Digraph Contrastive Learning

Abstract

Graph Contrastive Learning (GCL) has emerged to learn generalizable representations from contrastive views. However, it is still in its infancy with two concerns: 1) changing the graph structure through data augmentation to generate contrastive views may mislead the message passing scheme, as such graph changing action deprives the intrinsic graph structural information, especially the directional structure in digraphs; 2) since GCL usually generates a limited number of contrastive views, it does not take full advantage of the contrastive information provided by data augmentation, resulting in incomplete structure information for models learning. In this paper, we design a digraph data augmentation method called Laplacian perturbation and theoretically analyze how it provides contrastive information without changing the digraph structure. Moreover, we present the multi-view digraph contrastive learning framework, which learns from all possible contrastive views generated by Laplacian perturbation. Then we train it using multi-task curriculum learning to progressively learn from multiple easy-to-hard contrastive views. We empirically show that our model can retain more structural features of digraphs than other GCL models because of its ability to provide complete contrastive information. Experiments on various benchmarks reveal our dominance over the state-of-the-art approaches.

1 Introduction

There is a growing interest in learning from **digraphs (directed graphs)** using Graph Neural Networks (GNNs) [16, 11, 17] to tackle practical problems [1, 6, 20]. Current GNNs designed for learning from digraphs [22, 38, 39] are trained end-to-end under supervision. This training scheme shows excellent performance by virtue of enough labeled data. Correspondingly, to make use of rich unlabeled data, several Graph Contrastive Learning (GCL) [53, 58, 32, 12] works are proposed based on GNNs and Contrastive Learning (CL) [4, 40, 13]. They utilize data augmentation methods to generate contrastive views from the original graphs, then force views generated from the same instance (node or graph) closer while views from different instances apart using *InfoNCE-like* [27] objective function. However, current GCL methods encounter some issues in processing digraphs, mainly in *data augmentation method* and *contrastive learning framework*.

Firstly, most current *data augmentation* methods used in GCL [53, 58, 57] do not take the digraph structure into account and may discard distinctive direction information. For example, the idea of dropping nodes/edges [57, 53] is borrowed from random erasing used in images [56] , which overlooks the discrepancy of nodes and edges in different graph structures. Moreover, by grasping contrastive information through changing graph structure, part of the distinctive direction information, *e.g.,* irreversible time-series relationships in digraphs, will inevitably be lost. The message passing scheme will also be mislead, making it difficult for GNN-based encoders to learn from digraphs. To the best of our knowledge, there is a lack of data augmentation methods that are specifically designed for digraphs to retain the original digraph structure while providing enough contrastive information.

Besides, common *contrastive learning framework* can only learn from a limited number of contrastive views [53, 58, 12]. However, because of the complex structure of digraphs, it is insufficient to use a small number of contrastive views to fully understand their structural characteristics [38]. Furthermore, since the number of contrastive views has to be determined before the training, it becomes another problem to select the ideal views for the downstream task during data pre-processing, about which we actually have no knowledge in advance [37, 48]. Several works [36, 50] try to obtain more information by increasing the number of contrastive views. However, they still have to pre-define the views and trade off between the number of views and the training difficulty.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

To address these two issues, we first propose a digraph data augmentation method called **Laplacian perturbation**. Since the contrastive views are passed to the encoder in the form of Laplacian matrix, a desirable data augmentation method is to perturb the Laplacian matrix without changing the digraph structure. To achieve this, we adopt the approximate digraph Laplacian [38] where a teleport probability is introduced to control the degree of approximation. Thus, by adding a perturbation term to this probability, we can simply augment the Laplacian matrix without altering the digraph structure. As it is time-consuming to calculate the Laplacian matrix, we then speed it up using the power method [23]. In addition, from the theoretically analysis, we find that the Laplacian perturbation is essentially a perturbation to the digraph entropy. The perturbation term essentially determines the magnitude of the entropy perturbation error, which affects the magnitude of the contrastive information.

To learn the complete structural characteristics of digraphs, the first step is generating an arbitrary number of contrastive views with Laplacian perturbation. Next, we introduce a generalized multi-view contrastive learning objective function and adopt it to design a **digraph contrastive learning framework**. This objective function maximizes the sum of the mutual information between the representations of all contrastive views, which motivates the encoder to learn a generalized representation from all these views. However, this multi-view objective function is hard to optimize. Thus, we leverage the multi-task curriculum learning strategy [31, 33, 9, 25] to divide multiple contrastive views into sub-tasks with various difficulties and progressively learn from easy-to-hard sub-tasks. In this way, we can learn all views step-by-step to reach the final multi-view objective. Moreover, learning from all contrastive views eliminates the needs to adjust the augmentation parameters anymore.

Through experiments, we empirically show that our **Di**graph **C**ontrastive **L**earning (**DiGCL**) outperforms the competitive baselines in the settings of unsupervised and supervised learning. Systematic analysis is also carried out to analyze the performance of various augmentations on the mainstream benchmarks and the impact of different pacing functions on the performance of digraph contrastive learning. The source codes are provided in the Supplementary Material for reproducibility.

2 Digraph Data Augmentation

In this section, we first design a digraph data augmentation scheme named Laplacian perturbation. As it is time-consuming to calculate the Laplacian matrix, we then speed it up using the power method [23]. Finally, we analyze the proposed data augmentation scheme theoretically.

2.1 Digraph Laplacian and its Approximation

Formally, let a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, its adjacency matrix can be denoted as $\mathbf{A} = \{0,1\}^{n\times n}$, where $n = |\mathcal{V}|$. The nodes are described by the feature matrix $\mathbf{X} \in \mathbb{R}^{n\times c}$, with the number of features c per node. Intuitively, the data augmentation can be performed at the topological- and feature-level, operating on $\mathbf{A}$ and $\mathbf{X}$, respectively [53, 58]. The distinctive attribute of digraphs is the directionality of the edges, which leads us to focus on *topological node-level* data augmentation.

