Dynamic and Chemical Constraints to Enhance the Molecular Masked Graph Autoencoders

Jiahui Zhang[1,2], **Wenjie Du**[1,2,†], **Yang Wang**[1,2,†]
[1]University of Science and Technology of China, China
[2]Suzhou Institute for Advanced Research, USTC, China
kongping@mail.ustc.edu.cn {duwenjie, angyan}@ustc.edu.cn

Abstract

Masked Graph Autoencoders (MGAEs) have gained significant attention recently. Their proxy tasks typically involve random corruption of input graphs followed by reconstruction. However, in the molecular domain, two main issues arise: the predetermined mask ratio and reconstruction objectives can lead to suboptimal performance or negative transfer due to overly simplified or complex tasks, and these tasks may deviate from chemical priors. To tackle these challenges, we propose Dynamic and Chemical Constraints (DyCC) for MGAEs. This includes a masking strategy called GIBMS, which preserves essential semantic information during graph masking while adaptively adjusting the mask ratio and content for each molecule. Additionally, we introduce a Soft Label Generator (SLG) that reconstructs masked tokens as learnable prototypes (soft labels) rather than hard labels. These components adhere to chemical constraints and allow dynamic variation of proxy tasks during training. We integrate the model-agnostic DyCC into various MGAEs and conduct comprehensive experiments, demonstrating significant performance improvements. Our code is available at `https://github.com/forever-ly/DyCC`.

1 Introduction

Molecular Representation Learning (MRL) plays a pivotal role in many related applications such as drug discovery, material design, and reaction prediction [8, 11, 44]. By representing molecules as graphs, where atoms are treated as nodes and bonds as edges, Graph Neural Networks (GNNs) [43, 20, 14] have exhibited remarkable performance across a wide range of tasks. However, a significant challenge is the scarcity of labeled data, which limits the effectiveness of supervised learning. Inspired by the remarkable progress in self-supervised pretraining in natural language processing [12], Masked Graph Autoencoders (MGAEs) [17, 41, 25, 16] have arisen as a promising approach to addressing these challenges. The pioneering work [17] on this topic introduced the pretraining of GNNs using a mask-then-recontruction task called AttrMask. Specifically, they randomly mask some proportions of atoms and then pretrain the models to predict them. AttrMask has emerged as a fundamental pretraining task and many subsequent works [16, 54] adopt it as a subtask for pretraining. Despite their success, we have identified two limitations that still lack exploration.

The first limitation is that proxy tasks are predetermined and lack dynamic adaptability. The effectiveness of MGAEs is largely governed by their proxy tasks, which are defined by the graph-masking strategy and the reconstruction objective. Prior approaches rely on fixed mask strategies and tokenizers, which result in suboptimal performance. **From the perspective of input corruption**, the

† corresponding author

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

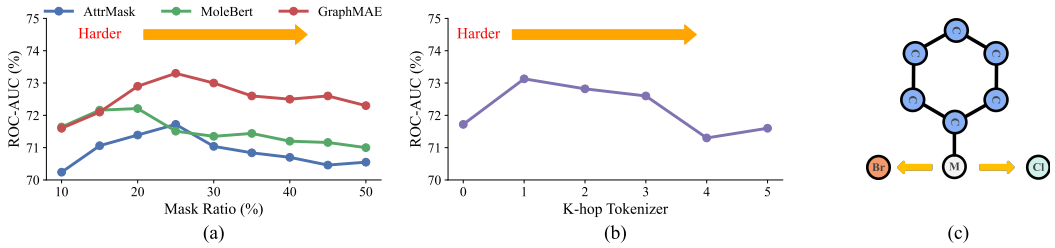

Figure 1: **(a)** The difficulty of proxy tasks for the AttrMask, MoleBert, and GraphMAE models was increased by raising the mask ratio. The average ROC-AUC scores were computed across the eight classification datasets in MoleculeNet [39]. **(b)** The average ROC-AUC scores of AttrMask with K-hop subgraph token as the reconstruction target. The larger the value of K, the more challenging the proxy task becomes. **(c)** Reconstructing the masked atom as either bromine or chlorine can generate valid molecules.

masking strategy significantly impacts pretraining performance. As shown in Fig. 1(a), involving GraphMAE[16], AttrMask [17], and MoleBert [41], we progressively increased the mask ratio to make the proxy task harder. Initially, as the mask ratio increased, pretraining performance improved; however, beyond a certain threshold, performance deteriorated. **From the perspective of the reconstruction target**, the choice of reconstruction tokens greatly affects pretraining performance. AttrMask treats atoms as tokens, but the simplicity of its reconstruction objective can cause suboptimal performance or even negative transfer. Consequently, several studies have proposed more challenging tokenizers [54, 41, 25]. For example, SimSGT [25] uses K-hop subgraphs as reconstruction targets and employs a simple GNN-based tokenizer to generate subgraph-level tokens. By adjusting the value of K, the complexity of the reconstruction task can be controlled—the larger the K, the more difficult the task. However, we observe that more challenging tokenizers do not necessarily lead to better pretraining performance. As shown in Fig. 1(b), we used K-hop subgraph tokens as the reconstruction target (with $K = 0$ corresponding to node-level tokens). While increasing K indicates a higher tokenizer complexity (i.e., a harder proxy task), the performance did not consistently improve. **In summary**, employing fixed masking and reconstruction strategies (i.e., a predetermined tokenizer) necessitates a cumbersome search for optimal hyperparameters—and even then yields only suboptimal results. A more effective solution is to introduce dynamic adaptability into proxy tasks.

The second limitation is that proxy tasks may not adhere to the constraints imposed by chemical priors, potentially leading to nonsensical or even harmful self-supervised learning signals. We outline three aspects that deviate from chemical constraints (CC), labeled as *CC1, CC2,* and *CC3*. *CC1: Each molecule has its specific mask ratio*. Currently, most approaches adopt a globally fixed mask ratio. However, different molecules exhibit varying degrees of structural redundancy, rendering a uniform mask ratio suboptimal—too high for certain molecules. *CC2: The importance of atoms within a molecule varies*, as certain key atoms play pivotal roles in determining molecular functional properties, reactivity, and biological interactions. Consequently, prioritizing the masking of these important atoms can encourage the model to engage in more challenging contextual reasoning, thereby fostering a deeper understanding of critical structural features. *CC3: The reconstruction target should not be unique*. Given the vast chemical compound space, many compounds exhibit significant structural similarity. As illustrated in Fig. 1(c), the masked atom could be reconstructed as either Cl or Br. However, current methods often constrain the reconstruction by favoring specific atom types (or substructures), producing conflicting self-supervised signals.

To address these issues, we propose Dynamic and Chemical Constraints (DyCC) MGAEs, a model-agnostic approach dynamically adjusts the proxy task while adhering to chemical constraints. Specifically, we leverage the Graph Information Bottleneck (GIB) theory [52, 51] to redefine graph masking as a graph compression problem, introducing the GIB-based Mask Strategy (GIBMS). The core concept is to identify a compressed core substructure within the molecular graph that encapsulates its key properties, and to place greater emphasis on reconstructing these core substructures during the graph masking stage. This design explicitly addresses *CC1* and *CC2* by encouraging the model to focus on essential structural information while learning robust molecular representations. However, traditional GIB relies on supervisory signals, which are unavailable during pretraining when downstream task

labels are not accessible. **Our contribution lies in extending GIB theory to the unsupervised setting**. We adopt the common multi-view assumption [30, 42], enabling us to demonstrate that the mutual information between the molecular graph and self-supervised signals acts as a lower bound for the mutual information between the graph and downstream task labels. As a result, we reformulate the problem as maximizing the mutual information between the molecular graph and self-supervised signals, which can be achieved through a contrastive learning paradigm.

Additionally, we introduce the Soft Label Generator (SLG) module, **which transforms the reconstruction objective from a specific token (hard label) to a soft cluster assignment (soft label), thereby fulfilling *CC3*[7, 4, 2].** We define potential clusters as prototypes represented by learnable vectors and subsequently map the hard labels to probability distributions of these prototypes using the Sinkhorn-Knopp algorithm[10]. Specifically, we randomly initialize a set of prototypes (learnable vectors), and both the token labels and the reconstruction predictions are evaluated for similarity against these prototypes, yielding two probability assignment matrices (soft labels). Minimizing the discrepancy between these two matrices is equivalent to minimizing the difference between the labels and the reconstruction predictions. During the training process, as the prototypes are updated, the mapped soft labels dynamically change, thereby enabling the proxy task to be adjusted throughout the training phase (fulfilling *dynamic*). In extreme cases, when the assignment probability distributions converge to the one-hot distribution, the soft labels degenerate into hard labels, resulting in high inter-class distinctiveness and simplifying the reconstruction task. Conversely, as the assignment probability distributions approach a uniform distribution, inter-class separability diminishes, rendering the reconstruction task exceptionally challenging.

In summary, our core contributions are as follows: First, we identified the lack of dynamism and the failure to adhere to chemical constraints in the proxy tasks of MGAEs. To address these challenges, we extend the supervised GIB theory to the unsupervised setting and design the GIBMS module, generating the optimal mask ratio and mask content for each molecule. Additionally, we introduce the concept of soft assignment into the graph reconstruction stages, avoiding conflicting self-supervised signals and dynamically adjusting the tokenizer. Lastly, we integrate these two model-agnostic modules into multiple MGAEs. Extensive experimentation shows consistent improvements across the integrated models, validating the effectiveness and generality of our approach.

