Pairwise Half-graph Discrimination: A Simple Graph-level Self-supervised Strategy for Pre-training Graph Neural Networks

Pengyong Li1,2*, Jun Wang2*, Ziliang Li3,4, Yixuan Qiao2, Xianggen Liu1, Fei Ma3, Peng Gao2, Sen Song1† and Guotong Xie2,5,6†

1Department of Biomedical Engineering, Tsinghua University, Beijing, China
2Ping An Healthcare Technology, Beijing, China
3Chinese Academy of Medical Sciences, Beijing, China
4Central University of Finance and Economics, Beijing, China
5Ping An Cloud Company Limited, Shenzhen, China
6Ping An International Smart City Technology Co., Ltd., Shenzhen, China

lipy0628@163.com, junwang.deeplearning@gmail.com, {liziliang908, qiaoyixuan528, gaopeng712, xieguotong}@pingan.com.cn, drmafei@126.com, {liuxg16, songsen}@mail.tsinghua.edu.cn

Abstract

Self-supervised learning has gradually emerged as a powerful technique for graph representation learning. However, transferable, generalizable, and robust representation learning on graph data still remains a challenge for pre-training graph neural networks. In this paper, we propose a simple and effective self-supervised pre-training strategy, named Pairwise Half-graph Discrimination (PHD), that explicitly pre-trains a graph neural network at graph-level. PHD is designed as a simple binary classification task to discriminate whether two half-graphs come from the same source. Experiments demonstrate that the PHD is an effective pre-training strategy that offers comparable or superior performance on 13 graph classification tasks compared with state-of-the-art strategies, and achieves notable improvements when combined with node-level strategies. Moreover, the visualization of learned representation revealed that PHD strategy indeed empowers the model to learn graph-level knowledge like the molecular scaffold. These results have established PHD as a powerful and effective self-supervised learning strategy in graph-level representation learning.

1 Introduction

Graph modeling has recently received broad interest because of the increasing number of non-Euclidean data that needs to be analyzed across various areas, including social networks, physics, and bioinformatics [Hamilton et al., 2017b]. The graph neural network (GNN) [Kipf and Welling, 2017; Velickovic et al., 2018; Hamilton et al., 2017a], a deep learning-based method, has been reported to be a powerful tool for graph representation learning. However, supervised training of GNN usually requires labor-intensive labeling and relies on domain expert knowledge. One way to alleviate the need for large labeled data is to pre-train a GNN on unlabeled data via self-supervised learning, and then transfer the learned model to downstream tasks. This transfer learning methodology has achieved great success in nature language process (NLP) and computer vision (CV) [Liu et al., 2020b]. But, there is less exploration [Hu et al., 2019; Hu et al., 2020; Rong et al., 2020; You et al., 2020] of pre-training schemes for GNNs compared to NLP and CV domains. Current pre-training schemes for GNN mainly focus on unsupervised representation learning [Sun et al., 2019], which learn graph embeddings on a dataset and the embeddings are fed into a classifier for the downstream task on this same dataset. The state-of-the-art method in unsupervised node and graph classification is contrastive learning. Contrastive learning techniques are used to train an encoder that builds discriminative representations by comparing positive and negative samples to maximize the mutual information [Liu et al., 2020b]. Most of them employ the batch-wise positive/negative samples generation for contrastive discrimination, which bring huge computation costs and unsuitability for pre-training on large-scale datasets, while large-scale dataset is essential for pre-training.

