Disentangled Graph Contrastive Learning with Independence Promotion

Haoyang Li, Ziwei Zhang, *Member, IEEE,* Xin Wang, *Member, IEEE,* Wenwu Zhu, *Fellow, IEEE*

Abstract—Self-supervised learning for graph neural networks has attracted considerable attention and shows notable successes in graph representation learning. However, the formation of a real-world graph typically arises from highly complex interactions of many latent factors. The existing self-supervised learning methods for GNNs are inherently holistic and neglect the entanglement of the latent factors, resulting in suboptimal learned representations for downstream tasks and difficult to be interpreted. Learning disentangled graph representations with self-supervised learning poses great challenges and remains largely ignored by the existing literature. In this paper, we introduce Independence Promoted Disentangled Graph Contrastive Learning (**IDGCL**) method, which can learn disentangled graph-level representations with self-supervision. In particular, we first identify the latent factors of the input graph and derive its factorized representations. Then we propose a factor-wise discrimination objective in a contrastive learning manner, which can force the factorized representations to independently reflect the expressive information from different latent factors. To further promote the independence between the representations, we employ the Hilbert-Schmidt Independence Criterion to eliminate the dependence among different representations, which is effectively integrated into the self-supervised framework as a regularizer. Extensive experiments on synthetic and real-world datasets demonstrate the superiority of our method against several state-of-the-art baselines.

Index Terms—Graph Data Mining, Graph Neural Network, Self-supervised Learning, Disentangled Representation Learning.

1 Introduction

Graph structured data is ubiquitous in the real world, e.g., social networks, biology networks, traffic networks, etc. Recently, graph neural networks (GNNs) have become increasingly prevalent in learning graph representations in a supervised manner, demonstrating their strength in a wide variety of research fields [1-4]. GNNs require task-dependent annotated labels to learn effective representations, which are extremely scarce, or even unavailable in practice, thus motivating the advent of self-supervised graph representation learning.

Contrastive learning, as a discriminative approach pulling similar samples close and pushing dissimilar samples far away, has become a dominant strategy in self-supervised graph representation learning [5-11]. Despite their notable successes, the existing graph contrastive learning methods generally adopt a holistic scheme, i.e., the learned representations characterize graphs as a perceptual whole, ignoring the nuances between different aspects of the graph. In fact, the formation of a graph typically follows a relational process in the real world, driven by many complex latent factors. For example, in social networks, a social group may have several communities originated from different relations (e.g., friends, colleagues, etc.) or interests (e.g., sports, games, etc.) [12]. And a molecular graph may consist of various groups of atoms and bonds representing different functional units [13]. The complex relations among the multiple latent factors bring an urge for disentangling these factors in contrastive graph representation learning, which remains unexplored by the existing holistic works. As a result, the graph representations learned by the existing methods contain a mixture of entangled factors, harming interpretability and leading to suboptimal performance for predictive tasks involving whole graph representations.

In this paper, we propose to learn disentangled contrastive graph representation. Although disentangled representation learning, which aims to characterize the various underlying explanatory factors behind the observed data in different parts of the factorized representations [14, 15], has been demonstrated to be more explainable [12] and generalizable [14], disentangled graph contrastive learning faces the following three challenges. (1) Tailored graph encoder for disentangled contrastive learning. The graph encoder should be carefully designed so that it can be sufficiently expressive to infer the disentangled latent factors in the graph. (2) Tailored discrimination tasks designed for disentangled graph contrastive learning. Since task-dependent labels are not available in the self-supervised setting, disentangled graph contrastive learning can only utilize the limited amount of self-supervision information. This implies that the discrimination tasks should be well-designed for disentangled contrastive representation learning on graphs. (3) Tailored training scheme to enforce the independence of the representations. The disentangled graph representations are expected to capture mutually exclusive information in terms of the latent factors. Therefore, the statistical independence among different latent representations should be effectively formulated, which can promote the quality of disentangled representations of the graph.

To tackle these challenges, we propose a novel independence promoted disentangled graph contrastive learning model (**IDGCL**) capable of disentangled contrastive learning

- *Haoyang Li, Ziwei Zhang, Xin Wang, and Wenwu Zhu are with the Department of Computer Science and Technology in Tsinghua University, Beijing 100084, China.*
  E-mail: lihy18@mails.tsinghua.edu.cn, zwzhang@tsinghua.edu.cn, xin_wang@tsinghua.edu.cn, wwzhu@tsinghua.edu.cn
- *Corresponding authors: Xin Wang and Wenwu Zhu*

on graphs. In particular, we first design a disentangled graph encoder whose key ingredient is a multi-channel message-passing layer. Each channel is tailored to aggregate features only from one disentangled latent factor. Then a separate readout operation in each channel summarizes the specific aspect of the graph according to the corresponding latent factor, so as to produce the disentangled graph representation. Next, we conduct contrastive learning in each representation subspace characterized by each factor independently instead of in the whole representation space. This novel factor-wise contrastive approach can ensure that each disentangled factor of the vectorized representations is sufficiently discriminative only under one specific aspect of the whole graph. Thus the representations are encouraged to be disentangled and best characterize the aspect pertinent to a latent factor of the graph. In addition, to further promote the independence among different latent representations, we eliminate the statistical dependence among different channels of the representations with Hilbert-Schmidt Independence Criterion (HSIC) [16], a kernel-based metric. The factor-wise contrastive representation learning and independence regularization are jointly optimized in a unified framework, so that the disentangled graph encoder can produce better disentangled graph representations. Compared with the existing methods, our proposed model encodes a graph with multiple disentangled representations, making it possible to explore the meaning of each channel, which benefits in more explainability for producing graph representations.

We conduct extensive experiments on both synthetic graph dataset and empirical well-known graph benchmarks. The results show that the representations learned from **IDGCL** can achieve substantial performance gains on the downstream graph classification task compared with various state-of-the-art baselines.

The contributions of this paper are summarized as follows:

- We propose a novel independence promoted disentangled graph contrastive learning model (**IDGCL**), which is able to learn disentangled graph representation via factor-wise contrastive learning and independence regularization. To the best of our knowledge, we are the first to study disentangled self-supervised graph representation learning with independence promotion.
- We propose a disentangled graph encoder to capture multiple aspects of graphs through learning disentangled latent factors on graphs. We further present the factor-wise contrastive learning approach on tailored discrimination tasks in terms of each latent factor independently.
- We present a kernel-based Hilbert-Schmidt Independence Criterion (HSIC) to measure dependence among the representations in terms of different latent factors effectively and accurately. The factor-wise contrastive learning and independence regularization are jointly optimized in a unified framework so that the learned representation can better capture predictive and mutually independent information.
- We conduct extensive experiments to verify the efficacy of our proposed model for the graph classification task. The results on several graph classification datasets demonstrate that **IDGCL** achieves state-of-the-art performance by significantly outperforming the baselines.

This manuscript is an extension of our paper published at NeurIPS 2021 [17]. Compared with the conference version, we make significant contributions from the following aspects:

- The newly proposed **IDGCL** model is able to learn disentangled self-supervised graph representation via explicit enforcing independence between the latent representations so as to improve the quality of disentangled graph representations.
- The proposed independence regularization can measure dependence among the representations in terms of different latent factors accurately and without inducing high time complexity costs.
- **IDGCL** can simultaneously integrate factor-wise contrastive learning and independence among different representations into a unified framework for joint optimization.
- More extensive experiments demonstrate that **IDGCL** is able to outperform baseline approaches and the original model proposed in the earlier conference paper.

We introduce the problem formulation and preliminaries in Section 2. In Section 3, we describe the details of our proposed method. Section 4 presents the experimental results, including quantitative and qualitative comparisons. We review the related work in Section 5. Finally, we conclude our work in Section 6.

2 Problem Formulation and Preliminaries

2.1 Problem Formulation

Let $\mathbf{G} = \{G_i\}_{i=1}^N$ be a graph dataset with N graphs. The key of most self-supervised graph representation learning methods, including ours, is to derive a graph encoder $f(\cdot)$, which outputs a d-dimensional representation $\mathbf{z}_i = f(G_i) \in \mathbb{R}^d$ for each input graph, such that $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^N$ best describes $\mathbf{G}$. In this work, we aim to learn a multi-channel graph encoder $f_\theta(\cdot)$ with parameters θ, so that the output $\mathbf{z}_i$ is a disentangled representation, i.e. $f_\theta(\cdot) = \{f_\theta^{(k)}(\cdot)\}_{k=1}^K$, where K is the number of channels. To be specific, $\mathbf{z}_i$ is expected to be composed of K independent components, i.e., $\mathbf{z}_i = [\mathbf{z}_{i,1}, \mathbf{z}_{i,2}, \ldots, \mathbf{z}_{i,K}]$, where $\mathbf{z}_{i,k} = f_\theta^{(k)}(G_i) \in \mathbb{R}^{\Delta d}$, $k \in [1, K], \Delta d = d/K$, assuming that there are K latent factors behind the graph instances to be disentangled. The k^{th} component $\mathbf{z}_{i,k}$ is for characterizing the aspect of G_i that is pertinent to factor k accurately. We also assume that the value of $\mathbf{z}_{i,k}$ will be merely a white noise vector if the input graph G_i does not contain any information of factor k. Here we follow the notion in disentangled representation learning literature [12, 14, 15, 18, 19], which assumes the existence of various natural factors that vary independently behind the observed data. The goal is to learn factorized representations where each of them independently reflects the expressive information specific to only one single ground truth factor. Therefore, the setting where the raw data has highly related features is out of scope for this work.