2 Related Work

Graph Contrastive Self-supervised Learning Contrastive self-supervised learning, follows the principle of mutual information maximization [1], which typically works to maximize the correspondence between the representations of an instance (e.g., node, subgraph, or graph) in its different augmentation views. GraphCL [49] performs graph-level contrastive learning with combinations of four graph augmentations, namely node dropping, edge perturbation, subgraph cropping, and feature masking. InfoGraph [32] conducts graph representation learning by maximizing the mutual information between graph-level representations and local substructures. GraphLOG [33] leverages clustering to construct hierarchical prototypes of graph samples. They further contrast each local instance with its corresponding higher prototype for contrastive learning. JOAO [48] proposes a framework to automatically search proper data augmentations for GCL. GraphMVP [24] uses a contrastive loss and a generative loss to connect the 2-dimensional view and 3-dimensional view of the same molecule, in order to inject the 3-dimensional knowledge into the 2-dimensional graph encoder. RGCL [22] trains a rationale generator to identify the causal subgraph in graph augmentation. Although the contrastive learning paradigm is very successful, it relies on data augmentation, which depends on domain knowledge.

Graph Generative self-supervised learning Generative self-supervised learning focuses on recovering missing parts of input data. It can be further divided into two families: autoregressive and autoencoding models. Autoregressive models break down joint probability distributions into a product of conditionals. In supervised graph generation, earlier methods like GraphRNN [47] and GCPN [46] have been proposed. More recently, GPT-GNN [18] represents an attempt to incorporate graph generation as a training objective. On the other hand, graph autoencoders are designed to reconstruct input data without enforcing a decoding order. Early work in this field includes GAE and VGAE [21], which use 2-layer GNN as encoders and dot-product decoding for link prediction. AttrMask [17] adopts a random masking strategy, where a portion of the nodes are randomly masked,

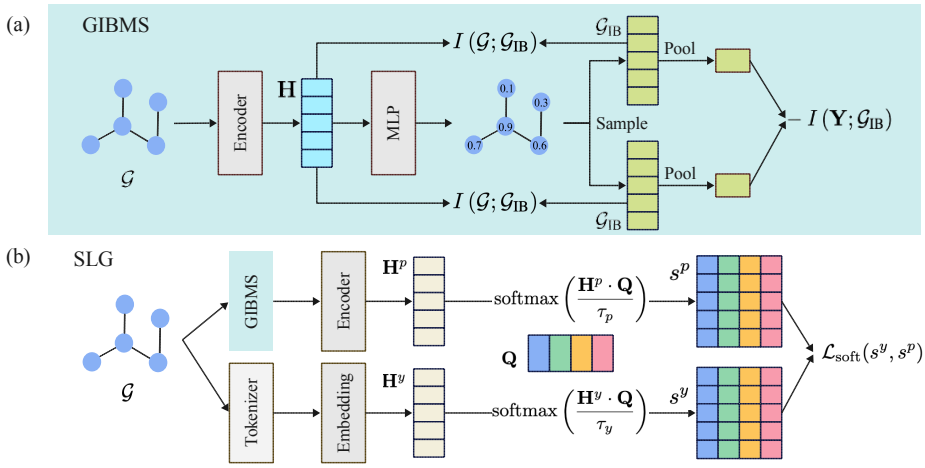

Figure 2: Overall architecture of DyCC. **(a):** The GIBMS module identifies the sampling probabilities of nodes in the molecular graph, based on graph information bottleneck theory. **(b):** During reconstruction, both the hard labels of the tokens $\mathbf{H}^y$ and the predictions $\mathbf{H}^p$ are mapped into soft labels s^y and soft predictions s^p through the SLG module, forming probability distributions. The training objective $\mathcal{L}_{\text{soft}}$ is to minimize the distance between these two distributions.

and the goal is to reconstruct them. GraphMAE [16] propose to focus on feature reconstruction with both a masking strategy and scaled cosine error that benefit the robust training of GraphMAE. Mole-BERT [41] observes that mask atom prediction is an overly easy pretraining task. Therefore, they employ a GNN tokenizer pretrained by VQ-VAE to generate more complex reconstruction targets for masked atom modeling. SimSGT [25] utilizes a simple GNN-based tokenizer, which removes the nonlinear update function in each GNN layer to derive subgraph level tokens. These works all adopt a fixed proxy task setup and do not take into account the three chemical constraints we proposed.

3 Method

To meet the constraints of chemical priors and enable dynamic adaptability in proxy tasks, we have designed the GIBMS and SLG, as depicted in Fig. 2. Next, we will provide detailed explanations of these two modules.

3.1 GIBMS: Graph Information Bottleneck for Mask Strategy

Subgraph recognition [52, 51, 53] aims to identify a condensed core substructure within a graph that maximizes its informativeness regarding the graph property while discarding redundant information. Inspired by this, we propose to dynamically generate a core subgraph for each molecule (CC1) and place greater emphasis on masking the important atoms within this core substructure (CC2). Subgraph recognition can be formulated by optimizing GIB [52] with a mutual information estimator. For a graph $\mathcal{G}$ and its label information $\mathbf{Y}$, the optimal IB-graph $\mathcal{G}_{\text{IB}}$ which keeps minimal sufficient information:

$$\mathcal{G}_{\text{IB}} = \arg\min_{\mathcal{G}_{\text{IB}}} -I\left(\mathbf{Y};\mathcal{G}_{\text{IB}}\right) + \beta I\left(\mathcal{G};\mathcal{G}_{\text{IB}}\right) \tag{1}$$

where β serves as a Lagrangian multiplier to balance the two mutual information terms.

Obtaining node representations of molecular graph To begin, it is essential to convert the molecular graph $\mathcal{G}$ into a vector representation. GNNs have emerged as the predominant approach for handling molecular graphs due to their effectiveness in capturing graph-structured data. Let Φ denote a GNN encoder, which is used to generate the node-level representations of the graph. Formally, this process can be expressed as:

$$\mathbf{H} = \Phi(\mathcal{G}) \tag{2}$$

where $\mathbf{H}$ represents the node representations of the graph.

Injecting Noise to Obtain the IB-Graph The discrete nature of graphs renders direct acquisition of the $\mathcal{G}_{\text{IB}}$ impractical due to the exponential proliferation of candidates (2^N) for a given graph $\mathcal{G}$ with N nodes. Therefore, we propose a relaxation by assuming a gilbert random graph [13], where node selection from the input graph $\mathcal{G}$ is conditionally independent. This assumption enables us to factorize the probability of the $\mathcal{G}_{\text{IB}}$ as:

$$\mathcal{P}(\mathcal{G}_{\text{IB}}|\mathcal{G}) = \prod_{i \in \mathcal{V}} \mathcal{P}\left(\mathcal{V}_i|\mathcal{G}\right) \tag{3}$$

Here, $\mathcal{V}$ denotes the nodes of $\mathcal{G}$, and $\mathcal{P}\left(\mathcal{V}_i|\mathcal{G}\right)$ signifies the probability distribution for node $\mathcal{V}_i$. A straightforward instantiation of $\mathcal{P}\left(\mathcal{V}_i|\mathcal{G}\right)$ is the Bernoulli distribution $\mathcal{V}_i \sim \text{Bernoulli}(p_i)$ parameterized by p_i. We compute the probability p for each node based on its embedding $\mathbf{H}$ using a Multi-Layer Perceptron network $\mathcal{M}$, expressed as:

$$p = \text{Sigmoid}(\mathcal{M}(\mathbf{H})) \tag{4}$$

Here, the Sigmoid function ensures the probabilities are normalized. The graph $\mathcal{G}_{\text{IB}}$ is then derived by performing Bernoulli sampling on all nodes:

$$\mathcal{G}_{\text{IB}} = \{\mathcal{V}_i \mid \mathcal{V}_i \sim \text{Bernoulli}(p_i), i = 1, 2, \ldots, N\} \tag{5}$$

Here $\mathcal{G}_{\text{IB}}$ is a sampled combination of nodes. Furthermore, we adopt noise injection following VGIB [51] to optimize the subgraph, as it has been proven to mitigate inefficiency and instability in GIB optimization caused by mutual information estimation. The key idea is to introduce more noise into less informative subgraphs while injecting less noise into more informative ones. Specifically, with the calculated probability p, we perturb the representation $\mathbf{H}$ by adding noise ϵ:

$$\hat{\mathbf{H}} = \lambda \mathbf{H} + (1 - \lambda)\, \epsilon \tag{6}$$

where $\lambda \sim \text{Bernoulli}\,(p)$ and $\epsilon \sim \mathcal{N}\left(\mu_{\mathbf{H}}, \sigma^2_{\mathbf{H}}\right)$. Here, $\mu_{\mathbf{H}}$ and $\sigma^2_{\mathbf{H}}$ are the mean and variance of $\mathbf{H}$, respectively, and Bernoulli represents the Bernoulli distribution. Thus, the information of $\mathcal{G}$ is compressed into $\mathcal{G}_{\text{IB}}$ with the probability of λ by replacing non-important nodes with noise. Moreover, to make the sampling process differentiable, we adopt a gumbel-sigmoid [26, 19] for discrete random variable λ, i.e.,

$$\lambda = \text{Sigmoid}(\frac{1}{t} \log \left[\frac{p}{(1-p)}\right] + \log \left[\frac{u}{(1-u)}\right]) \tag{7}$$

where $u \sim \text{Uniform}\,(0, 1)$, and t is the temperature hyperparameter.To ensure that the obtained $\mathcal{G}_{\text{IB}}$ is meaningful, we need to solve Eq. (1). Next, we will discuss the optimization of the first prediction term $-I\left(\mathbf{Y}; \mathcal{G}_{\text{IB}}\right)$ and the second compression term $I\left(\mathcal{G}; \mathcal{G}_{\text{IB}}\right)$ separately.

