Delayed Bottlenecking: Alleviating Forgetting in Pre-trained Graph Neural Networks

Zhe Zhao, Pengkun Wang, *Member, IEEE*, Xu Wang, Haibin Wen, Xiaolong Xie, Zhengyang Zhou, *Member, IEEE*, Qingfu Zhang, *Fellow, IEEE*, and Yang Wang, *Senior Member, IEEE*

***Abstract*—Pre-training GNNs to extract transferable knowledge and apply it to downstream tasks has become the de facto standard of graph representation learning. Recent works focused on designing self-supervised pre-training tasks to extract useful and universal transferable knowledge from large-scale unlabeled data. However, they have to face an inevitable question: traditional pre-training strategies that aim at extracting useful information about pre-training tasks, may not extract all useful information about the downstream task. In this paper, we reexamine the pre-training process within traditional pre-training and fine-tuning frameworks from the perspective of Information Bottleneck (IB) and confirm that the forgetting phenomenon in pre-training phase may cause detrimental effects on downstream tasks. Therefore, we propose a novel Delayed Bottlenecking Pre-training (DBP) framework which maintains as much as possible mutual information between latent representations and training data during pre-training phase by suppressing the compression operation and delays the compression operation to fine-tuning phase to make sure the compression can be guided with labeled fine-tuning data and downstream tasks. To achieve this, we design two information control objectives that can be directly optimized and further integrate them into the actual model design. Extensive experiments on both chemistry and biology domains demonstrate the effectiveness of DBP.**

***Index Terms*—Pre-training, graph neural networks, information bottleneck, forget.**

Received 23 April 2024; revised 29 September 2024; accepted 25 November 2024. Date of publication 12 December 2024; date of current version 5 February 2025. This work was supported in part by the Natural Science Foundation of China Youth Project under Grant 62402472, in part by the Natural Science Foundation of Jiangsu Province of China Youth Project under Grant BK20240461, and Grant BK20240460, in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant CityU 11215723, in part by the National Natural Science Foundation of China under Grant 62072427, and Grant 12227901, in part by the Project of Stable Support for Youth Team in Basic Research Field, CAS, under Grant YSBR-005, in part by the Academic Leaders Cultivation Program, USTC, and in part by the the Key Basic Research Foundation of Shenzhen, China under Grant JCYJ20220818100005011. Recommended for acceptance by X. Yi. *(Corresponding authors: Yang Wang; Pengkun Wang.)*

Zhe Zhao, Pengkun Wang, Xu Wang, Zhengyang Zhou, and Yang Wang are with the University of Science and Technology of China (USTC), Heifei 230022, China (e-mail: zz4543@mail.ustc.edu.cn; pengkun@ustc.edu.cn; wx309@ustc.edu.cn; zzy0929@ustc.edu.cn; angyan@ustc.edu.cn).

Haibin Wen is with the Shaoguan University, Shaoguan 512158, China (e-mail: haibin65535@gmail.com).

Xiaolong Xie is with the Nanchang University, Nanchang 330047, China (e-mail: 416100210092@email.ncu.edu.cn).

Qingfu Zhang is with the Department of Computer Science, City University of Hong Kong, Hong Kong, and also with the Shenzhen Research Institute, City University of Hong Kong, Shenzhen 518057, China (e-mail: qingfu.zhang@cityu.edu.hk).

The code is available in https://anonymous.4open.science/r/TKDE-DBP.

Digital Object Identifier 10.1109/TKDE.2024.3516192

I. Introduction

In RECENT years, Graph Neural Networks (GNNs) have shown prominent performances in various fields including social networking [1], [2], [3], [4], [5], molecular computing [6], [7], [8], [9], [10], web recommendation [9], [11], [12], [13], [14], [15], [16], and bioinformatics [17], [18], [19]. Meanwhile, pre-training GNN, which is capable of enhancing the performance of GNN on specific-data-required downstream tasks by extracting universal transferable knowledge from large-scale unlabeled graph-structured data, has also attracted the great attention of both academic and industrial communities [20], [21], [22].

