Prototypical Graph Contrastive Learning

Shuai Lin, Chen Liu, Pan Zhou, Zi-Yuan Hu, Shuojia Wang, Ruihui Zhao, Yefeng Zheng, *Fellow, IEEE*, Liang Lin, *Senior Member, IEEE*, Eric Xing, *Fellow, IEEE*, and Xiaodan Liang, *Senior Member, IEEE*

***Abstract*— Graph-level representations are critical in various real-world applications, such as predicting the properties of molecules. However, in practice, precise graph annotations are generally very expensive and time-consuming. To address this issue, graph contrastive learning constructs an instance discrimination task, which pulls together positive pairs (augmentation pairs of the same graph) and pushes away negative pairs (augmentation pairs of different graphs) for unsupervised representation learning. However, since for a query, its negatives are uniformly sampled from all graphs, existing methods suffer from the critical sampling bias issue, i.e., the negatives likely having the same semantic structure with the query, leading to performance degradation. To mitigate this sampling bias issue, in this article, we propose a prototypical graph contrastive learning (PGCL) approach. Specifically, PGCL models the underlying semantic structure of the graph data via clustering semantically similar graphs into the same group and simultaneously encourages the clustering consistency for different augmentations of the same graph. Then, given a query, it performs negative sampling via drawing the graphs from those clusters that differ from the cluster of query, which ensures the semantic difference between query and its negative samples. Moreover, for a query, PGCL further reweights its negative samples based on the distance between their prototypes (cluster centroids) and the query prototype such that those negatives having moderate prototype distance enjoy relatively large weights. This reweighting strategy is proven to be more effective than uniform sampling. Experimental results on various graph benchmarks testify the advantages of our PGCL over state-of-the-art methods. The code is publicly available at https://github.com/ha-lins/PGCL.**

Manuscript received 21 February 2022; revised 23 June 2022; accepted 7 July 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0109700, in part by the National Natural Science Foundation of China (NSFC) under Grant 61976233, in part by the Guangdong Province Basic and Applied Basic Research (Regional Joint Fund-Key) under Grant 2019B1515120039, in part by the Guangdong Outstanding Youth Fund under Grant 2021B1515020061, in part by the Shenzhen Fundamental Research Program under Project RCYX20200714114642083 and Project JCYJ20190807154211365, and in part by the CAAI-Huawei MindSpore Open Fund. *(Shuai Lin and Chen Liu contributed equally to this work.) (Corresponding author: Xiaodan Liang.)*

Shuai Lin, Chen Liu, and Xiaodan Liang are with the School of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen 518107, China (e-mail: shuailin97@gmail.com; cathyliu41@gmail.com; xdliang328@gmail.com).

Pan Zhou is with Sea AI Laboratory, Singapore 138522 (e-mail: panzhou3@gmail.com).

Zi-Yuan Hu and Liang Lin are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China.

Shuojia Wang, Ruihui Zhao, and Yefeng Zheng are with the Tencent Jarvis Lab, Shenzhen 518000, China (e-mail: skylawang@tencent.com; zacharyzhao@tencent.com; yefengzheng@tencent.com).

Eric Xing is with the School of Computer Science, Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates (e-mail: eric.xing@mbzuai.ac.ae).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3191086.

Digital Object Identifier 10.1109/TNNLS.2022.3191086

***Index Terms*— Contrastive learning, graph representation learning, self-supervised learning.**

I. Introduction

LEARNING graph representations is a fundamental problem in a variety of domains and tasks, such as molecular properties prediction in drug discovery [1], [2], protein function forecast in biological networks [3], [9], and community analysis in social networks [10]. Recently, graph neural networks (GNNs) [1], [11], [12] have attracted a surge of interest and showed the effectiveness in learning graph representations. These methods are usually trained in a supervised fashion, which demands the task-specific labeled data. However, there are some aspects of shortcomings for the supervised training of GNN. First, task-specific labels can be quite scarce for graph datasets (e.g., labeling biology and chemistry graph through human annotations are often resource-intensive). Second, due to the limited size of graph datasets, supervised GNNs are often confronted with the overfitting and oversmoothing problems [4], which limits their generalization capability to other tasks [76]. Therefore, it is highly desirable to learn the transferable and generalizable graph representations in a self-supervised way on the large-scale pretraining graph data. To this end, self-supervised approaches, such as generative methods [14], [71], predictive methods [68], and contrastive methods [13], [15], [18], are coupled with GNNs to enable the graph representation learning leveraging unlabeled data. The learned representations from well-designed self-supervised pretext tasks are then transferred to downstream tasks. Inspired by the advances of contrastive learning in those domains, graph contrastive learning (GCL) has been proposed and then attracted huge attention for graph representation learning.

GCL is mainly based on maximizing the agreement of two views extracted from the same graph against those from different graphs. The former are regarded as positive pairs and later as negative ones. Specifically, three sequential components should be well-designed for GCL, namely, data augmentation, pretext task, and contrastive objectives [76], [77]. The first is the key for generating multiple appropriate views. Due to the inherent non-Euclidean properties of graph data, it is difficult to directly apply data augmentations designed for images to graphs. Typical graph data augmentations includes shuffling node features [69], [70]; perturbing structural connectivity through adding, masking, and deleting nodes [68], [72]; and subgraph sampling [18]. The pretext task of GCL contrasts two

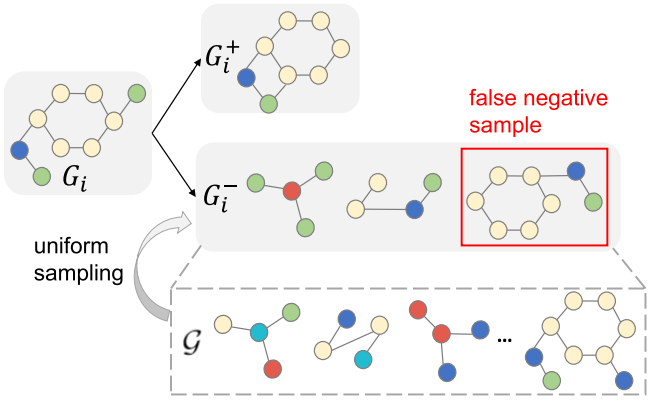


Fig. 1. "Sampling bias": the strategy of sampling negative examples uniformly from the data distribution $\mathcal{G}$ could result in that the sampled negatives G_i^- are semantically similar to the query G_i, e.g., they all contain the hexagonal structure that resembles a benzene ring.

graph views at the same scale or different scales. The scale of the view may be local, contextual, or global, corresponding to the node-, subgraph-, or graph-level information in the graph [13], [28], [59], [73]–[75]. The main way to optimize the graph contrastive objective is maximizing the mutual information (MI) or the lower bounds of MI (e.g., InfoNCE loss [6]) for two views. Typically, GraphCL [18] introduces four types of graph augmentations (namely, node dropping, edge perturbation, attribute masking, and subgraph sampling) and optimizes the InfoNCE loss on graph-level augmentations.

However, all these graph contrastive methods suffer from the following limitations. First, existing methods mainly focus on modeling the instance-level structure similarity but fail to discover the underlying global structure over the whole data distribution. However, in practice, there are underlying global structures in the graph data in most cases. For example, the graph MUTAG dataset [12] is a dataset of mutagenic aromatic and heteroaromatic nitro compounds with seven discrete categories that have underlying global structures but are not labeled to boost the representation learning. Second, as shown in Fig. 1, for a query, the common practice of sampling negatives uniformly from the whole data distribution could result in the fact that negatives are actually semantically similar to the query. However, these "false" negatives but really "right" positives are undesirably pushed apart by the contrastive loss. This phenomenon, which we call 0"sampling bias," can empirically lead to significant performance degradation [20]. Essentially, instancewise contrastive learning learns an embedding space that only preserves the local similarity around each instance but largely ignores the global-semantic structure of the whole graph data.

In this article, we propose prototypical graph contrastive learning (PGCL), a new framework that clusters semantically similar graphs into the same group and simultaneously encourages the clustering consistency between different augmentations of the same graph. The global-semantic structure of the entire dataset is depicted by PGCL in prototype vectors (i.e., trainable cluster centroids). Moreover, to address the sampling bias issue, we perform negative sampling via selecting the graphs from those clusters that differ from the query cluster. Specifically, we devise a reweighted contrastive objective, which reweights the negative samples based on the distance between their prototypes and the query prototype. In this way, those negative pairs having moderate prototype distance enjoy relatively large weights, which ensures the semantic difference between the query and its negative samples. In short, the contributions of this article can be summarized as follows.

1) We propose PGCL, a novel framework that clusters semantically similar graphs into the same group and simultaneously encourages the clustering consistency between different augmentations of the same graph.
2) We design a reweighted contrastive objective, which reweights the negative samples based on their prototype distance, to mitigate the sampling bias issue.
3) Combining both technical contributions into a single model, PGCL outperforms instancewise contrastive learning on multiple datasets in the task of unsupervised graph classification.