Minimizing the Prediction Term The first term, $-I\left(\mathbf{Y}; \mathcal{G}_{\text{IB}}\right)$, encourages $\mathcal{G}_{\text{IB}}$ to be informative about the label $\mathbf{Y}$. Since the goal of pre-training is to enhance the performance of downstream tasks, $\mathbf{Y}$ here refers to the labels of the downstream task dataset. To avoid confusion in notation moving forward, we will denote the labels of the downstream tasks as $\mathbf{Y}^{\text{sup}}$. Given the input $\mathcal{G}_{\text{IB}}$ and the downstream task labels $\mathbf{Y}^{\text{sup}}$, our objective is to learn a vector representation of $\mathcal{G}_{\text{IB}}$, denoted as $\mathbf{Z}^{\text{sup}}_{\mathcal{G}_{\text{IB}}} = \text{Pool}(\hat{\mathbf{H}})$, which can effectively predict the labels $\mathbf{Y}^{\text{sup}}$. Here, Pool represents the pooling function.

$$\mathbf{Z}^{\text{sup}}_{\mathcal{G}_{\text{IB}}} = \arg\max_{\mathbf{Z}_{\mathcal{G}_{\text{IB}}}} I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right) \tag{8}$$

However, since GIBMS is applied during the pretraining stage, obtaining downstream task labels $\mathbf{Y}^{\text{sup}}$ is not feasible. Here we approach it from a self-supervised learning perspective. Considering the input $\mathcal{G}_{\text{IB}}$ and self-supervised signals $\mathbf{S}$ (e.g., augmentations of $\mathcal{G}_{\text{IB}}$) as two different views of the data, we aim to derive sufficient self-supervised representations $\mathbf{Z}^{\text{ssl}}_{\mathcal{G}_{\text{IB}}}$ that can maximize the preservation of shared information between the views.

$$\mathbf{Z}^{\text{ssl}}_{\mathcal{G}_{\text{IB}}} = \arg\max_{\mathbf{Z}_{\mathcal{G}_{\text{IB}}}} I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S}\right) \tag{9}$$

where $\mathbf{Z}_{\mathcal{G}_{\text{IB}}}$ is the representation of the graph, obtained by pooling the node representations $\hat{\mathbf{H}}$, i.e., $\mathbf{Z}_{\mathcal{G}_{\text{IB}}} = \text{Pool}(\hat{\mathbf{H}})$, where Pool is the pooling function. By adopting the common multi-view assumption [30, 42], we have (Appendix E.1):

$$I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) = I\left(\mathbf{Z}^{\text{sup}}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right) \geq I\left(\mathbf{Z}^{\text{ssl}}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right) \geq I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) - \epsilon_{\text{info}}; \quad \epsilon_{\text{info}} > 0 \tag{10}$$

In this paper, we assume that with appropriate self-supervised signals $\mathbf{S}$, ϵ_{info} is negligible. Consequently, the above formula suggests that the self-supervised learned representations $\mathbf{Z}^{\text{ssl}}_{\mathcal{G}_{\text{IB}}}$ can capture almost as much task-relevant information about $\mathbf{Y}^{\text{sup}}$ as the supervised representations $\mathbf{Z}^{\text{sup}}_{\mathcal{G}_{\text{IB}}}$. In this case, minimizing $-I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}})$ is approximately equivalent to maximizing $I(\mathbf{Z}^{\text{sup}}_{\mathcal{G}_{\text{IB}}}; \mathbf{S})$. Since recently proposed contrastive learning methods [35, 15, 50], which aim to pull positive samples closer and push negative samples apart in the representation space, have been theoretically proven to maximize the mutual information between positive pairs, we leverage this approach. Given $\mathcal{G}_{\text{IB},i}$, we can repeatedly sample from Eq. (5) to obtain its positive sample $\mathcal{G}'_{\text{IB},i}$ as the self-supervised signal $\mathbf{S}$. Then, we can maximize $I(\mathbf{Z}^{\text{sup}}_{\mathcal{G}_{\text{IB}}}; \mathbf{S})$ using a contrastive learning loss, such as InfoNCE [37]:

$$I\left(\mathbf{Y}^{\text{sup}}; \mathcal{G}_{\text{IB}}\right) = \mathcal{L}_{\text{pred}}\left(\mathbf{Y}^{\text{sup}}, \mathcal{G}_{\text{IB}}\right) = -\frac{1}{K}\sum_{i=1}^{K} \log \frac{\exp\left(\text{sim}\left(\mathbf{Z}_{\mathcal{G}_{\text{IB},i}}, \mathbf{Z}_{\mathcal{G}'_{\text{IB},i}}\right)/\tau\right)}{\sum_{j=1, j\neq i}^{K} \exp\left(\text{sim}\left(\mathbf{Z}_{\mathcal{G}_{\text{IB},i}}, \mathbf{Z}_{\mathcal{G}_{\text{IB},j}}\right)/\tau\right)} \tag{11}$$

Here, the representations $\mathbf{Z}_{\mathcal{G}_{\text{IB},i}}$ and $\mathbf{Z}_{\mathcal{G}'_{\text{IB},i}}$ of two IB-graphs $\mathcal{G}_{\text{IB},i}$ and $\mathcal{G}'_{\text{IB},i}$ are considered as positive samples, while representations $\mathbf{Z}_{\mathcal{G}_{\text{IB},j}}$ of $\mathcal{G}_{\text{IB},j}$ from other graphs in the same batch are treated as negative samples. K and τ indicate the number of paired graphs in a batch and the temperature hyperparameter, respectively.

Optimizing the Compression Term The second term minimizes the mutual information of $\mathcal{G}$ and $\mathcal{G}_{\text{IB}}$ so that $\mathcal{G}_{\text{IB}}$ only receives limited information from the input graph $\mathcal{G}$. We can derive its variational upper bound (see Appendix E.2):

$$I\left(\mathcal{G}; \mathcal{G}_{\text{IB}}\right) \leq \mathbb{E}_{\mathcal{G}}\left(-\frac{1}{2}\log A + \frac{1}{2N}A + \frac{1}{2N}B^2\right) = \mathcal{L}_{\text{comp}}\left(\mathcal{G}, \mathcal{G}_{\text{IB}}\right) \tag{12}$$

where N represents the number of nodes in $\mathcal{G}$, $A = \sum_{j=1}^{N}\left(1-\lambda_j\right)^2$ and $B = \frac{\sum_{j=1}^{N}\lambda_j(\mathbf{H}_j - \mu_{\mathbf{H}})}{\sigma_{\mathbf{H}}}$.

The Final Training Objective Based on the aforementioned analysis, the training objective of GIBMS is as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{pred}}\left(\mathbf{Y}^{m}, \mathcal{G}_{\text{IB}}\right) + \mathcal{L}_{\text{pred}}\left(\mathcal{G}_{\text{IB}}, \mathbf{Y}^{\text{sup}}\right) + \beta\mathcal{L}_{\text{comp}}\left(\mathcal{G}, \mathcal{G}_{\text{IB}}\right) \tag{13}$$

Dynamic Masking Strategy The trained GIBMS is able to generate an importance score p_i for each atom node v_i in the molecular graph $\mathcal{G}$ according to Eq.(7), where p_i represents the probability of retaining the node under the information bottleneck framework. Subsequently, we adopt the Gumbel-Sigmoid approximation to sample a dynamic masking factor λ_i for each node, as shown in Eq.(7). Here, λ_i is a continuous variable in the range $[0, 1]$, which is further converted into a binary masking decision by applying a threshold r, where $\lambda_i > r$ indicates that the corresponding atom will be masked. This strategy allows the masking ratio to be adaptively adjusted at the molecule level according to the specific structural characteristics of each graph, without the need to predefine a fixed global masking ratio (CC1). Moreover, the mechanism tends to preferentially mask nodes with higher importance scores (CC2), thereby encouraging the pretraining model to focus on capturing the key structures within molecular graphs and enhancing the discriminability and generalization of the learned representations.

3.2 Soft Label Generator

The output of the tokenizer can be either discrete values (e.g., atomic numbers) or continuous vectors (e.g., representations of subgraphs). For the sake of convenience in subsequent discussions, we map the categorical variable y of tokens through embedding into a vector, unifying the reconstruction label as a d-dimensional vector $\mathbf{H}^y \in \mathbb{R}^d$.