The reason for less works on graph transfer learning might be that some graph datasets are limited in size and GNNs often have shallow architectures [You et al., 2020]. In fact, the graph datasets are recently getting larger (e.g. molecular graph data), and even for shallow GNNs, pre-training could provide a better initialized parameters than random initialization. Moreover, recent researches have proposed many theories and architectures about deep GNNs [Li et al., 2020; Liu et al., 2020]. Nowadays, some works have proven the significance of GNN transfer learning. For example, Hu et al. [Hu et al., 2019] and GROVER [Rong et al., 2020] have pre-trained the GNN model on large-scale molecular graph data and achieved impressive performance on multiple molecular properties prediction tasks by transferring the
Figure 1: Overview of the PHD strategy for pre-training GNNs. PHD task is designed to identify whether two half-graphs come from the same source graph. The graph was firstly decomposed into two half-graphs, one of these two half-graphs has a 0.5 possibility to be replaced by a half-graph from another graph as the negative sample, while the unchanged half-graphs form the positive sample with a 0.5 possibility. Besides, a virtual node called the collection node is added to connect with all other nodes by virtual edges, so as to gather the information from the graph pairs and explicitly learn the graph-level features. Then, each node’s hidden states are updated in each GNN message-passing iteration, based on the messages from neighbouring nodes and edges. The binary output is whether two half-graphs are homologous couples.

2 Related Works

Transfer learning. Transfer learning refers to pre-training a model and re-purposing it on different tasks [Hu et al., 2019]. Although transfer learning is a common and effective approach for NLP and CV, it is rarely explored for GNNs, due to the diverse fields that graph-structured data sources are from [You et al., 2020]. Current transfer learning schemes on graph data are mainly inspired by language model, such as AttrMasking [Hu et al., 2019], ContextPred [Hu et al., 2019], GPT-GNN [Hu et al., 2020], and GROVER [Rong et al., 2020]. However, most of these strategies fell into node-level representation learning, which might not capture global information well. Besides the language model inspired methods, some works began to use contrastive learning for transfer learning on graph data, including GraphCL [You et al., 2020] and MICRO-Graph [Zhang et al., 2020].

Unsupervised learning. Traditional graph unsupervised learning methods are mainly based on graph kernels. Compared to graph kernel, contrastive learning methods can learn explicit embedding, and achieve better performance, which are the current state-of-the-art in unsupervised node and graph classification tasks. Generally, current contrastive graph learning employs a node-node contrast [Veličković et al., 2019] or node-graph contrast [Sun et al., 2019; Hassani and Khasahmadi, 2020] to maximize the mutual information.

3 Methodology

3.1 PHD Overview
In order to train a model that captures the global graph information, we pre-train the model with a binarized Pairwise Half-graph Discrimination (PHD) task that can be easily established from any graph database. Simply, PHD task is designed to discriminate whether two half-graphs come from the same source. As shown in Figure 1, the graph is firstly decomposed into two half-graphs, one of these two half-graphs has a 0.5 probability to be replaced by a half-graph...
disconnected from another graph which constitutes the negative sample, otherwise the positive sample. Actually, we can regard PHD as a jigsaw puzzle. We assume that two half-graphs decomposed from the same source can be combined into a valid graph, while two half-graphs from the different source can not. By making the model to combine two sub-graph and distinguish whether they can form a valid graph by piecing together, the model can learn the global topological structure in this training process of comparing and combining.

Figure 2: The graph decomposition sample. The left sub-figure is the adjacency matrix of the graph in the right sub-figure, where the green and blue represent the decomposed two half-graphs.

To collect the information from the half-graph pair and learn the graph-level features, a virtual node, called collection node, is added to connect with all other nodes by virtual edges. The embedding $h_c$ of collection node is learned by GNN. Formally, during each message-passing iteration $k$ in a GNN, the vertex update function $U_k$ updates each node’s hidden states $h_i$, based on the messages from representations of neighboring nodes and edges.

$$m^k_i = \sum_{j \in \mathcal{N}_i} M_k(h^{k-1}_i, h^{k-1}_j, e_{ij}),$$

$$h^{k}_i = U_k(h^{k-1}_i, m^{k}_i),$$

where $\mathcal{N}_i$ represents the neighbors of node $i$, $M_k$ is the message passing function, $e_{ij}$ denotes the edge between the node $i$ and node $j$. And $m^{k}_i$ denotes the message node $i$ receives during iteration $k$.