Great efforts [23], [24], [25], [26], [27], [28], [29] have been studied in the field of pre-training GNN to achieve knowledge extraction, and existing works can be roughly distinguished into two categories, contrastive self-supervised learning [26], [27], [28], [29] and generative self-supervised learning [23], [24], [25]. The previous one aims at learning knowledge in different semantic levels by contrasting the enhanced views of different data, while the latter one tries to recover and generate graph structure data to eventually learn the property patterns of vertexes and edges within the graph structure [30], [31]. In summary, all these methods have paid all their attention to the design of the self-supervised pre-training task to extract useful information with regard to the pre-training task from large-scale unlabeled datadata [26], [32], [33], [34], [35], [36]. However, considering the difference between the pre-training task and downstream tasks, we have to face an inevitable question: *can the pre-training process transfer all useful information to the downstream task from large-scale unlabeled data?*

To answer this question, we need to re-examine the pre-training process within the traditional pre-training and fine-tuning framework from the perspective of information extraction. Some previous work indicates that, given a specific learning task, animals choose to forget some remembered behaviors to better adapt to some specific tasks [37], [38], [39], [40]. Meanwhile, [41], [42], [43], [44] also indicate that this kind of biological forgetting phenomenon can also be found during the training process of neural networks. As illustrated in Fig. 1(a), the neural network quickly learns information from data during the first phase and compresses the representation by forgetting some learned information which is useless to the pre-training task in the second phase. According to [42], such forgetting behavior is to better fit the target of the pre-training task. Nevertheless, considering the pre-training task which is

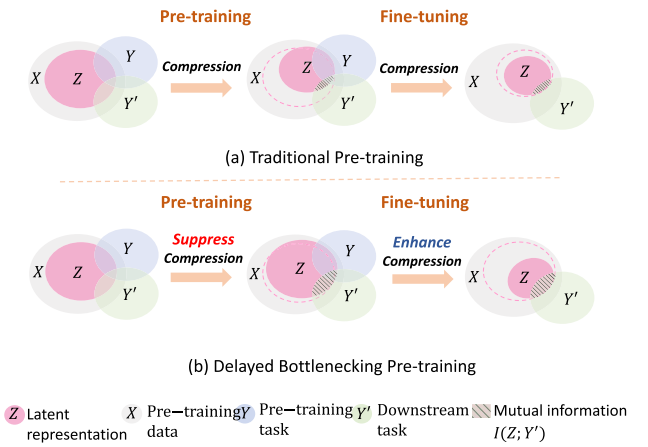

Fig. 1. Information-theoretic analysis of conventional and delayed bottlenecking pre-training in graph neural networks. Subfigure (a) presents the dynamics of information encoding in latent space during conventional pre-training, denoted as Z, relative to the pre-training data X and associated task Y, and its subsequent impact on downstream task Y'. In this regime, the latent representation Z undergoes a compression process, optimized for Y, which inadvertently discards non-salient features for Y but may be pertinent to Y', thereby diminishing the mutual information $I(Z;Y')$ post-compression. Subfigure (b) depicts an alternative approach with the proposed Delayed Bottlenecking Pre-Training, where the compression of Z during the pre-training phase is deliberately modulated. This control preserves a broader set of features in Z, allowing for enhanced mutual information $I(Z;Y')$ post-fine-tuning, which is refined under the guidance of labeled data specific to Y'.

artificially designed to extract universal transferable information from unlabeled data and is totally different from the downstream task in general [45], [46], such forgetting behavior is harmful to the learning and transformation of universal knowledge since those dropped information may be useful and of significance to downstream tasks.