The augmented digraph is fed to the GNN-based encoder in the form of Laplacian matrix to learn the representation. The transition matrix $\mathbf{P}$ of $\mathcal{G}$ is defined as $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$, where the diagonal degree matrix $\mathbf{D}(u,u) = \sum_{v\in\mathcal{V}} \mathbf{A}(u,v)$. As stated in Chung [5], the normalized Laplacian matrix of $\mathcal{G}$ can be defined as $\mathbf{L}(\mathcal{G}) = \mathbf{I} - \frac{1}{2}\left(\mathbf{\Pi}^{\frac{1}{2}}\mathbf{P}\mathbf{\Pi}^{-\frac{1}{2}} + \mathbf{\Pi}^{-\frac{1}{2}}\mathbf{P}^T\mathbf{\Pi}^{\frac{1}{2}}\right)$, where $\mathbf{\Pi} = \frac{1}{\|\pi\|_1}\mathrm{Diag}(\pi)$ and π is the eigenvector of the $\mathbf{P}$. However, $\mathcal{G}$ may contain isolated nodes or could be formed into bipartite graph. Thus, $\mathbf{P}$ is not necessarily *irreducible* and *aperiodic*, we can not guarantee that π is unique [38].

To relax this constraint, Tong *et al.* [38] add self-loops to the original digraph to ensure $\mathcal{G}$ to be *aperiodic*. Then they redefine the transition matrix based on *personalized* PageRank with the teleport probability $\alpha \in (0,1)$ to guarantee the redefined matrix to be *irreducible*. Finally, they obtain approximate eigenvector π_{appr} as

$$(1-\alpha)\pi_{\mathrm{appr}}\tilde{\mathbf{P}} + \frac{1}{n}\frac{\alpha}{1+\alpha}\mathbf{1}^{1\times n} = \pi_{\mathrm{appr}}, \tag{1}$$

where $\tilde{\mathbf{P}} = \tilde{\mathbf{D}}^{-1}\tilde{\mathbf{A}}$, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}^{n\times n}$ denotes the adjacency matrix with added self-loops and $\tilde{\mathbf{D}}(u,u) = \sum_{v\in\mathcal{V}} \tilde{\mathbf{A}}(u,v)$. They define the approximate digraph Laplacian $\mathbf{L}_{\mathrm{appr}}(\mathcal{G},\alpha)$ as

$$\mathbf{L}(\mathcal{G}) \approx \mathbf{L}_{\mathrm{appr}}(\mathcal{G},\alpha) = \mathbf{I} - \frac{1}{2}\left(\mathbf{\Pi}_{\mathrm{appr}}^{\frac{1}{2}}\tilde{\mathbf{P}}\mathbf{\Pi}_{\mathrm{appr}}^{-\frac{1}{2}} + \mathbf{\Pi}_{\mathrm{appr}}^{-\frac{1}{2}}\tilde{\mathbf{P}}^T\mathbf{\Pi}_{\mathrm{appr}}^{\frac{1}{2}}\right), \tag{2}$$

where $\mathbf{\Pi}_{\text{appr}} = \frac{1}{\|\pi_{\text{appr}}\|_1}\text{Diag}(\pi_{\text{appr}})$. Note that we follow the [38] and set $\alpha = 0.1$ in this paper.

According to Eq. (2), existing data augmentation methods [53, 58], e.g., *node/edge dropping, subgraph sampling*, need to obtain the contrastive information by changing $\tilde{\mathbf{P}}$. Their use of sampling-based methods inevitably corrupts the digraph structure and thus misleads the message passing in the GNN-based encoder. A desirable data augmentation method on $\mathbf{L}_{\text{appr}}$ is to perturb π_{appr} reasonably without altering the digraph structure $\tilde{\mathbf{P}}$. [1]

2.2 Digraph Data Augmentation with Laplacian Perturbation

From Eq. (1), it is easy to find that the eigenvector π_{appr} depends on the $\tilde{\mathbf{P}}$ and α. In other words, we can shift the teleport probability α without changing the digraph structure $\tilde{\mathbf{P}}$, thus altering the eigenvector π_{appr} and finally perturbing the Laplacian matrix $\mathbf{L}_{\text{appr}}$.

Definition 1 ***Laplacian perturbation***. *Given a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and the teleport probability α, we define the Laplacian perturbation opertation $\Phi(\cdot)$ on the $\mathbf{L}_{\text{appr}}(\tilde{\mathbf{P}}, \alpha)$ as*

$$\Phi_{\Delta\alpha}(\mathcal{G}, \alpha) = \mathbf{L}_{\text{appr}}(\mathcal{G}, \alpha + \Delta\alpha), \tag{3}$$

where $\Delta\alpha$ is a perturbation term that satisfies $\Delta\alpha \geqslant 0$ and $\alpha + \Delta\alpha \in (0, 1)$.

Through this operation, the digraph structure and the sparsity of the Laplacian matrix are maintained, which means there is no training burden to the subsequent GNN-based encoder. However, using this operation for data augmentation is very time-consuming, since the time complexity of computing Eq. (1) is $\mathcal{O}\left(n^2\right)$. To deal with it, we design an accelerating algorithm for computing the approximate eigenvalue based on the power method [23].

We take advantage of the sparsity of the transition matrix $\tilde{\mathbf{P}}$ to compute the approximate eigenvector π_{appr} and rewrite Eq. (1) to the form of discrete-time Markiv chain at time t as

$$\pi_{\text{appr}}^{t+1} = (1-\alpha)\pi_{\text{appr}}^{t}\tilde{\mathbf{P}} + \frac{1}{n}\frac{\alpha}{1+\alpha}\mathbf{1}^{1\times n}. \tag{4}$$

As stated in Tong *et al.* [38], π_{appr}^{t} has a special property that $\pi_{\text{appr}}^{t}\mathbf{1}^{n\times 1} = \frac{1}{1+\alpha}$. We then multiply the last term of Eq. (4) with $(1+\alpha)\pi_{\text{appr}}^{t}\mathbf{1}^{n\times 1}$ to

$$\pi_{\text{appr}}^{t+1} = (1-\alpha)\pi_{\text{appr}}^{t}\tilde{\mathbf{P}} + \frac{1}{n}\frac{\alpha}{1+\alpha}\mathbf{1}^{1\times n}\left[(1+\alpha)\pi_{\text{appr}}^{t}\mathbf{1}^{n\times 1}\right] = (1-\alpha)\pi_{\text{appr}}^{t}\tilde{\mathbf{P}} + \frac{\alpha}{n}\pi_{\text{appr}}^{t}. \tag{5}$$

We can solve Eq. (5) iteratively using the power method [23]. Therefore, the complexity decreases to $\mathcal{O}(nk)$, where k is the number of iteration times. Due to page limit, we do not discuss the number of iterations and convergence rate here. Please refer to [21] for more details. We take $k = 100$ and the tolerance as $1\text{e}{-6}$ throughout. It can be seen easily that for large-scale graphs, our method can significantly improve the speed and makes it possible to do Laplacian perturbation during the training, which is the basis for the framework proposed in Section 3. In addition, we compare the running time of our fast Laplacian perturbation with different data augmentation methods in Section 5.2.