The message passing runs for $K$ iterations and sequentially updates each node’s representation. The final embedding $h^K_c$ of collection node is fed into a simple linear discriminator to make a binary prediction,

$$p = \sigma(\text{NN}(h^K_c))$$

where $\sigma$ represents the sigmoid activation function, $\text{NN}$ is a single-layer perceptron network. We employed the cross-entropy loss function for simple computation to optimize the parameters of the network as follows:

$$L = -\sum_{i=1}^{m} y \log(p) + (1 - y) \log(1 - p)$$

where $m$ is the number of samples. After pre-trained, the collection node embedding can be regarded as a graph-level representation for the graph and used for downstream tasks. In addition, graph representation can also be obtained by averaging the nodes’ embeddings or other global graph pooling methods.

In the following sections, we describe the important components of PHD in detail.

### 3.2 Graph Decomposition and Negative Sampling

We decompose the graph into two half-graphs to generate the half-graph pairs, served as the positive sample, and replace one of the half-graphs to produce the negative sample. As the example shown in Figure 2, given a graph $G = (V, E)$ where $V$ represents nodes and $E$ represents edges. A sampled node $v_3$ is employed as the border node to separate $G$ into two half-graphs $G_{s,1}$ and $G_{s,2}$, where $G_{s,1}$ contains nodes $\{v_0, v_1, v_2\}$ and $G_{s,2}$ contains nodes $\{v_3, v_4, \cdots, v_7\}$. The edges in these two half-graphs correspond to the top-left sub-matrix and bottom-right sub-matrix of the adjacency matrix respectively. In order to produce half-graphs with balanced and various size, the border node index is randomly sampled in the range of 1/3 to 2/3 of the total number of nodes.

For negative sampling, we randomly sample another graph in the dataset and separate it into two half-graphs using the above method, and $G_{s,2}$ is replaced with one of these two half-graphs to generate a negative sample. How negative samples are generated can have a large impact on the quality of the learned embeddings. It may drive the model to identify whether the two graphs are homologous or estimate whether the two graphs can be combined into a valid graph. In this way, the model can learn the valuable graph-level features of graphs from the nodes and edges which is essential for the downstream tasks.

### 3.3 Virtual Collection Node

The half-graph pair obtained via the above approach are two independent graphs without any connection. We concatenate these two half-graphs into a single whole graph, and introduce a virtual collection node to derive the global graph-level representation by aggregating every node information. The collection node is linked with all the other nodes by virtual directed edges, pointing from the other nodes to the collection node. During the message passing process of GNN, the collection node learns its representation from all the other nodes but does not affect the feature update procedure of them. Consequently, the collection node’s feature can grasp the global representation of the half-graphs pair and be fed into a feed-forward neural network for the final prediction.

### 3.4 Input Representation

As shown in Figure 3, the input representation consists of two parts: feature embedding and segment embedding. A graph is generally described by a set of nodes features and edges features. Besides the feature embedding, we add a learned segmentation embedding to every node and every edge indicating which half-graph it belongs to. Specifically, we label each node and edge of $G_{s,1}$ as 0, each node and edge of $G_{s,2}$ as 1, collection node and edges connected to collection node as 2. Then these segment labels are fed into the embedding layer that yields segment embedding. The final input representation is constructed by summing the segment embedding and feature embedding. In this way, the model could distinguish the nodes and edges from different segments, thus enables simultaneous input of two graphs.
4 Experiments on Transfer Learning

We performed experiments on transfer learning on molecular property prediction following [Hu et al., 2019; You et al., 2020], which pre-trains GNNs on a large-scale molecular graph data and finetunes the model in different datasets to evaluate the transferability of the pre-training scheme.

4.1 Datasets

For pre-training, our PHD is performed on 2 million unlabelled molecules sampled from the ZINC15 [Sterling and Irwin, 2015] database. After pre-training, we fine-tune the model on 8 downstream task datasets including MUV, HIV, BACE, BBBP, Tox21, ToxCast, SIDER, and ClinTox. All the molecules in these datasets are described by a set of node and bond features obtained by RDKit as same as Hu et al. [Hu et al., 2019].