II. Related Work

A. Graph Representation Learning

Traditionally, graph kernels are widely used for learning node and graph representations. This common process includes meticulous designs such as decomposing graphs into substructures and using kernel functions such as the Weisfeiler–Leman graph kernel [21] to measure graph similarity between them. However, they usually require nontrivial handcrafted substructures and domain-specific kernel functions to measure the similarity while yielding inferior performance on downstream tasks, such as node classification and graph classification. Moreover, they often suffer from poor scalability [22] and huge memory consumption [23] due to some procedures such as path extraction and recursive subgraph construction. Recently, there has been increasing interest in GNN approaches for graph representation learning and many GNN variants have been proposed [11], [12], [24]. A general GNN framework involves two key computations for each node at every layer: 1) AGGREGATE operation: aggregating messages from neighborhood and 2) UPDATE operation: updating node representation from its representation in the previous layer and the aggregated messages. However, they mainly focus on supervised settings and differ from our unsupervised representation learning.

B. Contrastive Learning for GNNs

There are some recent works that explored GCL with GNNs in the aspects of data augmentations [18], [27], pretext task designs [25], [26], and contrastive objective [16]. These methods can be mainly categorized into two types: global–local contrast and global–global contrast. The first category [13], [16], [28], [29], [62] follows the InfoMax principle to maximize the MI between the local feature and the context representation. Another line of GCL approaches called global–global contrast [8], [17], [18], [25] directly studies the relationships between the global context representations of

different samples as what metric learning does. Specifically, deep graph infomax (DGI) [28] adapted the idea from deep InfoMax (DIM) to graphs for node representation learning via contrasting local node and global graph encodings with a summary vector. Further inspired by DGI, InfoGraph [13] extends DIM to learn graph-level representations by maximizing the agreements between the representations of entire graphs and the representations of substructures of different scales (e.g., nodes, edges, and triangles). MVGRL [16] trains the encoders through maximizing the MI from different structural views of graphs as well, while the same graph's adjacency and diffusion matrix were assumed as local and global views of a graph. Contrasting encodings between node representations of one view and graph representations of another view and vice versa yield better results compared with contrasting on local–local and global–global levels. DiGCL [62] extends the constractive paradigm to directed graphs and aims at learning on abundant views while retaining the original structure information. Candidate views at topological and feature levels are generated by a Laplacian perturbation operation on adjacency and node feature matrix. A GNN-based encoder could therefore be adopted to the augmented views afterward. CGCN [84] explores whether unsupervised graph learning can boost semisupervised learning. In contrast, PGCL focuses on unsupervised GCL.

More recently, MoCL [7] and KCL [8] both introduce the domain knowledge (e.g., manual rules and knowledge graph) into GCL and design knowledge-guided graph augmentation approaches to extract views. They boost the performance on the molecular graph since they have learned from the knowledge-guided augmented graphs. However, all these methods above are only able to model the discriminative relations between different graph instances, while they fail to discover the underlying semantic structure of the data distribution. Meanwhile, randomly uniform negative sampling could lead to obtain the "false" negative pairs [20], [30], [86]. This sampling bias phenomenon can empirically lead to significant performance degradation [20]. Therefore, pondering how to sampling negative pairs with both structural and global-semantic information carefully is the key process for GCL. To mitigate the sampling bias issue, AFGRL [86] proposes an augmentation-free self-supervised method by merely generating positive pairs given a target node embedding for node-level tasks, while PGCL focuses on the graph-level tasks.

C. Clustering-Based Contrastive Learning

Our work is also related to clustering-based representation learning methods [32]–[35], [46], [58], [60], [61], [79]–[81], [90], [91]. Among them, DeepCluster [58] and PCL [81] show that K-means assignments can be used as pseudo-labels to learn visual representations. PCL [81] introduces prototypes as latent variables to help find the maximum-likelihood estimation of the network parameters in an expectation–maximization (EM) framework, which encourages representations to be closer to their prototypes. Other works [32], [34] show how to cast the pseudo-label assignment problem as an instance of the optimal transport problem. However, these methods are mainly developed for images instead of graph-structured data. Different data require distinct solutions, e.g., data augmentations. In contrast, the works [64]-[67], [82], [83], [87] recently adapt the clustering idea to graph domain.

Concretely, a self-supervised contrastive attributed graph clustering approach [67] is proposed to benefit from imprecise clustering labels for the node classification task. With stochastic graph augmentation schemes, augmented node attribute and topological graph structure are projected to low-dimensional vectors. Intracluster nodes were pulled together and intercluster nodes were pushed away in this process. Pre-GNN [66] designs a novel iterative feature clustering module that could be easily plugged into GCN. It is based on feature clustering and the pseudo labels predicted can both be updated in an EM-like style, which can further facilitate the node classification. Liu *et al.* [64] proposed a multilayer graph contrastive clustering network, which clusters the nodes into different communities according to their relation types. Representations of the same node in different layers and different nodes were pulled closer and pushed away by a contrastive objective. NCL [89] utilizes the cluster-based GCL in the area of recommendation. For the multiview attributed graph data, MCGC [65] learns the consensus graph by weighing different views and regularizes by graph contrastive loss. Jiang *et al.* [82] proposed a graph pretraining approach for the heterogeneous graph, i.e., containing different types of nodes and edges. Hou *et al.* [83] utilized the neural graph matching to pretrain GNN. Concurrent to our work, GraphLoG [61] also brings together a clustering objective with graph representation learning. Similar to PCL [81], GraphLoG applies K-means clustering to capture the graph semantic structure, but utilizing K-means trivially could lead to imbalanced assignments of prototypes [32]. Compared to GraphLoG, the proposed PGCL adds the constraint that the prototype assignments must be partitioned in equally sized subsets and formulates it as an optimal transport problem. Moreover, PGCL aims to solve the sampling bias via sampling negatives from the clusters that differ from the query cluster and also reweighting negatives according to their prototype distances.

III. Preliminaries

A. Problem Definition

A desirable representation should preserve the local similarity among different data instances. We give the more detailed discussions following [61].

1) Local-Instance Structure: We refer to the local pairwise similarity between various graph examples as the local-instance structure [36], [61]. In the paradigm of contrastive learning, the embeddings of similar graph pairs are expected to be close in the latent space, while the dissimilar pairs should be mapped far apart.

The modeling of local-instance structure alone is usually insufficient to discover the global semantics underlying the entire dataset. It is highly desirable to capture the global-semantic structure of the data, which is defined as follows.

2) Global-Semantic Structure: Graph data from the real world can usually be organized as semantic clusters [37], [61].

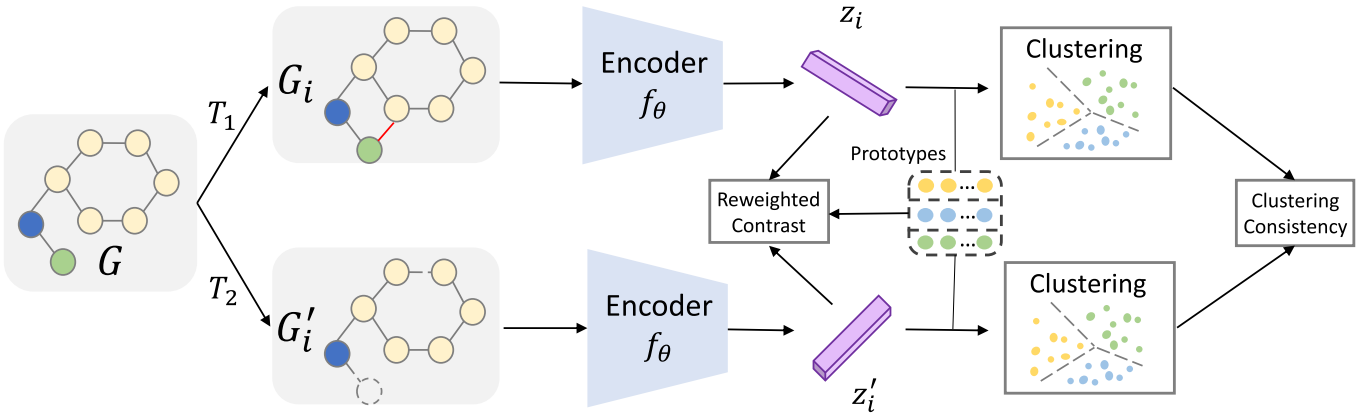


Fig. 2. Overview of PGCL. Two graph data augmentations T_1 and T_2 are applied to the input graph G. Then, two graph views G_i and G_i' are fed into the shared encoder f_θ (including GNNs and a projection head) to extract the graph representations z_i and z_i'. We perform the online clustering via assigning the representations of samples within a batch to prototype vectors (cluster centroids). The representations are learned via encouraging the clustering consistency between correlated views (Section IV-A) and a reweighted contrastive objective (Section IV-B), where prototype vectors are also updated along with the encoder parameters by backpropagation.

The embeddings of nearby graphs in the latent space should embody the global structures, which reflects the semantic patterns of the original data.