Regardless of the tokenizer used, existing methods reconstruct masked atoms into specific types. In the preceding discussion, we emphasized the importance of soft labels. The question now arises: *how do we obtain soft label s^y?* Since we cannot access all molecules in the world, it is not feasible to directly acquire the probability of reconstructing a masked atom into different tokens. To address

this limitation, we introduce soft label assignment [7, 4, 2]. We assume the existence of n learnable latent prototypes $\mathbf{Q} \in \mathbb{R}^{n \times d}$, where each masked atom has a probability of being reconstructed into all these prototypes. Firstly, we map hard labels $\mathbf{H}^y$ to soft labels s^y:

$$s^y = \text{softmax}\left(\frac{\mathbf{H}^y \cdot \mathbf{Q}}{\tau_y}\right) \tag{14}$$

where $\tau_y \in (0, 1)$ is a temperature. Similarly, for the node representations $\mathbf{H}^p$ outputted by the encoder Φ', we generate their predictions s^p by measuring the cosine similarity to the same prototype matrix $\mathbf{Q}$ with temperature $\tau_p \in (0, 1)$.

$$s^p = \text{softmax}\left(\frac{\mathbf{H}^p \cdot \mathbf{Q}}{\tau_p}\right) \tag{15}$$

Note, we always choose $\tau_y < \tau_p$ to encourage sharper target predictions, which implicitly guides the model to produce confident low entropy anchor predictions. We penalize when the prediction s^p is different from the soft label s^y. We enforce this criterion using a standard cross-entropy loss $H\left(s^y, s^p\right)$. We also incorporate the mean entropy maximization (ME-MAX) regularizer [3, 5] to encourage the model to utilize the full set of prototypes. Denote the average prediction of a batch of M samples as $\bar{s^p}$:

$$\bar{s^p} = \frac{1}{M} \sum_{i=1}^{K} s_i^p \tag{16}$$

The ME-MAX regularizer simply seeks to maximize the entropy of $\bar{s^p}$, denoted $H(\bar{s^p})$, or equivalently, minimize the negative entropy of $\bar{s^p}$. Thus, the overall objective is:

$$\mathcal{L}_{\text{soft}}(s^y, s^p) = \frac{1}{K} \sum_{i=1}^{K} H\left(s_i^y, s_i^p\right) - \alpha H(\bar{s^p}) \tag{17}$$

where $\alpha > 0$ controls the weight of the ME-MAX regularization.

4 Experiments

4.1 Experiments setup

Pretraining setup For the pretraining stage, we utilized 2 million molecules sourced from the ZINC15 database [31], following the precedent of prior studies [17]. The GIBMS module was trained using the loss function defined in Eq. (13), where the temperature factor $\tau = 0.1$ for the InfoNCE loss, and $\beta = 0.01$ controls the trade-off between prediction and compression. After training the GIBMS module, we utilized it to generate corresponding mask probabilities for each atom of the 2 million molecules in ZINC15 and sampled masked atoms based on these probabilities. In the reconstruction phase, we mapped the hard labels outputted by the tokenizer to soft labels using the SLG module. By default, we set temptures $\tau_p = 0.25$, $\tau_y = 0.1$, and the number of prototypes $n = 128$. After pretraining, we employed the widely-adopted 8 binary classification datasets within MoleculeNet [39] to evaluate performance on downstream molecular property prediction tasks (see Appendix B). These downstream datasets are divided into train/valid/test sets using scaffold split by 8:1:1 to facilitate an out-of-distribution evaluation setting. We report the mean performances (ROC-AUC) and standard deviations on the downstream datasets across ten random seeds.

Baselines We integrated our method into three MGAEs: AttrMask [17], MoleBert [41], and SimSGT [25]. All our settings remain consistent with the configurations of these models. It is noteworthy that, for a fairer evaluation of DyCC on the masked atom modeling proxy task, we have excluded irrelevant enhancements of MoleBert and SimSGT. Specifically, we removed the GraphTrans variant from SimSGT to ensure consistency in using the GIN architecture. Additionally, we eliminated the triplet masked contrastive learning (TMCL) from MoleBert as it is unrelated to MGAEs (see Appendix C.5). Furthermore, we selected several other self-supervised graph pretraining models for further comparison, including InfoGraph [32], GPT-GNN [18], EdgePred [17], ContextPred [17], GraphLOG [33], G-Contextual [29], G-Motif [29], AD-GCL [34], JOAO [48], SimGRACE [40], GraphCL [49], GraphMAE [16], GraphMVP [24] and MGSSL [54]. The results are collect from MoleBert [41].

Table 1: Transfer learning ROC-AUC (%) scores on eight MoleculeNet datasets. The suffix "DyCC" implies the introduction of both the GIBMS and SLG modules.

Dataset	Tox21	ToxCast	Sider	ClinTox	MUV	HIV	BBBP	Bace	Avg.(↑)
No Pretrain	74.6±0.4	61.7±0.5	58.2±1.7	58.4±6.4	70.7 ±1.8	75.5±0.8	65.7±3.3	72.4±3.8	67.0
InfoGraph	73.3±0.6	61.8±0.4	58.7±0.6	75.4±4.3	74.4±1.8	74.2±0.9	68.7±0.6	74.3±2.6	70.1
GPT-GNN	74.9±0.3	62.5±0.4	58.1±0.3	58.3±5.2	75.9±2.3	65.2±2.1	64.5±1.4	77.9±3.2	68.5
EdgePred	76.0±0.6	64.1±0.6	60.4±0.7	64.1±3.7	75.1±1.2	76.3±1.0	67.3±2.4	77.3±3.5	70.1
ContextPred	73.6±0.3	62.6±0.6	59.7±1.8	74.0±3.4	72.5±1.5	75.6±1.0	70.6±1.5	78.8±1.2	70.1
GraphLoG	75.0±0.6	63.4±0.6	59.6±1.9	75.7±2.4	75.5±1.6	76.1±0.8	68.7±1.6	78.6±1.0	71.6
G-Contextual	75.0±0.6	62.8±0.7	58.7±1.0	60.6±5.2	72.1±0.7	76.3±1.5	69.9±2.1	79.3±1.1	69.3
G-Motif	73.6±0.7	62.3±0.6	61.0±1.5	77.7±2.7	73.0±1.8	73.8±1.1	66.9±3.1	73.0±3.3	70.2
AD-GCL	74.9±0.4	63.4±0.7	61.5±0.9	77.2±2.7	76.3±1.4	76.7±1.2	70.7±0.3	76.6±1.5	72.2
JOAO	74.8±0.6	62.8±0.7	60.4±1.5	66.6±3.1	76.6±1.7	76.9±0.7	66.4±1.0	73.2±1.6	69.7
SimGRACE	74.4±0.3	62.6±0.7	60.2±0.9	75.5±2.0	75.4±1.3	75.0±0.6	71.2±1.1	74.9±2.0	71.2
GraphCL	75.1±0.7	63.0±0.4	59.8±1.3	77.5±3.8	76.4±0.4	75.1±0.7	67.8±2.4	74.6±2.1	71.2
GraphMAE	75.2±0.9	63.6±0.3	60.5±1.2	76.5±3.0	76.4±2.0	76.8±0.6	71.2±1.0	78.2±1.5	72.3
GraphMVP	74.9±0.8	63.1±0.2	60.2±1.1	79.1±2.8	77.7±0.6	76.0±0.1	70.8±0.5	79.3±1.5	72.6
MGSSL	75.2±0.6	63.3±0.5	61.6±1.0	77.1±4.5	77.6±0.4	75.8±0.4	68.8±0.6	78.8±0.9	72.3
AttrMask	75.1±0.9	63.3±0.6	60.5±0.9	73.5±4.3	75.8±1.0	75.3±1.5	65.2±1.4	77.8±1.8	70.8
AttrMask-DyCC	**76.6±0.5**	**64.6±0.4**	**61.3±0.6**	**79.8±3.5**	**76.7±0.9**	**77.6±1.2**	**70.5±1.0**	**82.1±2.0**	**73.7**
Mole-BERT	76.2±0.5	63.9±0.3	61.4±1.9	75.1±3.0	77.4±2.1	77.5±1.0	66.8±1.5	78.9±0.9	72.2
Mole-BERT-DyCC	**76.3±0.5**	**64.4±0.5**	**61.4±0.9**	**78.9±2.4**	**78.6±1.9**	**77.7±0.9**	**70.8±0.6**	**82.2±0.9**	**73.8**
SimSGT	75.1±0.5	63.5±0.4	61.0±0.4	79.1±2.6	76.0±0.5	76.3±0.5	70.9±0.6	82.5±0.9	73.0
SimSGT-DyCC	**76.0±0.5**	**64.6±0.4**	**61.6±0.6**	**80.5±2.2**	**77.7±0.9**	**77.3±0.8**	**71.4±0.7**	**83.4±1.0**	**74.1**

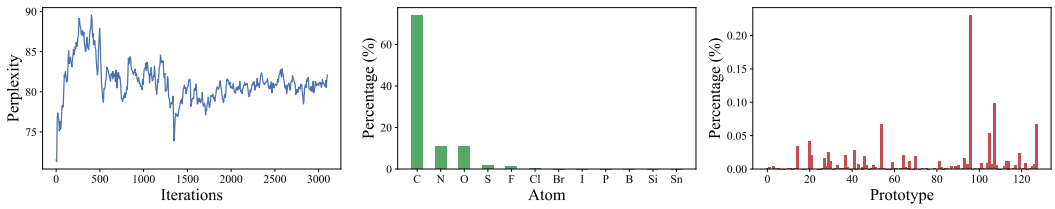

Figure 3: **(a)**: The variation of perplexity of prototypes during the training process. **(b)**: The atoms ratios of various chemical elements in the ZINC datasets. **(c)**: The distribution of 128 prototypes in the ZINC datasets after training.