### Table 1: Test ROC-AUC score of different pre-training strategies on transfer learning

| Dataset | BBBP | Tox21 | ToxCast | SIDER | ClinTox | MUV | HIV | BACE | Average |
|---------|------|-------|---------|-------|---------|-----|-----|------|---------|
| No. Molecules | 2039 | 7831 | 8575 | 1427 | 1478 | 93087 | 41127 | 1513 | / |
| No. Binary prediction tasks | 1 | 12 | 617 | 27 | 2 | 17 | 1 | 1 | / |

| Pre-training strategy | Graph-level | Node-level | Average ROC-AUC across 10 random seeds with scaffold split |
|-----------------------|-------------|-------------|----------------------------------------------------------|
| No Pre-train | 63.3 ±2.5 | 75.2 ±0.7 | 64.0 ±0.4 | 53.3 ±2.5 | 62.9 ±2.7 | 71.8 ±1.6 | 75.3 ±1.9 | 74.6 ±2.4 | 67.5 |
| – Informax | 68.8 ±0.8 | 75.3 ±0.5 | 62.7 ±0.4 | 58.4 ±0.8 | 69.9 ±3.0 | 75.3 ±2.5 | 76.0 ±0.7 | 75.9 ±1.6 | 70.3 |
| – EdgePred | 67.3 ±2.4 | 76.0 ±0.6 | 64.1 ±0.6 | 60.4 ±0.7 | 64.1 ±3.7 | 74.1 ±2.1 | 76.3 ±1.0 | 79.9 ±0.9 | 70.3 |
| – ContextPred | 68.0 ±2.0 | 75.7 ±0.7 | 63.9 ±0.6 | 60.9 ±0.6 | 65.9 ±3.8 | 75.8 ±1.7 | 77.3 ±1.0 | 79.6 ±1.2 | 70.9 |
| – AttrMasking | 64.3 ±2.8 | 76.7 ±0.4 | 64.2 ±0.5 | 61.0 ±0.7 | 71.8 ±4.1 | 74.7 ±1.4 | 77.2 ±1.1 | 79.3 ±1.6 | 71.1 |
| GraphCL | 69.7 ±0.7 | 73.9 ±0.7 | 62.4 ±0.6 | 60.5 ±0.9 | 76.0 ±2.7 | 69.8 ±2.7 | 78.5 ±1.2 | 75.4 ±1.4 | 70.7 |
| bi-PHD | 66.8 ±2.2 | 78.1 ±0.6 | 66.4 ±0.3 | 64.5 ±1.2 | 65.7±1.8 | 75.4 ±1.4 | 77.0±0.5 | 79.3±0.8 | 71.6 |
| PHD | 68.8 ±0.7 | 77.9 ±0.7 | 66.4 ±0.6 | 60.4 ±0.6 | 69.5 ±3.8 | 76.5 ±2.0 | 78.6 ±1.3 | 77.5 ±2.0 | 71.9 |

For perspective 1), we systematically compared PHD with some strong baselines including Infomax [Veličković et al., 2019], EdgePred [Hamilton et al., 2017b], AttrMasking [Hu et al., 2019], ContextPred [Hu et al., 2019], and GraphCL [You et al., 2020]. For perspective 2), we combined PHD with the node-level strategies—AttrMasking and ContextPred to test whether the performances are improved. For perspective 3), we chose four popular GNN architectures including GIN [Xu et al., 2018], GCN [Kipf and Welling, 2017], GAT [Veličkovic et al., 2018] and GraphSAGE [Hamilton et al., 2017a] to evaluate PHD.

4.2 Experimental Setup

We evaluate the effectiveness of PHD from three perspective on large-scale graph data: 1) Whether PHD can pre-train better GNNs that generalize well on graph classification tasks; 2) Whether our graph-level strategy—PHD can cooperate well with node-level strategies; 3) Whether PHD can generalize well to different GNNs.