Challenges: To solve the above-mentioned deficiencies of traditional pre-training GNNs, there is a key challenge needs to be addressed: *How to improve existing pre-training and fine-tuning strategy to make sure that the useful information with regard to a downstream task can be maintained as much as possible?*

To address these challenges, as demonstrated in Fig. 1(b), we propose a novel **D**elayed **B**ottlenecking **P**re-training (DBP) framework to address the issue of information forgetting during pre-training. In particular, as illustrated in Fig. 1(b), we first re-analyze the whole procedure of pre-training from the perspective of IB, and formulate the information dropping during pre-training. Based on this, we first design a novel information compression delayed pre-training strategy that maintains as much as possible mutual information between latent representations and training data during the pre-training phase by suppressing the compression operation. Then, we delay the compression operation to fine-tuning phase to make sure the compression can be guided with labeled fine-tuning data and downstream tasks. In the pre-training phase, DBP includes a newly designed information-based representation reconstruction which can maintain the mutual information between latent representation and training data by decoding the learned latent representation into the features of vertexes and edges. In the fine-tuning phase, we borrow the core idea of Depth Variational Information Bottleneck (DVIB) [42] and extend it to be adapted to graph-structured data, hence making sure that the delayed information compression is optimized with the guidance of labeled fine-tuning data and downstream task. Extensive experiments on both chemistry and biology domains verify the effectiveness of our proposed strategy on various pre-training GNNs.

The main contributions are summarized as follows:

- *New theoretical analysis:* For the first time, we analyze the information forgetting of the compression operation in pre-training GNNs from the perspective of the Information Bottleneck (IB) theory. To our knowledge, this is the first paper that aims at alleviating the influence of such kind of inevitable information forgetting on downstream tasks.
- *Novel framework and methods:* We propose a DBP framework that includes a novel information compression delayed pre-training strategy to enhance the performances of pre-training GNNs. In DBP, we propose a novel information-based representation reconstruction and an extended Graph-DVIB to respectively achieve information maintaining in the pre-training phase, and labeled fine-tuning data and downstream task-guided information compression in the fine-tuning phase.
- *Extensive empirical evaluation:* Extensive experiments on both chemistry and biology domains demonstrate the effectiveness of our proposed framework while incorporating different pre-training GNNs.

II. Related Work

Pre-training GNNs: Recently, Pre-training GNNs have received significant attention since they can alleviate the heavy reliance of traditional GNNs on data with fine-grained labels. Generally, pre-training GNNs usually consist of two phases: i) *Pre-training*: learning model parameters and node embeddings from large-scale unlabeled graph data; ii) *Fine-tuning*: fine-tuning the learned parameters and embeddings with labeled graph data to make the network more applicable to downstream tasks. Existing methods, which mostly follow such a two-phase framework, can be roughly divided into two categories: contrastive pre-training [26], [27], [47] and generative pre-training [23], [24]. Contrastive pre-training learns graph representation by contrasting the semantic differences between pre-define positive and negative samples. In particular, DGI [47] focuses on the correspondences between nodes and subgraphs, GraphLoG [27] pays attention to correlogram and subgraph pairs, and GraphCL [27] contrasts subgraph level with different data augmentations. On the other hand, generative pre-training captures the intrinsic dependencies between node attributes and graph structure by generating node attributes and edges, e.g., jointly generating nodes and edges [23] or reconstructing shadowed nodes [24]. However, all these methods paid all their attention to designing self-supervised tasks to maximally extract useful information with regard to the pre-training task from large-scale unlabeled data, ignoring the issue that the knowledge extracted by the pre-training task cannot be completely transferred to the downstream task due to the differences between pre-training and downstream tasks.

Mutual Information and Its Application: Mutual information (MI) is a measure of the degree of interdependence between random variables based on Shannon entropy. It is often used to measure the nonlinear correlation between variables, so it can be regarded as a measure of the true dependence between variables. For two random variables X and Y, the mutual information between them is as follows:

$$I(X;Y) = H(X) - H(X|Y) \tag{1}$$

Here, $H(X)$ is the information entropy of X, and $H(X|Y)$ is the conditional entropy of random variable X under the condition of known random variable Y. From a probabilistic perspective, mutual information is derived from the joint probability distribution $p(x, y)$ and the marginal probability distribution $p(x)$ and $p(y)$ of the random variables X and Y. The dependence between X and Y is stronger when the divergence between the joint probability distribution $p(x, y)$ and the marginal product $p(x)p(y)$ is larger. It is widely used in deep learning because it can measure the real dependencies between variables.