2.3 Theoretical Justification of Laplacian Perturbation

Noticing the inherent connections between graph Laplacian and von Neumann entropy [51], we can use the proprieties of von Neumann entropy to help us to analyze the Laplacian perturbation operation quantitatively. We start with defining the von Neumann entropy of digraphs based on the $\mathbf{L}_{\text{appr}}$.

Definition 2 ***Digraph von Neumann entropy***. *Given a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, its von Neumann entropy [51] based on the $\mathbf{L}_{\text{appr}}$ is defined as*

$$\tilde{\mathcal{H}}_{\text{VN}}(\mathcal{G}, \alpha) = 1 - \frac{1}{n} - \frac{1}{2n^2}\left\{\sum_{(u,v)\in\mathcal{E}}\left(\frac{1}{d_u^{out}\, d_v^{out}} + \frac{\pi_{\text{appr}}(u)}{\pi_{\text{appr}}(v) d_u^{out^2}}\right) - \sum_{(u,v)\in\tilde{\mathcal{E}}}\frac{1}{d_u^{out}\, d_v^{out}}\right\}, \tag{6}$$

where $\tilde{\mathcal{E}} = \{(u,v) \mid (u,v) \in \mathcal{E}$ and $(v,u) \notin \mathcal{E}\}$. $d_u^{\text{in}} = \sum_{v\in\mathcal{V}}\mathbf{A}(v,u)$, $d_u^{out} = \sum_{v\in\mathcal{V}}\mathbf{A}(u,v)$ are the in- and out-degree of the node u.

[1] Note that adding self-loops to transform $\mathbf{P}$ into $\tilde{\mathbf{P}}$ is a common trick [16, 45], and we ignore the effect of this operation on the digraph structure.

Referring to Eq. (6), since the Laplacian perturbation does not involve the nodes/edges, the number of nodes n and the degrees of nodes d_u (edge distribution) will not change. Conversely, the approximate eigenvector π_{appr} has been changed, which means the essence of this operation is a perturbation to the digraph entropy. We further introduce the perturbation error to quantify this impact.

DEFINITION 3 ***Perturbation error***. *Given a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the teleport probability α and the perturbation term $\Delta\alpha$, we define the perturbation error of digraph von Neumann entropy caused by Laplacian perturbation as*

$$\Delta\tilde{\mathcal{H}}_{\text{VN}}(\alpha, \alpha+\Delta\alpha) = \tilde{\mathcal{H}}_{\text{VN}}(\mathcal{G}, \alpha) - \tilde{\mathcal{H}}_{\text{VN}}(\mathcal{G}, \alpha+\Delta\alpha) = \frac{1}{2n^2}\left\{\sum_{(u,v)\in\mathcal{E}}\left(\frac{\pi^{\alpha+\Delta\alpha}_{\text{appr}}(u)}{\pi^{\alpha+\Delta\alpha}_{\text{appr}}(v)d_u^{out^2}} - \frac{\pi^{\alpha}_{\text{appr}}(u)}{\pi^{\alpha}_{\text{appr}}(v)d_u^{out^2}}\right)\right\}, \tag{7}$$

where we remark π_{appr} with its parameters $\alpha + \Delta\alpha$, α as $\pi^{\alpha+\Delta\alpha}_{\text{appr}}$, $\pi^{\alpha}_{\text{appr}}$ separately.

THEOREM 1 ***Monotonicity of the perturbation error***. *The perturbation error $\Delta\tilde{\mathcal{H}}_{\text{VN}}$ increases monotonically with the Laplacian perturbation term $\Delta\alpha$.*

THEOREM 2 ***Bounds on the perturbation error***. *Given a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and the teleport probability α, the inequality*

$$0 < \Delta\tilde{\mathcal{H}}_{\text{VN}}(\alpha, \alpha+\Delta\alpha) < \frac{1}{2n^2}\left\{\sum_{(u,v)\in\mathcal{E}}\left(\frac{1}{d_u^{out^2}} - \frac{\pi^{\alpha}_{\text{appr}}(u)}{\pi^{\alpha}_{\text{appr}}(v)d_u^{out^2}}\right)\right\} \tag{8}$$

holds. When the perturbation term $\Delta\alpha = 0$, $\Delta\tilde{\mathcal{H}}_{\text{VN}} = 0$ and when $\Delta\alpha \to 1-\alpha$, the perturbation error $\Delta\tilde{\mathcal{H}}_{\text{VN}}$ towards the upper bound.

We show in THEOREM 1 that Laplacian perturbation can provide contrastive information in various magnitude for GNN-based encoder to learn the representations by changing $\Delta\alpha$. Meanwhile, since Laplacian perturbation does not need to alter the digraph structure and the number of nodes n is generally high, the perturbation error will be in a very small range as shown in THEOREM 2. This can help the encoder to focus more on the digraph structure rather than just learning from the errors. Proofs of THEOREM 1 and 2 are attached in the Supplementary Material.

3 Digraph Contrastive Learning

In this section, we introduce the digraph contrastive learning framework with a multi-view objective, and then we present **DiGCL** to progressively learn from multiple easy-to-hard contrastive views.

3.1 Multi-view Digraph Contrastive Learning

As we have defined the digraph data augmentation in Eq. (3), we can follow the common graph contrastive learning (GCL) paradigm to construct a digraph contrastive learning framework.

Digraph Contrastive Learning. GCL seeks to maximize the mutual information (MI) between the representations of augmented graphs under different views [53, 58]. Inspired by it, we design a framework for digraph contrastive learning, which contains three steps. (1) First, given a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and the teleport probability α, we generate two correlated views using for the $\mathcal{G}$: $U = \Phi_{\Delta\alpha_1}(\mathcal{G}, \alpha)$ and $V = \Phi_{\Delta\alpha_2}(\mathcal{G}, \alpha)$ as a contrastive pair. This generation is through executing Laplacian perturbation Φ on the input $\mathcal{G}$ with the parameters $\Delta\alpha_1, \Delta\alpha_2$ separately. (2) Second, we take a GNN-based encoder $f(\cdot)$ to extract representation $\mathbf{H}_{\Delta\alpha_1} = f(U)$, $\mathbf{H}_{\Delta\alpha_2} = f(V)$ of two contrastive views. (3) Finally, the encoder is trained with the contrastive objective as

$$\max_f I\left(\mathbf{H}_{\Delta\alpha_1}; \mathbf{H}_{\Delta\alpha_2}\right) = \max_f I\left(f(\Phi_{\Delta\alpha_1}(\mathcal{G}, \alpha)); f(\Phi_{\Delta\alpha_2}(\mathcal{G}, \alpha))\right), \tag{9}$$

where $I(\cdot)$ is the mutual information. This framework lets the encoder $f(\cdot)$ learn the representation of two different views U, V while preserving as much MI as possible.