3) Problem Setup: Given a set of unlabeled graphs $\mathcal{G} = \{G_i\}_{i=1}^N$, the problem of unsupervised graph representation learning aims at learning the low-dimensional vector $z_i \in \mathbb{R}^D$ of every graph $G_i \in \mathcal{G}$, which is favorable for downstream tasks, such as graph classification.

B. Graph Neural Networks

In recent years, GNNs [11], [12], [38] have emerged as a promising approach for learning representations of graph data. We represent a graph instance as $G = (\mathcal{V}, \mathcal{E})$ with the node set $\mathcal{V}$ and the edge set $\mathcal{E}$. The dominant ways of graph representation learning are GNNs with neural message passing mechanisms [39]: at the kth iteration/layer, node representation $\mathbf{h}_v^k$ for every node $v \in \mathcal{V}$ is iteratively computed from the features of their neighbor nodes $\mathcal{N}(v)$ using a differentiable aggregation function. Specifically, at the iteration k, we get the embedding of node v in the kth layer as

$$\mathbf{h}_v^k = \text{COMBINE}^k\left(\mathbf{h}_v^{k-1}, \text{AGGREGATE}_k\left(\left\{\mathbf{h}_u^{k-1} \forall u \in \mathcal{N}(v)\right\}\right)\right). \tag{1}$$

Then, the graph-level representations can be attained by aggregating all node representations using a readout function, i.e.,

$$f_\theta(G_i) = \text{READOUT}\left(\left\{\text{CONCAT}\left(\left\{\mathbf{h}_j^k\right\}_{k=1}^K\right)\right\}_{j=1}^N\right) \tag{2}$$

where $f_\theta(G_i)$ is the entire graph's embedding and READOUT represents averaging or a more sophisticated graph-level pooling function [40], [41].

C. Graph Contrastive Learning

To empower the GNN pretraining with unlabeled data, GCL has been explored a lot recently [13], [16]–[18]. GCL performs pretraining through maximizing the agreement between two augmented views of the same graph via a contrastive loss in the latent space. GCL first augments the given graph to get augmented views G_i and G_i', which are correlated (positive) pairs. Then, G_i and G_i' are fed into a shared encoder f_θ (including GNNs and a following projection head) for extracting graph representations z_i, $z_i' = f_\theta(G_i)$, $f_\theta(G_i')$ Then, a contrastive loss function $\mathcal{L}(\cdot)$ is defined to enforce maximizing the consistency between positive pairs z_i and z_i' compared with negative pairs, such as InfoNCE loss [19], [42], [43]

$$\mathcal{L}_{\text{InfoNCE}} = -\sum_{i=1}^{n} \log \frac{\exp\left(z_i \cdot z_i'/\tau\right)}{\exp\left(z_i \cdot z_i'/\tau\right) + \sum_{j=1, j\neq i}^{2N} \exp\left(z_i \cdot z_j/\tau\right)} \tag{3}$$

where z_i and z_i' are positive embeddings for graph G_i, z_j denotes the embedding of a different graph G_j (i.e., negative embeddings), and τ is the temperature hyperparameter. Similar to [61] and [63], in the graph-structured data, there is an underlying set of discrete latent classes $\mathcal{C}$ that represent semantic structures, which could result in that G_i and G_j are actually similar.

IV. Prototypical Graph Contrastive Learning

In this section, we introduce the PGCL approach. Our goal is to cluster semantically similar graphs into the same group and simultaneously encourage the clustering consistency between different augmentations of the same graph (i.e., correlated views). As shown in Fig. 2, the representations of correlated views are encouraged to be clustered to have the same prototype (cluster centroid). Moreover, PGCL also designs a reweighted contrastive objective to sample the negatives from different clusters and reweight them according to their prototype distance. In the following, we introduce the PGCL in detail.

A. Clustering Consistency for Correlated Views

Formally, consider a GNN $z_i = f_\theta(G_i)$ mapping graph example G_i to representation vectors $z_i \in \mathbb{R}^D$. We can cluster

all representations z_i into K clusters whose centroids are denoted by a set of K trainable prototype vectors $\{c_1, \ldots, c_K\}$. Prototype vectors are trainable weight matrix of a feedforward network and initialized with He initialization [78]. For brevity, we denote by $C \in \mathbb{R}^{K\times D}$ the matrix whose columns are $c_1, \ldots, c_K$. In practice, C could be implemented by a single linear layer. In this way, given a graph G_i, we can perform clustering by computing the similarity between the representation $z_i = f_\theta(G_i)$ and the K prototype as follows:

$$p(y|z_i) = \text{softmax}(C \cdot f_\theta(G_i)). \tag{4}$$

Similarly, the prediction, i.e., $p(y|z_i')$, of assigning G_i' to prototypes can also be computed with its representation z_i'. To encourage the clustering consistency between two correlated views G_i and G_i', we predict the cluster assignments of G_i' with the representation z_i (rather than z_i') from the correlated view and vice versa. Formally, we define the clustering consistency objective via minimizing the average cross-entropy loss

$$\ell(p_i, q_{i'}) = -\sum_{y=1}^{K} q\left(y\middle|z_i'\right) \log p(y|z_i) \tag{5}$$

where $q(y|z_i')$ is the prototype assignment of the view G_i' and can serve as the target of the prediction $p(y|z_i)$ with z_i. The consistency objective acts as a regularizer to encourage the similarity of views from the same graph. We can obtain another similar objective if we swap the positions of z_i and z_i' in (5) and the ultimate consistency regularizer can be derived by the sum of two objectives

$$\mathcal{L}_{\textbf{consistency}} = \sum_{i=1}^{n} [\ell(p_i, q_{i'}) + \ell(p_{i'}, q_i)]. \tag{6}$$

The consistency regularizer can be interpreted as a way of contrasting between multiple graph views by comparing their cluster assignments rather than their representations. In practice, optimizing the distribution q faces the degeneracy problem since (5) can be trivially minimized by allocating all data samples to a single prototype. To avoid this, we add the constraint that the prototype assignments must be equally partitioned following [34]. We calculate the objective in a minibatch manner for an efficient online optimization as

$$\begin{aligned} &\min_{p,q} \mathcal{L}_{\textbf{consistency}} \\ &\text{s.t. } \forall y : q(y|z_i) \in [0, 1] \text{ and } \sum_{i=1}^{N} q(y|z_i) = \frac{N}{K}. \end{aligned} \tag{7}$$

The constraints mean that the prototype assignments to clusters $q(y|z_i)$ of each graph example x_i are soft labels and that, overall, the N graph examples within a minibatch are split uniformly among the K prototypes. The objective in (7) is an instance of the optimal transport problem, which can be addressed relatively efficiently. For more clarity, we denote two $K \times N$ matrices of joint probabilities as

$$P = \frac{1}{N} p(y|z_i); \quad Q = \frac{1}{N} q(y|z_i). \tag{8}$$

Then, we can impose an equal partition by enforcing the matrix Q to be a transportation polytope following [32], [34] in the minibatch manner:

$$T = \left\{ Q \in \mathbb{R}_+^{K\times N} \mid Q\mathbb{1}_N = \frac{1}{K}\mathbb{1}_K, Q^\top \mathbb{1}_K = \frac{1}{N}\mathbb{1}_N \right\} \tag{9}$$

where $\mathbb{1}_N$ and $\mathbb{1}_K$ denote the vector of all ones with dimension of N and K, respectively. Then, the loss function in (7) can be rewritten as

$$\min_{p,q} \mathcal{L}_{\textbf{consistency}} = \min_{Q\in\mathbf{T}} \langle Q, -\log P\rangle - \log N \tag{10}$$

where $\langle\cdot\rangle$ is the Frobenius dot product between two matrices and log is applied elementwise. Optimizing (10) always leads to an integral solution despite having relaxed Q to the continuous polytope T instead of the discrete one. We solve the transport problem via utilizing the Sinkhorn–Knopp algorithm [45] and the solution of (10) takes the form as

$$Q = \text{Diag}(\alpha) P^\eta \text{Diag}(\beta) \tag{11}$$

where α and β are two renormalization vectors and the exponentiation is elementwise. Here, η is chosen to trade off convergence speed with closeness to the original transport problem and it is a fixed value in our case. The renormalization vectors can be calculated using matrix multiplications with the Sinkhorn–Knopp algorithm [45]. Note that the first term of (10) is $\langle Q, -\log P\rangle$, while that is $\langle Q, P\rangle$ in [45, eq. (2)]. Thus, the original exponential term is replaced with P^η.