However, since the true distribution in neural networks is difficult to know, the calculation and optimization of mutual information is a difficult problem. In graph learning, many studies on optimizing neural networks through information theory choose MINE [47] or variational methods [42] to approximate the upper and lower bounds of mutual information to achieve the goal of optimizing mutual information. Many self-supervised learning methods on graphs also utilize mutual information. For example, DGI [47] relies on maximizing mutual information between patch representations and corresponding high-level graph summaries to learn node representations in graph-structured data. GraphMVP [48] performs self-supervised learning by optimizing the mutual information between molecular 2D topology and 3D geometric views to improve correspondence and consistency between these views. Besides representation learning and self-supervised learning, mutual information has also been used in the study of neural network interpretability and training dynamics. Typically, [42] investigates the correlation between changes in mutual information between representation and training data and labels during neural network training and the generalization and robustness of neural networks. These studies inspire us whether there will be related problems in the pre-training process of GNN.

IB Theory: IB theory can be used in deep learning to seek the balance between fitting and generalization by controlling the mutual information between latent representation and training data. The main idea of such equilibrium can be summarized into two points: i) enlarging the information that is useful to the task within the representation, and ii) suppressing the information that is irrelevant to the task within the representation [42], [49], [50]. Given the significant potential of IB in enhancing model interpretability and generalization, recent researchers attempt to explore its effect on extracting graph representations [50], [51], [52]. Specifically, GIB [50] extends general IB to graph data as a modified regularization on both structure and feature information, hence achieving more robust node representations. And, SIB [51] and VIB-GSL [52] apply IB to subgraph recognition and graph structure learning, respectively. Collectively, these methods directly utilize the IB principle to learn minimal but sufficient information. However, directly using IB in pre-training GNNs to learn minimal but sufficient information with regard to the pre-training task will definitely result in the issue of information forgetting during the procedure of seeking the minimal information subset.

III. Delayed Bottlenecking Pre-Training: Causation, Strategy, and Derivation

A key insight of this paper is *information suppressing in pre-training may lose useful information with regard to the downstream task.* In this section, we first conduct a theoretical analysis to demonstrate the existence of this effect and formulate the information forgetting during pre-training, and further pointedly propose the improvement strategy.

A. Re-analyzing Parameter Transfer in Pre-training

In this subsection, we re-analyze the information forgetting problem during pre-training and its impact on parameter transfer. Ideally, the essence of pre-training is to extract transferable knowledge from pre-training data, so the objective optimization process of pre-training can be described as:

$$\begin{aligned}\theta_0 &= \arg\min_{\theta} \ \mathcal{L}_p(f_\theta; \mathcal{D}^{pre}) \\ &= \arg\max_{\theta} \ I_\theta(f_\theta; \mathcal{D}^{pre}),\end{aligned} \tag{2}$$

where $\mathcal{L}_p$ represents the optimization objective of pre-training, and $I_\theta(f_\theta; \mathcal{D}^{pre})$ denotes the information extracted by model f_θ from pre-training dataset $\mathcal{D}^{pre}$. After training, the optimal model parameter set θ_0 and the corresponding extracted information $I_\theta(f_\theta; \mathcal{D}^{pre})$ are transferred to the downstream task through parameter initialization.