For perspective 1), we systematically compared PHD with some strong baselines including Infomax [Veličković et al., 2019], EdgePred [Hamilton et al., 2017b], AttrMasking [Hu et al., 2019], ContextPred [Hu et al., 2019], and GraphCL [You et al., 2020]. For perspective 2), we combined PHD with the node-level strategies—AttrMasking and ContextPred to test whether the performances are improved. For perspective 3), we chose four popular GNN architectures including GIN [Xu et al., 2018], GCN [Kipf and Welling, 2017], GAT [Veličkovic et al., 2018] and GraphSAGE [Hamilton et al., 2017a] to evaluate PHD.

4.3 Experiment Configuration

We adopt same experimental setting as Hu et al. [Hu et al., 2019]. The different GNN architectures including GIN, GCN, GAT, and GraphSAGE were adapted from the implementation in Hu et al. [Hu et al., 2019]. All the GNN architectures have 5 GNN message passing layers with 300 embedding dimension. We run all pre-training methods for 100 epochs with a learning rate of 0.001 and a batch size of 256 on 2 million molecules from ZINC15. After pre-training, we add a linear classifier to fine-tune on 8 datasets above. We split these datasets via scaffold split [Hu et al., 2019] with the ratio of 8:1:1 (train:validation:test). We train models for 100 epochs with a learning rate of 0.001 and the dropout rate of 50%. The validation sets were used for model selection and the ROC-AUC scores on test sets are reported. We report the mean ROC-AUC and standard deviation of experiments with 10 random seeds. We report results from previous papers with the same experimental setup if available.
other four recent proposed strategies as InfoGraph [Sun al., 2019], GPT-GNN [Hu et al., 2020], GROVER [Rong et al., 2020], and MICRO-Graph [Zhang et al., 2020]. Because InfoGraph has a huge computation cost that unsuited to pre-train on large-scale dataset, we pre-trained on a small-scale dataset following the same experimental setting as MICRO-Graph. Specifically, we pre-trained DeeperGCN [Li et al., 2020] on the HIV dataset, and fine-tuned the pre-trained model on each downstream tasks. The results of baselines are taken from MICRO-Graph.

Table 2: 10-fold cross validation ROC-AUC score comparison results (following the same experimental setting as MICRO-Graph, which pre-trained the DeeperGCN model on the HIV dataset, and fine-tuned the pre-trained model on each downstream tasks. The results of baselines are taken from MICRO-Graph.)

| Architecture | Pretrain strategy | BBBP   | Tox21  | ToxCast | SIDER  | ClinTox | MUV    | HIV    | BACE    | Average |
|--------------|-------------------|--------|--------|---------|--------|---------|--------|--------|---------|---------|
| GCN          | No Pre-train      | 66.5 ±0.5 | 74.9 ±0.4 | 64.2 ±0.4 | 58.2 ±0.4 | 55.2 ±0.9 | 72.3 ±0.9 | 73.2 ±1.6 | 76.3 ±0.7 | 67.6 ±0.7 |
|              | PHD + node-level  | 66.2 ±1.3 | 74.5 ±0.4 | 64.6 ±0.2 | 57.0 ±0.2 | 62.5 ±0.6 | 72.6 ±1.2 | 71.4 ±0.9 | 77.6 ±0.9 | 68.2 ±0.2 |
| GAT          | No Pre-train      | 65.4 ±1.2 | 71.8 ±0.6 | 57.4 ±0.6 | 60.3 ±1.0 | 62.0 ±3.1 | 67.1 ±1.1 | 73.9 ±1.9 | 67.4 ±2.3 | 65.6 ±1.0 |
|              | PHD + node-level  | 66.6 ±1.3 | 74.5 ±0.6 | 64.1 ±0.4 | 57.5 ±2.4 | 69.4 ±1.9 | 68.6 ±0.9 | 72.4 ±0.8 | 73.1 ±1.3 | 67.9 ±0.0 |
| GraphSAGE    | No Pre-train      | 68.6 ±1.5 | 75.4 ±0.5 | 64.7 ±0.6 | 58.8 ±1.4 | 59.5 ±2.9 | 74.2 ±1.0 | 75.9 ±1.1 | 65.1 ±1.0 | 67.8 ±0.9 |
|              | PHD + node-level  | 71.7 ±0.9 | 76.1 ±0.4 | 65.5 ±0.4 | 62.7 ±0.8 | 60.0 ±3.7 | 76.5 ±0.8 | 78.1 ±0.7 | 80.1 ±0.7 | 71.3 ±1.1 |

