GradGCL: Gradient Graph Contrastive Learning

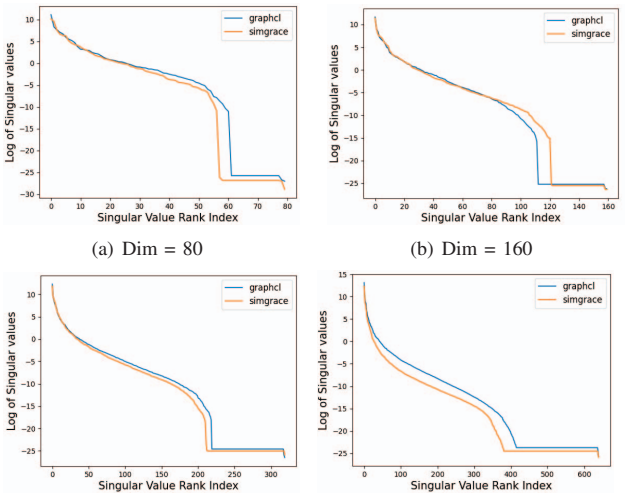
Ran LI[2], Shimin DI[2]†, Lei CHEN[1,2], Xiaofang ZHOU[2]
[1]The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China
[2]The Hong Kong University of Science and Technology, Hong Kong SAR, China
rlibb@connect.ust.hk, sdiaa@connect.ust.hk, leichen@hkust-gz.edu.cn, zxf@cse.ust.hk

***Abstract*—Graph self-supervised learning aiming to learn the graph representation without much label information is an important tasks in data mining and machine learning since labeled graph data is scarce and expensive to obtain in the real world. Contrastive learning emerges as a promising solution. However, we show existing graph contrastive learning (GCL) models have a significant issue: they generate representations that collapse into a low-dimensional subspace, resulting in a loss of information and diversity. We believe this issue arises from the strong assumption in current GCL methods that all positive samples should be close and all negative samples should be far in the representation space. From a data engineering view, this assumption fails to deeply mine the graph data and oversimplifies the complexity and heterogeneity of graph data, leading to clustered and redundant representations. To address this issue, we propose GradGCL, a novel method that leverages intrinsic gradient information as an additional input signal to regularize GCL training. The gradient information reflects the optimization process of the representations with respect to the contrastive loss, providing a complementary perspective to the representations. Furthermore, we have designed a soft separation strategy that relaxes the hard separation strategy between positive and negative samples, allowing for more flexibility and diversity in the representation space. We have conducted extensive experiments on various graph-related tasks, using different types of contrastive losses, datasets, and model architectures. We demonstrate that gradients alone can learn graph information and achieve competitive results with representation-based GCL methods. We also show that GradGCL can enhance existing GCL models and prevent the issue of dimensional collapse.**

***Index Terms*—contrastive learning; gradient analysis; graphs**

I. Introduction

Deep learning models have achieved remarkable performance in various tasks, such as computer vision, natural language processing, speech recognition [27] and database [54]. However, these models require a large amount of labeled data, which is often scarce, expensive, or unavailable in the real world [21]. In data mining and machine learning community, learning data representations with minimal label information is a critical task. To address this challenge, contrastive learning [3] has emerged as a promising technique for learning from unlabeled data. It has been successfully applied to various tasks [21] including data management [2], [52], recommendation systems [28], [60], community search [30], and temporal data mining [62], [64]. Graph contrastive learning (GCL) is a branch of contrastive learning that focuses on learning graph representations without much label information. Graphs are ubiquitous data structures that can capture complex relationships among entities, such as molecular structures [55] and knowledge graphs [6], [8]. How to mine the graph information unsupervisedly is an improtant question in data mining and machine learning community [32]. GCL has shown its effectiveness in various graph-related tasks, such as node classification [10], [13], [29], [69], [70], graph classification [13], [33], [47], [59], [65], [66] and transfer learning tasks [7], [59], [65], [66]. The main idea of GCL is to apply graph-specific data augmentation strategies, such as node dropping, edge deletion, or subgraph sampling, to generate different views of the same graph. Then, the agreement between the views is maximized in the representation space, while the disagreement between views from different graphs is minimized. This can be formulated as maximizing the mutual information (MI) between the views [16].



(a) Dim = 80 (b) Dim = 160

(c) Dim = 320 (d) Dim = 640

Fig. 1. The spectrum of Singular value of the representaion space. The representation are obtained from the pre-train SimGRACE and GraphCL model on the IMDB-BINARY dataset. The sorted singular values are computed from the representaiton covariance matrix shown in logarithmic scale. The detail of calculation are shown in III-A.The length of embedding vector is 80 in (a), 160 in (b), 320 in (c) and 640 in (d). The zero singualr values on the right suggest dimension collapse.

However, this assumption alone might not fully exploited the graph information and harm the performance on downstream tasks. The relatively strong assumption that all similar samples should be close to each other and all dissimilar samples should be apart in the representation space of existing GCL methods may cause the dimensional collapse issue [22]. It means that the learned representations tend to be concen-

†Corresponding author

2024 IEEE 40th International Conference on Data Engineering (ICDE) | 979-8-3503-1715-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICDE60146.2024.00095

2375-026X/24/$31.00 ©2024 IEEE
DOI 10.1109/ICDE60146.2024.00095

trated in a low-dimensional subspace, losing the richness and diversity of the original data features as shown in Fig. 1. The spectrum of singular values of the representation space are sorted and plotted in a logarithmic scale. The zero singular values on the right indicate dimension collapse. We suppose the current strong assumption of GCL that lack of mining the graph data deeply leads to a hard separation strategy. It tries to minimize the distance in the representation space between all positive samples created by different data augmentation techniques which may overlook the graph data information. This strategy may encourage the model to prefer low-rank representations that can easily separate the samples, but ignore some important information of the original data.

To alleviate the issues from a data engineering view, it would be ideal to obtain more graph data information that can help discriminate the samples while maintaining the intrinsic similarity unsupervisedly. More information can relax the hard separation strategy by providing more complementary view to the model. This way, the model can preserve more information of the original data and avoid shrinking the representation space. However, this idea also raises some questions and challenges that need to be answered: How to provide the complementary information? Since the label information is scarce and limited, we may not have enough supervised data for training. One possible way is to leverage other self-supervised information other than the current ones. How to balance the trade-off between separation and preservation? If we focus more on the preservation, the model may lose the discriminative power and fail to distinguish different classes or clusters. If we focus more on the separation, the model may still suffer from dimensional collapse and ignore some important features. We need to find a suitable balance point that can achieve both objectives. How to optimize the objective function that incorporates the new strategy and complementary information? One can design a new loss function that directly accounts for the new strategy and information, but this may require careful analysis and derivation and make it compatible with the contrastive loss.

In this paper, we propose a soft separation strategy to alleviate the above issues from the data engineering view by further utilizing the intrinsic gradients information to provide complementary view other than representations. Gradients of neural networks are widely used in neural network visualization [67], explanation [44] and data selection [23], [35], [51]. For example, the samples with gradient close to the average gradient of a group of data points can be regarded as the representative samples [23]. In the context of contrastive learning , gradients, revealing the specific relation between data and model, can be regarded as the "strength" of how the model push or pull different samples [12]. Other than the representations, gradients can provide complementary information about data-model relations. Thus, instead of using the hard separation strategy that pulls all the positive samples closer, gradients can potentially provide a soft separation one by adjusting the forces to pull them. In other words, previous models only rely on the similarity of data representations to forcibly distinguish the positive or negative samples. Instead, the gradients may provide complementary information about how the representations are optimized by the model.

Following these ideas about gradients, we propose a novel method, named Gradient Contrastive Learning (abbreviated to GradGCL), to enhance the performance of existing GCL models. More specifically, since the general neural networks are based on backpropagation [15] with gradient descent [43], we calculate the gradient vectors concerning the given features and use them as input for contrastive loss. Then, we introduce this new contrastive loss into existing designs so that GradGCL can be a plug-in module for different GCL models. The contributions are summarized as follows:

- We reveal the dimensional collapse issue in graph contrastive learning. To alleviate such an issue, we propose a novel graph contrastive learning method GradGCL based on gradients.
- We show that GradGCL can maximize the mutual information between the data and its representation and alleviate the dimensional collapse issue mathematically. GradGCL also empirically improves the representation quality compared with the previous methods.
- We demonstrate that GradGCL can improve the model performance of many existing graph contrastive learning methods across different tasks on different datasets.

II. Related Work

A. Gradients and data engineering

Gradients of neural networks are widely used in different tasks. In neural network visualization [67] and explanation [44], for example, Grad-CAM [44] maps the gradients w.r.t the weights of neural network with the original pixels of images to understand why the model makes the prediction. In data engineering, Grad-Match [23] use gradients to help the data selection process. The samples with gradient close to the average gradient of a group of data points can be regarded as the representative samples [23]. Moreover, some works [26], [35] try to use the gradients as features. For instance, [35] use gradients as features to approximate the neural network linearly. However, previous works mainly focus on image data and linear setting, gradients information for graph data are less explored. We further use it in the GCL domain to alleviate the dimension collapse issue.

B. Graph and Graph Neural networks

A graph can be represented by a tuple $G = (V, \boldsymbol{X}, \boldsymbol{A})$ where V is the node-set, $\boldsymbol{X} \in \mathbb{R}^{|V| \times d}$ is the node attributes and $\boldsymbol{A} \in \mathbb{R}^{|V| \times |V|}$ is corresponding adjacency matrix represents the connectivity of the graph. Let $a \in \mathcal{N}_a$ is the nodes connected with node a. Generally, the node classification task aims to classify a node in the graph into multi-classes such as $f : V \to Y$, and graph classification task aims to classify a graph into different categories as $f : \{G\} \to Y$. Graph neural networks [24] aim to map the graph data into the embedding space by first iteratively aggregating the neighborhood information to obtain the node

representation $\boldsymbol{h}_a$: $\boldsymbol{h}_a^{l+1} = \sigma(\boldsymbol{A}\boldsymbol{h}_a^l\boldsymbol{W}^l)$, where h_v^l is the node representation of node $\boldsymbol{v}$ in the l_{th} layer, $\boldsymbol{A}$ is the normalized adjacency matrix and σ is a nonlinear transformation. Then, a readout [61] phase can get the graph representation $\boldsymbol{h}_G$: $\boldsymbol{h}_G = \mathrm{READOUT}(h_a|a \in V)$, where READOUT is usually a permutation invariant function like MEAN function [56]. For simplicity, we note this process as a encoder $\boldsymbol{f}_\theta(\cdot)$ parameterized by θ. For a given graph G, a graph neural network maps it to a corresponding graph representation $\boldsymbol{h}_G = \boldsymbol{f}_\theta(G)$. $\boldsymbol{h}$ represents the graph representation in the following chapters.