4.2 Model performance

As depicted in Table 1, the incorporation of DyCC into AttrMask, MoleBert, and SimSGT significantly boosts the performance of pretraining. Particularly noteworthy is its integration into SimSGT, where it surpasses the "No pretrain" model by 10.4%, achieving a new state-of-the-art result. Moreover, DyCC demonstrates effective mitigation of the impact of tokenizers on pretraining. Previously, substantial performance variations were observed among the original AttrMask, MoleBert, and SimSGT models due to differences in tokenizers. However, following the integration of DyCC, these performance gaps narrow considerably, indicating reduced reliance on tokenizers. This can be attributed to DyCC's adaptive adjustment of the reconstruction targets, consequently diminishing dependency on tokenizers. In Appendix C.4, we verified the efficacy of DyCC across a broader spectrum of downstream tasks and datasets, including four molecular property prediction regression tasks and two Drug-Target Affinity (DTA) regression tasks [27, 28].

4.3 Detailed Analysis of GIBMS and SLG

Here, we primarily analyze the role of GIBMS and SLG, while experiments on other hyperparameters can be found in Appendix C.6.

The role of GIBMS module. To further evaluate GIBMS, we employed the MUTAG dataset, which includes 4,337 molecular graphs, each classified into one of two categories based on its mutagenic effect. As noted in GNNExplainer [45], carbon rings with chemical groups NH_2 or NO_2 are known to be mutagenic. We labeled the top 30% most important atoms identified by GIBMS, and if these

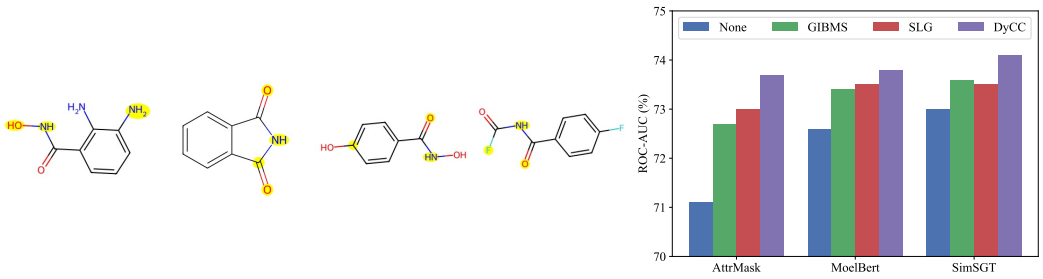

Figure 4: (Left) Core substructures (highlighted) extracted by GIBMS for four molecules. (Right) Component ablation of GIBMS and SLG.

atoms included NH_2 or NO_2, it was deemed a success. The success rate of GIBMS was 74%, demonstrating the effectiveness of the GIBMS module. In addition, we conducted a qualitative analysis of substructures based on our prior chemical knowledge. As depicted in Fig. 4, we randomly selected four molecules and utilized GIBMS to extract core substructures. The results indicate that the model tends not to focus on aromatic rings but rather tends to discover the substructures around them. This finding aligns with chemical knowledge, as aromatic rings, which contribute to the stability of molecules, are not directly related to chemical properties, whereas substructures in the side chains are more likely to contain chemical information.

The role of SLG module. SLG is proposed for dynamically adjusting task difficulty. To investigate its effectiveness, we utilize perplexity as an evaluation metric to assess the probability of different prototypes being utilized. A higher perplexity suggests a more uniform utilization of prototypes, implying increased difficulty in the reconstruction task. As depicted in Fig. 3(a), perplexity initially increases during training, then gradually decreases and converges to a stable value. This indicates that SLG enables our model to dynamically adjust the difficulty of the reconstruction task. Moreover, SLG effectively mitigates issues such as small vocabulary size and token imbalance. For instance, in the widely used ZINC15 dataset, which comprises 12 types of atoms, with 95% of the atoms distributed among the top three atom types (Fig. 3(b)), SLG allows flexible specification of the number of prototypes (determined by n), and yields a more uniform distribution of tokens, as illustrated in Fig. 3(c).

Are both GIBMS and SLG necessary. To investigate this, we separately added only one module, either GIBMS or SLG, into AttrMask, MoleBert, and SimSGT. As depicted in Fig. 4, we observed that while introducing either module alone improves the effectiveness of pretraining across all MGAEs, combining both strategies leads to better results.

5 Conclusion

We identified two significant issues when applying existing MGAEs methods to the molecular domain. On one hand, the proxy tasks are predetermined and lack the capability for dynamic adjustment during training. On the other hand, there are designs that do not align with chemical priors. To address these challenges, we propose the DyCC framework, which consists of two modules: GIBMS and SLG. The GIBMS module employs graph information bottleneck theory to identify nodes that preserve semantics during masking, enabling adaptive masking. The SLG module utilizes a set of learnable prototypes to map the hard labels of tokens to soft labels, dynamically updating these soft labels throughout the training process. This allows the reconstruction objectives to adaptively adjust as well. We integrated DyCC into various existing MGAEs, significantly enhancing pre-training performance while reducing reliance on tokenizers.

6 Acknowledgement

This paper is partially supported by the National Natural Science Foundation of China (No.12227901). The AI-driven experiments, simulations and model training were performed on the robotic AI-Scientist

platform of Chinese Academy of Sciences., Anhui Science Foundation for Distinguished Young Scholars (No.1908085J24), Natural Science Foundation of China (No.62502491).

References

[1] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. *CoRR*, abs/1612.00410, 2016.

[2] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXI*, volume 13691 of *Lecture Notes in Computer Science*, pages 456–473. Springer, 2022.

[3] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXI*, volume 13691 of *Lecture Notes in Computer Science*, pages 456–473. Springer, 2022.

[4] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Armand Joulin, Nicolas Ballas, and Michael G. Rabbat. Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 8423–8432. IEEE, 2021.

[5] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Armand Joulin, Nicolas Ballas, and Michael G. Rabbat. Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 8423–8432. IEEE, 2021.

[6] Simon Axelrod and Rafael Gómez-Bombarelli. GEOM: energy-annotated molecular conformations for property prediction and molecular generation. *CoRR*, abs/2006.05531, 2020.

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[8] K. Choudhary, B. DeCost, C. Chen, and et al. Recent advances and applications of deep learning methods in materials science. *npj Comput Mater*, 8:59, 2022.

[9] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[10] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2292–2300, 2013.

[11] Laurianne David, Amol Thakkar, Rocío Mercado, and Ola Engkvist. Molecular representations in ai-driven drug discovery: a review and practical guide. *J. Cheminformatics*, 12(1):56, 2020.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[13] Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

[14] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1024–1034, 2017.

[15] R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[16] Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In Aidong Zhang and Huzefa Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 594–604. ACM, 2022.

[17] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay S. Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[18] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. GPT-GNN: generative pre-training of graph neural networks. In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1857–1867. ACM, 2020.

[19] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

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

[21] Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.

[22] Sihang Li, Xiang Wang, An Zhang, Yingxin Wu, Xiangnan He, and Tat-Seng Chua. Let invariant rationale discovery inspire graph contrastive learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 13052–13065. PMLR, 2022.

[23] Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. Pre-training molecular graph representation with 3d geometry. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[24] Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. Pre-training molecular graph representation with 3d geometry. In *International Conference on Learning Representations*, 2022.

[25] Zhiyuan Liu, Yaorui Shi, An Zhang, Enzhi Zhang, Kenji Kawaguchi, Xiang Wang, and Tat-Seng Chua. Rethinking tokenizer and decoder in masked graph modeling for molecules. *CoRR*, abs/2310.14753, 2023.

[26] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[27] Thin Nguyen, Hang Le, Thomas P. Quinn, Tri Nguyen, Thuc Duy Le, and Svetha Venkatesh. Graphdta: predicting drug-target binding affinity with graph neural networks. *Bioinform.*, 37(8):1140–1147, 2021.

[28] Tapio Pahikkala, Antti Airola, Sami Pietilä, Sushil Kumar Shakyawar, Agnieszka Szwajda, Jing Tang, and Tero Aittokallio. Toward more realistic drug-target interaction predictions. *Briefings Bioinform.*, 16(2):325–337, 2015.

[29] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[30] Karthik Sridharan and Sham M. Kakade. An information theoretic framework for multi-view learning. In Rocco A. Servedio and Tong Zhang, editors, *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*, pages 403–414. Omnipress, 2008.

[31] Teague Sterling and John J. Irwin. ZINC 15 - ligand discovery for everyone. *J. Chem. Inf. Model.*, 55(11):2324–2337, 2015.

[32] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[33] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. Adversarial graph augmentation to improve graph contrastive learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 15920–15933, 2021.

[34] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. Adversarial graph augmentation to improve graph contrastive learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 15920–15933, 2021.

[35] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *CoRR*, abs/1906.05849, 2019.

[36] Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Self-supervised learning from a multi-view perspective. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[37] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[38] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. *CoRR*, abs/1809.10341, 2018.

[39] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay S. Pande. Moleculenet: A benchmark for molecular machine learning. *CoRR*, abs/1703.00564, 2017.

[40] Jun Xia, Lirong Wu, Jintao Chen, Bozhen Hu, and Stan Z. Li. Simgrace: A simple framework for graph contrastive learning without data augmentation. In *Proceedings of The Web Conference 2022*. Association for Computing Machinery, 2022.

[41] Jun Xia, Chengshuai Zhao, Bozhen Hu, Zhangyang Gao, Cheng Tan, Yue Liu, Siyuan Li, and Stan Z. Li. Mole-BERT: Rethinking pre-training graph neural networks for molecules. In *The Eleventh International Conference on Learning Representations*, 2023.

[42] Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *CoRR*, abs/1304.5634, 2013.

[43] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[44] Chaochao Yan, Qianggang Ding, Peilin Zhao, Shuangjia Zheng, Jinyu Yang, Yang Yu, and Junzhou Huang. Retroxpert: Decompose retrosynthesis prediction like A chemist. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[45] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9240–9251, 2019.