Table 3: Test ROC-AUC score performances on different GNN architectures. ContextPred is used as node-level strategy. The best results for each dataset and comparable results are in bold.

4.4 Experimental Results

Whether PHD Can Pre-train Better GNN?

The upper half of Table 1 shows that our PHD strategy gives better predictive performance than the previous strategies in terms of the average ROC-AUC score on eight datasets. For example, on Toxcast, we observe 3.4% relative improvement over previous state-of-the-art. These results indicate that leveraging global structural information plays a more essential role in self-supervised graph learning. In particular, the edges between the collection node and other nodes in our PHD strategy can be unidirectional or bidirectional. We conduct the ablation study of the edges direction. Table 1 shows that unidirectional edges outperform bidirectional edges (bi-PHD) in terms of average ROC-AUC. What is more, the bidirectional edges need more computation than unidirectional edges, so we employed the unidirectional message passing for collection node in all our experiments.

Aside from above methods, we also compare with the other four recent proposed strategies as InfoGraph [Sun et al., 2019], GPT-GNN [Hu et al., 2020], GROVER [Rong et al., 2020], and MICRO-Graph [Zhang et al., 2020]. Because InfoGraph has a huge computation cost that unsuited to pre-train on large-scale dataset, we pre-trained on a small-scale dataset following the same experimental setting as MICRO-Graph. Specifically, we pre-trained DeeperGCN [Li et al., 2020] on the HIV dataset which contains 40K molecules. Then we fine-tuned the pre-trained model on downstream tasks. The best 10-fold cross validation ROC-AUC scores averaged over 10 folds are reported in Table 2. Table 2 shows that PHD outperform previous best schemes on 6 of 7 datasets by a large margin (about 5.5% average improvement).

Whether PHD Can Cooperate Well with Node-level Strategies?

In general, to pre-train well on graph data, we need to pre-train a model at both graph and node-level to encode more information [Hu et al., 2019]. Lower half of table 1 summarizes the results of combining our PHD with node-level strategies. It shows that the combinations significantly outperform the strategies of only PHD or node-level strategy. Furthermore, the PHD combining with AttrMasking gives a better predictive performance than that with ContextPred, achieving state-of-the-art performance. These results reveal that combining PHD with node-level strategy has the potential of pre-training a GNN model well on graph data, especially on molecular representation learning.

Whether PHD Can Generalize Well to Different GNNs?

We conducted experiments with other popular GNN architectures: GCN, GAT, and GraphSAGE. Table 3 shows that PHD pre-training yields superior performance than random initialization over different GNN architectures on most datasets, and the performance could be further enhanced when cooperate with node-level strategy. Thus, PHD is architecture-free that could empower different GNN architectures to learn an expressive graph representation. It provides a favorable initialization for model and improves the performance of downstream tasks. Although our PHD strategy acts universally well to enhance many GNN-based models, different GNN architectures have different gains after pre-trained. Our experiments indicate that GraphSAGE and GIN achieve larger improvement than GCN and GAT. It remains an open challenge in research to explore the relationship between the pre-training strategies and the GNN architectures.
No Pre-structures of a molecule, which reveals the graph-level in-fold concept in chemistry, is proposed to represent the core global knowledge in a graph. Fortunately, the scaffolds. The primary problem to be solved is to define what can be visualized the representation extracted by the pre-trained models.

To intuitively prove our PHD is a graph-level strategy, we visualize the representation learned from the GIN model to better capture data’s globally inherent characteristics, which provides high-quality representations for the downstream graph-level tasks.