Lemma 1 (Representation Forgetting): According to the research of [42], in the normal training process, the mutual information $I(X; Z)$ between input data X and latent representation Z first increases and then decreases in the early stage of training, while the mutual information $I(X; Y)$ between input data X and output Y keeps increasing. Formally:

$$\begin{aligned}\exists \theta_{\max}, \text{ s.t. } \forall \theta < \theta_{\max}, \frac{\partial I(X; Z_\theta)}{\partial \theta} > 0; \\ \forall \theta > \theta_{\max}, \frac{\partial I(X; Z_\theta)}{\partial \theta} < 0.\end{aligned} \tag{3}$$

Theorem 1 (Pre-training Information Transfer): Since the pre-training task itself is also a training task, according to Lemma 1, for the existing pre-trained GNN model f_θ, the extracted information $I_\theta(f_\theta; \mathcal{D}^{pre})$ will gradually increase to $I_{\theta_{\max}}(f_{\theta_{\max}}; \mathcal{D}^{pre})$, and then decrease to $I_{\theta_0}(f_{\theta_0}; \mathcal{D}^{pre})$ when obtaining the optimal parameter θ_0, i.e.,

$$\begin{aligned}&\theta \to \theta_0 \Longrightarrow \\ &I_\theta(f_\theta; \mathcal{D}^{pre}) \to I_{\theta_{\max}}(f_{\theta_{\max}}; \mathcal{D}^{pre}) \to I_{\theta_0}(f_{\theta_0}; \mathcal{D}^{pre}).\end{aligned} \tag{4}$$

In this process, the forgotten information, probably contains some information which is useless to the pre-training task but useful to the downstream task. If such information is forgotten

and cannot be transferred to the downstream task, the parameter set θ_0 and extracted information $I_\theta(f_\theta; \mathcal{D}^{pre})$ are not optimal anymore.

B. Delayed Bottlenecking Strategy

In this subsection, we propose a novel strategy, Delayed Bottlenecking (DBP), to alleviate the above problem. The basic idea of such a strategy is to suppress the information compression imposed for the pre-training task in pre-training and enhance the compression based on the downstream task in fine-tuning, i.e., make sure $I_{\theta_0}(f_{\theta_0}; \mathcal{D}^{pre})$ be closer to $I_{\theta_{\max}}(f_{\theta_{\max}}; \mathcal{D}^{pre})$ while obtaining the optimal pre-training parameter set θ_0.

Such operation of maintaining as much as possible information to fine-tuning phase can also be viewed as that the pre-trained parameters and representation are skewed to the downstream task, and to achieve this target, we can formulate two information control objectives respectively for pre-training and fine-tuning phases, i.e.,

i) Pre-training phase:

$$\mathcal{L}_{pi} = -I(\mathcal{D}^{pre}; Z) \tag{5}$$

ii) Fine-tuning phase:

$$\mathcal{L}_{fine} = \mathcal{L}_{cls} + \beta \cdot \mathcal{L}_{fi}$$
$$where \quad \begin{cases} \mathcal{L}_{cls} = -I(Y; Z) \\ \mathcal{L}_{fi} = I(\mathcal{D}^{fine}; Z) \end{cases} \tag{6}$$

where $I(\cdot;\cdot)$ represents the mutual information, Z is the latent representation, Y is the downstream target, β is employed to control the degree of enhancing compression in fine-tuning and can be tuned based on task and dataset. $\mathcal{L}_{fine}$ is used to enhance compression with the guidance of the downstream task and labeled fine-tuning data. Therefore, it contains two components, $\mathcal{L}_{fi}$ is to enhance the compression based on labeled fine-tuning data, and $\mathcal{L}_{cls}$ is to enhance the mutual information between latent representation and downstream task, hence improving the final performance of our model in the downstream task. Different from traditional optimization problems, in the field of deep learning, the issue of optimizing such objectives can also be converted into seeking the variational upper bounds respectively for $\mathcal{L}_{pi}$ and $\mathcal{L}_{fine}$ to achieve the minimization constraint of mutual information in both pre-training and fine-tuning periods. We then detailedly discuss these in the next subsection.