[46] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay S. Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6412–6422, 2018.

[47] Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5694–5703. PMLR, 2018.

[48] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12121–12132. PMLR, 2021.

[49] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[50] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[51] Junchi Yu, Jie Cao, and Ran He. Improving subgraph recognition with variational graph information bottleneck. *CoRR*, abs/2112.09899, 2021.

[52] Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Graph information bottleneck for subgraph recognition. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[53] Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Recognizing predictive substructures with subgraph information bottleneck. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(3):1650–1663, 2024.

[54] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Chee-Kong Lee. Motif-based graph self-supervised learning for molecular property prediction. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 15870–15882, 2021.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have claimed the paper's contributions and scope in abstract and introduction

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Appendix A

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The proof could be found in Appendix E

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The code and data are provided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code and data are open.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have provided the data splits, hyperparameters, etc.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have provided the statistical significance of the experiments result.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide provide sufficient information on the computer resources in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research is within the Code of Ethics

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have discussed it.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: : All of them are properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Limitations

When training the GIBMS module, we adopt the common multi-view assumption [30, 42], assuming that our self-supervised proxy task is sufficiently effective to yield a small ϵ_{info}. However, this assumption may not always hold. Despite this, we empirically observe that leveraging this assumption indeed benefits the training of GIBMS. Therefore, we proceed with this assumption.

B Details of molecular datasets

We provide detailed information of the datasets for molecular property prediction (classification and regression) and drug target affinity prediction in Table 2.

Table 2: Summary for the molecule datasets for downstream tasks.

Dataset	Task	# Tasks	# Molecules	# Proteins	# Molecule-Protein
BBBP	Classification	1	2,039	–	–
Tox21	Classification	12	7,831	–	–
ToxCast	Classification	617	8,576	–	–
Sider	Classification	27	1,427	–	–
ClinTox	Classification	2	1,478	–	–
MUV	Classification	17	93,087	–	–
HIV	Classification	1	41,127	–	–
Bace	Classification	1	1,513	–	–
Delaney	Regression	1	1,128	–	–
Lipo	Regression	1	4,200	–	–
Malaria	Regression	1	9,999	–	–
CEP	Regression	1	29,978	–	–
Davis	Regression	1	68	379	30056
KIBA	Regression	1	2,068	229	118,254

Molecule representations. For simplicity, we use a minimal set of node and bond features that unambiguously describe the two-dimensional structure of molecules following previous works. We use RDKit o obtain these features, as show in Table 3 and Table 4.

Table 3: Atom features.

features	size	description
atom type	100	type of atom (e.g., C, N, O), by atomic number
formal charge	5	integer electronic charge assigned to atom
number of bonds	6	number of bonds the atom is involved in
chirality	5	number of bonded hydrogen atoms
number of H	5	number of bonded hydrogen atoms
atomic mass	1	mass of the atom, divided by 100
aromaticity	1	whether this atom is part of an aromatic system
hybridization	5	sp, sp2, sp3, sp3d, or sp3d2

Table 4: Bond features.

features	size	description
bond type	4	single, double, triple, or aromatic
stereo	6	none, any, E/Z or cis/trans
in ring	1	whether the bond is part of a ring
conjugated	1	whether the bond is conjugated

Dataset Splitting. We apply the scaffold splitting for all tasks on all datasets. It splits the molecules with distinct two-dimensional structural frameworks into different subsets. It is a more challenging but practical setting since the test molecular can be structurally different from training set. Here we apply the scaffold splitting to construct the train/validation/test sets.

C Experimental Details

C.1 Computational resources

Our experiments are conducted using an NVIDIA DGX A100 server. Each experiment can be executed on a single GPU while staying within the limit of 30 GB of GPU memory consumption.

C.2 Implementation and pretraining Details

We used the official source code provided by AttrMask, MoleBert, and SimSGT, retaining the exact same settings. Building upon this foundation, we introduced the GIBMS and SLG modules. The three additional hyperparameters for GIBMS were set to $t = 1$, $\beta = 0.01$, and $\tau = 0.1$, respectively. The four additional hyperparameters for SLG were set to $\tau_y = 0.1$, $\tau_p = 0.25$, $\alpha = 1$, and $n = 128$.

C.3 Baselines

We now describe the details of our reported baseline methods:

- **InfoGraph** [32] conducts graph representation learning by maximizing the mutual information between graph-level representations and local substructures of various scales.
- **GPT-GNN** introduces a self-supervised attributed graph generation task to pre-train a GNN so that it can capture the structural and semantic properties of the graph. They factorize the likelihood of the graph generation into two components: 1) Attribute Generation and 2) Edge Generation.
- **ContextPred** [17] uses the embeddings of subgraphs to predict their context graph structures.
- **GraphLOG** [33] leverages clustering to construct hierarchical prototypes of graph samples. They further contrast each local instance with its corresponding higher prototype for contrastive learning.
- **Infomax** [38] learns node representations by maximizing the mutual information between the local summaries of node patches and the patches' graph-level global summaries.
- **G-Contextual** [29] views the prediction problem as a multi-class prediction task, where each class corresponds to one contextual property.
- **G-Motif** [29] formulates the prediction task as a multi-label classification problem, where each motif corresponds to one label.
- **AD-GCL** [34] applies adversarial learning for adaptive graph augmentation to remove the redundant information in graph samples.
- **JOAO** [48] proposes a framework to automatically search proper data augmentations for GCL.
- **SimGRACE** [40] take original graph as input and GNN model with its perturbed version as two encoders to obtain two correlated views for contrast. SimGRACE is inspired by the observation that graph data can preserve their semantics well during encoder perturbations while not requiring manual trial-and-errors, cumbersome search or expensive domain knowledge for augmentations selection.
- **GraphCL** [49] performs graph-level contrastive learning with combinations of four graph augmentations, namely node dropping, edge perturbation, subgraph cropping, and feature masking.
- **GraphMAE** [16] shows that a linear classifier is insufficient for decoding node types. It applies a GNN for decoding and proposes remask to decouple the functions of the encoder and decoder in the autoencoder.

Table 5: Transfer learning performance for molecular property prediction (regression) and drug target affinity (regression). **Bold** indicates the best performance.

	Molecular Property Prediction (RMSE ↓)					Drug-Target Affinity (MSE ↓)		
	ESOL	Lipo	Malaria	CEP	Avg.	Davis	KIBA	Avg.
No Pre-train	1.178±0.044	0.744±0.007	1.127±0.003	1.254±0.030	1.076	0.286±0.006	0.206±0.004	0.246
ContextPred	1.196±0.037	0.702±0.020	1.101±0.015	1.243±0.025	1.061	0.279±0.002	0.198±0.004	0.238
AttrMask	1.112±0.048	0.730±0.004	1.119±0.014	1.256±0.000	1.054	0.291±0.007	0.203±0.003	0.248
JOAO	1.120±0.019	0.708±0.007	1.145±0.010	1.293±0.003	1.066	0.281±0.004	0.196±0.005	0.239
GraphMVP	1.064±0.045	0.691±0.013	1.106±0.013	1.228±0.001	1.022	0.274±0.002	0.175±0.001	0.225
Mole-BERT	1.192±0.028	0.706±0.008	1.117±0.008	1.078±0.002	1.024	0.277±0.004	0.210±0.003	0.243
SimSGT-G	1.039±0.012	**0.670±0.015**	1.090±0.013	1.060±0.011	0.965	0.263±0.006	0.144±0.001	0.204
SimSGT-G-DyCC	**0.988±0.023**	0.672±0.016	**1.082±0.012**	**1.035±0.012**	**0.944**	**0.256±0.003**	**0.140±0.001**	**0.198**

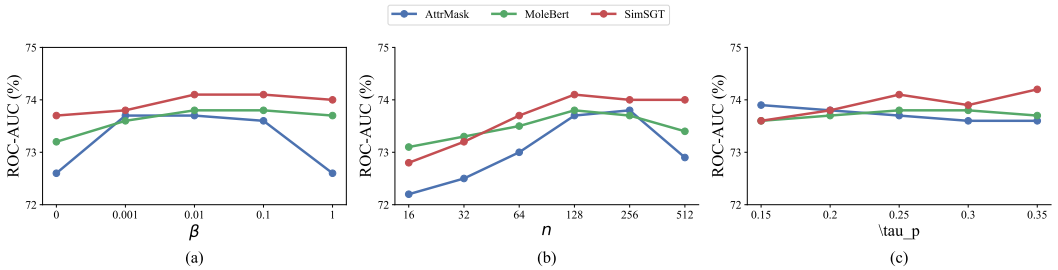

Figure 5: The impact of three hyperparameters β, n, and τ_p.

- **GraphMVP** [24] uses a contrastive loss and a generative loss to connect the 2-dimensional view and 3-dimensional view of the same molecule, in order to inject the 3-dimensional knowledge into the 2-dimensional graph encoder.
- **MGSSL** [54] introdue a novel self-supervised motif generation framework for GNNs. First, for motif extraction from molecular graphs, they design a molecule fragmentation method that leverages a retrosynthesis-based algorithm BRICS and additional rules for controlling the size of motif vocabulary. Second, they design a general motif-based generative pretraining framework in which GNNs are asked to make topological and label predictions.
- **RGCL** [22] trains a rationale generator to identify the causal subgraph in graph augmentation. Each graph's causal subgraph and its complement are leveraged in contrastive learning.
- **Mole-BERT** [41] combines a contrastive learning objective and a masked atom modeling objective for MRL. Specifically, they observe that mask atom prediction is an overly easy pretraining task. Therefore, they employ a GNN tokenizer pretrained by VQ-VAEto generate more complex reconstruction targets for masked atom modeling.