Table 4: Comparison of 10-fold cross validation classification accuracy on unsupervised learning over 5 datasets. The result in bold indicates the best reported classification accuracy.

| Dataset | MUTAG | PTC-MR | RDT-B | IMDB-B | IMDB-M |
|---------|-------|--------|-------|--------|--------|
| (No. Graphs) | (No. classes) | (Avg. Nodes) | | | |
| GLK | 81.7 ±2.1 | 57.3 ±1.0 | 77.3 ±0.2 | 65.9 ±1.0 | 43.9 ±0.4 |
| WL | 80.7 ±3.0 | 58.0 ±0.9 | 68.8 ±0.4 | 72.3 ±3.4 | 47.0 ±0.6 |
| DGK | 87.4 ±0.9 | 60.1 ±1.5 | 78.0 ±1.0 | 67.0 ±0.6 | 44.6 ±0.5 |
| sub2vec | 61.1 ±0.6 | 60.0 ±1.1 | 71.5 ±1.0 | 55.3 ±0.8 | 36.7 ±0.7 |
| graph2vec | 83.2 ±1.7 | 60.2 ±1.3 | 75.8 ±0.8 | 71.1 ±0.9 | 50.4 ±0.8 |
| InfoGraph | 89.0 ±1.1 | 61.6 ±1.8 | 82.5 ±1.4 | 73.0 ±0.9 | 49.7 ±0.8 |
| GraphCL | 86.8 ±1.3 | - | 89.5 ±0.9 | 71.1 ±0.4 | - |
| CMV | 89.7 ±1.1 | 62.5 ±1.7 | 84.5 ±0.6 | 74.2 ±0.7 | 51.2 ±0.5 |
| PHD | 90.5 ±0.9 | 65.5 ±0.8 | 89.2 ±0.4 | 72.5 ±0.3 | 49.8 ±0.7 |

5 Experiments on Unsupervised Learning

Besides transfer learning, we then evaluated PHD in the unsupervised representation learning, where unsupervised methods generate graph embeddings that are fed into a linear SVM classifier. We conduct experiments on five small-scale benchmark datasets: MUTAG, PTC-MR, REDDIT-BINARY, IMDB-BINARY, and IMDB-MULTI, to compare PHD with previous state-of-the-art approaches including graph kernel methods—GLK [Shervashidze et al., 2009], WL [Shervashidze et al., 2011], DGK [Yanardag, 2015], and some other unsupervised graph learning methods—sub2vec [Adhikari et al., 2018], graph2vec [Narayanan et al., 2017], InfoGraph [Sun et al., 2019], GraphCL [You et al., 2020], and Contrastive Multi-View (CMV) [Hassani and Khasahmadi, 2020].

We adopt the same procedure of previous works [Sun et al., 2019; You et al., 2020; Hassani and Khasahmadi, 2020] to make a fair comparison, and used 10-fold cross validation accuracy to report the classification performance. Specifically, we first pre-trained GIN model implemented by InfoGraph. The embedding dimension is set to 512. The graph representation learned by the GIN model was fed into a linear SVM to obtain the final prediction. The C parameter of SVM was selected from \(\{10^{-3}, 10^{-2}, \ldots, 10^2, 10^3\}\).

Table 4 shows that PHD still achieves state-of-the-art performance on MUTAG and PTC-MR and comparable performance on the rest datasets. The great improvement on MUTAG and PTC-MR indicates that PHD is good at handling the small attributed graphs.

6 Explainable Representation Visualization

To intuitively prove our PHD is a graph-level strategy, we visualize the representation extracted by the pre-trained models. The primary problem to be solved is to define what can represent global knowledge in a graph. Fortunately, the scaffold concept in chemistry is proposed to represent the core structures of a molecule, which reveals the graph-level intrinsic patterns to some extent [Bemis and Murcko, 1996]. Here, we employed UMAP [McInnes et al., 2018] for molecular graph representation visualization coloring with the corresponding scaffold. Specifically, we chose 9 most common scaffolds from ZINC dataset, and randomly sampled 1000 molecules from each selected scaffold. Finally, we have 9000 molecules labeled with nine different scaffolds. The molecular representations are obtained by averaging the node embeddings extracted by the last layer of a GIN model, we compared the UMAP visualization of representation results including (a) no pre-trained, (b) PHD, (c) AttrMasking and (d) combining AttrMasking with PHD.