C. Graph self-supervised learning

There are generally two kinds of self-supervised methods [57] to learn the graph representation without much label information: the generative self-supervised learning and contrastive learning. Followed by success in computer version [14] and natural [5], generative self-supervised learning evolves in the graph domain and aims to predict the missing information in the original data. GROVER [41] leverages the transformer-like architecture and obtains the representation by predicting graph context and subgraph information. GPT-GNN [20] leverages the autoregressive model framework and maximizes the likelihood of graph data by generating the edges, and the node attributes orderly. Other methods [25], [38], [50] utilize VAE (variational autoencoder) and reconstruct the structural information or features. Recently, GraphMAE [17] has improved the performance of generative graph self-supervised methods to be comparable with contrastive learning ones that we will introduce in the following part. It focuses on feature reconstruction with masking, replaces the MSE loss with scaled cosine loss, and re-masks the embedding of the encoder before decoding.

Compared with graph generative self-supervised learning, the contrastive learning is another popular graph self-supervised learning methods that is simple and effective. Graph contrastive learning aims to pull positive samples closer and push negative samples away by maximizing the mutual information between different data information versions as:

$$\arg\max_{\theta} \mathcal{I}(\mathbf{I}_\theta(\boldsymbol{x}), \mathbf{I}'_\theta(\boldsymbol{x})), \tag{1}$$

where $\mathbf{I}$ is the information of data parameterized by θ and $\mathbf{I}'$ is the modified data information. Note that $\boldsymbol{x}$ and $\boldsymbol{u}$ is denoted as the raw features and representations of one data instance because different models may focus on either node or graph classification tasks.The data information can be extracted with different ways. In the previous methods, the data information is represented by the extracted representations from the encoder, that is, use representation as the information extractor. The contrastive learning goal can be formulated as:

$$\arg\max_{\theta} \mathcal{I}(\boldsymbol{f}_\theta(\boldsymbol{x}), \boldsymbol{f}_\theta(\boldsymbol{x}')), \tag{2}$$

where θ is the model parameters, $\mathbf{x}, \mathbf{x}'$ is the different data versions. Various methods have different modelings of Eq. (2).

Perturbation $\mathrm{Pert}(\cdot)$ is defined as the technique to produce a slightly different representation of the original data point. For a given graph g, $\boldsymbol{h}$ is the original representation and $\boldsymbol{h}' = \mathrm{Pert}(\boldsymbol{f}(g))$ is the perturbed representation. Perturbation techniques used include data augmentation, feature enhancement, and encoder perturbation. Based on how to create or find out different versions of information, the methods can be divided into three groups. The first is the data-augmentation-based methods which perturb the graph information on the data instance level to create a new version of data. For example, GraphCL uses strategies like node dropping, edge deletion, and attribute masking to augment the input and obtain the augmented version. Then it maximizes the agreement between two input versions to learn the data information and produce graph representation for downstream tasks. The second is feature enhancement-based or feature-level perturbation methods that use feature information from different levels or angles and contrast these features to learn the data information. For example, they can contrast between local and global features or features from multi-views. The third one perturbs the encoder level and is classified as an encoder perturbation-based method. Instead of changing the original data or features, it adds noise to the encoder to obtain another perturbed encoder. It produces data representation by the two encoders to do contrastive learning.

Then extract graph representations accordingly. This process usually follows by a projection phase. Projection head is a neural network Proj, usually a MLP, projects the graph representation $\boldsymbol{h}$ to another embedding space: $\boldsymbol{u} = \mathrm{Proj}(\boldsymbol{h}), \boldsymbol{u}' = \mathrm{Proj}(\boldsymbol{h}')$. The transformation improves the performance by comparing the similarity in the new vector space [3], [4].

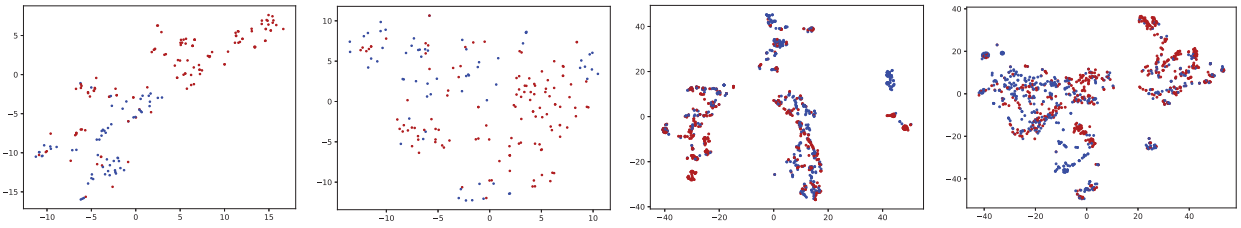
Finally, the model will maximize the mutual information. However, MI is usually intractable. People try to estimate it with different approaches and losses like InfoNCE [16], [37], JSD [16] are proposed. It is shown that optimizing the losses is equivalent to maximize the lower bound of MI. That is, this loss serve as good estimators for MI:

$$\mathcal{I}(\boldsymbol{u}_1, \boldsymbol{u}_2) \geq -\ell^{InfoNCE}(\boldsymbol{u}_1, \boldsymbol{u}_2) + \log(N) \tag{3}$$

To maximize the lower bound of mutual information is equilvalent to minimize the losses as shown in (3). Contrastive loss aims to attract the positive samples and push the negative samples away. Mathematically, it usually contains a similarity function and tends to increase the similarity between positive samples and punish one of the negative samples. The InfoNCE loss is the most widely used one.

$$\ell_n^f = -\log \frac{\exp\left(\mathrm{sim}\left(\boldsymbol{u}_n, \boldsymbol{u}'_n\right)\right)/\tau)}{\sum_{n'=1, n' \neq n}^{N} \exp\left(\mathrm{sim}\left(\boldsymbol{u}_n, \boldsymbol{u}'_n\right)/\tau\right)}, \tag{4}$$

where τ is the hyperparameter temperature and sim is a similarity function like cosine similarity. There are other kinds of contrastive loss but share a similar definition. These methods contrasting at the representation level can miss some data information contained in the encoder and use a hard separation strategy that may cause some trouble based on their assumption.



(a) Representations of MUTAG (b) Gradients of MUTAG (c) Representations of IMDB (d) Gradients of IMDB

Fig. 2. t-SNE visualization of representation and gradient distribution of MUTAG and IMDB-BINARY datasets with SimGRACE as backbone. The plots (a) and (c) show the model classified the two classes shown with different colors. The plots (b) and (d) show the corresponding gradient distribution of these representations with different patterns. This suggests that the gradient provide complementary information of the data.

III. Methodology

In this section, we first present some potential issues of existing graph contrastive learning models by motivational experiments, then we show how gradients may alleviate such issues. Based on the empirical observation, we formally introduce the proposed technique to enhance the general contrastive learning models and show some empirical evidences to demonstrate the advantages of the gradients.

A. Motivational Experiments

1) The Dimensional Collapse Issue: Previous research [22] has shown that current contrastive learning models have a dimensional collapse issue. The encoder model projects the original data into the representation space. However, it is found that the representation obtained only spans in a lower dimensional subspace among the entire representation space [22]. Some collapsed dimension suggest the representations are less informative.

We employ SimGRACE [59], one state of art contrastive learning model, to show the preliminary results. The model is trained on IMDB-BINARY [34] data set on unsupervised graph classification tasks. We output the representations $\boldsymbol{U} \in \mathbb{R}^{|U| \times d}$ from the trained model representing n data points of IMDB-BINARY, where $\boldsymbol{U} = [\boldsymbol{u}_1^\top, \boldsymbol{u}_2^\top, ..., \boldsymbol{u}_n^\top]$. We calculate the covariance matrix $\boldsymbol{C} \in \mathbb{R}^{d \times d}$of the representations $\boldsymbol{U}$ as:

$$\boldsymbol{C} = \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{u}_i - \bar{\boldsymbol{u}})(\boldsymbol{u}_i - \bar{\boldsymbol{u}})^T, \tag{5}$$

where $\bar{\boldsymbol{u}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{u_i}$. Then, we output the singular value of it, $C = LSV^T$ and $S = diag(\sigma^k)$. We plot the sorted singular value in logarithmic scale $\{\log(\sigma^k)\}$ in Fig. 1. The dimension d of represention is chosen from $\{160, 320, 640\}$.

Obviously, the results in Fig. 1 show that the a part of singular values is zero suggesting that some dimension collapse. We argue that this problem comes from the hard assumption behind the current contrastive learning goal. The general graph contrastive learning model aims to maximize the agreement between positive pairs by attracting all the similar samples closer and push away all the negative samples in the representation space. To achieve this, the model may "cheat" itself to provide low-rank representations to make instance wise separation easier. For example, if there are 20 data samples where 10 of them belong to class 1 and the rest 10 belong to class 2, it is easy to represent them with one-hot vectors and for class 1, the non-zero elements appear only in the first 10 positions of the vectors. However, low-rank representations may contain less information of the original data and be harmful for downstream tasks. Thus, existing works may need to be enhanced.