As shown in Eq. (9), the view generation is controlled by two perturbation terms $\Delta\alpha_1$ and $\Delta\alpha_2$, which may affect the objective function. The current GCLs manually select these parameters in

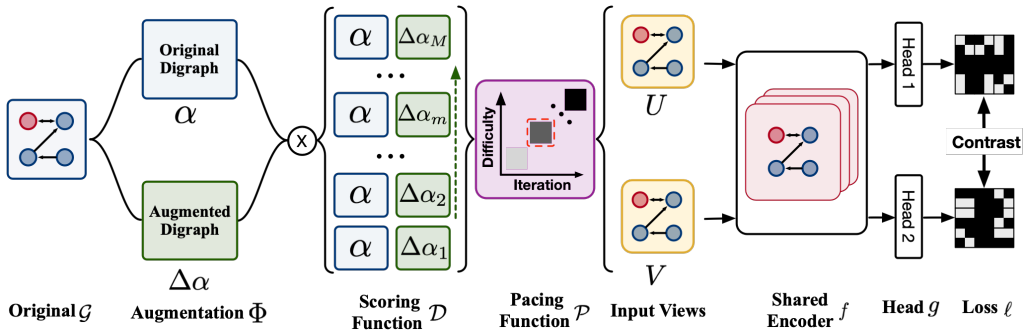

Figure 1: Illustration of our DiGCL model using Laplacian perturbation.

advance, e.g., grid search, and fix contrastive views during training. This parameter selection strategy could make the encoder obtained by Eq. (9) learn only the contrastive information in some particular views and lead to a reduced generalisation of the model [48]. To deal with it, we propose a more generalized multi-view objective of digraph contrastive learning.

Multi-view Objective. We rewrite Eq. (9) to the generalized multi-view objective as

$$\max_{f} \mathop{\mathbb{E}}_{\Delta\alpha_1,\Delta\alpha_2} I\big(f(\Phi_{\Delta\alpha_1}(\mathcal{G},\alpha)); f(\Phi_{\Delta\alpha_2}(\mathcal{G},\alpha))\big), \tag{10}$$

which requires the obtained optimal encoder works well with $\forall \Delta\alpha_1, \Delta\alpha_2 \in [0, 1-\alpha)$ to learn a more balanced and generalized representation. Note that unlike some contrastive learning methods that also use the multi-view strategy [36, 12, 50], there is no fixed number of our multi-views in our approach, and we do not preset the parameters to fix the contrastive views but let them dynamically change during training to learn a optimal encoder.

However, maximizing Eq. (10) is not easy. Varying the perturbation term brings changes to the MI of views, which may result in large variance during the training process and may not converge [37]. To increase the stability of the training, we make one of the views unperturbed in this paper, i.e. $\Delta\alpha_1 = 0$, and design a multi-task training scheme to optimize this multi-view objective.

3.2 Training using Multi-task Curriculum Learning

To tackle this problem, we propose a training scheme using curriculum learning, which utilizes *prior knowledge about the difficulty of the learning tasks* to learn from easy-to-hard contrastive views. The model illustration is in Figure 1 and the pseudo-code is given in the Supplementary Material.

Multi-task Curriculum Learning. Curriculum Learning facilitates the optimization on such a complex objective by scheduling the sub-objectives in a certain order [2, 31]. Inspired by it, we regard the multi-view contrastive learning as a multi-task problem [31, 33, 9, 10, 25, 19] and decompose it into M sub-tasks $\eta_1, \ldots, \eta_M$. Thus, the objective Eq. (10) can be rewritten as

$$\max_{f} \frac{1}{m} \sum_{m=1}^{M} \mathop{\mathbb{E}}_{\Delta\alpha_m} I\big(f(\mathbf{L}_{\text{appr}}(\mathcal{G},\alpha)); f(\Phi_{\Delta\alpha_m}(\mathcal{G},\alpha))\big), \tag{11}$$

where $\Delta\alpha_m \in \{\Theta_{1,\ldots,M}\}$ is a uniform partition of $\Theta = [0, 1-\alpha)$ and the sub-task η_m is associated with the $\Delta\alpha_m$. Then we consider sub-tasks that have less difficulty to learn in the optimization process as **easy** sub-tasks and vice versa as **hard** sub-tasks. We quantify this difficulty into *difficulty score* by means of a scoring function $\mathcal{D}(\cdot)$. Based on it, we learn sub-tasks sequentially in a certain order decided by the pacing function $\mathcal{P}(\cdot)$ to solve the main multi-task problem. Via information transmission by a shared encoder, the previously solved easy sub-tasks will assist in solving the next hard sub-tasks, while the latter can fine-tune the encoder learned by the former.

Scoring Function. The scoring function $\mathcal{D}(\cdot)$ measures how difficult the sub-task η_m is, and can be any function $\mathcal{D}(\cdot) : M \to d$, where d denotes the difficulty score. In each sub-task, there exists a pair of contrastive views as defined in Eq. (11). We empirically assume that the difficulty d_m of the sub-task η_m depend on how difficult it is for the encoder $f(\cdot)$ to learn the contrastive information from this sub-task. Referring to DEFINITION 3, we can find Laplacaian perturbation provides contrastive information to the encoder, and the perturbation term $\Delta\alpha$ controls how much of this information is. Therefore, we can define the scoring function $\mathcal{D}(\cdot)$ to calculate the difficulty d_m of a sub-task η_m as

$$d_m = \mathcal{D}(\eta_m) = 1 - \frac{\Delta\alpha_m}{1-\alpha}. \tag{12}$$

The higher this score represents the smaller the perturbation error, the less contrastive information provided, and the less the encoder is able to distinguish the difference between the two views, resulting in a more difficult sub-task. This idea coincides with the design of the discriminator in GAN [55, 8].

