Graph Self-supervised Learning via Proximity Divergence Minimization

Tianyi Zhang[1] **Zhenwei Dai**[2] **Zhaozhuo Xu**[1] **Anshumali Shrivastava**[1]

[1]Department of Computer Science, Rice University, Houston, Texas, USA
[2]Amazon, Palo Alto, California, USA

Abstract

Self-supervised learning (SSL) for graphs is an essential problem since graph data are ubiquitous and labeling can be costly. We argue that existing SSL approaches for graphs have two limitations. First, they rely on corruption techniques such as node attribute perturbation and edge dropping to generate graph views for contrastive learning. These unnatural corruption techniques require extensive tuning efforts and provide marginal improvements. Second, the current approaches require the computation of multiple graph views, which is memory and computationally inefficient. These shortcomings of graph SSL call for a corruption-free single-view learning approach, but the strawman approach of using neighboring nodes as positive examples suffers two problems: it ignores the strength of connections between nodes implied by the graph structure on a macro level, and cannot deal with the high noise in real-world graphs. We propose Proximity Divergence Minimization (PDM), a corruption-free single-view graph SSL approach that overcomes these problems by leveraging node proximity to measure connection strength and denoise the graph structure. Through extensive experiments, we show that PDM achieves up to 4.55% absolute improvement in ROC-AUC on graph SSL tasks over state-of-the-art approaches while being more memory efficient. Moreover, PDM even outperforms supervised training on node classification tasks of ogbn-proteins dataset. Our code is publicly available[a].

[a] `https://github.com/tonyzhang617/pdm`

1 INTRODUCTION

Graph Neural Networks (GNN) [Welling and Kipf, 2016, Hamilton et al., 2017, Chen et al., 2022b, Duan et al., 2022] are neural network architectures that extract meaningful and useful representations out of graph data. GNNs have shown great potential in a wide range of applications, including social networks [Fan et al., 2019, Min et al., 2021, Liu et al., 2021], recommendation systems [Wu et al., 2020, Chang et al., 2021, Chen et al., 2022a], and drug discovery [Chen et al., 2018, Xiong et al., 2019, Zhou et al., 2019].

The Need for Self-Supervised Learning: Traditional supervised GNN training strategies require intensive data labeling, which is prohibitively expensive in important fields such as biochemistry [Xiong et al., 2019]. As an alternative, Self-Supervised Learning (SSL) strategies do not rely on labels and have shown promising potential in graph learning. Prior SSL approaches such as DGI [Velickovic et al., 2019], GRACE [Zhu et al., 2020], BGRL [Thakoor et al., 2021] can learn meaningful representations that are useful in downstream tasks such as academic paper categorization, molecule classification, and product recommendation.

Problems of Existing Graph SSL Approaches: In this paper, we identify two problems in the current graph SSL approaches. First, prior competitive SSL approaches for graphs rely on corruption techniques, which perturb node attributes or the adjacency matrix. The corruption techniques are inspired by data augmentation tricks from the computer vision [Shorten and Khoshgoftaar, 2019]. However, unlike images, corrupted graphs may not maintain the original semantics at the node level or graph level. As a result, the encoder may not be able to learn meaningful representations because the learning goal is flawed. Second, existing graph SSL approaches need to compute multiple views of the graph, which increases the memory and computation complexity during training. This efficiency issue would be exacerbated when we train on large graphs with a limited memory budget.

Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023), PMLR 216:2498–2508.

Given the limitation of current graph SSL approaches, a natural question arises:

Can we have a corruption-free single-view approach for graph SSL with promising performance?

In this paper, we answer this question positively by proposing Proximity Divergence Minimization (PDM), a corruption-free single-view graph SSL approach. In particular, we summarize our contributions as:

1. We propose a novel graph SSL framework by leveraging node proximity as the learning target for node representation similarity from a single uncorrupted graph view. Without corruption, our proposed method is a natural and more informative learning objective that achieves significantly better accuracy with minimal tuning. Using only a single view, our approach is much more memory-efficient than previous methods and able to scale to large graphs that are impractical for multi-view methods.
2. We extend our approach to easily scale PDM to large-scale graphs that leverages recent advances in efficient graph training. We scale SSL to large-scale graphs that are difficult to tune and time-consuming to train for the existing SSL methods.
3. We demonstrate the effectiveness of PDM by achieving state-of-the-art accuracy on a variety of real-world graph datasets. We highlight that our approach achieves 2.6%, 4.55% and 3.04% absolute accuracy improvement on PubMed, ogbn-proteins and ogbn-products respectively compared to the previous best.

The following sections are organized as follows. We introduce the graph SSL problem and existing SSL methods' two major drawbacks in Section 2. The motivation behind our approach and the details of our proposed method are included in Section 3. We report the setup and results of our extensive experiments for evaluating our method in Section 4.

2 GRAPH SELF-SUPERVISED LEARNING

In this section, we introduce the graph self-supervised learning problem studied in this paper. Next, we identify two problems of existing graph SSL methods.

2.1 PROBLEM FORMULATION

Graph SSL aims to learn a GNN-based encoder that produces high-quality representations for graph data without using labels. We follow the standard problem setup of graph SSL [Velickovic et al., 2019, Zhu et al., 2020, Thakoor et al., 2021] to keep the training and evaluation procedures consistent with prior approaches. During the training stage, we have access to graph data $(\mathbf{X}, \mathbf{A})$ for training a GNN-based encoder $\mathcal{E}$, where $\mathbf{X} \in \mathbb{R}^{n \times k}$ is the node feature matrix and $\mathbf{A} \in \mathbb{R}^{n \times n}$ is the adjacency matrix. We denote each row of $\mathbf{X}$ as $\mathbf{x}_i$, which corresponds to a k-dimensional feature vector of node i, where $i \in [n]$. There should be an SSL objective to update the parameters of the encoder $\mathcal{E} : \mathbb{R}^{n \times k} \times \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n \times d}$, which encodes the graph $(\mathbf{X}, \mathbf{A})$ into node representation matrix $\mathbf{Z} \in \mathbb{R}^{n \times d}$. Here each row of $\mathbf{Z}$, denoted as $\mathbf{z}_i$, corresponds to the d-dimensional representation of node i. We evaluate graph SSL methods by training and testing a linear classifier with the learned node representation matrix $\mathbf{Z}$ on downstream tasks.