C. Information Control Objectives for Optimization

Due to the intractability of mutual information, the IB objectives in (5) and (6) are hard to be directly used in optimization. Therefore, in this subsection, we derive the tractable upper bounds of $\mathcal{L}_{pi}$ and $\mathcal{L}_{fi}$. The variational upper bounds ensure that the original mutual information objective can be reduced in case the empirical risks of $\mathcal{L}_{fine}$ and $\mathcal{L}_{pi}$ are reduced.

Proposition 1 (Upper bound of $\mathcal{L}_{pi}$): Given pre-training dataset $\mathcal{D}^{pre}$, latent representation Z_p learned from $\mathcal{D}^{pre}$, and graph $G_p = (X_p, \mathcal{E}_p) \in \mathcal{D}^{pre}$, we have

$$\begin{aligned} \mathcal{L}_{pi} &= -I(\mathcal{D}^{pre}; Z) \\ &\le -\mathbb{E}_{Z_p \sim p_\theta(Z_p|X_p,\mathcal{E}_p)}[\log q_\varphi(X_p, \mathcal{E}_p|Z_p)] \end{aligned} \tag{7}$$

where $p_\theta(Z_p|X_p, \mathcal{E}_p)$ is the variational approximation of true conditional probability $p(Z_p|X_p, \mathcal{E}_p)$ in the encoder during pre-training, and $q_\varphi(X_p, \mathcal{E}_p|Z_p)$ is the variational approximation of the true conditional probability $q(X_p, \mathcal{E}_p|Z_p)$ in the decoder during pre-training.

Proof: For pre-training dataset $\mathcal{D}^{pre}$, latent representation Z_p learned from $\mathcal{D}^{pre}$, and graph $G_p = (X_p, \mathcal{E}_p) \in \mathcal{D}^{pre}$, we have:

$$\begin{aligned} I(\mathcal{D}^{pre}; Z_p) &= H(\mathcal{D}^{pre}) - H(\mathcal{D}^{pre}|Z_p) \\ &\ge -H(\mathcal{D}^{pre}|Z_p) \\ &\overset{(1)}{=} \int dZ_p dG_p \, p(Z_p, G_p) \log p(G_p|Z_p) \\ &\overset{(2)}{=} \int dZ_p dG_p \, p(Z_p, G_p) \log q_\varphi(G_p|Z_p) \\ &\quad + \int dZ_p dG_p \, p(Z_p, G_p) \log \frac{p(G_p|Z_p)}{q_\varphi(G_p|Z_p)} \\ &\overset{(3)}{=} \int dZ_p dG_p \, p(Z_p, G_p) \log q_\varphi(G_p|Z_p) \\ &\quad + \int dG_p \, p(G_p|Z_p) \log \frac{p(G_p|Z_p)}{q_\varphi(G_p|Z_p)} \\ &\overset{(4)}{=} \int dZ_p \, p_\theta(Z_p|G_p) \log q_\varphi(G_p|Z_p) \\ &\quad + \mathrm{KL}\left[p(G_p|Z_p)||q(G_p|Z_p)\right] \\ &\overset{(5)}{\ge} \int dZ_p \, p_\theta(Z_p|G_p) \log q_(G_p|Z_p) \\ &\overset{(6)}{=} \mathbb{E}_{Z_p \sim p_\theta(Z_p|G_p)}[\log q_\varphi(G_p|Z_p)] \\ &\overset{(7)}{=} \mathbb{E}_{Z_p \sim p_\theta(Z_p|X_p,\mathcal{E}_p)}[\log q_\varphi(X_p, \mathcal{E}_p|Z_p)]. \end{aligned} \tag{8}$$

Among (8), step (1) is the definition of mutual information. Steps (2) and (3) are based on the integral property. Steps (4) and (5) are defined according to the KL divergence. Steps (6) and (7) are based on the properties of integrals and expectations.

$$\begin{aligned} \mathcal{L}_{pi} &= -\alpha \cdot I(\mathcal{D}^{pre}; Z_p) \\ &\le -\alpha \cdot \mathbb{E}_{Z_p \sim p_\theta(Z_p|X_p,\mathcal{E}_p)}[\log q_\varphi(X_p, \mathcal{E}_p|Z_p)] \end{aligned} \tag{9}$$

In the fine-tuning phase, we encourage the downstream task can compress information with ground-truth labels, so that the learned knowledge during pre-training can be transferred and generalized on the downstream task more quickly.