Pacing Function. The pacing function decides the learning sequence of sub-tasks and can be any function $\mathcal{P}(\cdot) : L \to d$, where L denotes all the learning iterations. We consider three function families [47]: logarithmic, exponential, and linear. Table 2 illustrates the pacing functions used in our experiments, which are parameterized by (d_a, d_b). Here d_a is the initial difficulty and d_b is the difficulty at the end of training, thus any pacing function with $d_a = d_b$ is equivalent to fixed view contrastive learning. We mark their corresponding perturbation terms as $(\Delta\alpha_a, \Delta\alpha_b)$. Considering the score in Eq. (12) is defined in a continuous domain, we set the step size of the pacing function to 1, the finest-grained unit, i.e., $L = M$. This can help the model to switch between different training views in a more delicate way.

Name	**Expression** $\mathcal{P}_{(d_a,d_b)}(l)$
log	$d_a + (d_b - d_a)\left(1 + 1/3 \log\left(\frac{l}{L} + e^{-3}\right)\right)$
exp	$d_a + \frac{d_b - d_a}{e^3 - 1}\left(\exp\left(\frac{3l}{L}\right) - 1\right)$
linear	$d_a + (d_b - d_a)\frac{l}{L}$

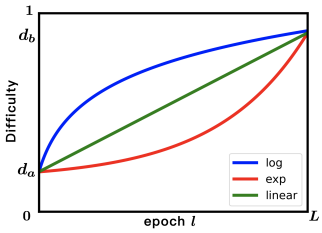

Figure 2: (left) pacing function definitions for the three families of pacing functions used throughout; (right) the plot of pacing function curves from each family. l is the training epoch.

It is desired to find the optimal solution over the entire definition domain of $\Delta\alpha \in [0, 1 - \alpha)$, we set the start point $\Delta\alpha_a = 0.9 - \alpha$ (0.9 is used because the boundary value cannot be obtained) and ending point $\Delta\alpha_b = 0$ in this paper, i.e. $d_a = 1/9, d_b = 1$. More analysis on the pacing functions can be seen in the Supplementary Material.

Contrastive loss. After discussing how to measure expectations in the objective, our attention shifts to maximizing the MI. Several works have investigated the lower bound of MI in contrastive learning, here we adopt *InfoNCE* [27]. For i^{th} node u_i in view U, the node-wise objective is defined as

$$\ell(u_i, v_i) = -\log \frac{\exp\left(\mathcal{S}\left(\boldsymbol{z}_U^i, \boldsymbol{z}_V^i\right)/\tau\right)}{\sum_{j=1}^{n} \exp\left(\mathcal{S}\left(\boldsymbol{z}_U^i, \boldsymbol{z}_V^j\right)/\tau\right)}, \tag{13}$$

where v_i is its corresponding positive node in view V and other nodes in view V is negative nodes of u_i. $\boldsymbol{z}_U^i$ is the projection of node feature that $\boldsymbol{z}_U^i \in \mathbf{Z}_U = g(\mathbf{H}_U)$. A non-linear transformation $g(\cdot)$ named projection head maps augmented representations $\mathbf{H}$ to another lower dimension where the contrastive loss is calculated. $\mathcal{S}(\cdot)$ denotes cosine similarity function and τ denotes the temperature parameter. The final loss is computed across all positive node pairs in the views. Note that GRACE [57] proposes similar *InfoNCE-like* loss function. Different with it, we do not treat intra-view nodes (nodes in U except u_i) as the negative samples since GNN-based encoder $f(\cdot)$ is already able to learn intra-graph structure well, we want to focus on inter-graph contrastive information.

4 Related Work

Graph Neural Networks for digraphs. Several works have tried to learn the complex graph structure by defining motifs [24], inheritance relationships [15], redefining the propagation scheme from Markov process view [38, 22] and second-order proximity[39]. However, these methods are supervised, and they cannot learn the structural features of digraphs without labels or with fewer labels. Our model DiGCL is unsupervised, which is the most significant difference between them.

Digraph Data Augmentation. Good data augmentations are the prerequisite for contrastive learning [53, 43], which provide contrastive information for the subsequent encoder. Most current data augmentation methods, such as dropping nodes or edges [53, 57, 58] and attribute masking [14, 53] are migrated from methods of visual representation [46, 56] with graph structural improvements. Other methods learn graph structure features by combining with traditional graph algorithms to go self-supervised, for example, subgraph sampling [53, 32], graph diffusion based on PageRank/Heat kernel [12], multi-hop neighbor prediction [29], and et al.. However, most of these methods are not

optimized for digraphs and thus cannot efficiently obtain structure information that can be used for self-supervised learning. The performance of existing methods on digraphs can be seen in Table 2.

Multi-view Contrastive Learning and Multi-task Curriculum Learning. Previous works [36, 12, 50] have found the powerful capabilities of using multiple contrastive views at the same time in contrastive learning. However, they only consider a few limited number of views, such as two in [12] and four in [36]. Besides, their methods require to set contrastive views in advance, which means the views are fixed during training. Unlike them, there is no constant number of our method's views, and we do not preset the parameters to fix the contrastive views but let them dynamically change during training. However, our multi-view contrastive learning objective is hard to optimize, thus we adopt the idea of multi-task curriculum learning [31, 33, 9, 10, 25, 19] to progressively learn from multiple easy-to-hard contrastive views. Besides, using curriculum learning [7, 52] in graph is not a new idea and we are the first to use curriculum learning as a solution to the self-supervised multi-view graph contrastive learning problem.

5 Experiments

We conduct extensive experiments to evaluate the effectiveness of our model. The Supplementary Material reports further details on the experiments and reproducibility.

5.1 Experimental Settings

Experimental Task. As explained in Section 2.1, our approach is more applicable to node-level contrastive learning, thus, we adopt the task of **Node Classification in Digraphs** [38] to verify the learning ability of models. Compared with the common experiments for undirected graphs [16, 45, 17, 44], the challenge is that the given adjacency matrix $\mathbf{A}$ is asymmetric, which means message passing has its direction. We will also use datasets of undirected graphs for controlled trials. Task definition, model structures and configurations are included in the Supplementary Material.

Datasets and Splitting. We use several widely-used datasets including digraph datasets: CORA-ML [3], CITESEER [34] and AM-PHOTO [35]; undirected graph datasets: PUBMED [26] and DBLP [28]. The split of the datasets would have a significant effect on models' results [35]. Thus, we divide the datasets randomly and conduct multiple tests to achieve consistent outcomes. For train/validation/test split, following the rules in GCN [16], we choose 20 labels per class for training set, 500 labels for validation set, and rest for the test set. The detailed descriptions are in the Supplementary Material.