2.2 GRAPH CORRUPTION TECHNIQUES

Prior competitive graph SSL methods rely on corrupting the input graph to generate positive and negative examples for learning. Graph corruption techniques perturb node attributes or the adjacency matrix to produce alternative graph views [Zhu et al., 2020]. In this way, the GNN-based encoder (see Section 2.1) can learn to produce invariant representations. Popular graph corruption techniques include node feature masking [Zhu et al., 2020], node feature shuffling [Velickovic et al., 2019], node dropping [You et al., 2020], edge dropping [Zhu et al., 2020], and subgraphing [You et al., 2020]. For example, CCA-SSG [Zhang et al., 2021] and BGRL [Thakoor et al., 2021] employ node feature masking and edge dropping to generate graph views and maximize the agreement between those views, DGI [Velickovic et al., 2019] uses node feature shuffling to produce negative examples, and GRACE [Zhu et al., 2020] uses node feature masking and edge removal for generating inter-view positives, inter-view negatives and intra-view negatives for contrastive learning. The graph corruption techniques are directly inspired by data augmentation methods from the computer vision domain, such as random erasing [Zhong et al., 2020] and cropping [Shorten and Khoshgoftaar, 2019].

However, corruption techniques for vision and graphs have a fundamental difference: corruptions of natural images preserve their underlying semantics, while the properties of a graph may alter significantly after minor corruptions. For example, in the application of a social network or citation graph, the semantics of a node could significantly change if its edge to a hub node is dropped by graph perturbation. Additionally, in the context of molecular graphs, perturbations to the nodes and edges can lead to drastic changes in the molecule's properties [Sun et al., 2021]. Through extensive experiments, You et al. [2020] demonstrate that edge perturbations in graph SSL significantly degrade the model performance on molecular graphs. It is unclear which

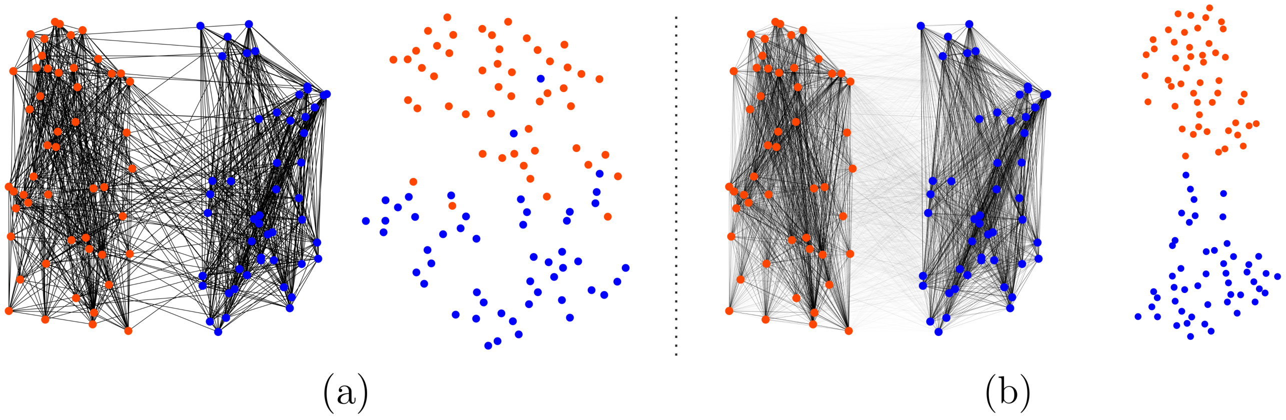

Figure 1: (a) Left: A graph of two densely connected clusters (orange and blue) with sparse edges across clusters. Right: Visualization of the learned node representation using the strawman approach, in which clusters are not perfectly separated. (b) Left: The connection strengths measured by the heat kernel proximity measure. Right: Visualization of the learned node representations using PDM, in which clusters are linearly separable.

corruption techniques are applicable in different graphs, and finding a decent graph corruption requires significant trials and errors since many graphs are highly sensitive to corruption techniques and parameters. As a result, previous works [You et al., 2021, Thakoor et al., 2021, Zhang et al., 2021] resort to extensive grid search for the best combinations of corruption schemes, and show that different datasets require vastly different corruption parameters since the performance of learned models differ greatly with slight changes to corruption schemes and parameters.

2.3 MULTI-VIEW REPRESENTATION LEARNING ON GRAPHS

In addition to over-reliance on corruption techniques, prior graph SSL approaches compute multiple views of the same graph, which has significant memory and computational overhead. The computation of multiple views are required for previous methods since they mine positive/negative examples from them. For example, DGI [Velickovic et al., 2019] computes an additional view through shuffling node features to produce negative examples, LaGraph [Xie et al., 2022] computes two views and minimizes the distance between them, and BGRL [Thakoor et al., 2021] computes four views for positive-only contrastive learning. This creates significant concerns related to computational efficiency and scalability. Modern hardware used for GNN training such as GPU has limited memory, and hence the computation of multiple views scale poorly to large graphs. Compared to supervised training which only computes a single view of the graph, prior self-supervised methods consume multiple times more memory and computation time. This is problematic in many real-world problems since common citation, co-purchasing, and social network graphs contain millions if not billions of nodes and edges [Hu et al., 2020]. Although sub-sampling techniques can fit multiple views of the graph in a limited memory budget, they have been demonstrated to hurt performance significantly [Thakoor et al., 2021]. As a result, it is ideal to have a graph SSL method that computes only a single view of the graph so that it can scale to larger graphs efficiently.