2.2 Preliminaries on Contrastive Learning

Unlike generative models, contrastive learning is an instance-wise discriminative approach that aims at making similar

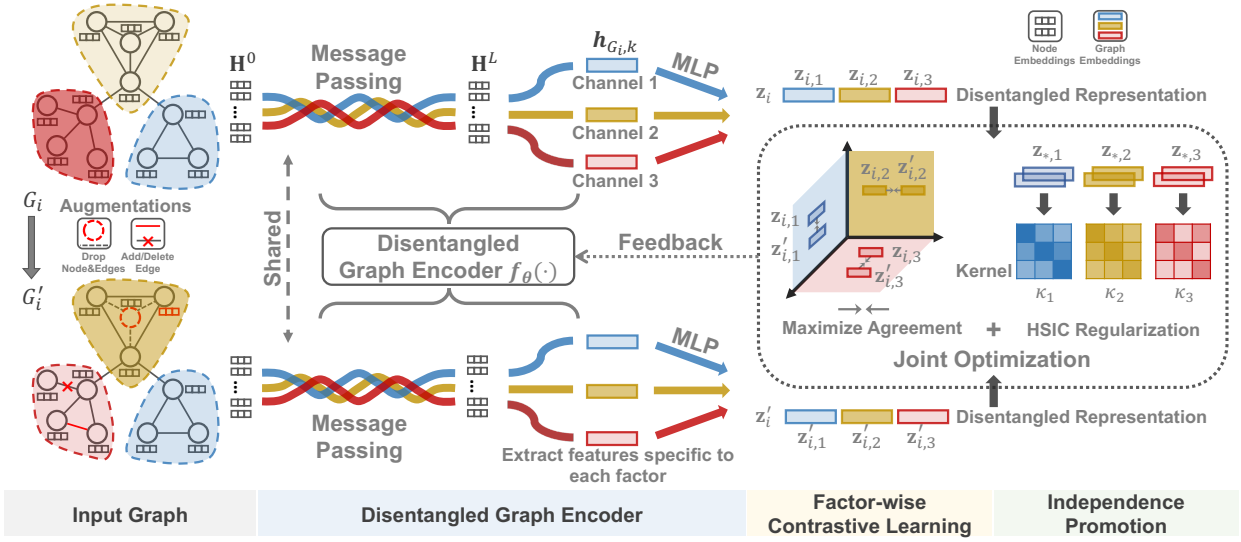


Fig. 1: The framework of our proposed **IDGCL** model. (1) The input graph G_i undergoes graph augmentations to produce G_i'. Then G_i and G_i' are fed into the shared disentangled graph encoder $f_\theta(\cdot)$. (2) In the encoder $f_\theta(\cdot)$, the node features $\mathbf{H}^0$ are first aggregated by L message-passing layers and then taken as the input of a multi-channel message-passing layer. (3) Based on the disentangled graph representation $\mathbf{z}_i$, the factor-wise contrastive learning aims to maximize the agreement under each latent factor. (4) The disentangled representations of different latent factors are encouraged to be sufficiently independent by using the HSIC regularization. The joint optimization of factor-wise contrastive learning and independence regularization provides feedback for the encoder to improve the disentanglement. The illustration assumes that there are three latent factors, corresponding to the three channels.

instances closer and dissimilar instances far from each other in representation space [20, 21]. It treats each instance in the dataset as a distinct class of its own and trains a classifier to distinguish between individual instance classes [22, 23]. Given a dataset $\mathbf{X} = \{x_i\}_{i=1}^N$, each instance x_i is assigned with a unique surrogate label y_i, since no ground-truth labels are given. y_i is often regarded as the ID of the instance in the dataset, i.e., $y_i = i$. So the probability classifier is defined as:

$$p_\theta(y_i|x_i) = \frac{\exp\, \phi(\mathrm{v}_i, \mathrm{v}'_{y_i})}{\sum_{j=1}^N \exp\, \phi(\mathrm{v}_i, \mathrm{v}'_{y_j})}, \tag{1}$$

where θ denotes the parameters of the encoder. Both v_i and v'_{y_i} are the embeddings from x_i, which are generated from two different encoders [24], or a shared encoder [25]. Before being passed into the encoder, the input x_i could undergo data augmentations [25], which play a critical role in defining effective predictive tasks for learning the encoder. ϕ is the similarity function, often adopting cosine similarity with temperature τ [26], i.e., $\phi(\mathrm{v}_i, \mathrm{v}'_{y_i}) = \mathrm{v}'^{\top}_{y_i}\mathrm{v}_i/\tau$, assuming the embeddings are ℓ^2-normalized. Then the learning objective is to maximize the joint probability $\prod_{i=1}^N p(y_i|x_i)$ over the dataset, namely minimize the negative log-likelihood function $\sum_{i=1}^N \ell_i$, if let $\ell_i = -\log\, p(y_i|x_i)$. Note that loss ℓ_i could be NCE loss [22], InfoNCE loss [27], or NT-Xent loss [25]. The encoder will be encouraged to learn a representation space where samples (e.g., augmented data) from the same instance (e.g., an image, a graph) are pulled closer and samples from different instances are pushed apart [20]. For convenience, we follow the settings above in this work.

3 Methodologies

In this section, we present the proposed **IDGCL** model. The framework of **IDGCL** is shown in Figure 1. In Section 3.1, we introduce the disentangled graph encoder to identify the complex latent factors and capture multiple aspects of graphs. Then in Section 3.2, we propose a factor-wise contrastive learning approach to conduct instance discrimination under each latent factor independently. We derive the Evidence Lower Bound in Section 3.3 and introduce the regularizer for independence promotion in Section 3.4. Finally we describe the objective, which jointly optimizes factor-wise contrastive learning and independence promotion in a unified framework in Section 3.5, followed by discussions regarding time complexity and number of parameters in Sectiion 3.6.

3.1 Disentangled Graph Encoder

The key of the disentangled graph encoder is to produce the factorized graph representation $\mathbf{z}_i = [\mathbf{z}_{i,1}, \mathbf{z}_{i,2}, \ldots, \mathbf{z}_{i,K}]$ for each input graph $G_i \in \mathbf{G}$. Based on the factorized representation, we can infer the latent factors of the graph.

Generally, GNNs use graph structure and node features to learn the representation vector $\boldsymbol{h}_v$ of node v with a message-passing mechanism, i.e., iteratively updating the node representation by aggregating representations of its neighbors. The propagation of the l^{th} layer is formulated as:

$$\boldsymbol{h}_v^l = \mathrm{COMBINE}^l(\boldsymbol{h}_v^{l-1}, \mathrm{AGGREGATE}^l(\{\boldsymbol{h}_u^{l-1} : u \in \mathcal{N}(v)\})), \tag{2}$$

where $\boldsymbol{h}_v^l$ is the representation of node v at the l^{th} layer and $\boldsymbol{h}_v^0$ is initialized with node features. $\mathcal{N}(v)$ is the neighborhood to node v. We use the term GNN to indicate the message-passing layer in Eq. (2).

Let $\mathbf{H}^l = \{\boldsymbol{h}_v^l | v \in V\}$ be the node embeddings after the l^{th} GNN, where V denotes the node set of the graph. After applying L traditional message-passing layers, we propose a graph-disentanglement layer to learn the disentangled representations. The goal is to extract features specific to each

latent factor with a separate channel. Specifically, we adopt K separate channels to identify the complex heterogeneous latent factors and capture multiple aspects of the input graph. For each channel, we first utilize a GNN_k to propagate information with its own parameters: $\mathbf{H}_k^{L+1} = \mathrm{GNN}_k(\mathbf{H}^L, A)$, where A is the adjacency matrix of the input graph. $\mathbf{H}_k^{L+1}$ is the node embeddings which is only pertinent to the k^{th} latent factor. Then the READOUT function (i.e., pooling function) of each channel is used to summarize all the obtained node representations into a fixed-length graph-level representation: $\boldsymbol{h}_{G_i,k} = \mathrm{READOUT}_k(\{\mathbf{H}_k^{L+1}\})$. Finally, each channel outputs the factorized graph representation with a separate MLP: $\mathbf{z}_{i,k} = \mathrm{MLP}_k(\boldsymbol{h}_{G_i,k})$.

Compared with the existing inherently holistic graph encoders, our disentangled graph encoder consists of K channels, rending the possibility to identify the complex heterogeneous latent factors and capture multiple aspects of graphs.

3.2 Disentangled Factor-wise Contrastive Learning

Unlike the existing contrastive learning methods, **IDGCL** designs a novel factor-wise instance discriminative task and learns to solve this task under each latent factor independently. This design not only makes similar samples closer and dissimilar samples far from each other in the representation space, but also encourages the learned representation to incorporate factor-level information for disentanglement.

Specifically, the formation of real-world graphs is usually driven by multiple latent heterogeneous factors. So the instance discriminative task should be represented as the expectation of several subtasks under the latent factors:

$$p_\theta(y_i|G_i) = \mathbb{E}_{p_\theta(k|G_i)}\left[p_\theta(y_i|G_i,k)\right]. \tag{3}$$

Here $p_\theta(k|G_i)$ is the probability distribution over latent factors for the input graph G_i. $p_\theta(y_i|G_i,k)$ denotes the instance discrimination subtask under the k^{th} latent factor.

Firstly, given the representation $\mathbf{z}_i$ of G_i derived from the disentangled graph encoder $f_\theta(\cdot)$, we present a prototype-based method to obtain $p_\theta(k|G_i)$. We introduce K latent factor prototypes $\{\mathbf{c}_k\}_{k=1}^K$, and the probability of the k^{th} latent factor reflected in G_i is parameterized as:

$$p_\theta(k|G_i) = \frac{\exp\,\phi(\mathbf{z}_{i,k}, \mathbf{c}_k)}{\sum_{k=1}^K \exp\,\phi(\mathbf{z}_{i,k}, \mathbf{c}_k)}, \tag{4}$$

where ϕ is the cosine similarity with temperature τ, i.e., $\phi(\mathbf{a},\mathbf{b}) = \textsc{Cosine}(\mathbf{a},\mathbf{b})/\tau$ and $\textsc{Cosine}(\mathbf{a},\mathbf{b}) = \mathbf{a}^\top\mathbf{b}/(\|\mathbf{a}\|_2\,\|\mathbf{b}\|_2)$.

Then, we define the instance discrimination subtask under the k^{th} latent factor as:

$$p_\theta(y_i|G_i,k) = \frac{\exp\,\phi(\mathbf{z}_{i,k}, \mathbf{z}'_{y_i,k})}{\sum_{j=1}^N \exp\,\phi(\mathbf{z}_{i,k}, \mathbf{z}'_{y_j,k})}, \tag{5}$$

where $\mathbf{z}_{i,k}$ and $\mathbf{z}'_{y_i,k}$ are the disentangled representations produced by the shared graph encoder, and y_i is the unique surrogate label (see Section 2) of the graph G_i. We follow [22, 23] to implement y_i as the ID of the graph in the dataset, i.e., $y_i = i$. For notation convenience, we do not distinguish y_i and i hereafter when there is no risk of confusion.

Next, we describe the process to get $\mathbf{z}'_{i,k}$ (i.e., $\mathbf{z}'_{y_i,k}$ in Eq. (5)). First, the input graph G_i undergoes graph data augmentations to obtain its correlated views G'_i, and they form a positive pair. Data augmentation is expected to create novel and realistically rational data by applying certain transformations that do not affect the label, and plays a critical role in defining effective predictive tasks [11, 25]. We follow [11] to adopt four types of graph augmentation strategies, including node dropping, edge perturbation, attribute masking, and subgraph sampling. More details of graph augmentations can be found in appendix. Then, the augmented graph G'_i is also fed into the shared disentangled graph encoder $f_\theta(\cdot)$ to produce $\mathbf{z}'_{i,k}$ (i.e., $\mathbf{z}'_{y_i,k}$). Given the disentangled representations $\mathbf{z}_{i,k}$ and $\mathbf{z}'_{i,k}$ of the G_i and G'_i respectively, we conduct factor-wise contrastive learning for each latent factor independently as Eq. (5).