Baselines. We compare our model to the 11 SOTA models divided into four main categories: 1) **supervised** models for undirected graph with GCN [16], GAT [41] and [17]; 2) **supervised** models for digraph with MagNet [54] and DiGCN [38]; 3) **self-supervised** models without augmentations including DGI [42] and GMI [30]; 4) **contrastive learning** models with augmentations containing MVGRL [12], GraphCL [53], GRACE [57], and GCA [58]. The descriptions and settings are introduced in the Supplementary Material.

5.2 Experimental Results

Overall accuracy. The performance comparisons between our model and baselines on five datasets are reported in Table 1. We use a two-layer GCN as our encoder and first train our model in an unsupervised manner to obtain the embedding. Then we take a simple ℓ_2-regularized logistic regression as the classifier [42]. For curriculum learning scheme, we select the log pacing function and the start and ending difficulties are set in Section 3.2. We train all models according to their default settings, then calculate mean test accuracy with STD in percent (%) averaged over 20 random dataset splits with random weight initialization.

It can be seen easily that our methods achieve the state-of-the-art results on all datasets. In general, the unsupervised methods including MVGRL, GRACE, and GCA, do not perform well on digraphs compared to their good performance in undirected graphs. This is mainly due to the models' data augmentation methods are not applicable to digaphs, resulting in the inability to learn contrastive information from complex directed structures. Notice that GraphCL performs relatively mediocre in unsupervised methods, most notably because it focuses mainly on graph-level unsupervised methods, and does not apply well to node-level contrastive learning task. Since DGI and GMI do

not require data augmentation to provide contrastive information, they perform very well on both undirected and directed graphs, which shows good suitability for different graph structure. Moreover, supervised methods such as GCN, GAT and APPNP are inferior to DiGCN and MagNet, which are specifically designed for digraphs, in terms of performance. Since our Laplacian perturbation uses the approximate Laplacian matrix proposed by DiGCN, we compare their performance. It is not difficult to find that our model outperforms DiGCN on all datasets, which shows that contrastive learning can learn good encoders by performing a certain data augmentation in an unsupervised manner.

Table 1: Overall accuracy (%) with STD. "No Curr" means do not use curriculum learning and we set $\Delta\alpha_a = \Delta\alpha_b = 1 - \alpha$. "Random" means random order, "Anti Curr" means using anti-curriculum order and "Curr" indicates using curriculum order. The best results are highlighted with **bold** and the second are marked with underline. OOM means out of memory on a 12GB GPU.

	Method	DIRECTED			UNDIRECTED	
		CORA-ML	CITESEER	AM-PHOTO	PUBMED	DBLP
SUPERVISED	GCN [16]	70.92 ± 0.39	63.00 ± 0.45	88.52 ± 0.47	78.78 ± 0.30	73.54 ± 0.77
	GAT [41]	72.22 ± 0.57	63.73 ± 0.57	88.36 ± 1.25	77.49 ± 0.47	76.08 ± 0.54
	APPNP [17]	70.31 ± 0.67	61.63 ± 0.63	87.43 ± 0.98	79.35 ± 0.48	77.92 ± 0.75
	MagNet [54]	76.32 ± 0.10	65.04 ± 0.47	86.80 ± 0.65	74.23 ± 0.46	69.73 ± 0.98
	DiGCN [38]	77.03 ± 0.70	64.60 ± 0.60	88.66 ± 0.51	76.79 ± 0.49	73.37 ± 0.72
UNSUPERVISED	DGI[42]	75.21 ± 1.29	64.58 ± 1.78	85.25 ± 0.59	74.11 ± 0.62	76.53 ± 1.24
	GMI[30]	76.59 ± 0.35	63.29 ± 0.70	81.12 ± 0.01	80.27 ± 0.16	76.66 ± 0.48
	MVGRL[12]	76.67 ± 0.12	62.22 ± 0.02	86.15 ± 0.21	79.98 ± 0.04	OOM
	GraphCL [53]	67.34 ± 0.12	57.84 ± 0.11	67.66 ± 0.05	75.29 ± 0.08	77.85 ± 0.22
	GRACE [57]	73.88 ± 0.25	61.20 ± 0.20	87.95 ± 0.32	79.54 ± 0.05	78.03 ± 0.09
	GCA [58]	76.32 ± 0.33	63.25 ± 0.10	87.35 ± 0.27	79.81 ± 0.61	77.83 ± 0.35
	Ours + No Curr	75.86 ± 0.09	66.99 ± 0.54	87.32 ± 0.14	79.57 ± 0.12	78.28 ± 0.05
	Ours + Random	76.52 ± 1.66	67.15 ± 0.82	89.03 ± 0.46	**80.75 ± 0.10**	79.58 ± 0.14
	Ours + Anti Curr	76.12 ± 1.04	66.83 ± 1.13	88.83 ± 0.73	80.22 ± 0.37	79.42 ± 0.15
	Ours + Curr	**77.53 ± 0.14**	**67.42 ± 0.14**	**89.41 ± 0.11**	80.69 ± 0.08	**79.70 ± 0.13**

Ablation study on curriculum learning. We validate the effectiveness of our curriculum learning strategy and the results are shown in Table 1. We find that even with the contrastive views containing the most contrastive information (with the largest perturbation term), the effect of the model without curriculum learning is significantly lower than the model with curriculum learning. This means that the information learned from fixed views is always incomplete. Besides, we also find that the order of curriculum learning is vital, the ACC of learning from easy-to-hard > random order > hard-to-easy. This substantially validates that our scoring function do help to learn from multi-view curriculars.

Impact of different pacing functions. We study the three different pacing functions defined in Table 2 and the results in Figure 3(a) show that using different path functions can have an impact on the performance but not as much as changing the order of curriculum learning. Among them, the log pacing function performs best as it speeds up learning on easy tasks and stays on harder tasks for more epochs, thus helping the model to grasp the more subtle differences between contrastive views.