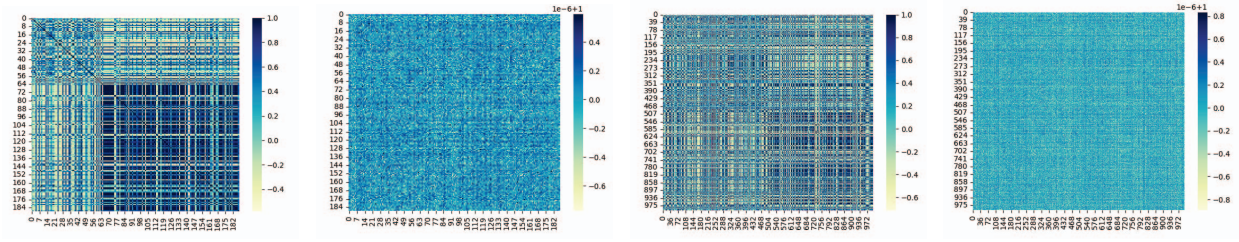
2) The Role of Gradient in Graph Contrastive Learning: As discussed previously, existing graph contrastive learning models may suffer from the dimension collapse issue because its hard assumption. In this subsection, we discuss the role of Gradient in Graph Contrastive Learning and its potential information. Formally, the gradient $g_\theta(\boldsymbol{u}) = \frac{\partial \ell_n^f}{\partial \boldsymbol{u}}$ of Eq. (4) w.r.t the representation $\boldsymbol{u}$ can be calculated as:

$$(1 - \frac{\exp(\boldsymbol{u}^T \boldsymbol{v}^+ / \tau)}{Z(\boldsymbol{u})}) / \tau \boldsymbol{v}^+ - \sum_{\boldsymbol{v}^-} \frac{\exp(\boldsymbol{u}^T \boldsymbol{v}^- / \tau)}{Z(\boldsymbol{u})} / \tau \boldsymbol{v}^-, \tag{6}$$

where $\boldsymbol{u} = f_\theta(x)$ and θ is model parameters. The partition function $Z(\boldsymbol{u_i})$ is defined as: $Z(\boldsymbol{u_i}) = \sum_{j=1, j \neq i}^{N} \exp(\boldsymbol{u_i}^T \boldsymbol{u_j} / \tau)$. Note that $\boldsymbol{v}^+$ and $\boldsymbol{v}^-$ are the representations of the positive sample and negative samples of $\boldsymbol{u}$. In Eq. (6), we may observe that several points: 1) for positive samples, if their similarity is low, the gradient w.r.t the samples are large. As a result, the model will increase their similarity to lower the contrastive loss; 2) for negative samples, if their similarity is large, the corresponding magnitude of the gradient will be significant. Thus, the model will tend to lower the similarity to obtain a smaller loss.

In summary, the calculation way of gradients in existing graph contrastive learning models can force the model to pull positives pairs closer and push the negative ones away.

But we also observe that Eq. (6) may contain more fine-grained information at the instance-level, i.e., the similarity among samples of one data instance. To verify this, we implement the advanced work SimGRACE [59] that pre-trained on the MUTAG [34] data set (binary classes) for the unsupervised graph classification task. Then, as shown in Fig. 2, we visualize the representations and gradients of SimGRACE with the help of TSNE technique. Note that we treat the outputs of Eq. 6 as the representations of gradients. The nodes with two colors in Fig. 2 represents the representation/gradient distribution of two classes of graphs in MUTAG. Compared with Fig. 2 (a), the representations in Fig. 2 (b) are



(a) Representation Similarty of MUTAG (b) Gradient Similarity of MUTAG (c) Representation Similarity of IMDB (d) Gradient Similarity of IMDB

Fig. 3. Instance-wise representation and its gradients similarity heatmap of MUTAG and IMDB-0BINARY datasets based on the pre-train SimGRACE model. (a) and (c) are for the representations and (b) and (d) represent the gradient. It is shown that the similarity of gradients is more diverse compared with the representations'.

more diverse, indicating the gradients have more fine-grained information regarding the correlations among data instances.

Thus, as shown in Fig. 3, we further visualize the instance-wise cosine similarity of representations $cosine\_sim(\boldsymbol{u}, \boldsymbol{u}')$ and gradients $cosine\_sim(g_\theta(\boldsymbol{u}), g_\theta(\boldsymbol{u}'))$ of 188 graphs in MUTAG. Two diagonal blocks in Fig. 3(a) with deep color indicate that the representations of graphs have high intra-class similarities, while another two blocks with light color mean that the representations have low inter-class similarities. This observation verifies that existing graph contrastive learning models tend to distinguish the classes while ignoring the internal correlation of samples in one class. In other words, existing models may not be able to distinguish samples that are similar in terms of features but do not belong to the same class, i.e., failing to handle hard negative samples. On the contrary, we may observe that the similarities in Fig. 3(b) are more diverse, which may be able to capture the correlation at the level of instances.

As a conclusion, $g_\theta(\boldsymbol{u})$ show in Eq (6) can serve as a information extractor just like the encoder $\boldsymbol{u} = f_\theta(x)$. We will show this from the mutual information view.

B. GradGCL

In this subsection, we introduce the proposed enhancement technique for graph contrastive learning methods as shown in Fig. 4 and show that gradients can maximize the mutual information between different data versions and gradients contrastive learning acts as a pairwise regularization module together with representations.

Based on the observation in Sec. III-A, we argue that the information of data can be shown into two parts: 1) representations by $f_\theta(\boldsymbol{x})$, 2) gradients in the learning procedure. Therefore, we can formulate the information of $\boldsymbol{I}_\theta(\boldsymbol{x})$ as:

$$\boldsymbol{I}_\theta(\boldsymbol{x}) \supseteq \boldsymbol{I}(\boldsymbol{f}_\theta(\boldsymbol{x})) \cup \boldsymbol{I}(\boldsymbol{g}_\theta(\boldsymbol{x})), \tag{7}$$

Eq. (7) suggests that we can maximize the mutual information based on $f_\theta(x)$ and gradients $g_\theta(x)$. Let

$$\begin{aligned}&\mathcal{I}(\boldsymbol{I}_\theta(\boldsymbol{x}), \boldsymbol{I}_\theta(\boldsymbol{x}'))\\ &= (1-a)\mathcal{I}(\boldsymbol{f}_\theta(\boldsymbol{x}), \boldsymbol{f}_\theta(\boldsymbol{x}')) + a\mathcal{I}(\boldsymbol{g}_\theta(\boldsymbol{x}), \boldsymbol{g}_\theta(\boldsymbol{x}'))\end{aligned} \tag{8}$$

be the mutual information. Then, the optimization goal can be formally formulated as:

$$\arg\max_\theta \ (1-a)\mathcal{I}(\boldsymbol{f}_\theta(\boldsymbol{x}), \boldsymbol{f}_\theta(\boldsymbol{x}')) + a\mathcal{I}(\boldsymbol{g}_\theta(\boldsymbol{x}), \boldsymbol{g}_\theta(\boldsymbol{x}')), \tag{9}$$

where a controlls the weight of the gradients part. When $a = 0$, it degrades to the classic contrastive learning goal. We further derive the lower bound of $\mathcal{I}(\boldsymbol{I}_\theta(\boldsymbol{x}), \boldsymbol{I}_\theta(\boldsymbol{x}'))$ as:

$$\begin{aligned}&(1-a)\mathcal{I}(\boldsymbol{f}_\theta(\boldsymbol{x}),\boldsymbol{f}_\theta(\boldsymbol{x}'))+a\mathcal{I}(\boldsymbol{g}_\theta(\boldsymbol{x}),\boldsymbol{g}_\theta(\boldsymbol{x}'))\\ &\geq -(1-a)\ell_n^f - a\ell_n^g + \log(N).\end{aligned} \tag{10}$$

1) Gradients and Mutual Information: The *infoNCE loss* is a type of contrastive loss for self-supervised learning. It is based on the idea of noise-contrastive estimation, which is a method to estimate the mutual information between two variables with a classifier to distinguish between positive and negative samples. The infoNCE loss can be written as:

$$-\mathbb{E}_X\left[\log \frac{\mathrm{sim}(I(x_t), I(x_t'))}{\sum_{x_j \in X} \mathrm{sim}(I(x_j), I(x_t'))}\right], \tag{11}$$

where $X = \{x_1, \ldots, x_N\}$ is a set of N random samples containing one positive sample from $p(x_t|x_t')$ and $N-1$ negative samples from the proposal distribution $p(x_t)$, x_t' is the context variable, sim is a function that measures the similarity between two variables and $I(x_t)$ is the information with the data, for example, $I(x_t) = g_\theta(\boldsymbol{x})$ means we use the gradient as the data information.

Lemma 1. *The infoNCE loss can be seen as a lower bound on the mutual information between $I(x_t)$ and $I(x_t')$:*

$$\mathcal{I}(I(x_t); I(x_t')) = \mathbb{E}_{p(I(x_t), I(x_t'))}\left[\log \frac{p(I(x_t), I(x_t'))}{p(I(x_t))p(I(x_t'))}\right] \tag{12}$$

Proof. To show this, we can use the following steps:

First, we rewrite the infoNCE loss as:

$$\begin{aligned}L_N = &-\mathbb{E}_X\left[\log \frac{p(I(x_t)|I(x_t'))\,\mathrm{sim}(I(x_t), I(x_t'))}{\sum_{x_j \in X} p(I(x_j))\,\mathrm{sim}(I(x_j), I(x_t'))}\right]\\ &+ \mathbb{E}_X\left[\log \frac{p(I(x_t))}{p(I(x_t)|I(x_t'))}\right]\end{aligned} \tag{13}$$

by multiplying and dividing by $p(I(x_t)|I(x_t'))$ inside the logarithm.

Next, we use Jensen's inequality, which states that for any convex function ϕ and any random variable Y, we have:

$$\phi(\mathbb{E}[Y]) \leq \mathbb{E}[\phi(Y)]. \tag{14}$$

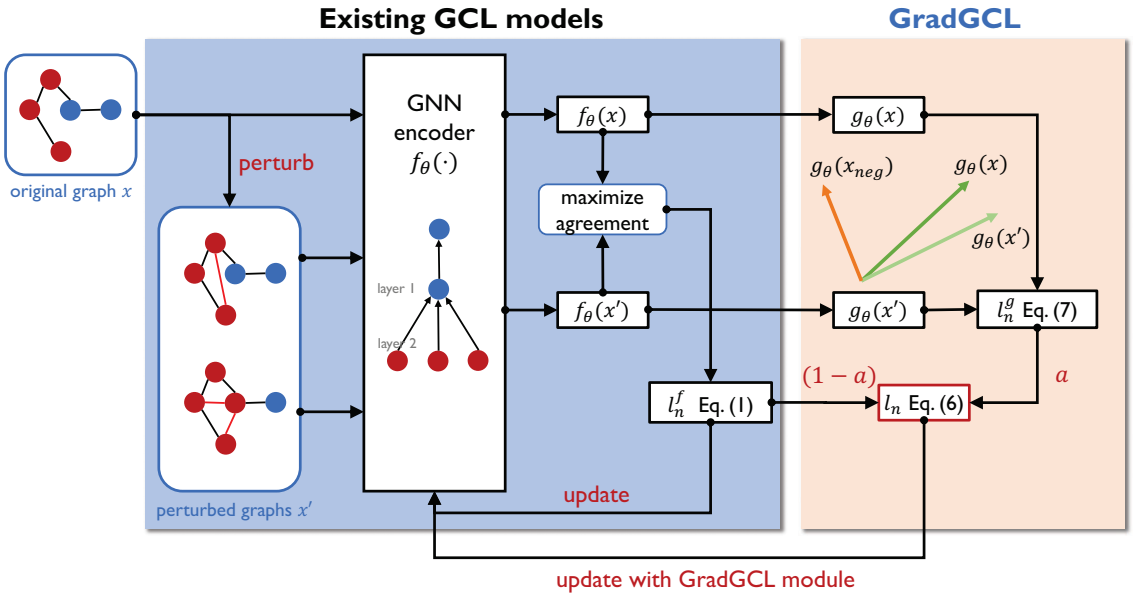


Fig. 4. The overview of GradGCL. Instead of contrasting between representations, we also contrast samples based on gradients. The agreement of gradients is maximized after projection. For sample x, x' is the positive samples and x_{neg} is the negative sample. When combined with representations, GradGCL serves as a pairwise regularization module, facilitating the alignment of positive samples during optimization.