3.3 Evidence Lower Bound (ELBO)

We present the objective of our method. Following the existing methods [22, 23], we aim to maximize the joint probability $\prod_{i=1}^N p(y_i|G_i)$ over the graph dataset $\mathbf{G} = \{G_i\}_{i=1}^N$. We learn the model parameters θ by maximizing the log-likelihood:

$$\begin{aligned}\theta^* &= \arg\max_\theta \sum_{i=1}^N \log\, p_\theta(y_i|G_i) \\ &= \arg\max_\theta \sum_{i=1}^N \log\, \mathbb{E}_{p_\theta(k|G_i)}\left[p_\theta(y_i|G_i,k)\right].\end{aligned} \tag{6}$$

However, directly maximizing the log-likelihood function is difficult because of the latent factors. Therefore, we instead optimize the evidence lower bound (ELBO) of the log-likelihood function given by Theorem 1.

Theorem 1. *The log likelihood function of each graph* $\log\, p_\theta(y_i|G_i)$ *is lower bounded by the ELBO:* $\mathcal{L}(\theta,i) = \mathbb{E}_{q_\theta(k|G_i,y_i)}[\log\, p_\theta(y_i|G_i,k)] - \mathrm{KL}(q_\theta(k|G_i,y_i) \parallel p_\theta(k|G_i))$.

Proof.

$$\begin{aligned}
&\log\, p_\theta(y_i|G_i) \\
&= \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log\, p_\theta(y_i|G_i)\right] \\
&= \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log \frac{p_\theta(y_i,k|G_i)}{p_\theta(k|G_i,y_i)}\right] \\
&= \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log \frac{p_\theta(y_i,k|G_i)}{q_\theta(k|G_i,y_i)}\frac{q_\theta(k|G_i,y_i)}{p_\theta(k|G_i,y_i)}\right] \\
&= \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log \frac{p_\theta(y_i,k|G_i)}{q_\theta(k|G_i,y_i)}\right] + \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log \frac{q_\theta(k|G_i,y_i)}{p_\theta(k|G_i,y_i)}\right] \\
&= \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log \frac{p_\theta(y_i,k|G_i)}{q_\theta(k|G_i,y_i)}\right] + \mathrm{KL}(q_\theta(k|G_i,y_i)\|p_\theta(k|G_i,y_i)) \\
&\geq \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log \frac{p_\theta(y_i,k|G_i)}{q_\theta(k|G_i,y_i)}\right] \\
&= \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log\, p_\theta(y_i|G_i,k)\frac{p_\theta(k|G_i)}{q_\theta(k|G_i,y_i)}\right] \\
&= \mathbb{E}_{q_\theta(k|G_i,y_i)}\left[\log\, p_\theta(y_i|G_i,k)\right] - \mathrm{KL}\left(q_\theta\left(k|G_i,y_i\right) \parallel p_\theta\left(k|G_i\right)\right) \\
&= \mathcal{L}(\theta,i).
\end{aligned}$$

$\mathrm{KL}(\cdot\|\cdot)$ means Kullback–Leibler divergence [28]. The equality holds when $\mathrm{KL}(q_\theta(k|G_i,y_i) \parallel p_\theta(k|G_i,y_i)) = 0$. Note that in the third-to-last line above, we have used $p_\theta(y_i,k|G_i) = p_\theta(k|G_i)p_\theta(y_i|G_i,k)$. □

To make the ELBO as tight as possible, we require that $q_\theta(k|G_i,y_i)$ is close to $p_\theta(k|G_i,y_i)$, whose detailed implementations are provided in Eq. (9) and Eq. (7). In the ELBO

$\mathcal{L}(\theta, i)$, $p_\theta(y_i|G_i, k)$ and $p_\theta(k|G_i)$ have been introduced in Eq. (5) and Eq. (4), respectively, and $q_\theta(k|G_i, y_i)$ is a variational distribution to infer the posterior distribution of the latent factors after observing both G_i and its correlated view G'_{y_i}.

We introduce a variational distribution $q_\theta(k|G_i, y_i)$ to infer the posterior probability $p_\theta(k|G_i, y_i)$ that is defined with Bayes' theorem as follows:

$$p_\theta(k|G_i, y_i) = \frac{p_\theta(k|G_i)p_\theta(y_i|G_i, k)}{\sum_{k=1}^{K} p_\theta(k|G_i)p_\theta(y_i|G_i, k)}. \tag{7}$$

$p_\theta(k|G_i, y_i)$ is the probability of the k^{th} latent factor pertinent to both G_i and the augmented G'_i simultaneously. Compared with the prior distribution $p_\theta(k|G_i)$ in Eq. (4), $p_\theta(k|G_i, y_i)$ incorporates more useful information (i.e., factor-wise similarity) from $p_\theta(y_i|G_i, k)$. Although both $p_\theta(k|G_i)$ and $p_\theta(k|G_i, y_i)$ are designed to infer the latent factor distribution, $p_\theta(k|G_i)$ is calculated only given the graph G_i, but $p_\theta(k|G_i, y_i)$ is calculated after observing G_i, the augmented version G'_{y_i}, and their similarities under the specific latent factor.

However, we cannot compute the posterior probability tractably because of the term $p_\theta(y_i|G_i, k)$. If we directly calculate $p_\theta(y_i|G_i, k)$ according to Eq. (5), all the instances in the dataset $\mathbf{G}$ are needed for computing the denominator in Eq. (5) , which could be computationally prohibitive [22, 23, 29]. To tackle this obstacle, several strategies are proposed in the literature, including memory bank [22, 24], dynamic dictionary [30], NT-Xent loss [25]. Here, we adopt NT-Xent loss on a minibatch $\mathcal{B} \subseteq \mathbf{G}$. So in practice, the instance discrimination under each latent factor is calculated by:

$$\hat{p}_\theta(y_i|G_i, k) = \frac{\exp \phi(\mathbf{z}_{i,k}, \mathbf{z}'_{i,k})}{\sum_{j\in\mathcal{B}, j\neq i}^{|\mathcal{B}|} \exp \phi(\mathbf{z}_{i,k}, \mathbf{z}'_{j,k})}. \tag{8}$$

We approximate the posterior probability $p_\theta(k|G_i, y_i)$ with a variational distribution defined as:

$$q_\theta(k|G_i, y_i) = \frac{p_\theta(k|G_i)\hat{p}_\theta(y_i|G_i, k)}{\sum_{k=1}^{K} p_\theta(k|G_i)\hat{p}_\theta(y_i|G_i, k)}. \tag{9}$$

Our method can inherently encourage disentanglement since factorizing the instance discrimination into K factor-wise subtasks will enforce the independence of the learned graph representation $\mathbf{z}_i$. Besides, $q_\theta(k|G_i, y_i)$ is computed based on k^{th} and other $K-1$ latent factors. Thus, the graph encoder is forced to preserve exclusive information in each channel to get more accurate approximation to the posterior, if a tighter ELBO is expected. The strong inductive biases in our proposed method encourage to learn disentangled graph representations that match the ground truth factors behind the graphs.

3.4 Statistical Independence of Representations

Enhancing the independence of the disentangled graph representations explicitly will encourage the graph encoder to better capture predictive and mutually independent information in terms of different latent factors, which can lead to improved performance for downstream tasks. Next we elaborate on the details of independence regularization.

Recall that the goal of our method is to empower the graph encoder to produce the disentangled graph representations $\mathbf{z}_i = [\mathbf{z}_{i,1}, \mathbf{z}_{i,2}, \ldots, \mathbf{z}_{i,K}]$ for each input graph $G_i \in \mathbf{G}$ that the K channels capture mutually exclusive information in terms of the latent factors. This means the statistical independence among the disentangled graph representations should be further promoted to enhance the disentanglement. For measuring the independence among the disentangled graph representations, it is infeasible to resort to histogram-based measures unless the dimensionality of representations is small enough [31]. Therefore, we introduce Hilbert-Schmidt Independence Criterion (HSIC) [16] for promoting the representations of different factors to be sufficiently independent.

Specifically, let $\mathbf{z}_{*,k}$ be the Δd-dimensional ($\Delta d = d/K$) random variable denoting the disentangled representations corresponding to the latent factor k. Consider a measurable, positive definite kernel κ_k on the domain of random variable $\mathbf{z}_{*,k}$ and denote the corresponding Reproducing Kernel Hilbert Spaces (RKHS) by $\mathcal{H}_k$. $\phi_k(\cdot)$ is the transformation function mapping $\mathbf{z}_{*,k}$ into $\mathcal{H}_k$ with respect to the kernel κ_k.

For a pair of latent factors $k_A, k_B \in [1, K]$, $k_A \neq k_B$, $\mathbf{z}_{*,k_A}$ and $\mathbf{z}_{*,k_B}$ are jointly drawn from a distribution $p(\mathbf{z}_{*,k_A}, \mathbf{z}_{*,k_B})$. $\mathcal{C}_{\mathbf{z}_{*,k_A},\mathbf{z}_{*,k_B}}$, the cross-covariance operator in the RKHS of κ_{k_A} and κ_{k_B}, is defined as follows:

$$\mathcal{C}_{\mathbf{z}_{*,k_A},\mathbf{z}_{*,k_B}} = \mathbb{E}_{p(\mathbf{z}_{*,k_A},\mathbf{z}_{*,k_B})}\Big[(\phi_{k_A}(\mathbf{z}_{*,k_A}) - \mu_{\mathbf{z}_{*,k_A}})^\top \cdot (\phi_{k_B}(\mathbf{z}_{*,k_B}) - \mu_{\mathbf{z}_{*,k_B}})\Big], \tag{10}$$

where $\mu_{\mathbf{z}_{*,k_A}} = \mathbb{E}_{p(\mathbf{z}_{*,k_A})}[\phi_{k_A}(\mathbf{z}_{*,k_A})]$ and $\mu_{\mathbf{z}_{*,k_B}} = \mathbb{E}_{p(\mathbf{z}_{*,k_B})}[\phi_{k_B}(\mathbf{z}_{*,k_B})]$. HSIC, the Hilbert-Schmidt norm of the associated cross-covariance operator, is defined as:

$$\text{HSIC}(\mathbf{z}_{*,k_A}, \mathbf{z}_{*,k_B}) := \|\mathcal{C}_{\mathbf{z}_{*,k_A},\mathbf{z}_{*,k_B}}\|^2_{\text{HS}}, \tag{11}$$

where $\|\mathbf{M}\|^2_{\text{HS}} = \sum_{i,j} \mathbf{M}^2_{i,j}$. The independence can be determined by the following proposition.