B. Reweighted Contrastive Objective

In this section, we introduce how to mitigate the sampling bias issue via sampling graphs from distinct clusters to the query and reweighting the negative samples. In the image domain, some previous works [20], [47] propose to approximate the underlying "true" distribution of negative examples by adopting a PU-learning viewpoint [48]. However, such approximation is sensitive to the hyperparameter choice and cannot avoid sampling the semantically similar pairs essentially. Given a query (and its cluster), we can achieve this simply by drawing "true" negative samples from different clusters. Since different clusters represent distinct underlying semantics, such sampling strategy can ensure the semantic differences between the query and its negatives, and (3) can be extended to

$$\mathcal{L} = -\sum_{i=1}^{n} \log \frac{\exp\left(z_i \cdot z_i'/\tau\right)}{\exp\left(z_i \cdot z_i'/\tau\right) + \sum_{j=1, j\neq i}^{2N} \mathbb{1}_{c_i \neq c_j} \cdot \exp\left(z_i \cdot z_j'/\tau\right)} \tag{12}$$

where c_i and c_j are the prototype vectors of graphs G_i and G_j, respectively, and $\mathbb{1}_{c_i\neq c_j}$ is the indicator that represents whether two samples are from different clusters. In this way, selected negative samples can enjoy desirable semantic difference from the query and those similar ones are "masked" out in the objective.

Beyond selecting negative samples based on the distinction of their clusters, we would like to avoid selecting too easy

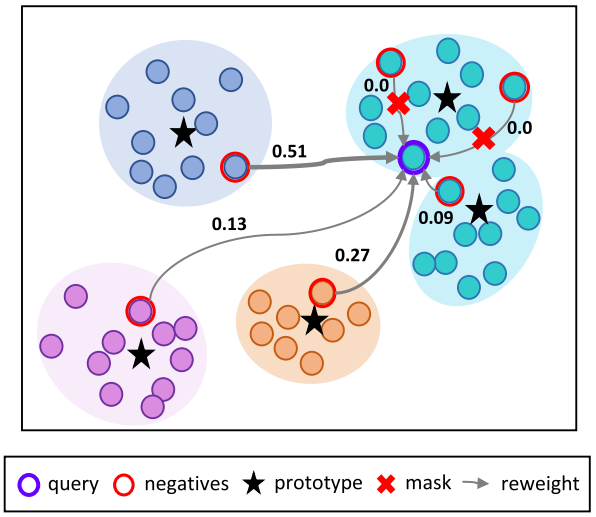


Fig. 3. Illustration of the negative sample reweighting. The linewidth of the arrow denotes the weight value.

samples that are far from the query in the latent space. Furthermore, intuitively, the desirable negative samples should have a moderate distance to the query. Empirically, we found that controlling their prototype distances performs better than using their direct sample distance. As shown in Fig. 3, on the one hand, if the prototypes of negatives are too close to the query's prototype, negatives could still share the similar semantic structure with the query (e.g., the nearby cyan cluster). On the other hand, if the prototypes of negatives (such as the purple cluster) are far from the prototype of a query, it means that negatives and query are far from each other and can be well distinguished, which actually does not help the representation learning

$$\mathcal{L}_{\mathbf{Reweighted}} = -\sum_{i=1}^{n} \times \log \frac{\exp\left(z_i \cdot z_i'/\tau\right)}{\exp\left(z_i \cdot z_i'/\tau\right) + M_i \sum_{j=1, j\neq i}^{2N} \mathbb{1}_{c_i \neq c_j} \cdot \boldsymbol{w}_{ij} \cdot \exp\left(z_i \cdot z_j'/\tau\right)} \tag{13}$$

where w_{ij} is the weight of negative pairs (G_i, G_j) and $M_i = (2N/(\sum_{j=1}^{2N} \boldsymbol{w}_{ij}))$ is the normalization factor. We utilize the cosine distance to measure the distance between two prototypes c_i and c_j as: $\mathcal{D}(c_i, c_j) = 1-(c_i \cdot c_j)/(\|c_i\|_2 \|c_j\|_2)$. Then, we define the weight based on the above prototype distance with the format of the Gaussian function as

$$\boldsymbol{w}_{ij} = \exp\left\{-\frac{\left[\mathcal{D}(c_i, c_j) - \mu_i\right]^2}{2\sigma_i^2}\right\} \tag{14}$$

where μ_i and σ_i are the mean and standard deviation of $\mathcal{D}(c_i, c_j)$ for query G_i, respectively.

As shown in Fig. 3, the reweighting strategy encourages that larger weights are assigned to meaningful negative samples (such as from the blue and orange clusters) with a moderate prototype distance to the query and smaller weights to too easy

Algorithm 1 Pseudocode of PGCL in Pytorch-Like Style

```
# C: prototypes (DxK)
# model: GIN + projection head
# temp: temperature

for x in loader: # load a batch x with N samples
 G1 = T1(x) # T1 is a random augmentation
 G2 = T2(x) # T2 is a another random augmentation
 z = model(cat(G1, G2)) # embeddings: 2NxD

 scores = mm(z, C) # prototype scores: 2NxK
 scores1 = scores[:N]
 scores2 = scores[N:]

 # cluster assignments
       with torch.no_grad():
 q1 = sinkhorn(scores)
 q2 = sinkhorn(scores2)

       # convert scores to probabilities
       p1 = Softmax(scores1 / temp)
       p2 = Softmax(scores2 / temp)

       # clustering consistency loss
       Loss_Consistency = - 0.5 * mean(q2 * log(p1) + q1 *
log(p2))

       # reweighted contrastive loss
       Compute Loss_Reweighted according to Eq.(13).

       # final loss
       loss = Loss_Reweighted + n * Loss_Consistency

       # SGD update: network and prototypes
       loss.backward()
       update(model.params)
       update(C)

       # normalize prototypes
       with torch.no_grad():
        C = normalize(C, dim=0, p=2)
```

negative samples (e.g., from the purple cluster) and "false" negative samples (from the nearby cyan cluster). The strategy is similar to those in [49] and [50], but they apply it on training samples under supervised learning, while we adopt it for selecting negative samples of self-supervised learning. The final training objective couples $\mathcal{L}_{\mathbf{Reweighted}}$ and $\mathcal{L}_{\mathbf{Consistency}}$ as

$$\mathcal{L} = \mathcal{L}_{\mathbf{Reweighted}} + \lambda \mathcal{L}_{\mathbf{Consistency}} \tag{15}$$

where the constant λ balances the reweighted contrastive loss $\mathcal{L}_{\mathbf{Reweighted}}$ and the consistency regularizer $\mathcal{L}_{\mathbf{Consistency}}$. This loss function is jointly minimized with respect to the prototypes $\mathbf{C}$ and the parameters θ of the graph encoder used to produce the representation z_i.

V. Experiments

This section is devoted to the empirical evaluation of the PGCL approach. Our initial focus is on unsupervised learning. We further apply PGCL to the transfer learning setting to test the out-of-distribution performance. Finally, we perform the extensive experiments for analysis, including ablation studies, sensitivity analysis, and visualization on the unsupervised learning datasets.

A. Unsupervised Learning

1) Task and Datasets: We conduct experiments by comparing with the state-of-the-art competitors on the unsupervised graph classification task [13], [18], where we only have access to all unlabeled samples in the dataset. We pretrain using the whole dataset to learn graph embeddings and feed them

TABLE I

GRAPH CLASSIFICATION ACCURACIES (%) OF KERNEL, SUPERVISED, AND UNSUPERVISED METHODS. WE REPORT THE MEAN TENFOLD CROSS-VALIDATION ACCURACY WITH FIVE RUNS. "> 1 DAY" REPRESENTS THAT THE COMPUTATION EXCEEDS 24 h

	Datasets	MUTAG	PTC	PROTEINS	NCI1	COLLAB	RDT-B	RDT-M5K
Datasets	# graphs	188	344	1113	4110	5000	2000	2000
	Avg # nodes	17.9	14.3	39.1	29.9	74.5	429.6	429.6
Supervised	GRAPHSAGE [39]	85.1±7.6	63.9±7.7	75.9±3.2	77.7±1.5	-	-	-
	GCN [11]	85.6±5.8	64.2±4.3	76.0±3.2	80.2±2.0	79.0±1.8	50.0±0.0	20.0 ± 0.0
	GIN-0 [12]	**89.4±5.6**	**64.6±7.0**	**76.2±2.8**	**82.7±1.7**	**80.2±1.9**	**92.4±2.5**	**57.5±1.5**
	GIN-ϵ [12]	89.0±6.0	63.7±8.2	75.9±3.8	**82.7±1.6**	80.1±1.9	92.2±2.3	57.0±1.7
Kernel	GL [52]	81.7±2.1	57.3±1.4	-	53.9±0.4	**56.3±0.6**	77.3±0.2	41.0±0.2
	WL [21]	80.7±3.0	58.0±0.5	72.9±0.6	80.0±0.5	-	68.8±0.4	46.1±0.2
	DGK [53]	87.4±2.7	60.1±2.6	**73.3±0.8**	**80.3±0.5**	-	**78.0±0.4**	41.3±0.2
	MLG [23]	87.9±1.6	**63.3±1.5**	41.2±0.0	>1 Day	>1 Day	63.3±1.5	**57.3±1.4**
	GCKN [54]	**87.2±6.8**	-	50.8±0.8	70.6±2.0	54.3±1.0	58.4±7.6	**57.3±1.4**
Unsupervised	GRAPH2VEC [55]	83.2±9.3	60.2±6.9	73.3±2.1	73.2±1.8	-	75.8±1.0	47.9±0.3
	INFOGRAPH [13]	89.0±1.1	61.7±1.7	74.4±0.3	73.8±0.7	67.6±1.2	82.5 ±1.4	53.5±1.0
	MVGRL [16]	89.7±1.1	62.5±1.7	-	75.0±0.7	68.9±1.9	84.5±0.6	-
	GCC [25]	86.4±0.5	58.4±1.2	72.9±0.5	66.9±0.2	75.2±0.3	88.4±0.3	52.6±0.2
	GRAPHCL [18]	86.8±1.3	58.4±1.7	74.4±0.5	77.9±0.4	71.4±1.2	89.5±0.8	56.0±0.3
	PGCL (ours)	**91.1±1.2**	**63.3±1.3**	**75.7±0.2**	**78.8±0.8**	**76.0±0.3**	**91.5±0.7**	**56.3±0.2**

into a downstream SVM classifier with tenfold cross validation. For this task, we conduct experiments on seven well-known benchmark datasets [51], including four bioinformatics datasets (MUTAG, PTC, PROTEINS, and NCI1) and three social network datasets (COLLAB, RDT-B, and RDT-M5K) with statistics summarized in Table I.