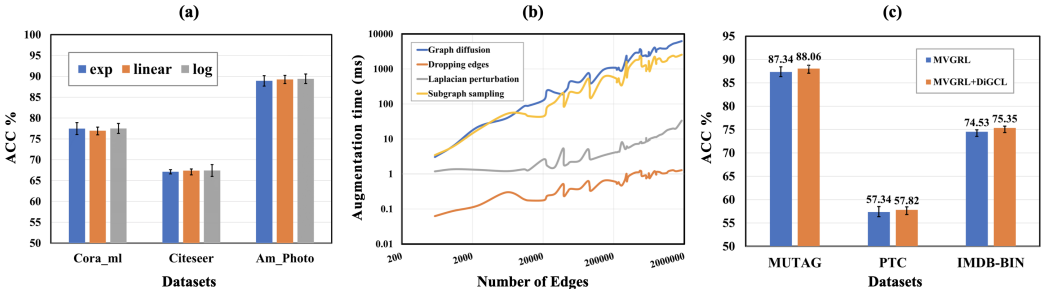

Figure 3: (a) node classification results using different pacing functions; (b) augmentation time on random digraphs with different size; (c) generalization to graph classification task based on MVGRL.

Effect of Laplacian perturbation and other data augmentation methods. Table 2 shows the effect of the different data augmentation methods on the five contrastive learning models. We our Laplacian perturbation works best on both directed and undirected graphs. But the difference with other methods is not significant on the undirected graph dataset PubMed. In addition to this, we also compare with random removing edges used in GRACE under our DiGCL framework. We find that the data augmentation approach without damaging the graph structure can achieve better performance.

Table 2: Overall accuracy (%) with STD with various data augmentation methods. Our model uses curriculum learning, and use the log pacing function. The best results are highlighted with **bold**.

Methods	Data Augmentation Method	CiteSeer	Am-Photo	PubMed
MVGRL [12]	Graph diffusion with heat kernel	61.22 ± 0.07	79.63 ± 0.31	78.54 ± 0.33
	Graph diffusion with PageRank kernel	62.22 ± 0.02	86.15 ± 0.21	79.98 ± 0.04
GraphCL [53]	Edge perturbation	57.84 ± 0.11	67.66 ± 0.05	75.29 ± 0.09
	Node dropping	57.45 ± 0.12	66.69 ± 0.07	75.25 ± 0.08
	Subgraph sampling	57.59 ± 0.10	66.75 ± 0.07	74.75 ± 0.11
GRACE [57]	Random removing edges	61.20 ± 0.20	87.95 ± 0.32	79.54 ± 0.05
GCA [58]	Removing edges by degree score	63.25 ± 0.10	87.35 ± 0.27	79.81 ± 0.61
	Removing edges by PageRank score	62.21 ± 0.16	86.88 ± 0.37	78.92 ± 0.37
	Removing edges by eigenvalue score	63.12 ± 0.08	87.02 ± 0.56	79.70 ± 0.09
Ours	Random removing edges	64.97 ± 0.08	88.45 ± 0.01	79.66 ± 0.13
	Laplacian perturbation	$\mathbf{67.42 \pm 0.14}$	$\mathbf{89.41 \pm 0.11}$	$\mathbf{80.69 \pm 0.08}$

Augmentation time at different graph scales. We construct 20 random digraphs of different scales, with the number of edges ranging from 1K to 2M, then test four different data augmentation methods' mean running time in milliseconds (ms) for 5 times. Figure 3(a) summarizes the results and shows that our Laplacian perturbation is very competitive in terms of runtime among these methods. In contrast to the graph diffusion, which also require the computation of eigenvectors, we can reduce the time by a factor of one hundred with our fast algorithm proposed in Section 2.2. This makes it possible to dynamically perform Laplacian perturbation to generate contrastive views during training.

Generalization to graph classification task. To validate the generalization ability of our DiGCL model, we migrate it to MVGRL [12] and test on three graph classification datasets: MUTAG [18], PTC [18], and IMDB-Bin [49]. The generalized MVGRL model is denoted by MVGRL+DiGCL and the results are shown in Figure 3(c). Using our model can improve the accuracy on all datasets, however, the improvement is not significant. We consider there are three main potential reasons for this: 1) the datasets are undirected graphs, and the effect of using Laplacian perturbation on them is not as obvious as in digraphs; 2) the graph classification problem requires more graph-level contrastive information, while our model focuses on node-level; 3) our scoring function designed for node-level curriculum learning is not suitable for graph-level difficulty measure.

6 Conclusion, Limitation and Future Work

In this paper, we design a digraph data augmentation scheme called Laplace perturbation and theoretically investigate how it can provide contrastive information without changing the digraph structure. Moreover, we present the Digraph Contrastive Learning (DiGCL) which utilizes Laplacian perturbation and curriculum learning to progressively learn from multiple easy-to-hard contrastive views. Finally, we use several tasks on various datasets to demonstrate the effectiveness and generalization ability of our proposed DiGCL. We empirically show that DiGCL can retain more structural features of digraphs than other GCL models while providing adequate contrastive information. Extensive experiments show that our DiGCL outperforms the state-of-the-art approaches.

As elaborated in Section 2.1 and 5.2, our approach is more suitable for obtaining node-level contrastive information. This limits us to use the model to solve some graph-level problems, such as pre-training on protein molecular, drug property prediction, etc. We will extend our approach to graph-level tasks in subsequent work. Meanwhile, we manually design the pacing function to preplan the training speed, such as linear or exponential functions. However, in many cases, the pacing during training is not perfectly planned by the ideal curve. We need to dynamically adjust the pacing function according to the training situation, such as using self-paced learning [19].

References

[1] M. Barigozzi and C. Brownlees, "Nets: Network estimation for time series," *Journal of Applied Econometrics*, vol. 34, no. 3, pp. 347–364, 2019.

[2] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[3] A. Bojchevski and S. Günnemann, "Deep gaussian embedding of attributed graphs: Unsupervised inductive learning via ranking," *arXiv preprint arXiv:1707.03815*, 2017.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[5] F. Chung, "Laplacians and the cheeger inequality for directed graphs," *Annals of Combinatorics*, vol. 9, no. 1, pp. 1–19, 2005.

[6] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[7] C. Gong, J. Yang, and D. Tao, "Multi-modal curriculum learning over graphs," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 4, pp. 1–25, 2019.

[8] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[9] H. Guo, R. Pasunuru, and M. Bansal, "Autosem: Automatic task selection and mixing in multi-task learning," *arXiv preprint arXiv:1904.04153*, 2019.

[10] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic task prioritization for multitask learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 270–287.

[11] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017, pp. 1024–1034.

[12] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," *arXiv preprint arXiv:2006.05582*, 2020.

[13] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *arXiv preprint arXiv:1911.05722*, 2019.

[14] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec, "Strategies for pre-training graph neural networks," *arXiv preprint arXiv:1905.12265*, 2019.