Proposition 1. *Assume* $\mathbb{E}[\kappa_{\mathbf{z}_{*,k_A}}(\mathbf{z}_{*,k_A}, \mathbf{z}_{*,k_A})] < \infty$ *and* $\mathbb{E}[\kappa_{\mathbf{z}_{*,k_B}}(\mathbf{z}_{*,k_B}, \mathbf{z}_{*,k_B})] < \infty$, *and* $\kappa_{\mathbf{z}_{*,k_A}}$ $\kappa_{\mathbf{z}_{*,k_B}}$ *is a characteristic kernel, then*

$$\text{HSIC}(\mathbf{z}_{*,k_A}, \mathbf{z}_{*,k_B}) = 0 \Leftrightarrow \mathbf{z}_{*,k_A} \perp\!\!\!\perp \mathbf{z}_{*,k_B}. \tag{12}$$

In practice, we employ an unbiased estimator of HSIC following [32]:

$$\text{HSIC}(\mathbf{z}_{*,k_A}, \mathbf{z}_{*,k_B}) = \frac{1}{m(m-3)}\left[\text{tr}\left(\widetilde{U}\widetilde{V}^T\right) + \frac{\mathbf{1}^T\widetilde{U}\mathbf{1}\mathbf{1}^T\widetilde{V}^T\mathbf{1}}{(m-1)(m-2)} - \frac{2}{m-2}\mathbf{1}^T\widetilde{U}\widetilde{V}^T\mathbf{1}\right] \tag{13}$$

where $\widetilde{U}$, $\widetilde{V}$ are the Gram matrices with $\kappa_{\mathbf{z}_{*,k_A}}$, $\kappa_{\mathbf{z}_{*,k_B}}$, whose diagonal entries are set to zero. We use the radial basis function (RBF) kernel in our implementation.

The main advantages of adopting the above criterion to measure the dependence of graph representations are three-fold [16, 33]. (1) Since graph representations are mapped into the RKHS to measure the dependence, the correlations measured in that space correspond to high-order joint moments between the original distributions and more complex nonlinear dependence can be addressed. (2) HSIC is effective yet efficient to estimate the dependence of graph representations since it avoids to estimate the joint distribution of the random variables explicitly. (3) The

estimator we adopted for HSIC is unbiased [32], as opposed to other empirical estimates [16].

3.5 Optimization

Finally, we seek to learn the parameters θ of the disentangled graph encoder under the unified framework of factor-wise contrastive learning and independence promotion. We optimize the following objective function by combining the ELBO and HSIC regularizer using mini-batch gradient descent:

$$\min_{\theta} -\mathcal{L}(\theta, \mathcal{B}) + \lambda \mathcal{L}_{reg} \tag{14}$$

where $\lambda \geq 0$ is a hyper-parameter that controls the impact of the regularizer. More specifically, the ELBO $\mathcal{L}(\theta, \mathcal{B})$ over a mini-batch $\mathcal{B}$ is calculated by:

$$\mathcal{L}(\theta, \mathcal{B}) = \sum_{i \in \mathcal{B}} \mathcal{L}(\theta, i), \tag{15}$$

where $\mathcal{L}(\theta, i)$ is defined in Theorem 1. The HSIC regularizer $\mathcal{L}_{reg}$ for promoting the statistical independence among the disentangled graph representations is calculated by:

$$\mathcal{L}_{reg} = \sum_{1 \leq k_A < k_B \leq K} \text{HSIC}(\mathbf{z}_{*,k_A}, \mathbf{z}_{*,k_B}). \tag{16}$$

Note that the HSIC regularizer can be easily calculated since it is only based on the output disentangled graph representation (i.e., the parameters θ of the disentangled graph encoder) without other assumptions.

We use **IDGCL** (Independence promoted Disentangled Graph Contrastive Learning) to refer to our proposed model and its detailed training procedure is shown in Algorithm 1.

3.6 Discussions

3.6.1 Time Complexity Analysis

The time complexity of our proposed **IDGCL** is $O(L\left|E\right|d + L\left|V\right|d^2)$, where $|V|$, $|E|$ denotes the total number of nodes and edges in the graphs, d is the dimensionality of the representation, and L is the number of message-passing layers of disentangled graph encoder. Specifically, we instantiate our method by adopting GIN [4] as the message-passing layers so the time complexity of each layer in the disentangled graph encoder is $O(\left|E\right|d + \left|V\right|d^2)$. As for the factor-wise contrastive learning, the positive and negative samples are drawn from graph data augmentations and graphs from the same minibatch respectively, so the time complexity is $O(|\mathcal{B}|^2 d)$, where $|\mathcal{B}|$ is the batch size. The calculation of HSIC regularizer in **IDGCL** has a time complexity of $O(|\mathcal{B}|^2 d^2)$. Notice that $|\mathcal{B}|$ and d are small constants that are unrelated to the dataset size.

In comparison, the time complexity of other representative self-supervised graph learning methods (e.g., GraphCL and MVGRL, see the Experiment Section for details) is also $O(L\left|E\right|d + L\left|V\right|d^2)$. Therefore, the time complexity of our **IDGCL** is on par with these baselines. Some unsupervised baseline, such as GVAE, has a $O(L\left|E\right|d + L\left|V\right|d^2 + \left|V\right|^2 d)$ time complexity due to the reconstruction of the adjacency matrix in the VAE framework. Since $O(|E|) \ll O(|V|^2)$ for sparse graphs, and L and d are small constants, our proposed method is much more scalable than GVAE.

Algorithm 1 The training procedure of **IDGCL**.

Input: A graph dataset $\mathbf{G} = \{G_i\}_{i=1}^N$
Output: The disentangled representations $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^N$

function DISENTANGLEDENCODER(G_i)
  for $l \leftarrow 1$ to L **do**
    $\mathbf{H}^l = \text{GNN}^l(\mathbf{H}^{l-1}, A)$
  end for
  for $k \leftarrow 1$ to K **do** ▷ separate K channels
    $\mathbf{H}_k^{L+1} = \text{GNN}_k(\mathbf{H}^L, A)$
    $\mathbf{z}_{i,k} = \text{MLP}_k(\text{READOUT}_k(\{\mathbf{H}_k^{L+1}\}))$
  end for
  return $\mathbf{z}_i$ ▷ disentangled graph representation
end function
for sampled minibatch $\mathcal{B} = \{G_i\}_{i=1}^{|\mathcal{B}|}$ **do**
  for $G_i \in \mathcal{B}$ **do** ▷ disentangled graph encoding
    $\mathbf{z}_i$ = DISENTANGLEDENCODER(G_i)
    $G_i' = \text{GRAPHAUGMENTATION}(G_i)$
    $\mathbf{z}_i'$ = DISENTANGLEDENCODER(G_i')
    Calculate $p_\theta(k|G_i)$ by Eq. (4)
  end for
  for $k \leftarrow 1$ to K **do** ▷ factor-wise contrastive learning
    for $i \leftarrow 1$ to $|\mathcal{B}|$ and $j \leftarrow 1$ to $|\mathcal{B}|$ **do**
      $s_{i,j}^{(k)} = \phi(\mathbf{z}_{i,k}, \mathbf{z}'_{j,k})$ ▷ similarity of k^{th} factor
    end for
    Calculate $\hat{p}_\theta(y_i|G_i, k)$ by Eq. (8)
  end for
  for $G_i \in \mathcal{B}$ **do** ▷ optimization objective
    Calculate $q_\theta(k|G_i, y_i)$ by Eq. (9)
    Calculate ELBO $\mathcal{L}(\theta, i)$
  end for
  Calculate ELBO over a minibatch $\mathcal{L}(\theta, \mathcal{B})$ by Eq. (15) and the independence regularizer $\mathcal{L}_{reg}$ by Eq. (16)
  Update θ to minimize $-\mathcal{L}(\theta, \mathcal{B}) + \lambda \mathcal{L}_{reg}$ by Eq. (14)
end for
$\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^N$, $\mathbf{z}_i$ = DISENTANGLEDENCODER(G_i), $G_i \in \mathbf{G}$

3.6.2 Number of Parameters Analysis

For our proposed **IDGCL**, the number of parameters is $O(Ld^2)$, where L is the number of message-passing layers of disentangled graph encoder, and d is the dimensionality of the representation. Specifically, because we adopt GIN as the message passing layers, the number of parameters of the disentangled graph encoder is $O(Ld^2)$. The number of parameters of K latent factor prototypes is $O(K * (d/K)) = O(d)$. The independence regularization does not involve extra learnable parameter. For the baselines that use GNNs (e.g., GCN or GIN) as the graph encoder, including GVAE, InfoGraph, GCC, MVGRL, and GraphCL, the number of parameters is $O(Ld^2)$. Therefore, the number of parameters of the proposed method and the baselines are comparable.

4 EXPERIMENTS

In this section, to demonstrate the effectiveness of our proposed **IDGCL** model, we empirically compare **IDGCL** with state-of-the-art (SOTA) methods for graph classification tasks. Concretely, in Section 4.1, we report the results of unsupervised learning settings. In Section 4.2, we conduct

TABLE 1: The statistics of the datasets used in the unsupervised learning setting. #Graphs is the number of graphs in the dataset. Avg #nodes/#edges are the average number of nodes and edges in a graph of the dataset, respectively.

	MUTAG	PTC-MR	PROTEINS	NCI1	IMDB-B	IMDB-M	RDT-B	RDT-M5K	COLLAB
#Graphs	188	344	1,113	4,110	1,000	1,500	2,000	4,999	5,000
#Classes	2	2	2	2	2	3	2	5	3
Avg #nodes	17.9	14.3	39.1	29.9	19.8	13.0	429.6	508.5	74.5
Avg #edges	19.8	14.7	72.8	32.3	96.5	65.9	497.8	594.9	2457.8

TABLE 2: Graph classification accuracy (%) of our proposed method and baselines in the unsupervised learning setting. In each column, the boldfaced score denotes the best result of all the methods and the underlined score represents the best result of baselines. "–" indicates the result is not reported in the paper.