2) Baselines: In the unsupervised graph classification, PGCL is evaluated following [13], [18]. We compare our results with five graph kernel Methods, including graphlet kernel (GL) [52], Weisfeiler–Lehman (WL) subtree kernel [21], deep graph kernel (DGK) [53], multiscale Laplacian kernel (MLG) [23], and graph convolutional kernel network (GCKN[1]) [54]. We also compare with four supervised GNNs reported in [12], including GraphSAGE [39] and GCN [11], and two variants of graph isomorphism network (GIN) [12], GIN-0 and GIN-ϵ. Finally, we compare with five unsupervised methods, including Graph2Vec [55], InfoGraph [13], MVGRL [16], GCC [25], and GraphCL [18]. We report the results of unsupervised methods based on the released code.

3) Model Configuration: We use the GIN [12] as the encoder following [18] to attain node representations for unsupervised graph classification. All projection heads are implemented as two-layer MLPs. For unsupervised graph classification, we adopt LIB-SVM [56] with C parameter selected in $\{10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}\}$ as our downstream classifier. Then, we use tenfold cross-validation accuracy as the classification performance and repeat the experiments five times to report the mean and standard deviation. We adopt "node dropping" and "edge perturbation" as the two types of graph augmentations, which performs better than other augmentations (e.g., "subgraph") empirically, referring to the implementation.[2] Prototype vectors are initialized with the default He initialization [78] in Pytorch. To help the very beginning of the optimization, we freeze the prototypes during the first few epochs of training and focus on learning the graph representation first. Then, the prototype vectors are involved in the optimization of PGCL progressively. The best hyperparameter λ to balance the consistency regularizer and the reweighted contrastive objective is 6. Also, the number of prototypes is set to 10. The source code of PGCL will be released for reproducibility.

4) Experimental Results: The results of unsupervised graph-level representations for downstream graph classification tasks are presented in Table I. Overall, from the table, we can see that our approach achieves state-of-the-art results with respect to other unsupervised models across all seven datasets. PGCL consistently performs better than unsupervised baselines by considerable margins. For example, on the RDT-B dataset [53], it achieves 91.5% accuracy, i.e., a 2.0% absolute improvement over previous the state-of-the-art method (GraphCL [18]). Our model also outperforms graph kernel methods in four out of seven datasets and outperforms the best supervised model in one of the datasets. For example, it harvests a 2.4% absolute improvement over the state-of-the-art graph kernel method (DGK [53]) on the PROTEINS dataset. When compared to supervised baselines individually, our model outperforms GraphSAGE in two out of four datasets and outperforms GCN in three out of seven datasets, e.g., a 5.5% absolute improvement over GCN on the MUTAG dataset. It is noteworthy that PGCL tightens the gap with respect to the supervised baseline of GIN [12] such that their performance gap on four out of seven datasets is less than 2%. The strong

[1]We report our reproduced results of GCKN for fair comparisons since the original GCKN paper adopts different train–test splits for nested tenfold cross validation, which is different from the Stratified10fold splits of other contrastive learning works [13], [16] and ours.

[2]https://github.com/Shen-Lab/GraphCL

TABLE II

TRANSFER LEARNING PERFORMANCE FOR CHEMICAL MOLECULES PROPERTY PREDICTION (MEAN ROC-AUC ± STD. OVER TEN RUNS). THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

Downstream Dataset	BBBP	Tox21	SIDER	ToxCast	ClinTox	BACE	MUV	Average
#Molecules	2039	7831	1427	8575	1478	1513	93087	Rank
#Tasks	1	12	27	617	2	1	17	(↓)
No Pre-Train	65.8 ± 4.5	74.0 ± 0.8	57.3 ± 1.6	63.4 ± 0.6	58.0 ± 4.4	70.1 ± 5.4	71.8 ± 2.5	6.1
EdgePred [68]	67.3 ± 2.4	76.0 ± 0.6	60.4 ± 0.7	64.1 ± 0.6	64.1 ± 3.7	**79.9 ± 0.9**	74.1 ± 2.1	3.7
AttrMasking [68]	64.3 ± 2.8	**76.7 ± 0.4**	61.0 ± 0.7	64.2 ± 0.5	71.8 ± 4.1	79.3 ± 1.6	74.7 ± 1.4	2.9
ContextPred [68]	68.0 ± 2.0	75.7 ± 0.7	60.9 ± 0.6	63.9 ± 0.6	65.9 ± 3.8	79.6 ± 1.2	**75.8 ± 1.7**	3.1
InfoGraph [13]	68.8 ± 0.8	75.3 ± 0.5	58.4 ± 0.8	62.7 ± 0.4	69.9 ± 3.0	75.9 ± 1.6	75.3 ± 2.5	4.4
GraphCL [18]	69.7 ± 0.7	73.9 ± 0.7	60.5 ± 0.9	62.4 ± 0.6	**75.9 ± 2.7**	75.4 ± 1.4	69.8 ± 2.7	5.0
PGCL (Ours)	**69.8 ± 1.3**	75.6 ± 0.5	**61.6 ± 1.1**	**66.4 ± 0.2**	69.4 ± 1.4	79.3 ± 1.5	71.2 ± 1.3	**2.9**

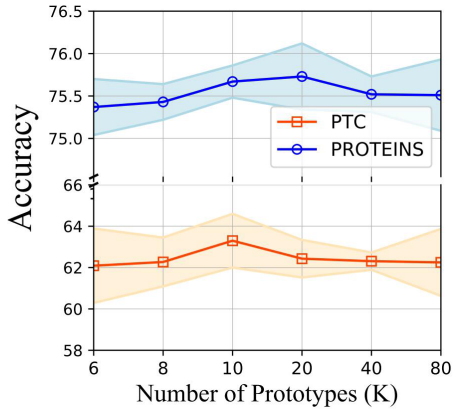


Fig. 4. Sensitivity analysis for the number of prototypes K.

performance verifies the superiority of the proposed PGCL framework.

B. Transfer Learning

1) Experimental Setup: Next, we evaluate the GNN encoders trained by PGCL on transfer learning to predict chemical molecule properties. We follow the setting in [17] and use the same datasets: GNNs are pretrained on the ZINC-2M dataset using self-supervised learning and later fine-tuned on another downstream dataset to test out-of-distribution performance. We adopt baselines, including no pretrained GIN (i.e., without self-supervised training on the first dataset and with only fine-tuning), InfoGraph [13], GraphCL [59], and three different pretrained strategies in [68], including edge prediction, node attribute masking, and context prediction. Note that Hu *et al.* [68] incorporated the domain knowledge heuristically that correlates with the specific downstream datasets.

2) Experimental Results: According to Table II, PGCL significantly outperforms all baselines in three out of seven datasets and achieves a mean rank of 2.9 across these seven datasets. Although without domain knowledge incorporated, PGCL still achieves competitive performance to heuristic self-supervised approaches [68]. Meanwhile, PGCL outperforms GraphCL [59] on unseen datasets with better generalizability. In contrast to InfoGraph [13] and GraphCL [59], PGCL achieves some performance closer to those heuristic graph pre-training baselines (EdgePred, AttrMasking, and ContextPred) based on domain knowledge in [68]. This is rather significant as our method utilizes only node dropping and edge perturbation as the data augmentation, which again shows the effectiveness of the PGCL.