Applying this to the negative logarithm function and the first term of the infoNCE loss, we get:

$$-\log \mathbb{E}_X \left[\sum_{x_j \in X} \frac{p(I(x_j))\,\mathrm{sim}(I(x_j), I(x_t'))}{p(I(x_t)|I(x_t'))\,\mathrm{sim}(I(x_t), I(x_t'))} \right] \leq -\mathbb{E}_X \left[\log \sum_{x_j \in X} \frac{p(I(x_j))\,\mathrm{sim}(I(x_j), I(x_t'))}{p(I(x_t)|I(x_t'))\,\mathrm{sim}(I(x_t), I(x_t'))} \right]. \tag{15}$$

Then, we notice that the left-hand side of the inequality can be simplified as:

$$\begin{aligned} &-\log \mathbb{E}_X \left[\sum_{I(x_j) \in X} \frac{p(I(x_j))\,\mathrm{sim}(I(x_j), I(x_t'))}{p(I(x_t)|I(x_t'))\,\mathrm{sim}(I(x_t), I(x_t'))} \right] \\ &= -\log p(I(x_t')) - \log \mathrm{sim}(I(x_t'), I(x_t')) + \log N, \end{aligned} \tag{16}$$

where we use the fact that $\mathbb{E}_X[p(I(x_j))] = 1$ and $\sum_{x_j \in X} \mathrm{sim}(I(x_j), I(x_t')) = N\,\mathrm{sim}(I(x_t'), I(x_t'))$.

Finally, we rearrange the terms and obtain:

$$\begin{aligned} &\ell_N + \log p(I(x_t')) + \log \mathrm{sim}(I(x_t'), I(x_t')) - \log N \\ &\leq -\mathbb{E}_X \left[\log \frac{p(I(x_t)|I(x_t'))}{p(I(x_t))} \right], \end{aligned} \tag{17}$$

the term on the right hand side is exactly the $\mathcal{I}(I(x_t); I(x_t'))$, which shows that the infoNCE loss is a lower bound. □

To maximize the lower bound of the mutual information between different versions of data information in Eq. (10) is equivalent to minimize the loss function as:

$$\ell_n = (1-a)\ell_n^f + a\ell_n^g. \tag{18}$$

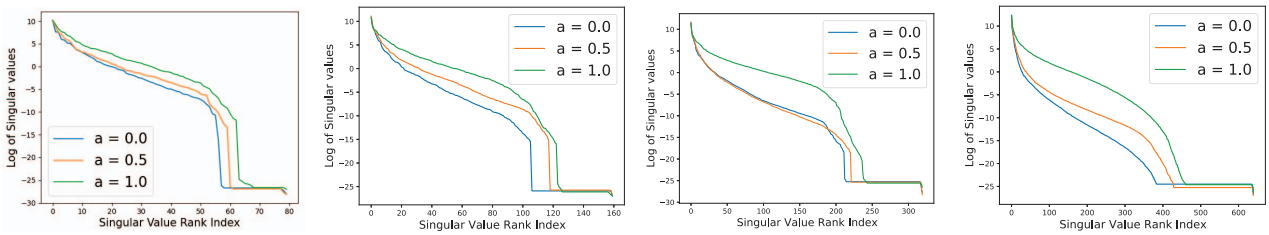
Note that ℓ_n^f is the classic general loss in graph contrastive learning as Eq. (4) presented. Similarly, we introduce the infoNCE loss to model the contrastive loss ℓ_n^g for gradient:

$$\ell_n^g = -\log \frac{\exp\left(\mathrm{sim}\left(\boldsymbol{g}_n, \boldsymbol{g}_n'\right)\right)/\tau)}{\sum_{n'=1, n' \neq n}^{N} \exp\left(\mathrm{sim}\left(\boldsymbol{g}_n, \boldsymbol{g}_{n'}\right)/\tau\right)}. \tag{19}$$

In conclusion, utilizing gradients alone can maximize the mutual information between different data versions, making them valuable for contrastive learning. As shown in Fig. 4 and Eq. (18), gradient contrastive learning for graphs (GradGCL) act as a pairwise regularization module, which can be implemented into existing graph contrastive learning models. We will demonstrate how the regularization mechanism of GradGCL effectively addresses the dimensional collapse issue and enhances the quality of representations with more experiments.

2) Gradients and dimensional collapse issue: In this part, we will show that the gradient loss can help alleviate the dimensional collapse issue. To illustrate how gradient loss works, we consider the case where we use Euclidean distance as the similarity measurement and linear networks as the encoders following the previous study [22]. The infoNCE loss function L is defined as follows:

$$-\sum_{i=1}^{N} \log \frac{\exp(-|\boldsymbol{u}_i - \boldsymbol{u}_i'|^2/2)}{\sum_{j \neq i} \exp(-|\boldsymbol{u}_i - \boldsymbol{u}_j|^2/2) + \exp(-|\boldsymbol{u}_i - \boldsymbol{u}_i'|^2/2)}, \tag{20}$$



(a) Dim = 80 (b) Dim = 160 (c) Dim = 320 (d) Dim = 640

Fig. 5. The spectrum of Singular value of the representaion space. The representation are obtained from the pre-train SimGRACE model on the IMDB-BINARY dataset. The sorted singular values are computed from the representaiton covariance matrix shown in logarithmic scale. Dim = 160 means the length of embedding vector is 160. The gradients increase the rank of representation vectors and alleviate the dimension collapse issue.

where $\boldsymbol{u}_i = W\boldsymbol{x}_i$, $\boldsymbol{u}'_i = W\boldsymbol{x}'_i$ are the embedding vectors of the positive data pair $\boldsymbol{x}_i, \boldsymbol{x}'_i$ respectively and $\boldsymbol{x}_j$ represents the negative samples.

We study the dynamics of the gradient flow, which is a continuous-time limit of gradient descent with an infinitesimal learning rate.

Lemma 2. *Based on the contrastive loss in Eq. 20 defined previously, the weight matrix in a linear scenario changes as follows:*

$$\dot{W} = -G, \tag{21}$$

$$G = \sum_i (\boldsymbol{g}_{\boldsymbol{u}_i} \boldsymbol{x}_i^T + \boldsymbol{g}_{\boldsymbol{u}'_i} \boldsymbol{x}_i'^T), \tag{22}$$

where $\boldsymbol{g}_{\boldsymbol{u}_i}$ and $\boldsymbol{g}_{\boldsymbol{u}'_i}$ are the gradient of $\boldsymbol{u}_i$ and $\boldsymbol{u}'_i$.

Proof. Take the derivative of the loss Eq. 20 with respect to the weight matrix based on the chain rule:

$$\frac{dL}{dW} = \sum_i \left(\frac{\partial L}{\partial \boldsymbol{u}_i}\frac{\partial \boldsymbol{u}_i}{\partial W} + \frac{\partial L}{\partial \boldsymbol{u}'_i}\frac{\partial \boldsymbol{u}'_i}{\partial W}\right)$$

Based on the linear setting of the network:

$$\frac{\partial \boldsymbol{u}_i}{\partial W} = \boldsymbol{x}_i, \frac{\partial \boldsymbol{u}'_i}{\partial W} = \boldsymbol{x}'_i,$$

we obtain:

$$\dot{W} = -\left(\frac{dL}{dW}\right)^T = -\sum_i (\boldsymbol{g}_{\boldsymbol{u}_i} \boldsymbol{x}_i^T + \boldsymbol{g}_{\boldsymbol{u}'_i} \boldsymbol{x}_i'^T).$$

The gradient $\boldsymbol{g}_{u_i}$ and $\boldsymbol{g}_{u'_i}$ can be obtain based on the infoNCE loss function with the linear encoder with respect to u_i and u'_i as :

$$\boldsymbol{g}_{u_i} = \sum_{j\neq i} \alpha_{ij}(\boldsymbol{u}_j - \boldsymbol{u}'_i) + \sum_{j\neq i} \alpha_{ji}(\boldsymbol{u}_j - \boldsymbol{u}_i),$$

$$\boldsymbol{g}_{u'_i} = \sum_{j\neq i} \alpha_{ij}(\boldsymbol{u}'_i - \boldsymbol{u}_i),$$

where $\alpha_{ij} = \exp(-||\boldsymbol{u}_i - \boldsymbol{u}_j||^2/2)/Z_i$, $\alpha_{ii} = \exp(-||\boldsymbol{u}_i - \boldsymbol{u}'_i||^2/2)/Z_i$, and $Z_i = \sum_{j\neq i}\exp(-||\boldsymbol{u}_i - \boldsymbol{u}_j||^2/2) + \exp(-||\boldsymbol{u}_i - \boldsymbol{u}'_i||^2/2)$. That is, $\sum_j \alpha_{ij} = 1$. Since $\boldsymbol{u}_i = W\boldsymbol{x}_i$, we can get:

$$G = -WX, \tag{23}$$

where X is the weighted covariance data matrix. □

Lemma 3. *The rank of G is N under the conditions:*

1) *For each $i = 1, 2, ..., N$, the vectors g_{u_i} and $g_{u'_i}$ are linearly dependent, i.e., there exists a scalar c_i such that $g_{u_i} = c_i g_{u'_i}$.*
2) *The vectors $g_{u_i} + g_{u'_i}$ for $i = 1, 2, ..., N$ are linearly independent of one another, i.e., they form a basis for a subspace of dimension N.*

Proof. Given the data and its augmentation, the difference between x' and x is assumed to be small, such that $x' = x + \delta x$.