	MUTAG	PTC-MR	PROTEINS	NCI1	IMDB-B	IMDB-M	RDT-B	RDT-M5K	COLLAB
SP	85.2±2.4	58.2±2.4	75.1±0.5	73.0±0.2	55.6±0.2	38.0±0.3	64.1±0.1	39.6±0.2	–
GK	81.7±2.1	57.3±1.4	71.7±0.6	62.3±0.3	65.9±1.0	43.9±0.4	77.3±0.2	41.0±0.2	72.8±0.3
WL	80.7±3.0	58.0±0.5	72.9±0.6	80.0±0.5	72.3±3.4	47.0±0.5	68.8±0.4	46.1±0.2	–
DGK	87.4±2.7	60.1±2.6	73.3±0.8	80.3±0.5	67.0±0.6	44.6±0.5	78.0±0.4	41.3±0.2	73.1±0.3
MLG	87.9±1.6	63.3±1.5	76.1±2.0	80.8±1.3	66.6±0.3	41.2±0.0	–	–	–
node2vec	72.6±10.2	58.6±8.0	57.5±3.6	54.9±1.6	–	–	–	–	–
sub2vec	61.1±15.8	60.0±6.4	53.0±5.6	52.8±1.5	55.3±1.5	36.7±0.8	71.5±0.4	36.7±0.4	–
graph2vec	83.2±9.3	60.2±6.9	73.3±2.1	73.2±1.8	71.1±0.5	50.4±0.9	75.8±1.0	47.9±0.3	–
GVAE	87.7±0.7	61.2±1.8	–	–	70.7±0.7	49.3±0.4	87.1±0.1	52.8±0.2	–
InfoGraph	89.0±1.1	61.7±1.4	74.4±0.3	76.2±1.1	73.0±0.9	49.7±0.5	82.5±1.4	53.5±1.0	70.7±1.1
GCC	–	–	–	–	72.0	49.4	89.8	53.7	78.9
MVGRL	89.7±1.1	62.5±1.7	–	–	74.2±0.7	51.2±0.5	84.5±0.6	–	–
GraphCL	86.8±1.3	63.6±1.8	74.4±0.5	77.9±0.4	71.1±0.4	50.7±0.4	89.5±0.8	56.0±0.3	71.4±1.2
JOAO	87.7±0.8	61.1±1.7	74.1±1.1	78.4±0.5	70.8±0.3	51.0±0.5	86.4±1.5	56.0±0.3	69.3±0.3
DGCL	92.1±0.8	65.8±1.5	76.4±0.5	81.9±0.2	75.9±0.7	51.9±0.4	91.8±0.2	56.1±0.2	81.2±0.3
IDGCL	**92.5±0.6**	**66.2±1.3**	**77.1±0.2**	**82.4±0.3**	**76.1±0.2**	**52.3±0.4**	**91.9±0.3**	**56.3±0.2**	**81.3±0.3**

experiments on semi-supervised settings. Finally, we report several analyses including ablation studies in Section 4.3.

4.1 Unsupervised Learning

We first evaluate our model in the unsupervised representation learning following the common evaluation protocols in the existing literature [5, 11, 34, 35], where graph representations are trained in an unsupervised setting and then fed into a downstream SVM classifier.

4.1.1 Experimental Setup

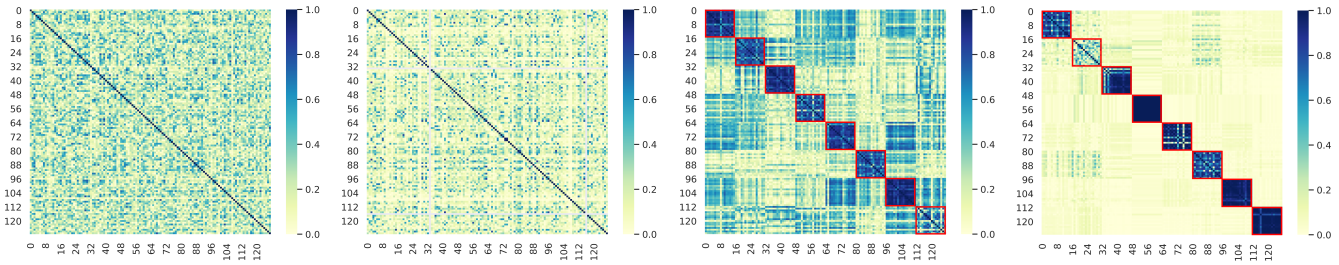
To demonstrate the advantages of our method, we conduct experiments on nine well-known graph classification datasets including five bioinformatics datasets, i.e., MUTAG, PTC-MR, PROTEINS, NCI1 and four social network datasets, i.e., IMDB-BINARY, IMDB-MULTI, REDDIT-BINARY, REDDIT-MULTI-5K, and COLLAB. The statistics of the datasets are summarized in Table 1.

We compare our proposed method with the following two groups of baselines. One group of baselines are graph kernels including Shortest Path Kernel (SP) [36], Graphlet Kernel (GK) [37], Weisfeiler-Lehman Sub-tree Kernel (WL) [38], Deep Graph Kernels (DGK) [35], and Multi-Scale Laplacian Kernel (MLG) [39]. The other group of baselines are classical unsupervised graph representation learning methods including node2vec [40], sub2vec [41], graph2vec [34], GVAE [42], and more recent contrastive graph representation learning methods including InfoGraph [5], GCC [10], MVGRL [9], GraphCL [11], JOAO [43]. For our proposed method, we consider two versions: the full **IDGCL** model and DGCL, a variant of our model when $\lambda = 0$ [17].

We implement our models in PyTorch and use Stochastic Gradient Descent (SGD) for the optimization. We use GIN [4] as the message-passing layers since it is shown to be one of the most expressive message-passing GNNs. The number of message-passing layers is chosen from $\{2, 3, 4, 5\}$. The dimensionality of the representations d is chosen from $\{64, 128, 256, 512\}$. Note that the ground-truth number of the latent factors is unknown, so we search the number of channels K from 1 to 10. For our **IDGCL**, the hyper-parameter λ controlling the impact of the HSIC regularizer is chosen from $\{0.0001, 0.001, 0.01, 0.1\}$. For a fair comparison, the hyper-parameters of the graph augmentations are kept consistent with GraphCL [11]. For the unsupervised setting, we use SVM as the downstream classifier. We adopt the 10-fold cross validation accuracy, and report the mean accuracy (%) with standard variation after five repeated runs.

4.1.2 Results on Real Benchmark Graphs

The results are reported in Table 2. We can see that the graph contrastive learning methods generally outperform the graph kernel methods or the classical unsupervised methods, which verify the effectiveness of contrastive learning. Our disentangled method **IDGCL** consistently achieves the best performances compared with other contrastive baselines (e.g., MVGRL, GrpahCL) and classical unsupervised baselines (e.g., graph2vec, GVAE), demonstrating the superiority of disentanglement. For example, **IDGCL** increases the classification accuracy by 2.6% and 1.6% against the strongest baselines on PTC-MR and NCI1 respectively. And **IDGCL** improves upon DGCL by a margin of 0.4% and 0.5% on PTC-MR and NCI1, suggesting that encouraging independence between latent factors is beneficial to learn informative graph



(a) MVGRL (b) GraphCL (c) DGCL (d) **IDGCL**

Fig. 2: An analysis of feature correlation on the synthetic graphs with eight latent factors. The figures show the absolute value of the correlations between the elements of the representations learned by MVGRL, GraphCL, DGCL, and **IDGCL** with eight channels, respectively. We can see that the representations generated from **IDGCL** present a more block-wise correlation pattern, indicating that the eight channels of the disentangled graph encoder in **IDGCL** are able to capture mutually exclusive information and the latent features have indeed been disentangled.

representations. We attribute the results to the fact that these existing methods fail to identify the underlying latent factors, which are important in preserving graph properties, and therefore cannot learn the disentangled representations. In contrast, we disentangle graph representations to explicitly consider the entanglement of heterogeneous factors. When compared to graph kernel methods, our method also has the best accuracy on all the datasets. Notice that none of these kernel methods is consistently competitive across all of the datasets, as opposed to our method.

4.1.3 Results on Synthetic Graphs

Since the formation processes of real-world graphs are usually unobserved to us, it is difficult to obtain the semantic information of latent factors, which is the common case in the graph disentanglement literature [12, 19]. To further investigate the behavior of our method, we generate a synthetic dataset consisting of 1,000 graphs with known latent factors. Specifically, we generate synthetic graphs using the stochastic block model [44]. Each graph contains four communities and each community consists of 10 nodes. We define the latent factor as the probability p that two nodes are connected in a community. p can take value from $\{0.2, 0.3, \ldots, 0.9\}$, meaning that there are eight latent factors in the dataset in total. The probability for each community is drawn from the eight possible choices without replacement. Two nodes in different communities are connected with probability 0.05. The rows of the adjacency matrices are used as node features, and the ground-truth communities are used as labels, i.e., there are 8 classes and each graph has 4 labels. We train **IDGCL**, DGCL and two baselines, i.e., MVGRL and GraphCL on the synthetic dataset with self-supervision. Then we adopt the SVM classifier on the learned graph representations and use the Micro-F1 (%) as the evaluation metric.

We vary the number of channels K of our method and report the results in Figure 3. Our methods **IDGCL** and DGCL report better performance than the baselines. We also find that as K increases from 1 to 8, the result of our methods improves, which verifies the importance of disentangling latent factors. After reaching the peak at $K = 8$, the performance slightly drops, but in general, our method is not very sensitive when K is not too large. When

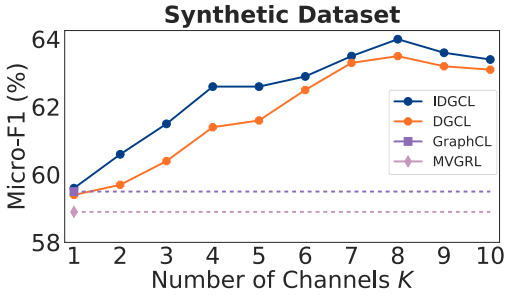


Fig. 3: Micro-F1 (%) of our proposed method and two baselines with different number of channels K.

K is equal to the ground-truth number of latent factors, our methods achieve the best results and **IDGCL** further improves on DGCL, indicating that our method can capture the underlying structure of this simulation dataset with the disentangled representations.

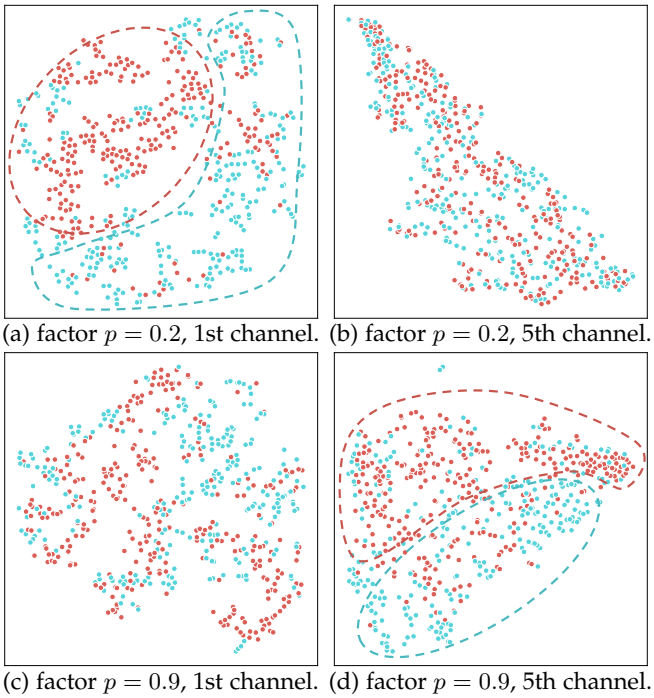
Besides the quantitative evaluation, we also provide a qualitative evaluation by plotting the correlation of the latent features in Figure 2. The figure shows the absolute values of the correlation between the elements of 128-dimensional graph representation obtained from MVGRL, GraphCL, our DGCL and **IDGCL** ($K = 8$) on the synthetic dataset. We can see from the results that the graph representations of MVGRL and GraphCL are entangled. In comparison, the correlation of **IDGCL** shows eight diagonal blocks, meaning that the channels of **IDGCL** likely extract mutually exclusive information and output disentangled representations. From Figure 2 (c)(d), we can observe that **IDGCL** can further reduce the correlation of different blocks compared to DGCL. The results indicate that optimizing our statistical independence metric can further promote the output representations of different factors to be sufficiently independent for disentanglement. To better show the effectiveness of the proposed method, we further consider the following evaluation way. Specifically, we first cluster the 128 dimensions by the k-means algorithm [45] (with 8 clusters) using the 128 row vectors of the feature correlation matrix, so that the cluster id of each dimension could indicate the block id. Then, we calculate the ratio of the average feature correlation inside the block structure divided by the average feature correlation

TABLE 3: The statistics of the OGB datasets in the semi-supervised setting. The datasets contain multiple binary classification tasks and #Tasks denotes the number of tasks (i.e., the dimensionality of output).