TABLE III

ABLATION STUDY FOR DIFFERENT OBJECTIVE FUNCTIONS ON DOWNSTREAM GRAPH CLASSIFICATION DATASETS. AS TWO VARIANTS OF THE VANILLA INFONCE LOSS, $\mathcal{L}_{\mathbf{S.R.}}$ DENOTES CALCULATING THE WEIGHT IN (14) WITH THE SAMPLE DISTANCE, WHILE $\mathcal{L}_{\mathbf{P.R.}}$ CORRESPONDS TO THE PROTOTYPE DISTANCE

$\mathcal{L}_{\mathbf{Inf.}}$	$\mathcal{L}_{\mathbf{Con.}}$	$\mathcal{L}_{\mathbf{S.R.}}$	$\mathcal{L}_{\mathbf{P.R.}}$	MUTAG	PTC	PRO.	COLLAB
✓				86.8±1.3	58.4±1.7	74.4±0.5	71.4±1.2
	✓			89.7±1.0	61.1±1.7	75.4±0.4	71.5±1.4
		✓		89.9±1.1	61.9±0.9	73.4±0.6	72.6±0.5
			✓	90.1±0.9	62.5±0.7	75.2±0.4	73.3±0.7
✓	✓			89.9±1.0	62.4±2.1	75.4±0.3	73.3±1.2
	✓	✓		91.0±1.4	**63.4±1.5**	73.6±1.1	74.6±0.6
	✓		✓	**91.1±1.2**	63.3±1.3	**75.7±0.2**	**76.0±0.3**

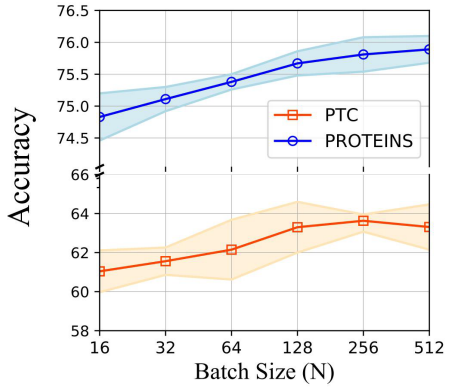


Fig. 5. Sensitivity analysis for batch size N.

C. Ablation Studies

In Table III, we analyze the effect of various objective functions, including the vanilla InfoNCE loss $\mathcal{L}_{\mathbf{Inf.}}$, its

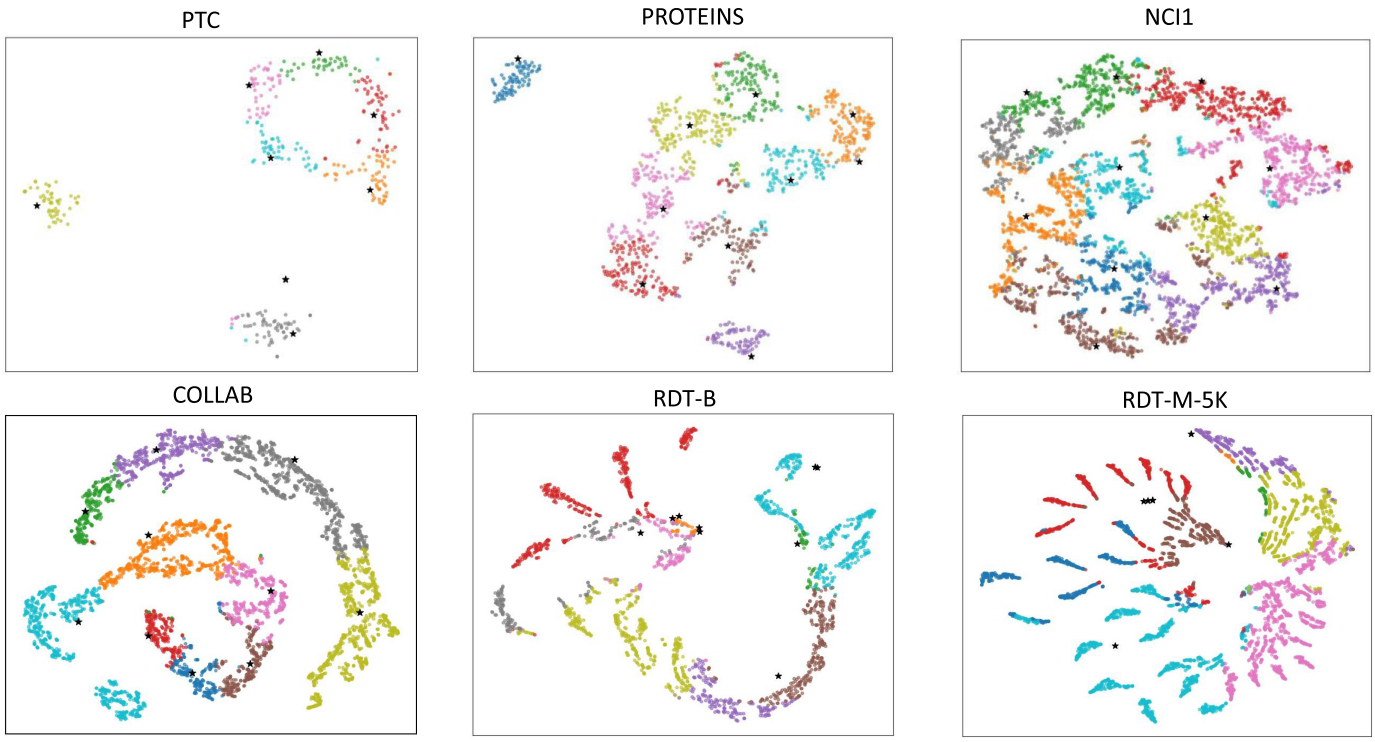


Fig. 6. T-SNE visualization of the learned representation on six datasets. "⋆" means the prototype vectors. Colors represent underlying classes that PGCL discovers.

two variants, i.e., reweighting with prototype distance $\mathcal{L}_{\mathbf{P.R.}}$ and sample distance $\mathcal{L}_{\mathbf{S.R.}}$, and the clustering consistency objective $\mathcal{L}_{\mathbf{Con.}}$. When the clustering consistency and the reweighted contrastive objective are individually applied, they perform better than the InfoNCE loss, which benefits from their explorations of the semantic structure of the data. The prototype-based reweighting objective $\mathcal{L}_{\mathbf{P.R.}}$ outperforms the sample-based one $\mathcal{L}_{\mathbf{S.R.}}$ in most datasets since the prototype plays an important role as the pseudo label during negative sampling and provides a more robust reweighting strategy. By simultaneously applying both objectives $\mathcal{L}_{\mathbf{Con.}}$ and $\mathcal{L}_{\mathbf{P.R.}}$, our full model (last row) achieves better performance than merely combining the InfoNCE loss and the clustering consistency, which indicates that the prototype-based reweighting strategy can mitigate the sampling bias problem.

D. Sensitivity Analysis

1) Sensitivity to Prototype Numbers K: In this section, we discuss the selection of parameter K that is the number of prototypes (clusters). Fig. 4 shows the performance of PGCL with a different number of prototypes K from 6 to 80 on PTC and PROTEINS. It can be observed that at beginning, increasing the number of prototypes improves the performance, while too many prototypes leads to slight performance drop, which we conjecture that the model degenerates to the case without prototypes (i.e., each graph acts as a prototype itself). Overall, our PGCL is robust to the prototype numbers.

2) Sensitivity to Batch Size N: In this experiment, we evaluate the effect of batch size N on our model performance. Fig. 5 shows the classification accuracy of our model using different batch sizes from 32 to 512 on PTC and PROTEINS. From the line chart, we can observe that a large batch size (i.e. $N > 32$) can consistently improve the performance of PGCL. This observation aligns with the case in the image domain [19].

E. Visualization Results

In Fig. 6, we utilize the t-SNE [57] to visualize the graph representations and prototype vectors with the number of clusters $K = 10$ on various datasets. Generally, the representations learned by the proposed PGCL forms separated clusters, where the prototypes fall into the center. It demonstrates that PGCL can discover the underlying global-semantic structure over the entire data distribution. Moreover, it can be observed that each cluster has a similar number of samples, which indicates the effectiveness of the equal-partition constraints during clustering.

VI. Conclusion

We proposed a clustering-based approach called PGCL for unsupervised graph-level representation learning. PGCL clusters semantically similar graphs into the same group and simultaneously encourages the clustering consistency for different graph views. Moreover, to mitigate the sampling bias issue, PGCL reweights its negative pairs based on the distance between their prototypes. Benefiting from modeling the global-semantic structure via clustering, we achieve a new state-of-the-art performance compared to previous unsupervised learning methods on seven graph classification benchmarks.

ACKNOWLEDGMENT

The authors thank MindSpore for the support of this work, which is a new deep learning computing framework (https://www.mindspore.cn).

REFERENCES

[1] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. ICML*, 2017, pp. 1263–1272.

[2] C. Chen, W. Ye, Y. Zuo, C. Zheng, and S. P. Ong, "Graph networks as a universal machine learning framework for molecules and crystals," *Chem. Mater.*, vol. 31, no. 9, pp. 3564–3572, May 2019.

[3] M. A. Alvarez and C. Yan, "A new protein graph model for function prediction," *Comput. Biol. Chem.*, vol. 37, pp. 6–10, Apr. 2012.

[4] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. AAAI*, 2018, pp. 3538–3545.