Expanding the expression for G:

$$\begin{aligned} G &= \sum_i (g_{u_i} x_i^T + g_{u'_i} x_i'^T) \\ &= \sum_i (g_{u_i} x_i^T + g_{u'_i} (x + \delta x')^T) \\ &\approx \sum_i (g_{u_i} x_i^T + g_{u'_i} x^T) \\ &= \sum_i (g_{u_i} + g_{u'_i}) x_i^T. \end{aligned}$$

It can be seen that:

$$\forall i, \quad g_{u_i} \neq 0, x_i \neq 0 \implies \text{rank}(g_{u_i} x_i^T) = 1.$$

For any index i, if the vectors g_{u_i} and x_i are both non-zero, then the matrix $g_{u_i} x_i^T$ has rank 1. This is because the column space of $g_{u_i} x_i^T$ is spanned by g_{u_i}, which is a non-zero vector. Therefore, the column space has dimension 1, and the rank of a matrix is equal to the dimension of its column space.

When g_{u_i} and $g_{u'_i}$ are linearly dependent:

$$g_{u_i} + g_{u'_i} = (1 + \lambda) g_{u_i},$$

hence $G_i = (g_{u_i} + g_{u'_i}) x_i^T$ remains rank 1. λ is a real number here.

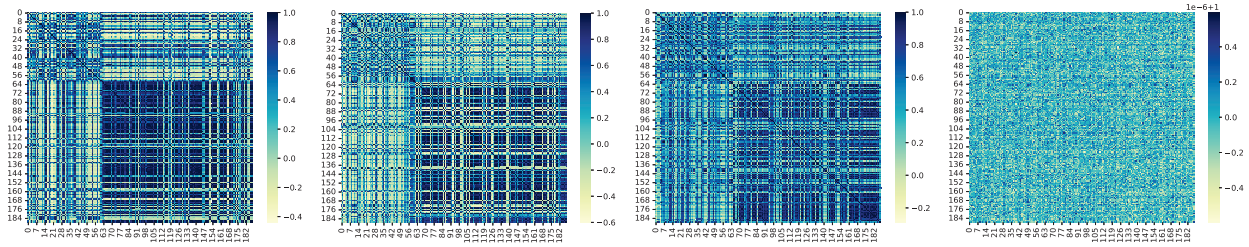
Moreover, if $g_{u_i} + g_{u'_i}$ vectors are linearly independent across all i:

$$rank(Col\{G\}) = N,$$

that is, the column space of G, $Col\{G\}$, spanned by G_i, has rank N.

For our GradGCL, we enforce the similarity among g_{u_i} for positive samples and difference for negative samples, i.e.,

$$g_{u_i}^T g_{u'_i} = 1, g_{u_i}^T g_{u_j} = 0$$



(a) Gradient weight = 0 (b) Gradient weight = 0.5 (c) Gradient weight = 1 (d) Gradient Similarity

Fig. 6. Instance-wise representation and its gradients similarity heatmap with various gradients weights. (a)Weight is 0 means the original model;(b)Weight is 0.5 means learning with both views of equal contribution; (c)Weight is 1 means the model is trained only using gradients signal. From left to right, the darker region is less centered indicating the representation similarity is more diverse.

Therefore,

$$rank(Col\{G\}) = rank(span\{g_{u_i}\}) = N$$

the column space $Col\{G\}$ spanned by $\{g_{u_i}\}$ also has rank N.

Given that $G = -WX$, X is a weighted covariance matrix and remains fixed, and leveraging the property

$$rank(AB) \leq min\{rank(A), rank(B)\},$$

it implies $rank(Col\{W\}) = N$.

The matrix C can be expressed as:

$$C = \sum_i (u_i - \bar{u})(u_i - \bar{u})^T / N = \sum_i W(x_i - \bar{x})(x_i - \bar{x})^T W^T / N.$$

Because $rank(Col\{W\}) = N$, C exhibits a high-rank characteristic. □

C. Key Observations of GradGCL

1) Representation Quality: To better show the benefit of the gradients contrastive learning, we evaluate different contrastive learning methods with the alignment and uniformity [53]. Alignment in Eq. (24) is calculated as the expected distance of the positive samples:

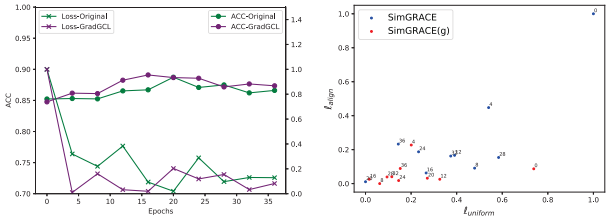
$$\ell_{\text{align}}(f;\alpha) \triangleq \mathop{\mathbb{E}}_{x \sim p_{\text{data}}} \left[\|f_\theta(x) - f_\theta(x')\|_2^\alpha\right], \quad \alpha > 0. \tag{24}$$

A good contrastive learning should be able to learn the common information in the positive pairs and lower their distance in the representation space. However, if all samples are close in the representation space, mode collapse may happen and the model can not be robust enough for downstream tasks. To balance the diversity, another metric, uniformity is proposed:

$$\ell_{\text{uniform}}(f;\alpha) \triangleq \log \mathop{\mathbb{E}}_{x,y \overset{i.i.d.}{\sim} p_{\text{data}}} \left[e^{-t\|f_\theta(x) - f_\theta(y)\|_2^2}\right], \quad t > 0. \tag{25}$$

It calculates the expected Gaussian potential between pairs. A small uniformity means the random samples are distributed in the representation space.

We introduce the technique proposed in Sec. III-B into SimGRACE, named SimGRACE(g) and train them on MUTAG. We plot the scores (Eq. (24) and Eq. (25)) of learned representations in Fig. 7(a) with every 4 epochs annotated by numbers. As shown in Fig. 7(a), we can see model trained with gradients contrastive goal produces better representations, which indicates the improved representation quality with the help of gradients. Also, for each epoch we also show the improvements on the performance by the loss and accuracy curve in Fig. 7.



(a) Loss and accuracy curve (b) Uniform Align relation

Fig. 7. Loss and accuracy curve and Uniform Align relation for SimGRACE and SimGRACE(g), the improved version with our method. Note that it is the lower, the better for both metrics.

2) Alleviating Dimensional Collapse Issues: Contrastive learning suffers from dimensional collapse, but using gradients can help alleviate this problem, as shown in Fig. 5. The representations in the figure are obtained using our method with the SimGRACE backbone and gradient weights of $a \in \{0, 0.5, 1.0\}$. The gradients help postpone the drop in singular values, alleviating the dimensional collapse shown by the vanishing singular values.

In Fig. 6, the previous model (weight 0) allows the model to successfully classify the two classes (darker diagonal blocks). However, the model exaggerates intra-class similarity, leading to underestimated diversity between classes or different samples. Improving the model with our method and increasing the gradient weight leads to more evenly distributed similarity, as shown in Fig. 7(a). We will further investigate if the improved representation benefits downstream tasks in our experiments.

IV. Experiment

We implement GradGCL based on Pytorch framework [39]. All experiments are performed using one single 32GB V100 GPU. And the code of GradGCL and other baselines are publicly available[1].

We empirically show that our GradGCL can be applied to different GCL models in 3 graph-related tasks, including graph classification, transfer learning, and node classification. We generally plug our methods GradGCL and its variant into previous methods and evaluate the performance following their

[1]https://github.com/ranlislz/Graph

TABLE I
DATASETS STATISTICS FOR UNSUPERVISED GRAPH CLASSIFICATION

Datasets	Category	Graph Num.	Classes	Avg. Node	Avg. Edges
NCI1	Biochemical	4,110	2	29.87	32.30
PROTEINS	Biochemical	1,113	2	39.06	72.82
DD	Biochemical	1178	2	284.32	715.66
MUTAG	Biochemical	188	2	17.93	19.79
COLLAB	Social Networks	5,000	2	74.49	2457.78
IMDB-B	Social Networks	1000	2	19.77	96.53
RDT-B	Social Networks	2000	2	429.63	497.75
RDT-M5K	Social Networks	4,999	5	508.52	594.87
RDT-M12K	Social Networks	11,929	11	391.41	456.89
TWITTER-RGP	Social Networks	144,033	2	4.03	4.98

TABLE II
STATISTICS OF DATASETS USED IN OUR EXPERIMENTS.

Datasets	Nodes	Edges	Features	Classes
Cora	2,708	5,429	1,433	7
Citeseer	3,327	4,732	3,703	6
PubMed	19,717	44,338	500	3
WikiCS	11,701	216,123	300	10
Amazon Computer	13,752	245,861	767	10
Amazon Photo	7,650	119,081	745	8
Coauthor CS	18,333	81,894	6,805	15
Coauthor Physics	34,493	247,962	8,415	5
ogbn-Arxiv	169,343	1,166,243	128	40

previous experimental settings in datasets, model architectures and evaluations. Due to space limits, the data statistics are summarized in the supplementary material. Note that XXX represent the raw performance of the baseline XXX (only activating the left part of Fig. 4, setting $a = 0$ in Eq. (18)), XXX(g) represents the gradient alone case of GradGCL with baseline XXX (discarding the left part of Fig. 4, setting $a = 1$ in Eq. (18)), mark XXX(f+g) represents full GradGCL with baseline XXX (whole framework shown in Fig. 4).

A. Graph Classification

Datasets and Experimental Setup: We applied our method to five previous graph contrastive learning methods and evaluated its effectiveness in two scenarios: using gradients alone and combining gradients with feature representations on unsupervised graph classification tasks. The baseline models include data augmentation-based methods (GraphCL [66], JOAO [65]), encoder perturbation-based methods (SimGRACE [59]), and feature enhancement-based methods (InfoGraph [47], MVGRL [13]). We also compared with kernel methods (WL [45], DGK [63]), other self-supervised methods like node2vec [11], sub2vec [1], graph2vec [36] and one most recent work RGCL [31]. We evaluated our methods on the TUdataset [34] benchmark with social network [42], [63] and biochemical datasets [9], [40]. Beside the smaller datasets with thousands of graphs used in the previous models, we conduct experiment on larger datasets with over hundreds of thousand of graphs. The performances of unsupervised graph classification with GCL on larger datasets are less studied in the previous research. The size of datasets (number of graphs) ranges from 188 to 144033. The datasets statistics are shown in Table I.