	BACE	BBBP	TOX21	CLINTOX	SIDER	TOXCAST	MUV	HIV
#Graphs	1,513	2,039	7,831	1,477	1,427	8,576	93,087	41,127
#Tasks	1	1	12	2	27	12	17	1
Avg #nodes	34.1	24.1	18.6	26.2	33.6	18.8	24.2	25.5
Avg #edges	36.9	26.0	19.3	27.9	35.4	19.3	26.3	27.5
Metric	ROC-AUC	ROC-AUC	ROC-AUC	ROC-AUC	ROC-AUC	ROC-AUC	AP	ROC-AUC

outside the block structure. Intuitively, a larger ratio denotes better disentanglement quantitatively. Finally, the ratio is 2.20, 2.45, 2.87, and 22.49 for MVGRL, GraphCL, DGCL, and **IDGCL**, respectively. The results further demonstrate that the proposed **IDGCL** can output disentangled graph representations compared with other methods.

To further explore the properties of the disentangled graph representations, we plot the graph distribution based on representations extracted from different channels with t-SNE [46] in Figure 4. For simplicity, we only take two latent factors and two channels as the example, while the others show similar patterns. Specifically, we plot the graph distribution on a 2D plane based on the graphs with (denoted by cyan points) and without (denoted by red points) the factor $p = 0.2$ or $p = 0.9$ by the representations extracted from the channel that is the only one among all channels whose output representations are discriminative to the factor, i.e., the 1st channel for factor $p = 0.2$ and the 5th channel for factor $p = 0.9$ in our experiments. Figure 4 shows that different channels capture different latent factors of the graphs. In Figure 4 (a)(b), we can observe whether the latent factor $p = 0.2$ is included in the graph can be discriminated obviously with the representations extracted only from the 1st channel (as shown in (a)), while they are mixed with representations from the 5th channel (as shown in (b)). However, whether the latent factor $p = 0.9$ is included in the graph can be discriminated with the representations extracted only from the 5th channel while they are hard to be separated with representations from the 1st channel, which is the opposite to the case of the latent factor $p = 0.2$. Therefore, although it is difficult to figure out the specific meaning that each channel represents without enough supervision, learning disentangled graph representations by our method makes it possible to explore the meaning of each channel.



(a) factor $p = 0.2$, 1st channel. (b) factor $p = 0.2$, 5th channel.
(c) factor $p = 0.9$, 1st channel. (d) factor $p = 0.9$, 5th channel.

Fig. 4: Visualization of the learned representations from two channels (i.e., the 1st channel and 5th channel) in terms of the two latent factors (i.e., factor $p = 0.2$ and factor $p = 0.9$). We can observe that different channels of the disentangled graph encoder capture different latent factors of the graphs.

4.2 Semi-supervised Learning

4.2.1 Experimental Setup

Furthermore, we evaluate our proposed method in the semi-supervised learning. We first perform pre-training with all training data without labels. Then we conduct fine-tuning on the partially labeled training data and evaluation on the validation/test sets.

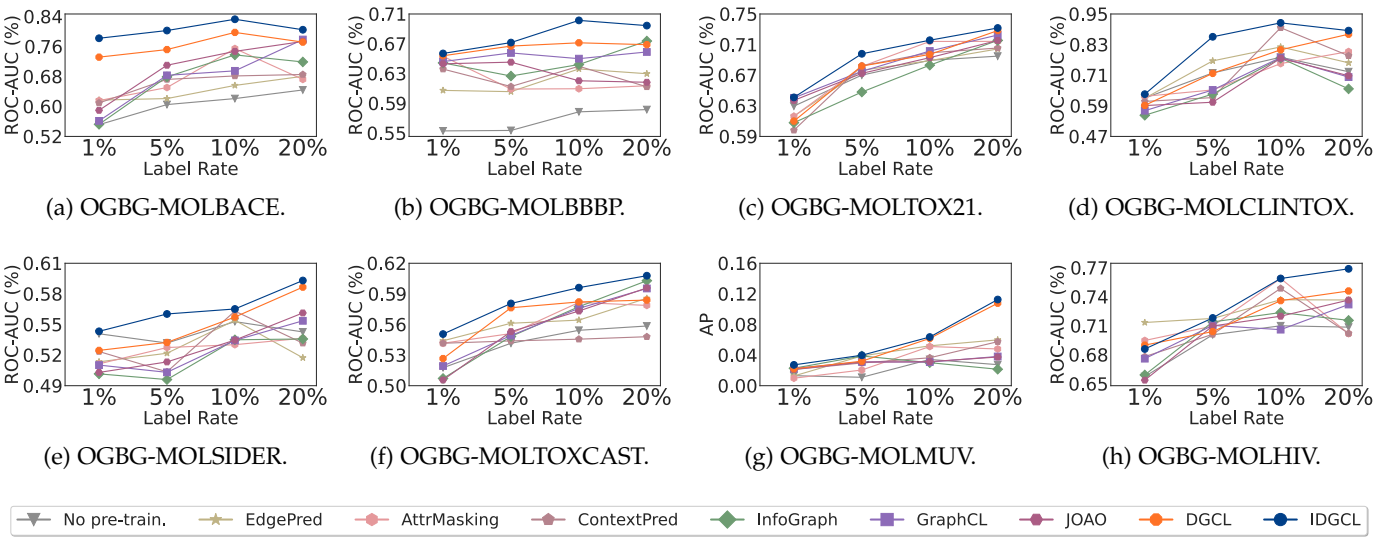
We consider 8 graph property prediction datasets from **OGBG-MOL**$*$ in Open Graph Benchmark (OGB) [47], i.e., BACE, BBBP, TOX21, CLINTOX, SIDER, TOXCAST, MUV, and HIV. The task is to predict the target molecular properties as accurately as possible. We adopt the evaluator and dataset splits provided by OGB for a fair comparison.

We compare our proposed method with the following baselines. The naive baseline is training from scratch without the pre-training stage, which we denote as "No pre-train". Besides, we consider three self-supervised methods: edge-based reconstruction method EdgePred, vertex feature masking & recovering method AttrMasking, and sub-structure information preserving method ContextPred [48]. They are designed based on certain domain knowledge, which work well when such knowledge is available and benefits downstream tasks [43, 48]. We also consider two representative self-supervised baselines InfoGraph [5] and GraphCL [11].

For our proposed method and baselines, we first perform pre-training and then conduct fine-tuning with 1%, 5%, 10%, or 20% labeled training data. The number of message-passing layers is set to 5 and the dimensionality of the representations is set to 300, following [47]. Others hyper-parameters are kept consistent with those in the unsupervised learning setting.

4.2.2 Results

The results are shown in Figure 5. We can find that **IDGCL** outperforms the baselines in most comparisons, which verifies that learning disentangled graph representation with independence promotion can benefit encoding useful



(a) OGBG-MOLBACE. (b) OGBG-MOLBBBP. (c) OGBG-MOLTOX21. (d) OGBG-MOLCLINTOX.

(e) OGBG-MOLSIDER. (f) OGBG-MOLTOXCAST. (g) OGBG-MOLMUV. (h) OGBG-MOLHIV.

Fig. 5: Graph classification results of our proposed method and baselines in the semi-supervised learning setting.

information of graphs into representations. For example, **IDGCL** increases the AUC score by 7.8% against the strongest baseline (i.e., AttrMasking) on OGBG-MOLBACE (10% label rate). We also observe that **IDGCL** improves upon DGCL by a margin of 3.5% on the same dataset. Besides, the improvement of **IDGCL** compared with DGCL is 5.1%, 14.3%, and 2.8% on BACE, CLINTOX, and SIDER, respectively, when the label rate is 5%. The results on OGBG-MOLMUV and OGBG-MOLHIV illustrate that our method is also able to handle large-scale graphs, demonstrating the benefit of learning disentangled graph representations in the contrastive manner.

4.3 Analysis

4.3.1 Ablation Studies

We perform ablation studies over the key components of our method to understand their functionalities more deeply. We compare **IDGCL** with the following three variants: (1) DGCL, i.e., setting $\lambda = 0$. (2) Variant 1: it sets $p_\theta(k|G_i) = 1/K$, i.e. a uniform distribution of latent factors. (3) Variant 2: it sets $K = 1$ directly, so that our method will degenerate to the entangled graph contrastive learning model. For simplicity, we only report the results on the datasets in the unsupervised setting, while the results in the semi-supervised setting show similar patterns.

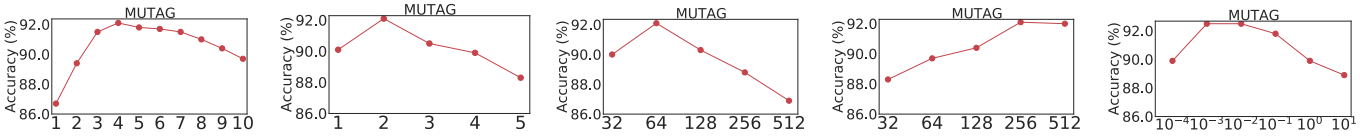
The results of **IDGCL**, DGCL and the variants are shown in Table 4. We can find that when setting λ to 0, the performance is consistently degraded on all datasets, i.e., **IDGCL** consistently outperforms DGCL. It means that although DGCL considered disentanglement of different latent factors implicitly by conducting factor-wise contrastive learning, it neglects to explicitly encourage the independence of the latent factors, which could lead to suboptimal performance for disentanglement. In contrast, IDGCL, adopting the HSIC to regularize the latent representations, can enforce them to be independent to produce more informative representations than DGCL, which is validated by the results in both unsupervised and semi-supervised settings. Besides, from Table 4, we observe a drop in performance of Variant 1, demonstrating the efficacy of inferring the latent factors of the graphs. In variant 2, the latent factors are entangled in the graph representation, making difficulties for characterizing different aspects of the graphs and conducting discrimination tasks in terms of each latent factor independently. The deterioration of performance verifies the significance of the proposed factor-wise contrastive learning.