From Figure 4, we observe that the no pre-trained GNN model (a) shows no obvious cluster and the molecules overlap in a mess without meaningful patterns. In contrast, the GNN model pre-trained with our PHD strategy (b) exhibits a discernible clustering. Moreover, although the model pre-trained with node-level AttrMasking (c) scatters the molecules with slight cluster, it still presents notable overlap. Alternatively, with a beneficial combination of PHD and node-level strategy, the representations extracted from the pre-trained model (d) exhibit a more favorable and distinctive clustering corresponding to the nine scaffolds. These results indicate that our PHD strategy prompts the pre-training model to better capture data’s globally inherent characteristics, which provides high-quality representations for the downstream graph-level tasks.

7 Conclusions

In this work, we present a self-supervised strategy named Pairwise Half-graph Discrimination (PHD), an effective and simple strategy that explicitly pre-trains the expressive GNN at graph level. Extensive experiments on multiple downstream benchmarks show that the PHD achieves superior performance than state-of-the-art self-supervised strategies on transfer learning and unsupervised representation learning. Moreover, we observe that the pre-trained GNN model with PHD strategy can capture the global graph-level knowledge like the molecular scaffold. In the future, we plan to use PHD for pre-training more expressive GNN models on larger datasets and evaluate it on more downstream tasks.
References

[Adhikari et al., 2018] Bijaya Adhikari, Yao Zhang, Naren Ramakrishnan, and B Aditya Prakash. Sub2vec: Feature learning for subgraphs. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 170–182. Springer, 2018.

[Bemis and Murcko, 1996] Guy W Bemis and Mark A Murcko. The properties of known drugs. 1. molecular frameworks. Journal of medicinal chemistry, 39(15):2887–2893, 1996.

[Hamilton et al., 2017a] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Advances in Neural Information Processing Systems (NeurIPS), pages 1025–1035, 2017.

[Hamilton et al., 2017b] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. IEEE Data Engineering Bulletin, 40(3):52–74, 2017.

[Hassani and Khasahmadi, 2020] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. arXiv preprint arXiv:2006.05582, 2020.

[Hu et al., 2019] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In International Conference on Learning Representations, 2019.

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

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

[Li et al., 2020] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. Deepergc: All you need to train deeper gcns. arXiv preprint arXiv:2006.07739, 2020.

[Liu et al., 2020a] Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 338–348, 2020.

[Liu et al., 2020b] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. arXiv, pages arXiv–2006, 2020.

[McInnes et al., 2018] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426, 2018.

[Narayanan et al., 2017] Annamalai Narayanan, Mahinthan Chandramohan, Rajasek Varakatesan, Lihui Chen and Y Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. CoRR, abs/1707.05005, 2017.

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

[Shervashidze et al., 2009] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In Artificial Intelligence and Statistics, pages 488–495, 2009.

[Shervashidze et al., 2011] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. Journal of Machine Learning Research, 12(Sep):2539–2561, 2011.

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

[Sun et al., 2019] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. arXiv preprint arXiv:1908.01000, 2019.

[Velickovic et al., 2018] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In International Conference on Learning Representations (ICLR), 2018.

[Veličković et al., 2019] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In International Conference on Learning Representations (ICLR), 2019.

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

[Yanardag and Vishwanathan, 2015] Pinar Yanardag. Deep graph kernels. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 1365–1374. ACM, 2015.

[You et al., 2020] 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, 2020.

[Zhang et al., 2020] Shichang Zhang, Ziniu Hu, Arjun Subramonian, and Yizhou Sun. Motif-driven contrastive learning of graph representations. arXiv preprint arXiv:2012.12533, 2020.