C.4 Broader Range of Downstream Tasks

We verified the efficacy of DyCC across a broader spectrum of downstream tasks and datasets, including four molecular property prediction regression tasks and two Drug-Target Affinity (DTA) regression tasks [27, 28]. DTA aims to predict the affinity scores between molecular drugs and target proteins. Following prior work [23], we pretrain SimSGT-DyCC on 50 thousand molecule samples from the GEOM dataset [6] and report the mean performances and standard deviations across three random seeds. We report the RMSE for the molecular property prediction datasets with scaffold splitting and report the MSE for the DTA datasets with random splitting. The results are summarized in Table 5. It is evident that SimSGT-DyCC surpasses the original version of SimSGT, achieving significant improvement over other baseline models. This suggests that DyCC can effectively enhance performance across a wider spectrum of downstream tasks.

C.5 Additional experimental results

Regression tasks We verified the efficacy of DyCC across a broader spectrum of downstream tasks and datasets, including four molecular property prediction regression tasks and two Drug-Target

Table 6: Transfer learning ROC-AUC (%) scores on eight MoleculeNet datasets.The suffix "DyCC" implies the introduction of both the GIBMS and SLG modules.

Dataset	Tox21	ToxCast	Sider	ClinTox	MUV	HIV	BBBP	Bace	Avg.
No Pretrain	74.6±0.4	61.7±0.5	58.2±1.7	58.4±6.4	70.7 ±1.8	75.5±0.8	65.7±3.3	72.4±3.8	67.0
Mole-BERT	76.2±0.5	63.9±0.3	61.4±1.9	75.1±3.0	77.4±2.1	77.5±1.0	66.8±1.5	78.9±0.9	72.2
Mole-BERT DyCC	76.3±0.5	64.4±0.5	61.4±0.9	78.9±2.4	78.6±1.9	77.7±0.9	70.8±0.6	82.2±0.9	73.8
Mole-BERT DyCC + TMCL	76.6±0.4	64.8±0.5	61.8±0.8	78.6±2.2	78.8 ±1.8	77.9±0.8	71.3±0.7	82.8±0.9	74.1
SimSGT	75.1±0.5	63.5±0.4	61.0±0.4	79.1±2.6	76.0±0.5	76.3±0.5	82.5±0.9	70.9±0.6	73.0
SimSGT DyCC	76.0±0.5	64.6±0.4	61.6±0.6	80.5±2.2	77.7±0.9	77.3±0.8	71.4±0.7	83.4±1.0	74.1
SimSGT DyCC GraphTrans	76.6 ±0.6	66.3±0.8	62.0±1.2	83.6 ±2.1	80.3±2.2	77.8±1.7	84.9 ±1.0	72.2±0.8	75.5

Table 7: Mean Average Error (MAE) performanceon the QM datasets.

	QM7	QM8	QM9
#Tasks	1	12	12
GraphCL	80.4±3.3	0.0200±0.0004	5.76±0.37
GraphMAE	78.4±2.3	0.0190±0.0003	5.84±0.16
Mole-BERT	79.8±2.6	0.0190±0.0003	5.75±0.16
Mole-BERT DyCC	78.7±2.2	0.0188±0.0003	5.70 ±0.16
SimSGT	78.8±2.2	0.0189±0.0004	5.73±0.18
SimSGT DyCC	**77.6±1.8**	**0.0180±0.0003**	**5.60±0.21**

Affinity (DTA) regression tasks [27, 28]. DTA aims to predict the affinity scores between molecular drugs and target proteins. Following prior work [23], we pretrain SimSGT-DyCC on 50 thousand molecule samples from the GEOM dataset [6] and report the mean performances and standard deviations across three random seeds. We report the RMSE for the molecular property prediction datasets with scaffold splitting and report the MSE for the DTA datasets with random splitting. The results are summarized in Table 5. It is evident that SimSGT-DyCC surpasses the original version of SimSGT, achieving significant improvement over other baseline models. This suggests that DyCC can effectively enhance performance across a wider spectrum of downstream tasks.

Additional Modules of SimSGT and MoleBert In the main paper, for fair comparison, we excluded the GraphTrans variant of SimSGT and the TMCL proxy task of MoleBert. We provide the complete version in Table 6, and the results indicate that restoring these strategies further improves the model's performance. This suggests that DyCC can work in conjunction with other enhancements to MGAEs.

Quantum chemistry property prediction. We report performances of predicting the quantum chemistry properties of molecules. We divide the downstream datasets by scaffold split. Specifically, we attach a two-layer MLP after the pretrained molecule encoders and fine-tune the models for property prediction. We report average performances and standard deviations across 10 random seeds. The performances are reported in Table 7. We observe a consistent enhancement of pre-trained model performance by DyCC.

C.6 Hyperparameter experiments

Here, we explore several crucial hyperparameters of DyCC. The first one is β, which controls the trade-off between prediction and compression in our final objectives. As shown in Fig 5(a), there exists an optimal point at $\beta = 0.01$ in terms of model performance, indicating the trade-off between prediction and information compression. Setting a larger $\beta = 1$ encourages aggressive information compression, leading to difficulties in capturing the core subgraph related to the target task. Conversely, decreasing β encourages the model to retain the original information of the given graph structure. In an extreme case, i.e., when $\beta = 0$, the model focuses solely on the prediction term, potentially leading to a lack of generalization ability. The second parameter is the number of prototypes n. We find that as n increases from 16 to 128, the pretraining performance gradually improves, while using more prototypes has little effect. Therefore, $n = 128$ is a suitable choice. Lastly, τ_y and τ_p control the soft label sharpening level. We always choose $\tau_y < \tau_p$ to encourage

sharper target predictions, implicitly guiding the model to produce confident low-entropy predictions. We fix $\tau_y = 0.1$ and set τ_p to $\{0.15, 0.2, 0.25, 0.3\}$. Experimental results show that different models have different optimal values of τ_p, which may be due to the different tokenizer types.

D Algorithm for GIBMS and SLT

Algorithm 1 and Algorithm 2 provide a comprehensive description of the GIBMS algorithm and the SLG process, respectively.

Algorithm 1 The training process of GIBMS

Input: Unlabeled molecular pre-trained graph dataset $\mathcal{D}_1 = \{\mathcal{G}_1, \mathcal{G}_2, \cdots\}$, GNN encoder Φ, and node importance evaluation MLP $\mathcal{M}$.
Initialize parameters of Φ and $\mathcal{M}$.
for each graph $\mathcal{G}$ in $\mathcal{D}_1$ **do**
  Encode $\mathcal{G}$ into node representations: $\mathbf{H} = \Phi(\mathcal{G})$.
  Generate a sampling probability for each node: $p = \mathcal{M}(\mathbf{H})$.
  Apply the Gumbel-Sigmoid function to sample λ from p based on Eq. (7).
  Inject noise into $\mathbf{H}$ to obtain $\hat{\mathbf{H}}$ based on Eq. (6),
  Compute the unsupervised prediction loss based on Eq. (11).
  Compute the loss for the compression term based on Eq. (12).
  Calculate the total loss for the first stage based on Eq. (13).
  Perform backpropagation to optimize the training objective.
end for
Return: The well trained Φ and $\mathcal{M}$ jointly constitute the GIBMS module $\mathcal{M}(\Phi(\mathcal{G}))$.

Algorithm 2 Soft Label Generator

Input: Unlabeled molecular pre-trained graph dataset $\mathcal{D}_1 = \{\mathcal{G}_1, \mathcal{G}_2, \cdots\}$, the pre-trained GIBMS model $P(\Phi(\mathcal{G}))$, GNN encoder Φ' for MGAEs, and learnable prototypes matrix $\mathbf{Q}$.
Initialize the parameters of Φ' and $\mathbf{Q}$.
for each graph $\mathcal{G}$ in $\mathcal{D}_1$ **do**
  Compute the sampling probability for each node as $p = \mathrm{Sigmoid}(\mathcal{M}(\mathbf{H}))$.
  Sample a set of important nodes $V_{\text{mask}} = \{V_i \mid V_i \sim \text{Bernoulli}(1 - p_i), i = 1, 2, \ldots, N\}$.
  Replace the nodes in V_{mask} within graph $\mathcal{G}$ with a MASK token to obtain $\mathcal{G}_{\text{mask}}$.
  Obtain the node representations $\mathbf{H} = \Phi_2(\mathcal{G}_{\text{mask}})$.
  Compute the soft label assignments s^p for all nodes by applying Eq. (15) to $\mathbf{H}$ and $\mathbf{Q}$.
  Compute the soft label assignments s_y for all nodes by applying Eq. (14) to the node labels y and $\mathbf{Q}$.
  Minimize the distance between s_y and s^p according to Eq. (17).
  Perform backpropagation to optimize the training objective.
end for
Return: The pre-trained GNN encoder Φ_2 for various downstream tasks.

E Proof

E.1 Proof of Eq. (10)

By adopting the common multi-view assumption [30, 42], we have:

$$\begin{aligned} I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) &= I\left(\mathbf{Z}^{\text{sup}}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right) \\ &\geq I\left(\mathbf{Z}^{\text{ssl}}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right) \\ &\geq I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) - \epsilon_{\text{info}}; \quad \epsilon_{\text{info}} > 0 \end{aligned}$$

The proofs contain two parts [36]. The first one is showing the results for the supervised learned representations and the second one is for the self-supervised learned representations.