[5] Q. Zhu, B. Du, and P. Yan, "Self-supervised training of graph convolutional networks," 2020, *arXiv:2006.02380*.

[6] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proc. AISTATS*, 2010, pp. 297–304.

[7] M. Sun, J. Xing, H. Wang, B. Chen, and J. Zhou, "MoCL: Contrastive learning on molecular graphs with multi-level domain knowledge," in *Proc. SIGKDD*, 2021, pp. 3585–3594.

[8] Y. Fang *et al.*, "Molecular contrastive learning with chemical element knowledge graph," in *Proc. AAAI*, 2022, pp. 3968–3976.

[9] B. Jiang, K. Kloster, D. F. Gleich, and M. Gribskov, "AptRank: An adaptive PageRank model for protein function prediction on bi-relational graphs," *Bioinformatics*, vol. 33, no. 12, pp. 1829–1836, Jun. 2017.

[10] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, Feb. 2004, Art. no. 026113.

[11] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017.

[12] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks," in *Proc. ICLR*, 2019.

[13] F.-Y. Sun, J. Hoffmann, V. Verma, and J. Tang, "Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," in *Proc. ICLR*, 2020.

[14] T. N. Kipf and M. Welling, "Variational graph auto-encoders," in *Proc. NeurIPS*, 2016.

[15] Y. Li, C. Gu, T. Dullien, O. Vinyals, and P. Kohli, "Graph matching networks for learning the similarity of graph structured objects," in *Proc. ICML*, 2019, pp. 3835–3845.

[16] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," in *Proc. ICML*, 2020, pp. 4116–4126.

[17] H. Zhang *et al.*, "Iterative graph self-distillation," in *Proc. Workshop Self-Supervised Learn. Web (SSL@WWW)*, 2020.

[18] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," in *Proc. NeurIPS*, vol. 33, 2020, pp. 5812–5823.

[19] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. ICML*, 2020, pp. 1597–1607.

[20] C.-Y. Chuang, J. Robinson, L. Yen-Chen, A. Torralba, and S. Jegelka, "Debiased contrastive learning," in *Proc. NeurIPS*, 2020, pp. 8765–8775.

[21] N. Shervashidze, P. Schweitzer, E. J. V. Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler–Lehman graph kernels," *J. Mach. Learn. Res.*, vol. 12, no. 9, pp. 1–23, 2011.

[22] K. M. Borgwardt and H. Kriegel, "Shortest-path kernels on graphs," in *Proc. 5th IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2005, p. 8.

[23] R. Kondor and H. Pan, "The multiscale Laplacian graph kernel," in *Proc. NeurIPS*, 2016, pp. 2990–2998.

[24] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld, "Quantum chemistry structures and properties of 134 kilo molecules," *Sci. Data*, vol. 1, no. 1, pp. 1–7, Dec. 2014.

[25] J. Qiu *et al.*, "GCC: Graph contrastive coding for graph neural network pre-training," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 1150–1160.

[26] Z. Hu, Y. Dong, K. Wang, K.-W. Chang, and Y. Sun, "GPT-GNN: Generative pre-training of graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 1857–1867.

[27] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, Apr. 2021, pp. 2069–2080.

[28] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. ICLR*, 2019.

[29] Q. Sun *et al.*, "SUGAR: Subgraph neural network with reinforcement pooling and self-supervised mutual information mechanism," in *Proc. Web Conf.*, Apr. 2021, pp. 2081–2091.

[30] M. Tschannen, J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic, "On mutual information maximization for representation learning," *ICLR*, 2019.

[31] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. ECCV*, Sep. 2018, pp. 132–149.

[32] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning," in *Proc. ICLR*, 2020.

[33] M. Caron, P. Bojanowski, J. Mairal, and A. Joulin, "Unsupervised pre-training of image features on non-curated data," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2959–2968.

[34] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Proc. NeurIPS*, 2020, pp. 9912–9924.

[35] J. Huang, Q. Dong, S. Gong, and X. Zhu, "Unsupervised deep learning by neighbourhood discovery," in *Proc. ICML*, 2019, pp. 2849–2858.

[36] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.

[37] G. W. Furnas *et al.*, "Information retrieval using a singular value decomposition model of latent semantic structure," in *Proc. 11th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 1988, pp. 90–105.

[38] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, 2018.

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

[40] Z. Su, Z. Hu, and Y. Li, "Hierarchical graph representation learning with local capsule pooling," in *Proc. ACM Multimedia Asia*, Dec. 2021, pp. 1–7.

[41] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proc. AAAI*, vol. 32, 2018.

[42] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[43] R. D. Hjelm *et al.*, "Learning deep representations by mutual information estimation and maximization," in *Proc. ICLR*, 2019, pp. 1–24.

[44] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi, "A theoretical analysis of contrastive unsupervised representation learning," in *Proc. ICML*, 2019, pp. 5628–5637.

[45] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Proc. Int. Conf. Neural. Infor. Process. Syst. (NIPS)*, vol. 26, no. 12. Lake Tahoe, NV, USA, Dec. 2013 pp. 2292–2300.

[46] Z. Li, F. Nie, X. Chang, Y. Yang, C. Zhang, and N. Sebe, "Dynamic affinity graph construction for spectral clustering using multiple features," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6323–6332, Dec. 2018.

[47] J. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka, "Contrastive learning with hard negative samples," in *Proc. ICLR*, 2020.

[48] C. Elkan and K. Noto, "Learning classifiers from only positive and unlabeled data," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2008, pp. 213–220.

[49] P. Zhao and T. Zhang, "Accelerating minibatch stochastic gradient descent using stratified sampling," 2014, *arXiv:1405.3080*.

[50] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[51] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "Tudataset: A collection of benchmark datasets for learning with graphs," in *Proc. ICML Workshop Graph Represent. Learn. Beyond*, 2020.

[52] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Proc. AISTATS*, 2009, pp. 488–495.

[53] P. Yanardag and S. V. N. Vishwanathan, "Deep graph kernels," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 1365–1374.

[54] D. Chen, L. Jacob, and J. Mairal, "Convolutional kernel networks for graph-structured data," in *Proc. ICML*, 2020, pp. 1576–1586.

[55] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "Graph2vec: Learning distributed representations of graphs," 2017, *arXiv:1707.05005*.

[56] C. C. Chang and C. J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, 2011.

[57] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *JMLR*, vol. 9, no. 11, pp. 2579–2605, 2008.

[58] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. ECCV*, Sep. 2018, pp. 132–149.

[59] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning automated," in *Proc. ICML*, 2021, pp. 12121–12132.

[60] H. Zhao, X. Yang, Z. Wang, E. Yang, and C. Deng, "Graph debiased contrastive learning with joint representation clustering," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 3434–3440.

[61] M. Xu, H. Wang, B. Ni, H. Guo, and J. Tang, "Self-supervised graph-level representation learning with local and global structure," in *Proc. ICML*, 2021, pp. 11548–11558.

[62] Z. Tong, Y. Liang, H. Ding, Y. Dai, X. Li, and C. Wang, "Directed graph contrastive learning," in *Proc. NeurIPS*, vol. 34, 2021.

[63] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi, "A theoretical analysis of contrastive unsupervised representation learning," 2019, *arXiv:1902.09229*.

[64] L. Liu, Z. Kang, L. Tian, W. Xu, and X. He, "Multilayer graph contrastive clustering network," 2021, *arXiv:2112.14021*.

[65] E. Pan *et al.*, "Multi-view contrastive graph clustering," in *Proc. NeurIPS*, 2021, pp. 2148–2159.

[66] Z. Hu, G. Kou, H. Zhang, N. Li, K. Yang, and L. Liu, "Rectifying pseudo labels: Iterative feature clustering for graph representation learning," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 720–729.

[67] W. Xia, Q. Gao, M. Yang, and X. Gao, "Self-supervised contrastive attributed graph clustering," 2021, *arXiv:2110.08264*.

[68] W. Hu *et al.*, "Strategies for pre-training graph neural networks," in *Proc. ICLR*, 2020.

[69] F. L. Opolka, A. Solomon, C. Cangea, P. Veličković, P. Liò, and R. Devon Hjelm, "Spatio-temporal deep graph infomax," 2019, *arXiv:1904.06316*.

[70] Y. Ren, B. Liu, C. Huang, P. Dai, L. Bo, and J. Zhang, "Heterogeneous deep graph infomax," 2019, *arXiv:1911.08538*.

[71] Q. Zhu, B. Du, and P. Yan, "Self-supervised training of graph convolutional networks," 2020, *arXiv:2006.02380*.

[72] J. Zeng and P. Xie, "Contrastive self-supervised learning for graph classification," in *Proc. AAAI*, 2021, pp. 10824–10832.

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

[74] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, Apr. 2021, pp. 2069–2080.

[75] Y. Jiao, Y. Xiong, J. Zhang, Y. Zhang, T. Zhang, and Y. Zhu, "Sub-graph contrast for scalable self-supervised graph representation learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2020, pp. 222–231.