For unsupervised graph tasks, we trained the new models with the given datasets to obtain graph representations and applied them to downstream classification tasks using an SVM classifier with 10-fold cross-validation for the smaller datasets. For larger datasets, we adopt stochastic gradient descent (SGD) classifier for better efficiency in the evaluation phase. The results are the average accuracy and standard deviation for 5 runs. To ensure fairness in comparison, we left the model architectures and loss types unchanged for each method and implemented our method to calculate gradients and the gradient contrastive loss.

TABLE III
DATASETS STATISTICS FOR TRANSFER LEARNING.

Datasets	Category	Utilization	Graph Num.	Avg. Node	Avg. Degree
ZINC-2M	Molecules	Pretrain	2,000,000	26.62	57.72
PPI-306K	Protein	Pretrain	306,925	39.82	729.62
BBBP	Biochemical	Finetuning	2,039	24.06	51.90
Tox21	Biochemical	Finetuning	7,831	18.57	38.58
ToxCast	Biochemical	Finetuning	8,576	18.78	38.52
SIDER	Biochemical	Finetuning	1,427	33.64	70.71
ClinTox	Biochemical	Finetuning	1,477	26.15	55.76
MUV	Biochemical	Finetuning	93,087	24.23	52.55
HIV	Biochemical	Finetuning	41,127	25.51	54.93
BACE	Biochemical	Finetuning	1,513	34.08	73.71

Results: The results are shown in Tab. IV. Compared with classic graph representation learning models, we may first observe that graph contrastive learning generally achieve better performance. As for XXX(g) (using gradients alone, i.e., $a = 1$ in Eq. (18)), we found that the learned representations can be successfully applied to downstream graph classification tasks and perform comparably or better than previous graph contrastive learning models, including MVGRL, InfoGraph, JOAO, and SimGRACE. When using both gradients and representations (i.e., XXX(f+g)), it is obvious that GradGCL can improve the performance of existing methods. The consistent improvements across different model architectures and datasets demonstrate the effectiveness of the soft separation strategy and the improved representation quality from using gradients.

Hyperparameter sensitivity: Generally, the weight of gradients loss is essential. When the weight is 0, the model degraded to the representation-only case. When the weight is 1, the model only uses gradients to learn helpful information. As shown in Figure 8, we can see the influence of the different gradients. Generally, for different models and datasets, the optimal weight may vary.

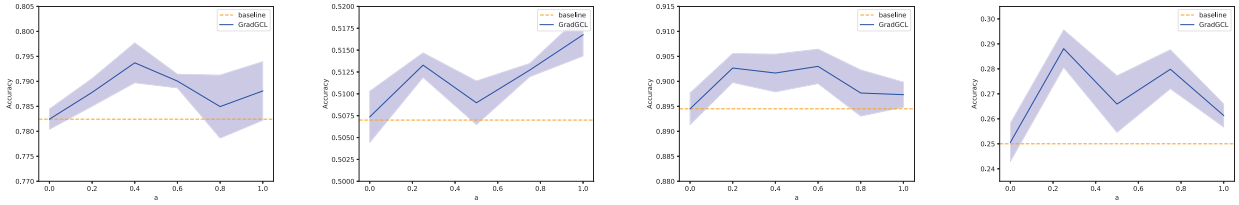
B. Node classification

Experiment Setup: This task focuses on the transductive setting. The model is first trained in an unsupervised manner and then used to produce the node embeddings. We evaluate the performance on the different datasets used in previous research. We evaluate the performance on 4 graph contrastive learning models on node classification: GRACE [69], GCA [70], BGRL [49], MVGRL [13] COSTA [68] and SGCL [48]. We conduct experiment on both smaller graphs and larger graphs like ogbn-Arxiv [18] with 169K nodes and the data statistics are shown in Tab. II. To evaluate the performance, we adopt the previous protocol [13], [69], [70].

TABLE IV

THE PERFORMANCE COMPARISON ON THE UNSUPERVISED GRAPH CLASSIFICATION TASK. WE USE THE RESULTS REPORTED IN THEIR PUBLISHED PAPERS. NOTE THAT "–" MEANS THE RESULTS ARE NOT AVAILABLE OR MISSING IN THE ORIGINAL PAPERS.

Methods	NCI1	PROTEINS	DD	MUTAG	COLLAB	RDT-B	RDT-M5K	IMDB-B	RDT-M12K	TWITTER-RGP
GL	–	–	–	81.66±2.11	–	77.34±0.18	41.01±0.17	65.87±0.98	–	–
WL	80.01±0.50	72.92±0.56	–	80.72±3.00	–	68.82±0.41	46.06±0.21	72.30±3.44	–	–
DGK	80.31±0.46	73.30±0.82	–	87.44±2.72	–	78.04±0.39	41.27±0.18	66.96±0.56	–	–
node2vec	54.89±1.61	57.49±3.57	–	72.63±10.20	–	–	–	–	–	–
sub2vec	52.84±1.47	53.03±5.55	–	61.05±15.80	–	71.48±0.41	36.68±0.42	55.26±1.54	–	–
graph2vec	73.22±1.81	73.30±2.05	–	83.15±9.25	–	75.78±1.03	47.86±0.26	71.10±0.54	–	–
RGCL	78.14±1.08	75.03±0.43	78.86±0.48	87.66±1.01	70.92±0.65	90.34±0.58	56.38±0.40	71.85±0.84	–	–
MVGRL	–	–	–	89.70±1.10	–	84.50±0.60	–	74.20±0.70	–	54.94±0.19
MVGRL(g)	–	–	–	89.52±1.22	–	83.08±0.80	–	73.13±0.65	–	54.67±1.1
MVGRL(f+g)	–	–	–	**90.59±1.21**	–	**84.90±0.56**	–	**74.60±0.74**	–	**55.53±0.37**
InfoGraph	76.20±1.06	74.44±0.31	72.85±1.78	89.01±1.13	70.65±1.13	82.50±1.42	53.46±1.03	**73.03**±0.87	30.21±1.09	52.78±0.04
InfoGraph(g)	77.00±0.65	**75.48±0.30**	73.31±0.16	88.46±1.10	70.78±0.93	**88.55±1.23**	55.63±1.08	71.73±0.61	32.75±1.58	52.77±0.04
InfoGraph(f+g)	**77.46±0.44**	**75.48±0.30**	**75.86±0.39**	**89.87±0.90**	**71.27±0.96**	**88.55±1.23**	**55.86±0.95**	72.16±**0.74**	**35.60±0.66**	**52.84±0.09**
GraphCL	77.87±0.41	74.39±0.45	78.62±0.40	86.80±1.34	71.36±1.15	89.53±0.84	55.99±0.28	71.14±0.44	32.86±0.24	52.20±0.25
GraphCL(g)	78.36±0.31	74.84±0.52	78.80±0.45	87.15±1.21	71.76±1.02	89.73±0.65	55.73±0.29	70.97±0.51	32.15±0.44	51.74±0.75
GraphCL(f+g)	**78.56±0.56**	**75.68±0.35**	**79.37±0.61**	**89.31±1.42**	**72.23±1.21**	**90.50±0.71**	**56.24±0.19**	**72.26±0.61**	**32.98±0.45**	**52.28±0.23**
JOAO	78.07±0.47	74.55±0.41	77.32±0.54	87.35±1.02	69.50±0.36	85.29±1.35	55.74±0.63	70.21±3.08	25.01±0.76	51.09±0.29
JOAO(g)	78.07±0.46	74.58±0.52	**78.95±0.47**	**88.46±0.98**	70.23±0.34	88.20±1.51	**56.20±0.31**	70.23±1.21	26.13± 0.46	**51.62±0.51**
JOAO(f+g)	**79.72±0.53**	**74.89±0.39**	**78.95±0.47**	**88.46±0.98**	**72.96±0.34**	**90.45±1.06**	**56.20±0.31**	**72.10±1.07**	**28.81±0.75**	**51.62±0.51**
SimGRACE	79.12±0.44	75.35±0.09	77.44±1.11	89.01±1.31	71.72±0.82	89.51±0.89	55.91±0.34	71.30±0.77	26.19±0.89	50.73±0.30
SimGRACE(g)	78.94±0.45	74.62±0.58	77.88±1.20	89.03±1.80	68.80±0.73	**89.82±0.97**	55.97±0.25	71.30±0.86	**31.15± 0.44**	**51.68±0.24**
SimGRACE(f+g)	**80.16±0.40**	**75.56±0.47**	**78.41±0.76**	**89.92±1.75**	**72.86±0.73**	**89.82±0.97**	**56.21±0.30**	**72.78±0.90**	**31.15± 0.44**	**51.68±0.24**



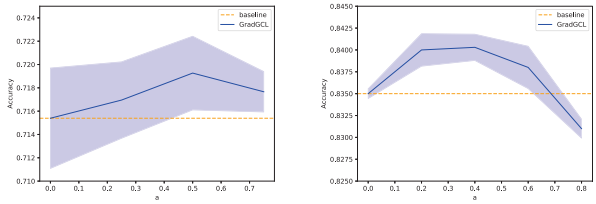
(a) DD (b) TWITTER-RGP (c) RDT-B (d) RDT-M12k

Fig. 8. Performance versus gradients loss weight (a) in graph classification tasks. The name (e.g. DD) is for the different datasets and the backbone models are GraphCL for (a) and (c), SimGRACE for (b) and JOAO for (d). The baseline in yellow dash line stand for the original model. The overall improvements in the accuracy with various gradients weight prove the effectiveness of our methods.

A linear classifier is trained and the results are reported by the mean accuracy and standard deviation on test node set.

Results: We follow the original settings of BGRL, GRACE, MVGRL, and COSTA, thus presenting two tables Tab. V and Tab. VII for the performance comparison. For BGRL, the improved model outperforms the original one shown in Tab. V. In Tab. VII for GRACE and MVGRL, we can see in the performance on the node classification improved on the three benchmark datasets except for the PubMed with GRACE model. However, the improvement on node classification tasks is less significant compared with that on the graph classification tasks. Node classification aims to make predictions at the node level, rather than the graph level. Unlike graph classification tasks, nodes are not independent of each other. The gradients may not capture much of this interdependent information because they are computed on an individual instance without aggregating neighborhood gradients.