Lemma 1 (Determinism) If $P\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}} \mid \mathcal{G}_{\text{IB}}\right)$ is Dirac, then the following conditional independence holds: $\mathbf{Y}^{\text{sup}} \perp \mathbf{Z}_{\mathcal{G}_{\text{IB}}} \mid \mathcal{G}_{\text{IB}}$ and $S \perp \mathbf{Z}_{\mathcal{G}_{\text{IB}}} \mid \mathcal{G}_{\text{IB}}$, inducing a Markov chain $\mathbf{S} \leftrightarrow \mathbf{Y}^{\text{sup}} \leftrightarrow \mathcal{G}_{\text{IB}} \rightarrow$

$\mathbf{Z}_{\mathcal{G}_{\text{IB}}}$. When $\mathbf{Z}_{\mathcal{G}_{\text{IB}}}$ is a deterministic function of $\mathcal{G}_{\text{IB}}$, for any A in the sigma-algebra induced by $\mathbf{Z}_{\mathcal{G}_{\text{IB}}}$ we have $\mathbb{E}\left[\mathbf{1}_{\left[\mathbf{Z}_{\mathcal{G}_{\text{IB}}} \in A\right]} \mid \mathcal{G}_{\text{IB}}, \{\mathbf{Y}^{\text{sup}}, \mathbf{S}\}\right] = \mathbb{E}\left[\mathbf{1}_{\left[\mathbf{Z}_{\mathcal{G}_{\text{IB}}} \in A\right]} \mid \mathcal{G}_{\text{IB}}, \mathbf{S}\right] = \mathbb{E}\left[\mathbf{1}_{\left[\mathbf{Z}_{\mathcal{G}_{\text{IB}}} \in A\right]} \mid \mathcal{G}_{\text{IB}}\right]$, which implies $\mathbf{Y}^{\text{sup}} \perp \mathbf{Z}_{\mathcal{G}_{\text{IB}}} \mid \mathcal{G}_{\text{IB}}$ and $\mathbf{S} \perp \mathbf{Z}_{\mathcal{G}_{\text{IB}}} \mid G_{\text{IB}}$.

Supervised Learned Representations Adopting Data Processing Inequality [9] in the Markov chain $\mathbf{S} \leftrightarrow \mathbf{Y}^{\text{sup}} \leftrightarrow \mathcal{G}_{\text{IB}} \rightarrow \mathbf{Z}_{\mathcal{G}_{\text{IB}}}$, $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right)$ is maximized at $I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}})$. Since the supervised learned representations $\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{sup}}$ maximize$I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right)$, we conclude $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{sup}}; \mathbf{Y}^{\text{sup}}\right) = I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}})$.

Self-supervised Learned Representations First, we have

$$\begin{aligned} I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S}\right) &= I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}\right) \\ &\quad - I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}\right) + I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S} \mid T\right) \\ &= I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{Y}^{\text{sup}}; \mathbf{S}\right) + I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S} \mid \mathbf{Y}^{\text{sup}}\right) \end{aligned}$$

and

$$\begin{aligned} I(\mathcal{G}_{\text{IB}}; \mathbf{S}) &= I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) \\ &\quad - I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}) + I(\mathcal{G}_{\text{IB}}; \mathbf{S} \mid \mathbf{Y}^{\text{sup}}) \\ &= I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}; \mathbf{S}) + I(\mathcal{G}_{\text{IB}}; \mathbf{S} \mid \mathbf{Y}^{\text{sup}}) \end{aligned}$$

By DPI in the Markov chain $\mathbf{S} \leftrightarrow \mathbf{Y}^{\text{sup}} \leftrightarrow \mathcal{G}_{\text{IB}} \rightarrow \mathbf{Z}_{\mathcal{G}_{\text{IB}}}$, we know

- $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S}\right)$ is maximized at $I(\mathcal{G}_{\text{IB}}; \mathbf{S})$
- $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S}; \mathbf{Y}^{\text{sup}}\right)$ is maximized at $I(\mathcal{G}_{\text{IB}}; \mathbf{S}; \mathbf{Y}^{\text{sup}})$
- $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S} \mid \mathbf{Y}^{\text{sup}}\right)$ is maximized at $I(\mathcal{G}_{\text{IB}}; \mathbf{S} \mid \mathbf{Y}^{\text{sup}})$

Since the self-supervised learned representations $\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}$ maximize $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}; \mathbf{S}\right)$, we have $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{S}\right) = I(\mathcal{G}_{\text{IB}}; \mathbf{S})$. Hence $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{S} \mid \mathbf{Y}^{\text{sup}}\right) = I(\mathcal{G}_{\text{IB}}; \mathbf{S} \mid \mathbf{Y}^{\text{sup}})$. Using the result $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{S}; \mathbf{Y}^{\text{sup}}\right) = I(\mathcal{G}_{\text{IB}}; \mathbf{S}; \mathbf{Y}^{\text{sup}})$, we get

$$\begin{aligned} I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{Y}^{\text{sup}}\right) &= I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) \\ &\quad - I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}) \\ &\quad + I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}\right) \end{aligned}$$

Now, we are ready to present the inequalities:

$I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) \geq I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{Y}^{\text{sup}}\right)$ due to $I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}) \geq I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}\right)$ by DPI and $I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{Y}^{\text{sup}}\right) \geq I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) - \epsilon_{\text{info}}$ due to

$$\begin{aligned} &I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) - I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}) + I\left(\mathbf{Z}_{\mathcal{G}_{\text{IB}}}^{\text{ssl}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}\right) \\ &\geq I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) \\ &\geq I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}}) - \epsilon_{\text{info}} \end{aligned} \tag{18}$$

where $I(\mathcal{G}_{\text{IB}}; \mathbf{Y}^{\text{sup}} \mid \mathbf{S}) \leq \epsilon_{\text{info}}$ by the redundancy assumption.

E.2 Proof of Eq. (12)

We derive the upper bound of $I\left(\mathcal{G}; \mathcal{G}_{\text{IB}}\right)$ by introducing the variation approximation $q\left(\mathcal{G}_{\text{IB}}\right)$ of distribution $p\left(\mathcal{G}_{\text{IB}}\right)$:

$$\begin{aligned} I\left(\mathcal{G}; \mathcal{G}_{\text{IB}}\right) &= \mathbb{E}_{\mathcal{G}, \mathcal{G}_{\text{IB}}}\left[\log \frac{p_\phi\left(\mathcal{G}_{\text{IB}} \mid \mathcal{G}\right)}{p(\mathcal{G})}\right] \\ &= \mathbb{E}_{\mathcal{G}, \mathcal{G}_{\text{IB}}}\left[\log \frac{p_\phi\left(\mathcal{G}_{\text{IB}} \mid \mathcal{G}\right)}{q(\mathcal{G}_{\text{IB}})}\right] \\ &\quad - \mathbb{E}_{\mathcal{G}_{\text{IB}}, \mathcal{G}}\left[KL\left(p\left(\mathcal{G}\right)\right) \| q\left(\mathcal{G}_{\text{IB}}\right)\right)] \end{aligned}$$

According to the non-negativity of KL divergence, we have:

$$I\left(\mathcal{G}_{\text{IB}}, \mathcal{G}\right) \leq \mathbb{E}_{\mathcal{G}}\left[KL\left(p_\phi\left(\left(\mathcal{G}_{\text{IB}} \mid \mathcal{G}\right) \| q\left(\mathcal{G}_{\text{IB}}\right)\right)\right)\right]$$

We assume that $q\left(\mathcal{G}_{\text{IB}}\right)$ is obtained by aggregating the node representations in a fully perturbed graph. The noise $\epsilon \sim \mathcal{N}\left(\mu_{\mathbf{H}}, \sigma_{\mathbf{H}}^2\right)$ is sampled from a Gaussian distribution where $\mu_{\mathbf{H}}$ and $\sigma_{\mathbf{H}}^2$ are mean and variance of $\mathbf{H}$. Choosing sum pooling as the aggregatiion function, since the summation of Gaussian distributions is a Gaussian, we have the following form:

$$q\left(\mathcal{G}_{\text{IB}}\right) = \mathcal{N}\left(N\mu_{\mathbf{H}}, N\sigma_{\mathbf{H}}^2\right)$$

Then for $p_\phi\left(\mathcal{G}_{\text{IB}} \mid \mathcal{G}\right)$, we have the following equation:

$$\mathcal{N}\left(N\mu_{\mathbf{H}} + \sum_{j=1}^{N}\lambda_j\mathbf{H}_j - \sum_{j=1}^{N}\lambda_j\mu_{\mathbf{H}}, \sum_{j=1}^{N}\left(1-\lambda_j\right)^2\sigma_{\mathbf{H}}^2\right)$$

Finally, we have following inequality:

$$I\left(\mathcal{G}_{\text{IB}}, \mathcal{G}\right) \leq \mathbb{E}_{\mathcal{G}}\left[-\frac{1}{2}\log A + \frac{1}{2N}A + \frac{1}{2N}B^2\right] + C$$

where $A = \sum_{j=1}^{N}\left(1-\lambda_j\right)^2$ and $B = \frac{\sum_{j=1}^{N}\lambda_j(\mathbf{H}_j - \mu_{\mathbf{H}})}{\sigma_{\mathbf{H}}}$.