[15] M. Kampffmeyer, Y. Chen, X. Liang, H. Wang, Y. Zhang, and E. P. Xing, "Rethinking knowledge graph propagation for zero-shot learning," in *CVPR*, 2019, pp. 11 487–11 496.

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

[17] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," *arXiv preprint arXiv:1810.05997*, 2018.

[18] N. Kriege and P. Mutzel, "Subgraph matching kernels for attributed graphs," *arXiv preprint arXiv:1206.6483*, 2012.

[19] C. Li, J. Yan, F. Wei, W. Dong, Q. Liu, and H. Zha, "Self-paced multi-task learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[20] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "Geoman: Multi-level attention networks for geo-sensory time series prediction." in *IJCAI*, 2018, pp. 3428–3434.

[21] C. Liu and Y. Li, "A parallel pagerank algorithm with power iteration acceleration," *International Journal of Grid and Distributed Computing*, vol. 8, no. 2, pp. 273–284, 2015.

[22] Y. Ma, J. Hao, Y. Yang, H. Li, J. Jin, and G. Chen, "Spectral-based graph convolutional network for directed graphs," *arXiv preprint arXiv:1907.08990*, 2019.

[23] C. B. Moler, *Experiments with MATLAB*. Society for Industrial and Applied Mathematics, 2011.

[24] F. Monti, K. Otness, and M. M. Bronstein, "Motifnet: a motif-based graph convolutional network for directed graphs," in *IEEE DSW*. IEEE, 2018, pp. 225–228.

[25] K. Murugesan and J. Carbonell, "Self-paced multitask learning with shared knowledge," *arXiv preprint arXiv:1703.00977*, 2017.

[26] G. Namata, B. London, L. Getoor, B. Huang, and U. EDU, "Query-driven active surveying for collective classification," in *10th International Workshop on Mining and Learning with Graphs*, vol. 8, 2012.

[27] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[28] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-party deep network representation," *Network*, vol. 11, no. 9, p. 12, 2016.

[29] Z. Peng, Y. Dong, M. Luo, X.-M. Wu, and Q. Zheng, "Self-supervised graph representation learning via global context prediction," *arXiv preprint arXiv:2003.01604*, 2020.

[30] Z. Peng, W. Huang, M. Luo, Q. Zheng, Y. Rong, T. Xu, and J. Huang, "Graph representation learning via graphical mutual information maximization," in *Proceedings of The Web Conference 2020*, 2020, pp. 259–270.

[31] A. Pentina, V. Sharmanska, and C. H. Lampert, "Curriculum learning of multiple tasks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5492–5500.

[32] J. Qiu, Q. Chen, Y. Dong, J. Zhang, H. Yang, M. Ding, K. Wang, and J. Tang, "Gcc: Graph contrastive coding for graph neural network pre-training," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1150–1160.

[33] N. Sarafianos, T. Giannakopoulos, C. Nikou, and I. A. Kakadiaris, "Curriculum learning of visual attribute clusters for multi-task classification," *Pattern Recognition*, vol. 80, pp. 94–108, 2018.

[34] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.

[35] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," *arXiv preprint arXiv:1811.05868*, 2018.

[36] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," *arXiv preprint arXiv:1906.05849*, 2019.

[37] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola, "What makes for good views for contrastive learning," *arXiv preprint arXiv:2005.10243*, 2020.

[38] Z. Tong, Y. Liang, C. Sun, X. Li, D. Rosenblum, and A. Lim, "Digraph inception convolutional networks," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[39] Z. Tong, Y. Liang, C. Sun, D. S. Rosenblum, and A. Lim, "Directed graph convolutional network," *arXiv preprint arXiv:2004.13970*, 2020.

[40] M. Tschannen, J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic, "On mutual information maximization for representation learning," in *International Conference on Learning Representations*, 2019.

[41] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[42] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," *arXiv preprint arXiv:1809.10341*, 2018.

[43] V. Verma, M. Qu, A. Lamb, Y. Bengio, J. Kannala, and J. Tang, "Graphmix: Regularized training of graph neural networks for semi-supervised learning," *arXiv preprint arXiv:1909.11715*, 2019.

[44] Y. Wang, W. Wang, Y. Liang, Y. Cai, and B. Hooi, "Graphcrop: Subgraph cropping for graph classification," *arXiv preprint arXiv:2009.10564*, 2020.

[45] F. Wu, T. Zhang, A. H. d. Souza Jr, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," *arXiv preprint arXiv:1902.07153*, 2019.

[46] M. Wu, C. Zhuang, M. Mosse, D. Yamins, and N. Goodman, "On mutual information in contrastive learning for visual representations," *arXiv preprint arXiv:2005.13149*, 2020.

[47] X. Wu, E. Dyer, and B. Neyshabur, "When do curricula work?" *arXiv preprint arXiv:2012.03107*, 2020.

[48] T. Xiao, X. Wang, A. A. Efros, and T. Darrell, "What should not be contrastive in contrastive learning," *arXiv preprint arXiv:2008.05659*, 2020.

[49] P. Yanardag and S. Vishwanathan, "Deep graph kernels," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 1365–1374.

[50] C. Yang, Z. An, and Y. Xu, "Multi-view contrastive learning for online knowledge distillation," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3750–3754.

[51] C. Ye, R. C. Wilson, C. H. Comin, L. d. F. Costa, and E. R. Hancock, "Approximate von neumann entropy for directed graphs," *Physical Review E*, vol. 89, no. 5, p. 052804, 2014.

[52] W. Yiwei, W. Wei, L. Yuxuan, C. Yujun, and H. Bryan, "Curgraph: Curriculum learning for graph classification," *The Web Conference (WebConf)*, 2021.

[53] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[54] X. Zhang, N. Brugnone, M. Perlmutter, and M. Hirn, "Magnet: A magnetic neural network for directed graphs," *arXiv preprint arXiv:2102.11391*, 2021.

[55] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," *arXiv preprint arXiv:1609.03126*, 2016.

[56] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation." in *AAAI*, 2020, pp. 13 001–13 008.

[57] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep graph contrastive representation learning," *arXiv preprint arXiv:2006.04131*, 2020.

[58] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," *arXiv preprint arXiv:2010.14945*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] The main content of this paper is theoretical.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 2 and 3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See the Supplementary Material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code, data, and ReadMe file are in the Supplementary Material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See the Supplementary Material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 5.2.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the Supplementary Material.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 5.2.
   (b) Did you mention the license of the assets? [Yes] See the Supplementary Material.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]