3 PROXIMITY DIVERGENCE MINIMIZATION

In this section, we first motivate our proposed method by describing a strawman approach and identify its flaws in 3.1. Then we describe our proposed method by explaining our learning target in Section 3.2.1 and training strategy in Section 3.2.2. We then analyse the memory efficiency of our proposed approach and compare it against existing methods in Section 3.3. Finally, we describe an extension to our method for scaling to very large graphs in Section 3.4.

3.1 OUR MOTIVATION: A CORRUPTION-FREE SINGLE-VIEW SSL ON GRAPHS

The drawbacks of corruption techniques and multi-view approaches call for a corruption-free single-view SSL approach for graphs. However, this is nontrivial in practice.

3.1.1 A Strawman Approach

A strawman approach for corruption-free single-view SSL is to perform contrastive learning using neighboring nodes as positive instances and non-neighboring nodes as negative instances. But this method has two major problems: it only sees the local structure and fails to take the graph structure

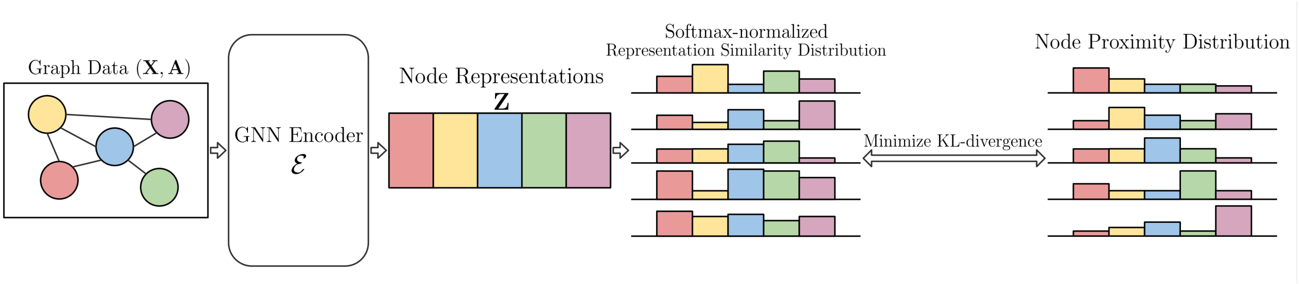
Graph Data $(\mathbf{X}, \mathbf{A})$
GNN Encoder
$\mathcal{E}$
Node Representations
$\mathbf{Z}$
Softmax-normalized
Representation Similarity Distribution
Minimize KL-divergence
Node Proximity Distribution

Figure 2: Our proposed training approach PDM for graph SSL. During each training step, we feed-forward the uncorrupted graph a single time, and minimize the divergence between node representation similarity and proximity distribution.

at a macro level into account, and it is impacted by noise in real-world datasets.

In this strawman approach, neighboring nodes are considered as positive examples and their mutual information is maximized. However, this is counterproductive because it ignores the rich information implied by the graph structure at a macro level. The strengths of connections between nodes vary greatly. For example, a graph may consists of a few densely connected node clusters with sparse edges across clusters. This is a common structure for many real-world graphs such as citation and social networks [Fan et al., 2019]. The graph structure on a macro level implies nodes are more strongly linked to nodes within the same cluster, with weaker connection to nodes in other clusters. However, this information is not captured in the strawman approach, in which each edge is considered as equally strong. This could mislead the feature learning target and discourage the encoder from understanding the global graph structure. As a result, the learned node representations are entangled, shown in Figure 1. Furthermore, real-world graphs often include a large amount of noisy edges [Kang et al., 2019]. The existence of noisy edges confuses the encoder and results in poorer representations.

3.2 OUR PROPOSAL

We propose Proximity Divergence Minimization (PDM), which views node proximity as a distribution and uses it as the learning objective for similarity between node representations. Our proposed method resolves the two aforementioned problems and incorporates the knowledge of graph structure into the learned representations.

3.2.1 Node Proximity Distribution

A node proximity score $P_u(v)$ measures the strength of direct and indirect connections between a pair of nodes u and v in a graph [Zhu et al.]. Unlike the classical distance metrics between nodes such as the shortest path distance, proximity measures take all connections into account to capture rich structural information between the relationship of a node pair [Chebotarev and Shamis, 2006]. Leveraging node proximity measures overcomes the aforementioned two problems. Since proximity measures take all connections into account, it smoothes out the noisy edges in real-world graphs and takes into account the structure of the graph on a macro level. Moreover, it captures the differences in strength of connections between node pairs, unlike the binary adjacency information.

We consider only the proximity measures that are normalized, i.e., $\sum_{v=1}^{n} P_u(v) = 1$. We view the proximity score for a certain node as a distribution over all nodes in the graph. The distributions of proximity are used as the learning target in our proposed method. We consider the following three types of node proximity measures in this work.

Heat Kernel Heat kernel is a technique commonly applied in natural sciences to measure the distribution of heat or diffusive matter [Vassilevich, 2003], and Chung [2007] generalized the heat kernel to discrete graph structures. The heat kernel matrix is a convergent, infinite sum of weighted i-hop adjacency matrices,

$$\mathbf{P}_{\text{heat}} = \sum_{i=0}^{\infty} \alpha_i^{\text{heat}} \hat{\mathbf{A}}^i \tag{1}$$

where $\hat{\mathbf{A}} = (\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})(\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}$, $\mathbf{D} \in \mathbb{R}^{n\times n}$ is the diagonal degree matrix, and $\alpha_i^{\text{heat}} = \frac{e^{-t}t^i}{i!}$ for diffusion time t [Gasteiger et al., 2019]. The node proximity score between nodes u, v, based on heat kernel, is the u, v entry of the heat matrix, i.e. $P_u(v) = [\mathbf{P}_{\text{heat}}]_{u,v}$.