Hyperparameter sensitivity: In Figure 9, when increasing the gradients weight, the performance curve first goes up and then drops with large gradients. The improvement is less significant compared with that on graph classification task. This corresponds to the definition of gradients without aggregating the neighborhood gradients and thus can not fully represent the node information.



(a) CiteSeer (b) Cora

Fig. 9. Performance versus gradients loss weight (a) in Node classification tasks with GRACE backbone on CiteSeer dataset and MVGRL backbone on Cora dataset. The baseline in yellow dash line stand for the original model. The overall improvements in the accuracy with various gradients weight prove the effectiveness of our methods.

C. Transfer learning

The performance improvement is significant in the unsupervised graph classification tasks. We further show the whether the improved representation can benefit the transfer-ability.

Experiment setup: Transfer learning includes two phases. The model is first pre-trained on a large datasets and then fine-tuned on the downstream tasks. To evaluate the robustness of

TABLE V
THE PERFORMANCE COMPARISON ON THE NODE CLASSIFICATION TASK WITH IMPROVED BGRL AND SGCL MODEL. THE RESULTS OF OTHER MODELS ARE REPORTED IN THEIR PAPER.

Methods	WikiCS	Am. Comp.	Am. Photos	Co.CS	Co.Phy	ogbn-Arxiv
Raw features	71.98 ± 0.00	73.81 ± 0.00	78.53 ± 0.00	90.37 ± 0.00	93.58 ± 0.00	-
deepwalk	74.35 ± 0.06	85.68 ± 0.06	89.44 ± 0.11	84.61 ± 0.22	91.77 ± 0.15	-
deepwalk + feat.	77.21 ± 0.03	86.28 ± 0.07	90.05 ± 0.08	87.70 ± 0.04	94.90 ± 0.09	-
Supervised GCN	77.19 ± 0.12	86.51 ± 0.54	92.42 ± 0.22	93.03 ± 0.31	95.65 ± 0.16	71.74 ± 0.29
DGI	75.35 ± 0.14	83.95 ± 0.47	91.61 ± 0.22	92.15 ± 0.63	94.51 ± 0.52	65.10 ± 0.40
GCA	78.35 ± 0.05	88.94 ± 0.15	92.53 ± 0.16	93.10 ± 0.01	95.73 ± 0.03	68.20 ± 0.20
BGRL	79.98 ± 0.10	**90.34** ± 0.19	93.17 ± 0.30	93.31 ± 0.13	95.73 ± 0.05	71.75 ± 0.17
BGRL(f+g)	**80.83 ± 0.13**	89.79 ± **0.22**	**93.32 ± 0.27**	**93.33 ± 0.12**	**95.96 ± 0.10**	-
SGCL	*79.83 ± 0.54*	*90.46 ± 0.31*	*93.44 ±0.28*	93.29 ± 0.17	95.78 ± 0.11	*70.75 ± 0.10*
SGCL(f+g)	**79.93 ± 0.50**	**90.56 ± 0.32**	**93.47 ±0.32**	**93.32 ± 0.18**	**95.79 ± 0.08**	**70.89 ± 0.06**

TABLE VI
THE PERFORMANCE COMPARISON ON THE TRANSFER LEARNING TASK. THE RESULTS OF OTHERS ARE REPORTED FROM THEIR PUBLISHED REPORTS.

Pre-train dataset	PPI-306K	ZINC2M								Avg.
Fine-tune dataset	PPI	BBBP	ToxCast	SIDER	BACE	ClinTox	MUV	Tox21	HIV	
No Pre-Train	64.8	74.0	63.4	57.3	70.1	58.0	71.8	65.8	75.3	66.76
AttrMasking	65.2	76.7	64.2	61.0	79.3	71.8	74.7	64.3	77.2	70.44
ContextPred	64.4	75.7	63.9	60.9	79.6	65.9	75.8	68.0	77.3	70.18
SimGRACE	70.25	**71.25**	**63.36**	60.59	75.00	75.60	**76.90**	75.60	75.20	71.52
SimGRACE(f+g)	**70.77**	71.08	62.92	**61.12**	**75.96**	**75.90**	76.21	**75.65**	**75.36**	**71.66**
GraphCL	67.88	68.00	**63.09**	**60.09**	75.38	**75.99**	69.80	73.87	**78.47**	70.28
GraphCL(f+g)	**69.57**	**70.05**	62.84	59.01	**76.6**	75.11	**75.20**	**75.58**	77.75	**71.30**

our methods, we conduct experiments for both proteins function for biology domain and molecular property predication with chemical datasets as [19], [59], [66] did.

Datasets: The datasets statistics are listed in Table III. Moreover,we pretrain the model on PPI-306K datasets [71] for protein function prediction and on ZINC15 [46] for molecule property prediction. And the model is then finetuned on the PPI datasets and MoleculeNet [58] respectively. PPI-306K represents protein-protein interaction network and is used for protein function prediction. ZINC15 are molecules sample dataset. For pre-training, 2 million molecule samples without labels are used. We select 3 benchmark datasets in MoleculeNet for finetuning as [59].

Baselines: We show the transferability of gradient contrastive learning on two previous contrastive learning model GraphCL [66] and SimGRACE [59].

Evaluation: Experiments results for each datasets are reported by mean and standard deviation of ROC-AUC scores after 10 times running as [66]. We use the same GIN as the GNN encoder with the original models introduced in [19].

Results: From Tab. VI, we can see our methods improve the average performance on the transfer learning tasks of previous methods. For the PPI dataset, the improvement is significant for both models. It demonstrates that the model trained by contrasting with gradients improved the robustness and transferability of contrastive learning models. However, for the performance that transferred from ZINC2M, there is not a universally beneficial strategy for transfer learning task, which is consistent with the observations in previous articles [59], [66].

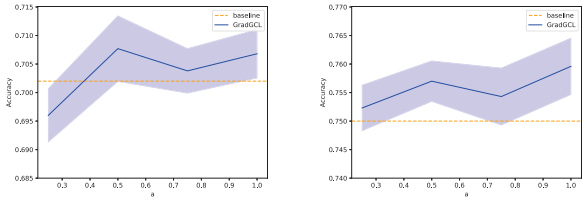
Hyperparameter sensitivity: Shown by the results in Fig. 10 reported from the PPI and BACE datasets, a larger gradient weight usually brings the improved transferability. The trend of performance is first increase then drops but the sweet zone relatively large. This shows that gradients generally improve the tranfer learning of previous methods.

TABLE VII
THE PERFORMANCE COMPARISON WITH OTHER BASELINES ON THE NODE CLASSIFICATION TASK. THE RESULTS OF MVGRL ARE FROM PREVIOUS PUBLISHED PAPERS. FOR GRACE, THE RESULTS ARE PRODUCED BY RUNNING THEIR PUBLISHED CODE.

Methods	Cora	CiteSeer	PubMed
GRACE	82.86 ± 0.55	71.53 ± 0.77	**86.70± 0.10**
GRACE(f+g)	**83.08 ± 0.70**	**71.92 ± 0.57**	86.21 ± 0.13
MVGRL	83.5 ± 0.40	73.30 ± 0.50	**80.10 ± 0.70**
MVGRL(f+g)	**84.03 ± 0.45**	**73.39 ± 0.60**	80.04 ± 0.35
COSTA	84.3±0.2	72.90±0.30	**86.0±0.20**
COSTA(f+g)	**85.01±0.30**	**73.05±0.13**	85.75±0.18

D. Hyperparameter sensitivity

Weight of Gradients Loss a**.** Generally, the weight of gradients loss is essential. When the weight is 0, the model degrades to the representation-only case When the weight is 1, the model only uses gradients to learn helpful information Generally, for different models and datasets, the optimal weight may vary. As shown in Fig. 10 reported from the PPI dataset, a larger gradient weight usually brings improved transferability. The trend of performance is first increasing and then dropping but the sweet zone is relatively large. This shows that gradients generally improved the tranfer learning of previous methods. In Fig. 9, when increasing the gradients weight, the performance curve first goes up and then drops with large gradients. The improvement is less significant compared with that on the graph classification task. This corresponds to the definition of gradients without aggregating the neighborhood gradients and thus can not fully represent the node information.



(a) PPI (b) BACE

Fig. 10. Performance versus gradients loss weight (a) in Transfer learning tasks with SimGRACE backbone on PPI dataset and GraphCl model on BACE datasets. The overall improvements in the accuracy with various gradients weight prove the effectiveness of our methods.

TABLE VIII
COMPARISON OF EFFICIENCY OF DIFFERENT BACKBONE MODELS ON VARIOUS DATASETS.

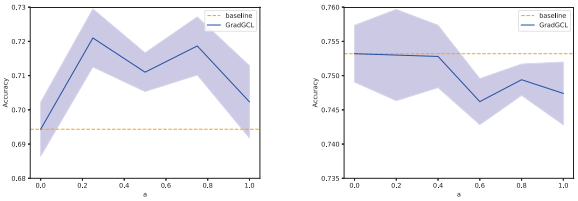
Dataset	Model	Training Time(s)
DD	InfoGraph	439
	InfoGraph(f+g)	445
PROTEINS	GraphCL	31
	GraphCL(f+g)	33
IMDB	JOAO	45
	JOAO(f+g)	46
RDT-B	SimGRACE	86
	SimGRACE(f+g)	91

E. Efficiency (Training time)

In Table VIII, we show the training time comparison with various backbone model on different datasets with the same epochs. The gradient loss will introduce extra computation and increase the training time by 2% - 6%.

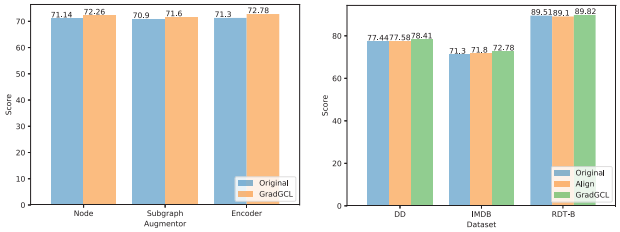
F. Ablation study

We first show that gradients alone can maximize the mutual information in Tab. IV with XXX(g). Shown by XXX(f+g), when combined with representations, the performance further improves. Further, since ours can improve the representation quality by alignment and uniformity, in Fig. 12(b), we compare the performance of using alignment loss directly in [53] as ablation studies. Alignment loss improve performance of SimGRACE but ours is better since we provide extra graph data information. Besides, we also show the results of different augmenters from different models, ours can improve the performance for different augmentation techniques. Moreover, we compared the performance of various augmentations techniques in Fig. 12(a). GradGCL can successfully work across the augmenters. For different loss function like InfoNCE, JSD [13], SCE [17], we show the result for various loss types since the gradients are calculated based on the different loss functions. InfoNCE loss can be used to estimate the mutual information, however SCE loss can not. We use GraphCL with infoNCE loss, MVGRL with JSD loss, GraphMAE with SCE loss and train them on the IMDB-BINARY dataset in unsupervised graph classification tasks. As shown in Fig. 11, our GradGCL works for contrastive loss like NCE loss but failed on SCE loss, which is from GraphMAE, a generative self-supervised learning without contrastive loss. Incorporating gradient weight degrades the model performance.