[76] L. Wu, H. Lin, C. Tan, Z. Gao, and S. Z. Li, "Self-supervised learning on graphs: Contrastive, generative, or predictive," *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 1, 2021, doi: 10.1109/TKDE.2021.3131584.

[77] Y. Liu *et al.*, "Graph self-supervised learning: A survey," *IEEE Trans. Knowl. Data Eng.*, early access, May 6, 2022, doi: 10.1109/TKDE.2022.3172903.

[78] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026–1034.

[79] D. Yuan, X. Chang, P. Huang, and Q. Liu, "Self-supervised deep correlation tracking," *IEEE Trans. Image Process.*, vol. 30, pp. 976–985, 2020.

[80] Z. Li, F. Nie, X. Chang, Y. Yang, C. Zhang, and N. Sebe, "Dynamic affinity graph construction for spectral clustering using multiple features," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6323–6332, Dec. 2018.

[81] J. Li, P. Zhou, C. Xiong, and S. Hoi, "Prototypical contrastive learning of unsupervised representations," in *Proc. ICLR*, 2020.

[82] X. Jiang, T. Jia, Y. Fang, C. Shi, Z. Lin, and H. Wang, "Pre-training on large-scale heterogeneous graph," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 756–766.

[83] Y. Hou, B. Hu, W. X. Zhao, Z. Zhang, J. Zhou, and J.-R. Wen, "Neural graph matching for pre-training graph neural networks," in *Proc. SDM*. Philadelphia, PA, USA: SIAM, 2022, pp. 172–180.

[84] B. Hui, P. Zhu, and Q. Hu, "Collaborative graph convolutional networks: Unsupervised learning meets semi-supervised learning," in *Proc. AAAI*, vol. 34, no. 4, 2020, pp. 4215–4222.

[85] H. Hafidi, M. Ghogho, P. Ciblat, and A. Swami, "Negative sampling strategies for contrastive self-supervised learning of graph representations," *Signal Process.*, vol. 190, Jan. 2022, Art. no. 108310.

[86] N. Lee, J. Lee, and C. Park, "Augmentation-free self-supervised learning on graphs," in *Proc. AAAI*, 2022, pp. 7372–7380.

[87] B. Hui, P. Zhu, and Q. Hu, "Collaborative graph convolutional networks: Unsupervised learning meets semi-supervised learning," in *Proc. AAAI*, 2020, pp. 4215–4222.

[88] N. Rethmeier and I. Augenstein, "A primer on contrastive pretraining in language processing: Methods, lessons learned and perspectives," 2021, *arXiv:2102.12982*.

[89] Z. Lin, C. Tian, Y. Hou, and W. X. Zhao, "Improving graph collaborative filtering with neighborhood-enriched contrastive learning," in *Proc. ACM Web Conf.*, Apr. 2022, pp. 2320–2329.

[90] P. Zhou, C. Xiong, X. Yuan, and S. Hoi, "A theory-driven self-labeling refinement method for contrastive representation learning," in *Proc. NeurIPS*, 2021, pp. 6183–8167.

[91] P. Zhou, Y. Zhou, C. Si, W. Yu, T. K. Ng, and S. Yan, "Mugs: A multi-granular self-supervised learning framework," 2022, *arXiv:2203.14415*.

Shuai Lin received the bachelor's degree in communication engineering from Xidian University, Xi'an, China, in 2019. He is currently pursuing the master's degree with the Department of Intelligent Engineering, Sun Yat-sen University, Guangzhou, China, under the supervision of Xiaodan Liang.

His main research interests include data mining and interpretable machine learning.

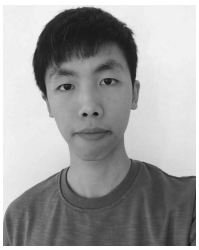
Chen Liu received the bachelor's and master's degrees in psychology from Nanjing University, Nanjing, China, in 2013, and the master's degree in psychology from the University of York, York, U.K., in 2016. She is currently pursuing the Ph.D. degree with the Department of Intelligent Engineering, Sun Yat-sen University, Guangzhou, China, under the supervision of Xiaodan Liang.

Her research interests include deep learning, graph-structured data mining, and their application in time neural signals.

Pan Zhou received the master's degree in computer science from Peking University, Beijing, China, in 2016, and the Ph.D. degree in computer science from the National University of Singapore, Singapore, in 2019.

He is currently the Senior Research Scientist with the SEA AI Laboratory, SEA Group, Singapore. From 2019 to 2020, he was a Research Scientist with Salesforce, Singapore. His research interests include computer vision, machine learning, and optimization.

Dr. Zhou was the winner of the Microsoft Research Asia Fellowship in 2018.



Zi-Yuan Hu is currently pursuing the bachelor's in computer science with Sun Yat-sen University, Guangzhou, China.

He is good at algorithm design and implementation. His research interest is data mining and cross-modality pretraining.

Shuojia Wang received the Ph.D. degree in epidemiology and health statistics from Zhejiang University, Hangzhou, China, in 2020.

She is currently the Data Scientist at the Tencent Jarvis Lab, Shenzhen, China. Her research interests include data mining, medical decision-making, and disease prediction.

Ruihui Zhao received the B.S. degree from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2015, and the M.S. degree at Waseda University, Tokyo, Japan, in 2017.

He joined the Tencent Jarvis Lab as a Senior Researcher in early 2018. Previously, he was an NLP Engineer at Sinovation Ventures, Beijing, China, from 2017 to 2018. He has several papers accepted by Annual Meeting of the Association for Computational Linguistics (ACL), International World Wide Web Conference (WWW), AAAI Conference on Artificial Intelligence (AAAI), Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), International Joint Conference on Neural Networks (IJCNN), ACM Transactions on Internet Technology (TOIT), and IEEE International Conference on Wireless Communications and Signal Processing (WCSP). His research and projects mainly focus on natural language processing (NLP) and information security.

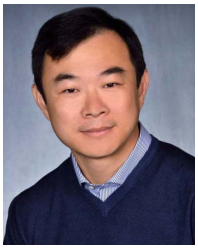
Yefeng Zheng (Fellow, IEEE) received the B.E. and M.E. degrees from Tsinghua University, Beijing, in 1998 and 2001, respectively, and the Ph.D. degree from the University of Maryland, College Park, MD, USA, in 2005.

After graduation, he joined Siemens Corporate Research, Princeton, NJ, USA. He is currently the Director and the Distinguished Scientist of Tencent Jarvis Laboratory, Shenzhen, China, leading the company's initiative on Medical AI. His research interests include medical image analysis, graph data mining, and deep learning.

Dr. Zheng is a fellow of the American Institute for Medical and Biological Engineering (AIMBE).

Liang Lin (Senior Member, IEEE) was the Executive Director of the SenseTime Group, Hong Kong, from 2016 to 2018, leading the research and development teams in developing cutting-edge, deliverable solutions in computer vision and data mining. He is currently a Full Professor of computer science with Sun Yat-sen University, Guangzhou, China. He has authored or coauthored more than 200 papers in leading academic journals and conferences, such as IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI), Conference on Neural Information Processing Systems (NeurIPS), International Conference on Machine Learning (ICML), and The Association for the Advancement of Artificial Intelligence (AAAI).

Dr. Lin is a fellow of IET. He served as the Area/Session Chair for numerous conferences, such as IEEE conference on computer vision and pattern Recognition (CVPR), International Conference on Multimedia & Expo (ICME), International Conference on Computer Vision (ICCV), and International Conference on Multimedia Retrieval (ICMR). He is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (IEEE TNNLS).



Eric Xing (Fellow, IEEE) received the Ph.D. degree in molecular biology from Rutgers University, New Brunswick, NJ, USA, in 1999, and the Ph.D. degree in computer science from the University of California at Berkeley, Berkeley, CA, USA, in 2004.

He is currently a Professor of machine learning with the School of Computer Science and the Director of the CMU Center for Machine Learning and Health, Carnegie Mellon University, Pittsburgh, PA, USA. His principal research interests lie in the development of machine learning and statistical methodology, especially for solving problems involving automated learning, reasoning, and decision-making in high-dimensional, multimodal, and dynamic possible worlds in social and biological systems.

Dr. Xing is a member of the DARPA Information Science and Technology (ISAT) Advisory Group and the Program Chair of the International Conference on Machine Learning (ICML) 2014. He is also an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (PAMI) and the *Machine Learning journal (MLJ)* and the *Journal of Machine Learning Research* (JMLR).

Xiaodan Liang (Senior Member, IEEE) received the Ph.D. degree from Sun Yat-sen University, Guangzhou, China, in 2016, under the supervision of Liang Lin.

She was a Post-Doctoral Researcher with the Department of Machine Learning, Carnegie Mellon University, Pittsburgh, PA, USA, working with Prof. Eric Xing from 2016 to 2018. She is currently an Associate Professor with Sun Yat-sen University. She has authored or coauthored several cutting-edge projects on interpretable machine learning, data mining, and graph neural network.