Personalized PageRank Personalized PageRank (PPR) was originally proposed to use in search engines to measure the personalized importance of a web page [Page et al., 1999]. It can be interpreted as a probability matrix in which the entry u, v is the probability of a random walk starting at node u eventually terminating at node v. The PPR matrix is defined as

$$\mathbf{P}_{\text{PPR}} = \sum_{i=0}^{\infty} \alpha_i^{\text{PPR}} \hat{\mathbf{A}}^i \tag{2}$$

where $\alpha_i^{\text{PPR}} = \beta(1-\beta)^i$ for the teleport probability β [Gasteiger et al., 2019]. The node proximity score between nodes u, v, based on PPR, is the u, v entry of the PPR matrix, i.e. $P_u(v) = [\mathbf{P}_{\text{PPR}}]_{u,v}$.

SimRank SimRank [Jeh and Widom, 2002] is a proximity measure that determines the similarity between nodes based on their structural contexts, which can be combined with domain-specific similarity to be made more informative. It is motivated by the insight that two nodes are similar if they are pointed to by similar nodes. The SimRank score between nodes u, v is defined recursively as

$$\mathbf{P}_{\text{SimRank}}(u,v) = \frac{C}{|\mathcal{N}_{\text{in}}(u)||\mathcal{N}_{\text{in}}(v)|} \sum_{a \in \mathcal{N}_{\text{in}}(u)} \sum_{b \in \mathcal{N}_{\text{in}}(v)} \mathbf{P}_{\text{SimRank}}(a,b) \quad (3)$$

where $\mathcal{N}_{\text{in}}(u)$ denotes the set of in-neighbors of u and $C \in (0,1)$ is a constant. In practice, the SimRank score is computed iteratively until convergence or for a fixed number of iterations, as follows

$$\mathbf{P}^{i+1}_{\text{SimRank}}(u,v) = \frac{C}{|\mathcal{N}_{\text{in}}(u)||\mathcal{N}_{\text{in}}(v)|} \sum_{a \in \mathcal{N}_{\text{in}}(u)} \sum_{b \in \mathcal{N}_{\text{in}}(v)} \mathbf{P}^{i}_{\text{SimRank}}(a,b) \quad (4)$$

with initialization

$$\mathbf{P}^{0}_{\text{SimRank}}(u,v) = \begin{cases} 0 & \text{if } u \neq v \\ 1 & \text{if } u = v \end{cases}$$

3.2.2 Training Strategy

The main idea of PDM is to minimize the divergence between the distribution of node representation similarity and the distribution of node proximity.

Node Representation Similarity To measure the similarity between the learned representations of two nodes u, v, we use the dot product $\text{sim}(\mathbf{z}_u, \mathbf{z}_v) = \mathbf{z}_u^\top \mathbf{z}_v$. We apply softmax normalization to normalize the similarity scores between node u and all other nodes into a distribution. Specifically, the representation similarity distribution of node u is

$$S_u(v) = \frac{\exp(\text{sim}(\mathbf{z}_u, \mathbf{z}_v))}{\sum_{i=1}^n \exp(\text{sim}(\mathbf{z}_u, \mathbf{z}_i))}, \text{where } v \in [n] \quad (5)$$

Loss Function Our proposed loss function is the mean Kullback-Leibler divergence [Kullback and Leibler, 1951] between the node proximity distribution and node representation similarity distribution,

$$\frac{1}{n}\sum_{u=1}^n D_{\text{KL}}(P_u \parallel S_u) = \frac{1}{n}\sum_{u=1}^n \left(\sum_{v=1}^n P_u(v)\log P_u(v) - \sum_{v=1}^n P_u(v)\log S_u(v)\right) \quad (6)$$

Since the entropy of the node proximity distribution of a given graph is fixed (i.e. $\sum_{v=1}^n P_u(v)\log P_u(v)$ is a constant), we omit it and equivalently minimize the following loss function.

$$\mathcal{L} = -\frac{1}{n}\sum_{u=1}^n \sum_{v=1}^n P_u(v)\log\frac{\exp(\text{sim}(\mathbf{z}_u, \mathbf{z}_v))}{\sum_{i=1}^n \exp(\text{sim}(\mathbf{z}_u, \mathbf{z}_i))} \quad (7)$$

An overview of PDM's training strategy is shown in Figure 2.

In practice, using the proximity distributions of a subsample of all nodes accelerates convergence and avoids loading all proximity scores into GPU RAM. Therefore, for a batch B of sampled indices, we minimize the following batched loss function.

$$\mathcal{L}_{\text{batch}} = -\frac{1}{|B|}\sum_{u \in B} \sum_{v=1}^n P_u(v)\log\frac{\exp(\text{sim}(\mathbf{z}_u, \mathbf{z}_v))}{\sum_{i=1}^n \exp(\text{sim}(\mathbf{z}_u, \mathbf{z}_i))} \quad (8)$$

Empirically, we found batch sizes of 1024 or 2048 for node proximity distributions work well.

Intuition Node proximity measures are higher-order connectivity scores for node pairs. It is a more informed measure of structural relationship between nodes than adjacency information. It is able to smooth out noisy connections and boost the signal-to-noise ratio of the graph spectrum. Minimizing the divergence between proximity distribution and representation similarity distribution ensures that the encoder learn to incorporate structural knowledge into the learned node representations. Instead of using hard positive/negative instances, we leverage proximity measures between nodes as supervision to tune the relationship between learned node representations using the connection strength implied by the graph structure at a macro level.

Limitations Our proposed method PDM is based on the assumption of homophily McPherson et al. [2001], meaning neighboring nodes tend to be more similar. Therefore, PDM may not perform well on non-homophilous graphs. Additionally, choosing a good proximity measure for learning on a particular graph may require trial and error, as there is a lack of theoretical guidance. Proximity measures may also be computationally expensive to compute.