(a) InfoNCE (b) SCE

Fig. 11. Results of unsupervised graph classification tasks with different weights on gradient contrastive learning on IMDB-BNARY datasets for different loss type: infoNCE, and SCE. The yellow dash line represents the original model as the baseline. Since SCE are reconstruction loss rather than contrastive loss, our methods fail to improve the performance.



(a) Ablation:augmentor (b) Ablation:alignment loss

Fig. 12. Ablation study on (a) Different augmenters: Node for node dropping, Subgraph for subgraph sampling with GraphCL and Encoder for encoder perturbation with SimGRACE on IMDB datasets. (b) The comparison with alignment loss (Align) with ours(GradGCL) and the original methods.

V. CONCLUSION

Previous GCL methods often suffer from the dimensional collapse issue, which leads to less informative representations. We found this problem came from the strong assumption that all positive samples should be close and all negative samples should be far apart in the representation space. In this paper, we propose using gradients in GCL to mitigate the dimensional collapse and improve graph representation quality. Our method, gradient contrastive learning (GradGCL), can be used alone or in combination with existing GCL models. Through experiments on various loss types, datasets, and model architectures for different graph tasks, we demonstrate that using gradients alone can perform similarly to previous GCL methods and combining GradGCL with current GCL models leads to improvements in most cases.

ACKNOWLEDGMENT

Lei Chen's work is partially supported by National Key Research and Development Program of China Grant No. 2023YFF0725100, National Science Foundation of China (NSFC) under Grant No. U22B2060, the Hong Kong RGC GRF Project 16213620, RIF Project R6020-19, AOE Project AoE/E-603/18, Theme-based project TRS T41-603/20R, CRF Project C2004-21G, Guangdong Province Science and Technology Plan Project 2023A0505030011, Hong Kong ITC ITF grants MHX/078/21 and PRP/004/22FX, Zhujiang scholar program 2021JC02X170, Microsoft Research Asia Collaborative Research Grant and HKUST-Webank joint research lab grants. Xiaofang Zhou's work is partially conducted in JC STEM Lab of Data Science Foundations funded by The Hong Kong Jockey Club Charities Trust.

REFERENCES

[1] Bijaya Adhikari, Yao Zhang, Naren Ramakrishnan, and B Aditya Prakash. Sub2vec: Feature learning for subgraphs. In *Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part II 22*, pages 170–182. Springer, 2018.

[2] Dmitrii Babaev, Nikita Ovsov, Ivan Kireev, Maria Ivanova, Gleb Gusev, Ivan Nazarov, and Alexander Tuzhilin. Coles: contrastive learning for event sequences with self-supervision. In *Proceedings of the 2022 International Conference on Management of Data*, pages 1190–1199, 2022.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020.

[4] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] Shimin Di and Lei Chen. Message function search for knowledge graph embedding. In Ying Ding, Jie Tang, Juan F. Sequeda, Lora Aroyo, Carlos Castillo, and Geert-Jan Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2633–2644. ACM, 2023.

[7] Shimin Di, Yanyan Shen, and Lei Chen. Relation extraction via domain-aware transfer learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 1348–1357. ACM, 2019.

[8] Shimin Di, Quanming Yao, and Lei Chen. Searching to sparsify tensor decomposition for n-ary relational data. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 4043–4054. ACM / IW3C2, 2021.

[9] Paul D Dobson and Andrew J Doig. Distinguishing enzyme structures from non-enzymes without alignments. *Journal of molecular biology*, 330(4):771–783, 2003.

[10] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[11] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[12] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[13] Kaveh Hassani and Amir Hosein Khas Ahmadi. Contrastive multi-view representation learning on graphs. *ArXiv*, abs/2006.05582, 2020.

[14] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[15] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier, 1992.

[16] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[17] Zhenyu Hou, Xiao Liu, Yuxiao Dong, Chunjie Wang, Jie Tang, et al. Graphmae: Self-supervised masked graph autoencoders. *arXiv preprint arXiv:2205.10803*, 2022.

[18] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[19] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[20] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. Gpt-gnn: Generative pre-training of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1857–1867, 2020.

[21] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

[22] Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. *arXiv preprint arXiv:2110.09348*, 2021.

[23] Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pages 5464–5474. PMLR, 2021.

[24] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[25] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[26] Gukyeong Kwon, Mohit Prabhushankar, Dogancan Temel, and Ghassan AlRegib. Backpropagated gradient representations for anomaly detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*, pages 206–226. Springer, 2020.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[28] Haoyang Li, Shimin Di, and Lei Chen. Revisiting injective attacks on recommender systems. In *Advances in Neural Information Processing Systems 35: NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[29] Haoyang Li, Shimin Di, Zijian Li, Lei Chen, and Jiannong Cao. Black-box adversarial attack and defense on graph neural networks. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*, pages 1017–1030. IEEE, 2022.

[30] Ling Li, Siqiang Luo, Yuhai Zhao, Caihua Shan, Zhengkui Wang, and Lu Qin. Coclep: Contrastive learning-based semi-supervised community search. *IEEE 39th ICDE*, 2023.

[31] Sihang Li, Xiang Wang, An zhang, Yingxin Wu, Xiangnan He, and Tat-Seng Chua. Let invariant rationale discovery inspire graph contrastive learning, 2022.

[32] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and S Yu Philip. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(6):5879–5900, 2022.

[33] Xiao Luo, Wei Ju, Meng Qu, Chong Chen, Minghua Deng, Xian-Sheng Hua, and Ming Zhang. Dualgraph: Improving semi-supervised graph classification via dual contrastive learning. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 699–712. IEEE, 2022.

[34] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

[35] Fangzhou Mu, Yingyu Liang, and Yin Li. Gradients as features for deep representation learning. *arXiv preprint arXiv:2004.05529*, 2020.

[36] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

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

[38] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. *arXiv preprint arXiv:1802.04407*, 2018.

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[40] Kaspar Riesen and Horst Bunke. Iam graph database repository for graph based pattern recognition and machine learning. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 287–297. Springer, 2008.

[41] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 33:12559–12571, 2020.

[42] Benedek Rozemberczki, Oliver Kiss, and Rik Sarkar. An api oriented open-source python framework for unsupervised learning on graphs. *arXiv preprint arXiv:2003.04819*, 10(3340531.3412757), 2020.

[43] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[44] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[45] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

[46] Teague Sterling and John J Irwin. Zinc 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11):2324–2337, 2015.

[47] Fan-Yun Sun, Jordan Hoffmann, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *ArXiv*, abs/1908.01000, 2020.

[48] Wangbin Sun, Jintang Li, Liang Chen, Bingzhe Wu, Yatao Bian, and Zibin Zheng. Rethinking and simplifying bootstrapped graph latents. *arXiv preprint arXiv:2312.02619*, 2023.

[49] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Rémi Munos, Petar Veličković, and Michal Valko. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*, 2021.

[50] Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang. Mgae: Marginalized graph autoencoder for graph clustering. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 889–898, 2017.

[51] Jiayi Wang, Chengliang Chai, Nan Tang, Jiabin Liu, and Guoliang Li. Coresets over multiple tables for feature-rich and data-efficient machine learning. *Proceedings of the VLDB Endowment*, 16(1):64–76, 2022.

[52] Runhui Wang, Yuliang Li, and Jin Wang. Sudowoodo: Contrastive self-supervised learning for multi-purpose data integration and preparation. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 1502–1515. IEEE, 2023.

[53] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

[54] Wei Wang, Meihui Zhang, Gang Chen, HV Jagadish, Beng Chin Ooi, and Kian-Lee Tan. Database meets deep learning: Challenges and opportunities. *ACM Sigmod Record*, 45(2):17–22, 2016.

[55] Zhili Wang, Shimin Di, and Lei Chen. Autogel: An automated graph neural network with explicit link information. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 24509–24522, 2021.

[56] Zhili Wang, Shimin Di, and Lei Chen. A message passing neural network space for better capturing data-dependent receptive fields. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 2489–2501. ACM, 2023.

[57] Zhili Wang, Shimin Di, Lei Chen, and Xiaofang Zhou. Search to fine-tune pre-trained graph neural networks for graph-level tasks. *arXiv preprint arXiv:2308.06960*, 2023.

[58] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[59] Jun Xia, Lirong Wu, Jintao Chen, Bozhen Hu, and Stan Z. Li. Simgrace: A simple framework for graph contrastive learning without data augmentation. *Proceedings of the ACM Web Conference 2022*, 2022.

[60] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 1259–1273. IEEE, 2022.

[61] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[62] Yiming Xu, Bin Shi, Teng Ma, Bo Dong, Haoyi Zhou, and Qinghua Zheng. Cldg: Contrastive learning on dynamic graphs. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 696–707. IEEE, 2023.

[63] Pinar Yanardag and SVN Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1365–1374, 2015.

[64] Sean Bin Yang, Chenjuan Guo, Jilin Hu, Bin Yang, Jian Tang, and Christian S Jensen. Temporal path representation learning with weakly-supervised contrastive curriculum learning. In *ICDE*, 2022.

[65] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In *ICML*, 2021.

[66] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *ArXiv*, abs/2010.13902, 2020.

[67] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[68] Yifei Zhang, Hao Zhu, Zixing Song, Piotr Koniusz, and Irwin King. Costa: covariance-preserving feature augmentation for graph contrastive learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2524–2534, 2022.

[69] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

[70] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph contrastive learning with adaptive. In *Proceedings of the Web Conference 2021*, pages 2069–2080, 2021.

[71] Marinka Zitnik, Rok Sosič, Marcus W Feldman, and Jure Leskovec. Evolution of resilience in protein interactomes across the tree of life. *Proceedings of the National Academy of Sciences*, 116(10):4426–4433, 2019.