2016, arXiv preprint arXiv:1611.07308.

[8] W.L. Hamilton, R. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 1025–1035.

[9] P. Velickovic, W. Fedus, W.L. Hamilton, P. Liò, Y. Bengio, R.D. Hjelm, Deep graph infomax, ICLR (Poster) 2 (3) (2019) 4.

[10] L.F. Ribeiro, P.H. Saverese, D.R. Figueiredo, struc2vec: Learning node representations from structural identity, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 385–394.

[11] R. Zhi, J. Hu, F. Wan, Micro-expression recognition with supervised contrastive learning, Pattern Recognit. Lett. 163 (2022) 25–31.
[12] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, Y. Shen, Graph contrastive learning with augmentations, Adv. Neural Inf. Process. Syst. 33 (2020) 5812–5823.
[13] Z. Tong, Y. Liang, H. Ding, Y. Dai, X. Li, C. Wang, Directed graph contrastive learning, Adv. Neural Inf. Process. Syst. 34 (2021).
[14] N. Jovanović, Z. Meng, L. Faber, R. Wattenhofer, Towards robust graph contrastive learning, 2021, arXiv preprint arXiv:2102.13085.
[15] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, N. Saunshi, A theoretical analysis of contrastive unsupervised representation learning, in: International Conference on Machine Learning, 2019, pp. 9904–9923.
[16] C.-Y. Chuang, J. Robinson, L. Yen-Chen, A. Torralba, S. Jegelka, Debiased contrastive learning, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020, pp. 8765–8775.
[17] A. Tejankar, S.A. Koohpayegani, V. Pillai, P. Favaro, H. Pirsiavash, ISD: Self-supervised learning by iterative similarity distillation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 9609–9618.
[18] T. Huynh, S. Kornblith, M.R. Walter, M. Maire, M. Khademi, Boosting contrastive self-supervised learning with false negative cancellation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 2785–2795.
[19] Y. You, T. Chen, Y. Shen, Z. Wang, Graph contrastive learning automated, in: International Conference on Machine Learning, 2021, pp. 12121–12132.
[20] Z. Wu, Y. Xiong, S.X. Yu, D. Lin, Unsupervised feature learning via non-parametric instance discrimination, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3733–3742.
[21] K. He, H. Fan, Y. Wu, S. Xie, R. Girshick, Momentum contrast for unsupervised visual representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9729–9738.
[22] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: International Conference on Machine Learning, 2020, pp. 1597–1607.
[23] J. Zbontar, L. Jing, I. Misra, Y. LeCun, S. Deny, Barlow twins: Self-supervised learning via redundancy reduction, in: International Conference on Machine Learning, 2021, pp. 12310–12320.
[24] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P.H. Richemond, E. Buchatskaya, C. Doersch, B.A. Pires, Z.D. Guo, M.G. Azar, et al., Bootstrap your own latent a new approach to self-supervised learning, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020, pp. 21271–21284.
[25] X. Chen, K. He, Exploring simple siamese representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15750–15758.
[26] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, A. Joulin, Unsupervised learning of visual features by contrasting cluster assignments, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020, pp. 9912–9924.
[27] K. Hassani, A.H. Khasahmadi, Contrastive multi-view representation learning on graphs, in: International Conference on Machine Learning, 2020, pp. 4116–4126.
[28] H. Zhong, J. Wu, C. Chen, J. Huang, M. Deng, L. Nie, Z. Lin, X.-S. Hua, Graph contrastive clustering, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 9224–9233.
[29] P. Bielak, T. Kajdanowicz, N.V. Chawla, Graph Barlow Twins: A self-supervised representation learning framework for graphs, Knowl.-Based Syst. 256 (2022) 109631.
[30] H. Zhang, Q. Wu, J. Yan, D. Wipf, P.S. Yu, From canonical correlation analysis to self-supervised graph neural networks, Adv. Neural Inf. Process. Syst. 34 (2021) 76–89.
[31] G. Chu, X. Wang, C. Shi, X. Jiang, CuCo: Graph representation with curriculum contrastive learning, in: IJCAI, 2021, pp. 2300–2306.
[32] M. Jin, Y. Zheng, Y.-F. Li, C. Gong, C. Zhou, S. Pan, Multi-scale contrastive siamese networks for self-supervised graph representation learning, 2021, arXiv preprint arXiv:2105.05682.
[33] J. Xia, L. Wu, G. Wang, J. Chen, S.Z. Li, Progcl: Rethinking hard negative mining in graph contrastive learning, in: International Conference on Machine Learning, 2022, pp. 24332–24346.
[34] S. Thakoor, C. Tallec, M.G. Azar, R. Munos, P. Veličković, M. Valko, Bootstrapped representation learning on graphs, in: ICLR 2021 Workshop on Geometrical and Topological Representation Learning, 2021.
[35] F.-Y. Sun, J. Hoffman, V. Verma, J. Tang, InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization, in: International Conference on Learning Representations, 2020.
[36] B. Li, B. Jing, H. Tong, Graph communal contrastive learning, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 1203–1213.
[37] H. Zhao, X. Yang, Z. Wang, E. Yang, C. Deng, Graph debiased contrastive learning with joint representation clustering, in: IJCAI, 2021, pp. 3434–3440.
[38] S. Lin, C. Liu, P. Zhou, Z.-Y. Hu, S. Wang, R. Zhao, Y. Zheng, L. Lin, E. Xing, X. Liang, Prototypical graph contrastive learning, IEEE Trans. Neural Netw. Learn. Syst. (2022).
[39] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Trans. Neural Netw. 20 (1) (2008) 61–80.
[40] M. Sajjadi, M. Javanmardi, T. Tasdizen, Regularization with stochastic transformations and perturbations for deep semi-supervised learning, in: Proceedings of the 30th International Conference on Neural Information Processing Systems, 2016, pp. 1171–1179.
[41] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, Q.V. Le, Unsupervised data augmentation for consistency training, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020, pp. 6256–6268.
[42] C. Morris, N.M. Kriege, F. Bause, K. Kersting, P. Mutzel, M. Neumann, Tudataset: A collection of benchmark datasets for learning with graphs, 2020, arXiv preprint arXiv:2007.08663.
[43] W.J. Conover, Practical Nonparametric Statistics, vol. 350, john wiley & sons, 1999.
[44] T. Chen, S. Bian, Y. Sun, Are powerful graph neural nets necessary? a dissection on graph classification, 2019, arXiv preprint arXiv:1905.04579.
[45] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, K. Borgwardt, Efficient graphlet kernels for large graph comparison, in: Artificial Intelligence and Statistics, PMLR, 2009, pp. 488–495.
[46] N. Shervashidze, P. Schweitzer, E.J. Van Leeuwen, K. Mehlhorn, K.M. Borgwardt, Weisfeiler-lehman graph kernels, J. Mach. Learn. Res. 12 (9) (2011).
[47] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.
[48] B. Adhikari, Y. Zhang, N. Ramakrishnan, B.A. Prakash, Sub2vec: Feature learning for subgraphs, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2018, pp. 170–182.
[49] P. Scherer, P. Lio, Learning distributed representations of graphs with Geo2DR, 2020, arXiv preprint arXiv:2003.05926.
[50] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (1) (2019) 1–48.