3.3 MEMORY ANALYSIS

Without the reliance on multiple graph views or extra MLP layers, our approach has clear advantage in memory efficiency over prior approaches. Table 1 presents the memory complexity and the empirical GPU memory usage of the most competitive graph SSL methods on ogbn-arxiv and ogbn-proteins datasets [Hu et al., 2020]. Each forward pass/back-propagation consumes $O(n+m)$ memory, where n is the number of nodes and m is the num-

Table 1: Memory complexity and empirical GPU memory usage of competitive graph SSL approaches on ogbn-arxiv and ogbn-proteins datasets. “OOM” means out of memory on a GPU with 32 GB of memory.

Method	**Memory Complexity**	**GPU RAM Usage**	
		ogbn-arxiv	ogbn-proteins
Supervised GCN	$C^{\text{fw}}_{\text{GCN}}(n+m) + C^{\text{bw}}_{\text{GCN}}(n+m) + C_{\text{Supervised}} \cdot n$	6.9G	17.2G
GRACE	$2C^{\text{fw}}_{\text{GNN}}(n+m) + C^{\text{bw}}_{\text{GNN}}(n+m) + 2C^{\text{fw}}_{\text{MLP}} \cdot n + C^{\text{bw}}_{\text{MLP}} \cdot n + C_{\text{GRACE}} \cdot n^2$	OOM	OOM
BGRL	$4C^{\text{fw}}_{\text{GNN}}(n+m) + C^{\text{bw}}_{\text{GNN}}(n+m) + 2C^{\text{fw}}_{\text{MLP}} \cdot n + C^{\text{bw}}_{\text{MLP}} \cdot n + C_{\text{BGRL}} \cdot n$	25.4G	OOM
LaGraph	$2C^{\text{fw}}_{\text{GNN}}(n+m) + C^{\text{fw}}_{\text{GNN}}(n+m) + C^{\text{bw}}_{\text{MLP}} \cdot n + C^{\text{bw}}_{\text{MLP}} \cdot n + C_{\text{LaGraph}} \cdot n$	16.9G	OOM
CCA-SSG	$2C^{\text{fw}}_{\text{GNN}}(n+m) + C^{\text{bw}}_{\text{GNN}}(n+m) + C_{\text{CCA-SSG}} \cdot n$	15.6G	OOM
PDM (ours)	$C^{\text{fw}}_{\text{GNN}}(n+m) + C^{\text{bw}}_{\text{GNN}}(n+m) + C_{\text{PDM}} \cdot b \cdot n$	7.8G	18.3G

ber of edges in the graph. We let C^{fw} be the constant factor for each forward pass and C^{bw} be the constant factor for each back-propagation. We consider the most memory-efficient graph SSL methods GRACE [Zhu et al., 2020], BGRL [Thakoor et al., 2021], LaGraph [Xie et al., 2022], and CCA-SSG [Zhang et al., 2021], all of which compute two or more graph views at each training step. GRACE uses intra-view and inter-view negative examples, and hence computing its loss function consumes $O(n^2)$ memory. BGRL does not use negative instances in its loss function to avoid quadratic blowup, but it computes 4 graph views in total, 2 by the online encoder and 2 by the target encoder. LaGraph and CCA-SSG both compute 2 graph views and maximize the invariance between views, and LaGraph uses an additional MLP component as the decoder. Our method computes a single graph view, with memory efficiency on par with supervised training theoretically and empirically.

We employ the same encoder for all approaches to ensure a fair comparison of memory usage. On ogbn-arxiv, the memory efficiency of our method is on par with supervised training, while other methods consume 2× or more memory. On ogbn-proteins, supervised training consumes more than half the memory on a 32GB GPU, which makes multi-view training impractical. Therefore, only our SSL method is able to train on ogbn-proteins without running out of memory.

3.4 SCALING TO LARGE GRAPHS

Real-world graphs are very large and pose a significant challenge in scalability [Tang et al., 2023]. Existing methods resort to sub-sampling techniques such as neighbor-sampling [Thakoor et al., 2021]. For PDM, neighbor sampling techniques may not work since node proximity measure may not be well-defined for the sampled neighborhood. To scale to very large graphs, we propose a natural extension to our method by leveraging recent advances in efficient GNN training. Cluster-GCN [Chiang et al., 2019] proposes to train on subgraphs of clusters partitioned from the original graph to avoid the exponential neighborhood expansion problem. We leverage Cluster-GCN to scale PDM to very large graphs by partitioning them into subgraphs and minimize the divergence between node proximity distribution and node representation distribution in each of the subgraphs. We evaluate the effectiveness of PDM scaled to large graphs in Section 4.3.

4 EXPERIMENTS

In this section, we evaluate the performance of PDM and compare it against the most competitive graph SSL methods on a variety of node classification datasets. We first introduce the setup, the datasets, and the baselines used for the evaluation in Section 4.1. Then we present the evaluation results on small to medium-scale datasets in Section 4.2 and results on large-scale datasets in Sectin 4.3. Finally, we present the results of the ablation study in Section 4.4.

4.1 SETTINGS

Evaluation Setup We take an untrained GNN encoder with randomly initialized parameters, and train it using PDM and baseline methods on the graph data $(\mathbf{X}, \mathbf{A})$ without labels until convergence. Then we freeze the parameters of the GNN and and use it to encode the nodes into learned node representations $\mathbf{Z}$. We then train a linear classifier (logistic regression classifier) on the labelled training set with $\mathbf{Z}$ as input, and report the evaluation metrics on the unseen test set. Our evaluation setup is identical to previous works [Velickovic et al., 2019, Thakoor et al., 2021] to keep the evaluation fair and consistent. Details about the testbed for performing the evaluation is given in Section 4.1, and the hyper-parameters and training details are presented in Table 1 in the Supplemental Materials.

Datasets Our baselines are evaluated on 6 datasets, including 3 small scale datasets (Cora, Citeser, PubMed [Sen et al., 2008]), 1 medium scale dataset (ogbn-arxiv [Hu et al., 2020]) and 2 large scale dataset (ogbn-proteins and ogbn-products [Hu et al., 2020]). The graph statistics are summarized in Table 3.