4.3.2 Hyper-parameter Sensitivity

We investigate the sensitivity of hyper-parameters of our method: the number of channels K, the number of message-passing layers L, the dimensionality of the representation d, the batch size B, and the regularization coefficient λ. Among them, the number of channels K is the most important hyper-parameter. For simplicity, we only report the results on the MUTAG (Figure 6), IMDB-B (Figure 7) datasets for the unsupervised learning setting and the results on OGBG-MOLBACE (20% label rate, Figure 8) for the semi-supervised learning setting, while the results on other datasets show similar patterns. From Figures 6, 7, and 8, we can observe the following results. The performance increases at first with a larger K and drops after reaching a peak, showing that a proper number of channels K which matches the real latent factors behind the observed data can lead to better results. Then, the number of message-passing layers L is also important because our model with a small L has a limited model capacity and may not be able to fuse enough information from neighbors, and a very large L could also lead to the over-smoothing problem [49]. In addition, the optimal dimensionality of representation d for MUTAG is relatively smaller than that for IMDB-B and OGBG-MOLBACE, since MUTAG only consists of 188 graphs but IMDB-B and OGBG-MOLBACE contain 1,000 and 1,513 graphs with more nodes and edges, respectively. A too large d may induce over-fitting and hurt the performance. Our method benefits from larger batch sizes, which is consistent with the literature on contrastive learning [25]. Finally, λ also influences the performance. A large λ overemphasizes the independence between latent factors and a too small λ limits the impact of the independence regularization. We

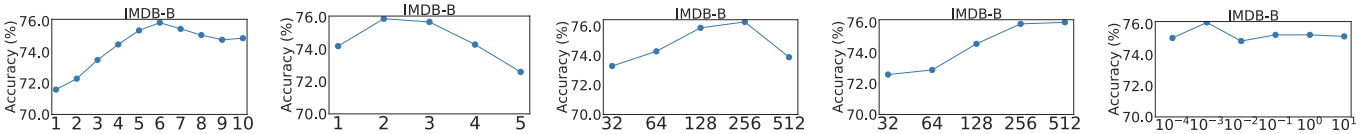
TABLE 4: Ablation studies on variants of our method. We report the accuracy (%) with standard variation on the datasets. The performance of Variant 1 and 2 are greatly degraded compared with **IDGCL**, demonstrating the significance to infer latent factors behind the graphs and conduct factor-wise contrastive learning.

	MUTAG	PTC-MR	PROTEINS	NCI1	IMDB-B	IMDB-M	RDT-B	RDT-M5K	COLLAB
IDGCL	**92.5±0.6**	**66.2±1.3**	**77.1±0.2**	**82.4±0.3**	**76.1±0.2**	**52.3±0.4**	**91.9±0.3**	**56.3±0.2**	**81.3±0.3**
DGCL	92.1±0.8	65.8±1.5	76.4±0.5	81.9±0.2	75.9±0.7	51.9±0.4	91.8±0.2	56.1±0.2	81.2±0.3
Variant 1	89.3±0.3	64.3±1.3	74.9±0.2	78.5±0.5	73.4±0.5	50.3±0.2	91.1±0.7	55.9±0.3	77.5±0.4
Variant 2	86.5±0.6	63.5±1.6	73.9±0.6	77.7±0.6	70.9±0.5	49.8±0.3	89.7±0.6	55.7±0.2	71.5±0.2



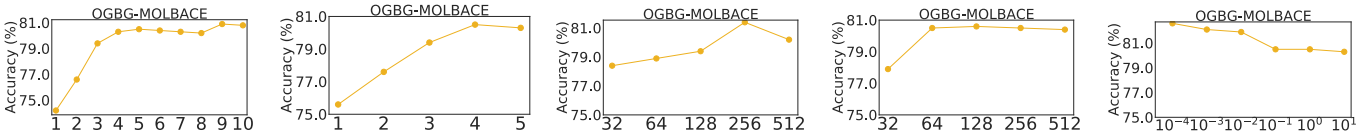
(a) Channel number K. (b) Number of layers L. (c) Dimensionality d. (d) Batch size B. (e) Regularization λ.

Fig. 6: Impact of different hyper-parameters on the MUTAG dataset.



(a) Channel number K. (b) Number of layers L. (c) Dimensionality d. (d) Batch size B. (e) Regularization λ.

Fig. 7: Impact of different hyper-parameters on the IMDB-B dataset.



(a) Channel number K. (b) Number of layers L. (c) Dimensionality d. (d) Batch size B. (e) Regularization λ.

Fig. 8: Impact of different hyper-parameters on the OGBG-MOLBACE dataset.

empirically find that setting λ to 10^{-3} achieves satisfactory results for most datasets.

5 Related Work

Graph Neural Networks. Graph structured data is ubiquitous in the real world [47, 50, 51]. Recently, graph neural networks (GNNs) [2–4] have revolutionized the field of graph representation learning [52]. GNNs show promising performance on various tasks, including node classification [2], link prediction [53], and graph classification [4]), and demonstrate profound successes in challenging applications, such as drug discovery [54], protein function prediction [55], traffic forecasting [56], etc. GNNs generally adopt a neighborhood aggregation (message passing) paradigm, i.e., the representation of node is iteratively updated by aggregating representations of its neighbors [3, 4]. The representation of the whole graph is summarized on node representation through the readout function, e.g., graph pooling [4]. However, in order to achieve state-of-the-art performance, most famous GNNs [57–62], are trained end-to-end with task-specific labels, which could be extremely scarce for some graph datasets. Compared with these supervised models, our proposed model is based on self-supervised contrastive learning and can largely reduce the over-dependence on the manual labels, which is crucial for graph representation learning.

Contrastive Learning on Graphs. Recently, contrastive learning, adopting the instance discrimination as the pretext task, has become a dominant component in self-supervised learning methods [22, 23, 25, 27, 30]. Some literatures utilizing contrastive learning for graph data are proposed [5, 7, 9–11, 63]. The key of these methods is to maximize the agreement (i.e., similarity) between proper transformations or different views of the input graph. However, the existing graph contrastive learning methods explore general settings where entanglement is severe and do not incorporate disentangled representation learning. They fail to recognize and disentangle the heterogeneous latent factors behind complex graph data. These holistic methods have limited capacity in preserving detailed graph properties, which easily results in suboptimal representations for downstream tasks.

Disentangled Representation Learning. Disentangled representation learning is to learn factorized representations that identify and disentangle the underlying explanatory factors hidden in the observed data [14]. The existing efforts for disentangled representation learning are mainly on computer vision [64, 65]. Graph disentangled representation learning has raised a surge of interests recently [12, 19, 33, 66–68]. This line of works attempts to learn disentangled representations

for graphs but heavily relies on the annotated labels, which largely restricts their applications to scenarios where labeled data are unavailable or expensive to collect. On the other hand, some works [42, 69] are based on the generative model, namely utilizing Variational Autoencoders (VAEs) on graph for disentanglement, since the hyper-parameter β of VAEs can balance the reconstruction and disentanglement [18, 70]. However, the reconstruction in generative methods could be computationally expensive [21, 71] and even introduce bias that has a negative effect on the learned representation [27]. In addition, the reconstruction for graph-structured data often involves discrete decisions that are not differentiable [69]. How to learn disentangled representation on graph-structured data with contrastive learning is largely unexplored.

Orthogonal Regularization. Several works find that imposing orthogonal regularization on the weighting parameters can improve neural network training for more efficient optimization or better training stability [72–74]. More recently, orthogonal regularization is also utilized to constrain the latent space for learning disentangled representation in computer vision [75–77]. For example, PrOSe [75] proposes to parameterize the latent space representation as a product of orthogonal spheres. OroJaR [76] introduces orthogonal regularization in deep generative models. However, the orthogonal regularizations can only encourage linear independence while the HSIC regularization can encourage complex nonlinear dependence among the graph representations.

6 Conclusions

In this paper, we propose the independence promoted disentangled graph contrastive learning model (**IDGCL**) to solve the problem of learning disentangled self-supervised graph representation. We design a disentangled graph encoder with a tailored multi-channel message-passing layer, which is capable of aggregating features in a disentangled manner. We further propose a factor-wise contrastive learning approach to solve the instance discrimination task under each latent factor independently, so that the learned representations of **IDGCL** are encouraged to not only best describe the graphs but also be disentangled. We also present an independence regularization to eliminate the statistical dependence among different latent representations. Utilizing these techniques, each component of the disentangled representations in **IDGCL** tends to characterize a disentangled aspect of the graph that is pertinent to a latent factor. Extensive experiments on both synthetic and real-world datasets demonstrate the superiority of our method against several state-of-the-art baselines in unsupervised and semi-supervised graph classification tasks.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China No. 2020AAA0106300 and National Natural Science Foundation of China (No. 62250008, 62222209, 62102222, 62206149), China National Postdoctoral Program for Innovative Talents No. BX20220185, China Postdoctoral Science Foundation No. 2022M711813.

References

[1] Y. Li, D. Tarlow, M. Brockschmidt, and R. S. Zemel, "Gated graph sequence neural networks," in *ICLR*, 2016.
[2] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.
[3] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.
[4] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *ICLR*, 2019.
[5] F.-Y. Sun, J. Hoffman, V. Verma, and J. Tang, "Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," in *ICLR*, 2019.
[6] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *ICLR*, 2019.
[7] Y. Ren and B. Liu, "Heterogeneous deep graph infomax," in *AAAI Workshop of Deep Learning on Graphs*, 2020.
[8] Z. Peng, W. Huang, M. Luo, Q. Zheng, Y. Rong, T. Xu, and J. Huang, "Graph representation learning via graphical mutual information maximization," in *WebConf*, 2020, pp. 259–270.
[9] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," in *ICML*. PMLR, 2020, pp. 4116–4126.
[10] J. Qiu, Q. Chen, Y. Dong, J. Zhang, H. Yang, M. Ding, K. Wang, and J. Tang, "Gcc: Graph contrastive coding for graph neural network pre-training," in *SIGKDD*, 2020, pp. 1150–1160.
[11] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," in *NeurIPS*, 2020.
[12] Y. Yang, Z. Feng, M. Song, and X. Wang, "Factorizable graph convolutional networks," in *NeurIPS*, 2020.
[13] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *NeurIPS*, 2018, pp. 4800–4810.
[14] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *TPAMI*, vol. 35, no. 8, pp. 1798–1828, 2013.
[15] I. Higgins, D. Amos, D. Pfau, S. Racaniere, L. Matthey, D. Rezende, and A. Lerchner, "Towards a definition of disentangled representations," *arXiv:1812.02230*, 2018.
[16] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with hilbert-schmidt norms," in *ALT*, 2005, pp. 63–77.
[17] H. Li, X. Wang, Z. Zhang, Z. Yuan, H. Li, and W. Zhu, "Disentangled contrastive learning on graphs," in *NeurIPS*, 2021.
[18] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in β-vae," *NeurIPS Workshop on Learning Disentangled Representations*, 2017.
[19] J. Ma, P. Cui, K. Kuang, X. Wang, and W. Zhu, "Disentangled graph convolutional networks," in *ICML*. PMLR, 2019, pp. 4212–4221.
[20] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised

learning," *Technologies*, vol. 9, no. 1, p. 2, 2021.
[21] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *IEEE Access*, 2020.
[22] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *CVPR*, 2018, pp. 3733–3742.
[23] T. W. Tsai, C. Li, and J. Zhu, "Mi{ce}: Mixture of contrastive experts for unsupervised image clustering," in *ICLR*, 2021.
[24] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *ECCV*. Springer, 2020, pp. 776–794.
[25] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*. PMLR, 2020, pp. 1597–1607.
[26] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L2 hypersphere embedding for face verification," in *MM*, 2017, pp. 1041–1049.
[27] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv:1807.03748*, 2018.
[28] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.
[29] Z. Ma and M. Collins, "Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency," in *EMNLP*, 2018.
[30] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *CVPR*, 2020, pp. 9729–9738.
[31] H. Bahng, S. Chun, S. Yun, J. Choo, and S. J. Oh, "Learning de-biased representations with biased representations," in *ICML*. PMLR, 2020, pp. 528–539.
[32] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt, "Feature selection via dependence maximization." *JMLR*, vol. 13, no. 5, 2012.
[33] Y. Liu, X. Wang, S. Wu, and Z. Xiao, "Independence promoted graph disentangled networks," in *AAAI*, vol. 34, no. 04, 2020, pp. 4916–4923.
[34] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," *arXiv:1707.05005*, 2017.
[35] P. Yanardag and S. Vishwanathan, "Deep graph kernels," in *SIGKDD*, 2015, pp. 1365–1374.
[36] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *ICDM*. IEEE, 2005, pp. 8–pp.
[37] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," in *AISTATS*. PMLR, 2009, pp. 488–495.
[38] N. Shervashidze, P. Schweitzer, E. J. Van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels." *JMLR*, vol. 12, no. 9, 2011.
[39] R. Kondor and H. Pan, "The multiscale laplacian graph kernel," *NeurIPS*, vol. 29, pp. 2990–2998, 2016.
[40] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *SIGKDD*, 2016, pp. 855–864.
[41] B. Adhikari, Y. Zhang, N. Ramakrishnan, and B. A. Prakash, "Sub2vec: Feature learning for subgraphs," in *PAKDD*. Springer, 2018, pp. 170–182.
[42] T. N. Kipf and M. Welling, "Variational graph auto-encoders," in *NeurIPS Workshop on Bayesian Deep Learning*, 2016.
[43] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning automated," in *ICML*, 2021, pp. 12 121–12 132.
[44] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social networks*, vol. 5, no. 2, pp. 109–137, 1983.
[45] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society.*, vol. 28, no. 1, pp. 100–108, 1979.
[46] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *JMLR*, vol. 9, no. 11, 2008.
[47] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," in *NeurIPS*, 2020.
[48] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec, "Strategies for pre-training graph neural networks," *arXiv preprint arXiv:1905.12265*, 2019.
[49] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *AAAI*, vol. 32, no. 1, 2018.
[50] H. Li, P. Cui, C. Zang, T. Zhang, W. Zhu, and Y. Lin, "Fates of microscopic social ecosystems: Keep alive or dead?" in *SIGKDD*, 2019, pp. 668–676.
[51] H. Li, X. Wang, Z. Zhang, J. Ma, P. Cui, and W. Zhu, "Intention-aware sequential recommendation with structured intent transition," *TKDE*, 2021.
[52] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *TKDE*, 2020.
[53] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," in *NeurIPS*, 2018, pp. 5165–5175.
[54] Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande, "Moleculenet: a benchmark for molecular machine learning," *Chemical science*, vol. 9, no. 2, pp. 513–530, 2018.
[55] 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, 2017.
[56] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," *arXiv preprint arXiv:2101.11174*, 2021.
[57] H. Ji, X. Wang, C. Shi, B. Wang, and P. Yu, "Heterogeneous graph propagation network," *TKDE*, 2021.
[58] X. Fan, M. Gong, Y. Wu, A. Qin, and Y. Xie, "Propagation enhanced neural message passing for graph representation learning," *TKDE*, 2021.
[59] J. Gao, J. Gao, X. Ying, M. Lu, and J. Wang, "Higher-order interaction goes neural: A substructure assembling graph attention network for graph classification," *TKDE*, 2021.
[60] X. Miao, W. Zhang, Y. Shao, B. Cui, L. Chen, C. Zhang, and J. Jiang, "Lasagne: A multi-layer graph convolutional network framework via node-aware deep architecture," *TKDE*, 2021.
[61] Y. Ye and S. Ji, "Sparse graph attention networks," *TKDE*, 2021.
[62] Z. Zhang, P. Cui, J. Pei, X. Wang, and W. Zhu, "Eigen-

gnn: A graph structure preserving plug-in for gnns," *TKDE*, 2021.
[63] S. Zhang, Z. Hu, A. Subramonian, and Y. Sun, "Motif-driven contrastive learning of graph representations," *arXiv:2012.12533*, 2021.
[64] J.-T. Hsieh, B. Liu, D.-A. Huang, L. F. Fei-Fei, and J. C. Niebles, "Learning to decompose and disentangle representations for video prediction," in *NeurIPS*, 2018, pp. 517–526.
[65] L. Ma, Q. Sun, S. Georgoulis, L. Van Gool, B. Schiele, and M. Fritz, "Disentangled person image generation," in *CVPR*, 2018, pp. 99–108.
[66] Z. Xinyi and L. Chen, "Capsule graph neural network," in *ICLR*, 2018.
[67] S. Fan, X. Wang, C. Shi, P. Cui, and B. Wang, "Generalizing graph neural networks on out-of-distribution graphs," *arXiv preprint arXiv:2111.10657*, 2021.
[68] Y. Sui, X. Wang, J. Wu, M. Lin, X. He, and T.-S. Chua, "Causal attention for interpretable and generalizable graph classification," *SIGKDD*, 2022.
[69] M. Simonovsky and N. Komodakis, "Graphvae: Towards generation of small graphs using variational autoencoders," in *ICANN*, 2018, pp. 412–422.
[70] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *ICLR*, 2016.
[71] X. Guo, L. Zhao, Z. Qin, L. Wu, A. Shehu, and Y. Ye, "Interpretable deep graph generation with node-edge co-disentanglement," in *SIGKDD*, 2020, pp. 1697–1707.
[72] N. Bansal, X. Chen, and Z. Wang, "Can we gain more from orthogonality regularizations in training deep networks?" *NeurIPS*, vol. 31, 2018.
[73] K. Jia, D. Tao, S. Gao, and X. Xu, "Improving training of deep neural networks via singular value bounding," in *CVPR*, 2017, pp. 4344–4352.
[74] J. Wang, Y. Chen, R. Chakraborty, and S. X. Yu, "Orthogonal convolutional neural networks," in *CVPR*, 2020, pp. 11 505–11 515.
[75] A. Shukla, S. Bhagat, S. Uppal, S. Anand, and P. Turaga, "Product of orthogonal spheres parameterization for disentangled representation learning," *arXiv preprint arXiv:1907.09554*, 2019.
[76] Y. Wei, Y. Shi, X. Liu, Z. Ji, Y. Gao, Z. Wu, and W. Zuo, "Orthogonal jacobian regularization for unsupervised disentanglement in image generation," in *ICCV*, 2021, pp. 6721–6730.
[77] M. H. Sarhan, N. Navab, A. Eslami, and S. Albarqouni, "Fairness by learning orthogonal disentangled representations," in *ECCV*, 2020, pp. 746–761.
[78] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.
[79] Z. Wang, T. Jian, A. Masoomi, S. Ioannidis, and J. Dy, "Revisiting hilbert-schmidt information bottleneck for adversarial robustness," *NeurIPS*, vol. 34, pp. 586–597, 2021.

Haoyang Li received his B.E. from the Department of Computer Science and Technology, Tsinghua University in 2018. He is a Ph.D. candidate in the Department of Computer Science and Technology of Tsinghua University. His research interests are mainly in machine learning on graphs and out-of-distribution generalization. He has published several papers in prestigious journals and conferences, e.g., TKDE, KDD, NeurIPS, ICLR, etc.

Ziwei Zhang received his Ph.D. from the Department of Computer Science and Technology, Tsinghua University, in 2021. He is currently a postdoc researcher in the Department of Computer Science and Technology at Tsinghua University. His research interests focus on machine learning on graphs, including graph neural network (GNN), network embedding, and automated graph learning. He has published over 20 papers in prestigious conferences and journals, including KDD, ICML, AAAI, and TKDE. He is the recipient of 2022 China National Postdoctoral Program for Innovative Talents.

Xin Wang is currently an Assistant Professor at the Department of Computer Science and Technology, Tsinghua University. He got both of his Ph.D. and B.E degrees in Computer Science and Technology from Zhejiang University, China. He also holds a Ph.D. degree in Computing Science from Simon Fraser University, Canada. His research interests include cross-modal multimedia intelligence and inferable recommendation in social media. He has published several high-quality research papers in top conferences including ICML, MM, KDD, WWW, SIGIR etc. He is the recipient of 2017 China Postdoctoral innovative talents supporting program. He receives the ACM China Rising Star Award in 2020.

Wenwu Zhu is currently a Professor in the Department of Computer Science and Technology at Tsinghua University, the Vice Dean of National Research Center for Information Science and Technology, and the Vice Director of Tsinghua Center for Big Data. Prior to his current post, he was a Senior Researcher and Research Manager at Microsoft Research Asia. He was the Chief Scientist and Director at Intel Research China from 2004 to 2008. He worked at Bell Labs New Jersey as Member of Technical Staff during 1996-1999. He received his Ph.D. degree from New York University in 1996.

His current research interests are in the area of data-driven multimedia networking and Cross-media big data computing. He has published over 350 referred papers, and is inventor or co-inventor of over 50 patents. He received eight Best Paper Awards, including ACM Multimedia 2012 and IEEE Transactions on Circuits and Systems for Video Technology in 2001 and 2019.

He served as EiC for IEEE Transactions on Multimedia (2017-2019). He served in the steering committee for IEEE Transactions on Multimedia (2015-2016) and IEEE Transactions on Mobile Computing (2007-2010), respectively. He serves as General Co-Chair for ACM Multimedia 2018 and ACM CIKM 2019, respectively. He is an AAAS Fellow, IEEE Fellow, SPIE Fellow, and a member of The Academy of Europe (Academia Europaea).