Table 2: Performance of self-supervised learning methods in terms of classification accuracy (along with standard deviations). The results of baselines are taken from official papers. "OOM" means out of memory on a GPU with 32 GB of memory.

Paradigm	Method	Cora	Citeseer	PubMed	ogbn-arxiv
Supervised	MLP	55.1	46.5	71.4	55.50
	GCN	81.5	70.3	79.0	71.74 ± 0.29
	GAT	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3	72.10 ± 0.13
Self-Supervised	DGI	82.3 ± 0.6	71.8 ± 0.7	76.8 ± 0.6	70.34 ± 0.16
	GATE	83.2 ± 0.6	71.8 ± 0.8	80.9 ± 0.3	OOM
	GRACE	81.9 ± 0.4	71.2 ± 0.5	80.6 ± 0.4	71.51 ± 0.11
	BGRL	82.7 ± 0.5	71.1 ± 0.8	79.6 ± 0.5	71.64 ± 0.12
	LaGraph	84.1 ± 0.3	73.0 ± 0.4	80.9 ± 0.3	71.71 ± 0.21
	GraphMAE	84.2 ± 0.4	73.4 ± 0.4	81.1 ± 0.4	71.75 ± 0.17
	InfoGCL	83.5 ± 0.3	73.5 ± 0.4	79.1 ± 0.2	OOM
	CCA-SSG	84.0 ± 0.4	73.1 ± 0.3	81.2 ± 0.3	71.24 ± 0.20
	PDM (ours)	**84.4 ± 0.1**	**74.6 ± 0.2**	**83.8 ± 0.2**	**72.08 ± 0.18**

Table 3: Statistics of the graph datasets used for evaluation.

Dataset	# of Nodes	# of Edges	Metric	# of Classes
Cora	2,708	5,429	Accuracy	7
Citeseer	3,327	4,732	Accuracy	6
PubMed	19,717	44,338	Accuracy	3
ogbn-arxiv	169,343	1,166,243	Accuracy	40
ogbn-proteins	132,534	39,561,252	ROC-AUC	112
ogbn-products	2,449,029	61,859,140	Accuracy	47

Testbed We implement our proposed method with the Deep Graph Library [Wang et al., 2019]. Our experiments are conducted on a machine with 1 NVIDIA Tesla V100 32GB GPU, 2 24-core/48-thread Intel Xeon Gold 5220R CPUs, and 1.5TB of RAM.

Baselines We perform a thorough comparison of PDM against the current most competitive graph SSL methods: *1*) DGI [Velickovic et al., 2019] proposes to learn node representations by maximizing the mutual information between node representations and the global representation through contrasting representations of a corrupted graph. *2*) GATE [Salehi and Davulcu, 2019] reconstructs the input graph with an auto-encoder architecture that uses self-attention. *3*) GRACE [Zhu et al., 2020] performs contrastive learning on positive and negative examples from two different corrupted graph views. *4*) BGRL [Thakoor et al., 2021] learns contrastively from positive examples only by leveraging bootstrapping. *5*) LaGraph [Xie et al., 2022] learns through a reconstruction loss and an invariance loss between the representations of the original graph and a corrupted graph. *6*) GraphMAE [Hou et al., 2022] learns through reconstructing node features using two GNNs as encoder and decoder. *7*) InfoGCL [Xu et al., 2021] maximizes the agreement between the learned representations of two corrupted graph views encoded by a GNN and MLP. *8*) CCA-SSG [Zhang et al., 2021] maximizes the agreement between two corrupted graph views using a loss function inspired by Canonical Correlation Analysis. We also include the most common supervised model baselines for reference, which are trained with the training set as supervision. *1*) MLP [Hu et al., 2020] is a multi-layer perceptron network with only the node features as input. *2*) GCN [Welling and Kipf, 2016] propagates node information through convolutional layers. *3*) GAT [Veličković et al., 2018] leverages self-attention to adaptively aggregate node information. *4*) GraphSage [Hamilton et al., 2017] aggregates node feature information to generalize to unseen data.

4.2 EVALUATION ON SMALL AND MEDIUM-SCALE GRAPHS

Table 2 presents the accuracy of PDM and baselines on Cora, Citeseer, Pubmed and ogbn-arxiv. PDM achieves state-of-the-art performance on all 4 datasets, and improves the previous best by 2.6% on PubMed, 1.1% on CiteSeer, 0.2% on Cora and 0.33% on ogbn-arxiv. Our method significantly exceeds the accuracy of supervised training on Cora, CiteSeer, PubMed, showing its potential in eliminating the reliance on labels in graph learning. On ogbn-arxiv, our method achieves accuracy competitive with the best supervised model GAT (within 0.02% difference) while using a simpler model (GCN).

4.3 EVALUATION ON LARGE-SCALE GRAPHS

We evaluate PDM and baselines on ogbn-proteins and ogbn-products, which are two challenging large-scale node classification datasets. Only PDM is able to train on ogbn-proteins using a single GPU without sub-sampling, other methods require sub-sampling to fit into 32 GB of GPU memory since they rely on multiple graph views (see Section 3.3). We

Table 4: AUC-ROC on ogbn-proteins and accuracy on ogbn-products of the best graph SSL methods. We consider GCN and GraphSage as the backbone model for each method.

Paradigm	**Method**	**ogbn-proteins**	**ogbn-products**
Supervised	GCN	72.51 ± 0.01	75.64 ± 0.01
	GraphSage	77.68 ± 0.01	78.29 ± 0.01
Self-Supervised	InfoGCL	OOM	OOM
	GraphMAE	62.52 ± 0.69	72.88 ± 0.37
	GRACE	68.40 ± 0.59	71.55 ± 0.88
	LaGraph	71.86 ± 0.28	73.23 ± 0.25
	BGRL	73.25 ± 0.79	72.86 ± 0.64
	CCA-SSG	73.08 ± 0.37	73.46 ± 0.26
	PDM (ours)	**77.80 ± 0.21**	**76.50 ± 0.18**

Table 5: Evaluate performance of PDM by varying the hyper-parameters for computing node proximity measures.

	Heat (t)					PPR (β)				
	3	4	5	6	7	0.02	0.04	0.06	0.08	0.10
Cora	83.9	84.1	84.0	84.0	**84.4**	84.1	84.0	84.3	84.1	84.0
Citeseer	73.3	74.1	74.3	74.1	74.1	74.0	**74.6**	74.2	74.0	74.2
PubMed	82.8	83.0	83.6	83.1	82.7	83.6	83.2	**83.8**	82.8	83.4

leverage the sub-sampling techniques described by Hamilton et al. [2017], Thakoor et al. [2021] to scale the baselines. By using full-graph training and no sub-sampling, PDM enjoys a significant advantage on accuracy over other baselines by achieving 4.55% improvement in AUC-ROC than the current best SSL method. More importantly, PDM beats supervised training: 5.29% and 0.12% better AUC-ROC than supervised GCN and GraphSage. As a biological graph of protein interactions, ogbn-proteins is sensitive to graph corruptions. Our method achieves the best accuracy by avoiding corruptions and training on the full graph, which maximally preserves semantics of the original graph.

The graph of ogbn-products is so large that even supervised training has to resort to sub-sampling. Therefore, we leverage Cluster-GCN [Chiang et al., 2019] to efficiently scale PDM by partitioning the graph into 100 clusters. Our method exceeds the previous best SSL method by 3.04%. Furthermore, our method beats supervised training by 0.86% when using the same GCN architecture, suggesting our method has potential of eliminating the need for costly labels in graph learning.

4.4 ABLATION STUDY

We study the sensitivity of our method to hyper-parameter changes. A robust SSL method should not be sensitive to hyper-parameters. This has been a weakness of prior SSL methods, which require vastly different corruption parameters for different datasets [You et al., 2021, Thakoor et al., 2021, Zhang et al., 2021]. We vary the hyper-parameters in computing the node proximity measures for heat kernel and PPR, and evaluate the test accuracy on Cora, Citeseer and PubMed. As shown in Table 5, our method is not sensitive to hyper-parameters of the node proximity scores, since the accuracy drops at most 1.3% from the best accuracy achieved. Therefore, our method is more robust than previous SSL approaches which are sensitive to hyper-parameter changes.

5 RELATED WORKS

Self-supervised Learning for Graphs The success of self-supervised contrastive learning in computer vision [Oord et al., 2018, Hjelm et al., 2018, Grill et al., 2020] inspired the development of contrastive learning methods for graph SSL based on mutual information maximization. For example, DGI [Velickovic et al., 2019] maximizes mutual information between local patch representations and global graph representation by contrasting with negative examples from shuffled node features. GRACE [Zhu et al., 2020] maximizes the mutual information between node representations of two corrupted graph views by contrasting with intra- and inter-view negatives. BGRL [Thakoor et al., 2021] leverages BYOL [Grill et al., 2020] to perform contrastive learning without negative examples. InfoGCL [Xu et al., 2021] proposes a contrastive framework to maintain task-relevant information at different levels and minimize the information loss during graph representation learning. MVGRL [Hassani and Khasahmadi, 2020] uses graph diffusion to produce an alternative graph view and maximize the mutual information between the local representation of one view and the global representation of the other view. Graph SSL methods based on the reconstruction objective have also been proposed in the past. For instance, GATE [Salehi and Davulcu, 2019] uses stacked self-attention-based encoder/decoder architecture to reconstruct node features and graph structure. GraphMAE [Hou et al., 2022] proposes to focus on feature reconstruction using a graph autoencoder. Recently, predictive graph SSL methods have also been proposed. LaGraph [Xie et al., 2022] proposes to learn through predicting unobserved latent graphs. CCA-SSG [Zhang et al., 2021] leverages a feature prediction objective inspired by canonical correlation analysis. Contrastive learning methods for graph SSL without data augmentation have been proposed before, such as AF-GCL [Wang et al., 2022] and AFGRL [Lee et al., 2022]. However, both methods use nodes with the most similar representations as positive instances for contrastive learning, while our method leverages proximity measures for node representation learning.

Node Proximity A variety of node proximity measures have been proposed in the past, including heat kernel [Chung, 2007], PageRank [Page et al., 1999], Cycle Free Effective Conductance [Koren et al., 2006], Katz [Katz, 1953], and SimRank [Jeh and Widom, 2002]. Node proximity measures have been leveraged for learning on graph data in previous works. Liben-Nowell and Kleinberg [2003] leverages different node proximity measures for link prediction in social networks. Murata and Moriyasu [2007] augments graph proximity measures with existing weights in social networks for more accurate link prediction. Zhu et al. computes structural and positional node embeddings using well-established proximity measures.

6 CONCLUSION

We propose PDM, a novel graph SSL framework that is corruption-free and uses a single view. By avoiding corruption techniques employed by prior SSL method, we achieve a natural SSL objective to attain significantly better accuracy on a variety of datasets with minimal tuning efforts. Without the computation of multiple views, our method is more memory-efficient and scalable than prior approaches and able to scale to large graphs that are difficult for existing methods. Through extensive experiments, we demonstrate the advantage of PDM over existing SSL methods with significant improvements in accuracy on popular graph benchmarks. Furthermore, our SSL approach is able to surpass or be competitive with the accuracy of fully supervised training on large-scale datasets, which is a crucial step towards eliminating the need for costly labels in graph learning.

References

Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. Sequential recommendation with graph neural networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 378–387, 2021.

Pavel Chebotarev and Elena Shamis. On proximity measures for graph vertices. 2006. doi: 10.48550/ARXIV.MATH/0602073. URL `https://arxiv.org/abs/math/0602073`.

Hongming Chen, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke. The rise of deep learning in drug discovery. *Drug discovery today*, 23(6):1241–1250, 2018.

Huiyuan Chen, Xiaoting Li, Kaixiong Zhou, Xia Hu, Chin-Chia Michael Yeh, Yan Zheng, and Hao Yang. Tinykg: Memory-efficient training framework for knowledge graph neural recommender systems. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 257–267, 2022a.

Tianlong Chen, Kaixiong Zhou, Keyu Duan, Wenqing Zheng, Peihao Wang, Xia Hu, and Zhangyang Wang. Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022b.

Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 257–266, 2019.

Fan Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 104(50):19735–19740, 2007. doi: 10.1073/pnas.0708838104. URL `https://www.pnas.org/doi/abs/10.1073/pnas.0708838104`.

Keyu Duan, Zirui Liu, Peihao Wang, Wenqing Zheng, Kaixiong Zhou, Tianlong Chen, Xia Hu, and Zhangyang Wang. A comprehensive study on large-scale graph training: Benchmarking and rethinking. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL `https://openreview.net/forum?id=2QrFr_U782Z`.

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The world wide web conference*, pages 417–426, 2019.

Johannes Gasteiger, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

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.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of International Conference on Machine Learning*, pages 3451–3461. 2020.

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.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders, 2022. URL https://arxiv.org/abs/2205.10803.

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.

Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543, 2002.

Zhao Kang, Haiqi Pan, Steven CH Hoi, and Zenglin Xu. Robust graph learning from noisy data. *IEEE transactions on cybernetics*, 50(5):1833–1843, 2019.

Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

Yehuda Koren, Stephen C. North, and Chris Volinsky. Measuring and extracting proximity in networks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 245–255, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933395. doi: 10.1145/1150402.1150432. URL https://doi.org/10.1145/1150402.1150432.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22 (1):79–86, 1951.

Namkyeong Lee, Junseok Lee, and Chanyoung Park. Augmentation-free self-supervised learning on graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7372–7380, 2022.

David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, page 556–559, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581137230. doi: 10.1145/956863.956972. URL https://doi.org/10.1145/956863.956972.

Zirui Liu, Kaixiong Zhou, Fan Yang, Li Li, Rui Chen, and Xia Hu. Exact: Scalable graph neural networks training via extreme activation compression. In *International Conference on Learning Representations*, 2021.

Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001.

Shengjie Min, Zhan Gao, Jing Peng, Liang Wang, Ke Qin, and Bo Fang. Stgsn — a spatial–temporal graph neural network framework for time-evolving social networks. *Knowledge-Based Systems*, 214:106746, 2021. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2021.106746. URL https://www.sciencedirect.com/science/article/pii/S0950705121000095.

Tsuyoshi Murata and Sakiko Moriyasu. Link prediction of social networks based on weighted proximity measures. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI'07)*, pages 85–88, 2007. doi: 10.1109/WI.2007.52.

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

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Amin Salehi and Hasan Davulcu. Graph attention autoencoders. *arXiv preprint arXiv:1905.10715*, 2019.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

Mengying Sun, Jing Xing, Huijun Wang, Bin Chen, and Jiayu Zhou. Mocl: data-driven molecular fingerprint via knowledge-aware contrastive learning from molecular graph. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3585–3594, 2021.

Yuxin Tang, Zhimin Ding, Dimitrije Jankov, Binhang Yuan, Daniel Bourgeois, and Chris Jermaine. Auto-differentiation of relational computations for very large scale machine learning. *arXiv preprint arXiv:2306.00088*, 2023.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L Dyer, Remi Munos, Petar Veličković, and Michal Valko. Large-scale representation learning on graphs via bootstrapping. *arXiv preprint arXiv:2102.06514*, 2021.

Dmitri V Vassilevich. Heat kernel expansion: user's manual. *Physics reports*, 388(5-6):279–360, 2003.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *ICLR (Poster)*, 2(3):4, 2019.

Haonan Wang, Jieyu Zhang, Qi Zhu, and Wei Huang. Augmentation-free graph contrastive learning. *arXiv preprint arXiv:2204.04874*, 2022.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, et al. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

Max Welling and Thomas N Kipf. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*, 2016.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys (CSUR)*, 2020.

Yaochen Xie, Zhao Xu, and Shuiwang Ji. Self-supervised representation learning via latent graph prediction, 2022. URL https://openreview.net/forum?id=Da3ZcbjRWy.

Zhaoping Xiong, Dingyan Wang, Xiaohong Liu, Feisheng Zhong, Xiaozhe Wan, Xutong Li, Zhaojun Li, Xiaomin Luo, Kaixian Chen, Hualiang Jiang, et al. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. *Journal of medicinal chemistry*, 63(16):8749–8760, 2019.

Dongkuan Xu, Wei Cheng, Dongsheng Luo, Haifeng Chen, and Xiang Zhang. Infogcl: Information-aware graph contrastive learning, 2021. URL https://arxiv.org/abs/2110.15438.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In *International Conference on Machine Learning*, pages 12121–12132. PMLR, 2021.

Hengrui Zhang, Qitian Wu, Junchi Yan, David Wipf, and Philip S. Yu. From canonical correlation analysis to self-supervised graph neural networks, 2021. URL https://arxiv.org/abs/2106.12484.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020.

Kaixiong Zhou, Qingquan Song, Xiao Huang, and Xia Hu. Auto-gnn: Neural architecture search of graph neural networks. *arXiv preprint arXiv:1909.03184*, 2019.

Jing Zhu, Xingyu Lu, Mark Heimann, and Danai Koutra. *Node Proximity Is All You Need: Unified Structural and Positional Node and Graph Embedding*, pages 163–171. doi: 10.1137/1.9781611976700.19. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611976700.19.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep Graph Contrastive Representation Learning. In *ICML Workshop on Graph Representation Learning and Beyond*, 2020. URL http://arxiv.org/abs/2006.04131.