Proposition 2 (Upper bound of $\mathcal{L}_{fine}$): Given fine-tuning dataset $\mathcal{D}^{fine}$, latent representation Z_f learned from $\mathcal{D}^{fine}$, the label y of downstream task Y, and graph $G_f = (X_f, \mathcal{E}_f) \in$

$\mathcal{D}^{fine}$, we have

$$\begin{aligned}\mathcal{L}_{fine} =& \ \beta \cdot I(\mathcal{D}^{fine}; Z) - I(Y; Z) \\ \leq& \ \beta \cdot \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)} \mathrm{KL}[p_\omega(Z_f|X_f,\mathcal{E}_f), r(Z_f)] \\ & - \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)}[\log \ q_\gamma(y|Z_f)]\end{aligned} \tag{10}$$

where $p_\omega(Z_f|X_f,\mathcal{E}_f)$ is the variational approximation of true conditional probability $p(Z_f|X_f,\mathcal{E})_f$ in the encoder during fine-tuning, $r(Z_f)$ is an estimation of prior probability $p(Z_f)$ of Z_f, and $q_\gamma(y|Z_f)$ is the variational approximation of the true conditional probability $q(y|Z_f)$ in the classifier.

Proof: For fine-tuning dataset $\mathcal{D}^{fine}$, latent representation Z_f learned from $\mathcal{D}^{fine}$, the label of downstream task y, and graph $G_f = (X_f, \mathcal{E}_f) \in \mathcal{D}^{fine}$, we have:

$$\begin{aligned}& I(y; Z_f) - \beta \cdot I(\mathcal{D}^{fine}; Z_f) \\ & \overset{(1)}{\geq} \int dX_f dy dZ_f \, p(X_f)\, p(y|X_f)\, p(Z_f|X_f) \log q(y|Z_f) \\ & \quad - \beta \cdot \int dX_f dZ_f \, p(X_f)\, p(Z_f|X_f) \log \frac{p(Z_f|X_f)}{r(Z_f)} \\ & \overset{(2)}{=} \int dZ_f \, p_\omega(Z_f|X_f) \log q_\gamma(y|Z_f) \\ & \quad - \beta \cdot \int dZ \, p_\omega(Z_f|X_f) \log \frac{p_\omega(Z_f|X_f)}{r(Z_f)} \\ & \overset{(3)}{=} \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)}[\log \ q_\gamma(y|Z_f)] \\ & \quad - \beta \cdot \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)} \mathrm{KL}[p_\omega(Z_f|X_f,\mathcal{E}_f), r(Z_f)].\end{aligned} \tag{11}$$

Among (11), Step (1) is derived from the application of Jensen's inequality and the variational lower bound for mutual information. This step provides a lower bound for both $I(y; Z_f)$ and $I(\mathcal{D}^{fine}; Z_f)$. Step (2) is based on the properties of conditional probability and marginal probability. Here, we introduce the parameterized distributions $p_\omega(Z_f|X_f)$ and $q_\gamma(y|Z_f)$ to represent our model. Step (3) follows from the definition of expectation and the Kullback-Leibler (KL) divergence. We express the integrals as expectations with respect to $p_\omega(Z_f|X_f,\mathcal{E}_f)$, and identify the KL divergence term. Thus, we can obtain an upper bound on the information control objective of the fine-tuning stage:

$$\begin{aligned}\mathcal{L}_{fine} =& \ \beta \cdot I(\mathcal{D}^{fine}; Z_f) - I(y; Z_f) \\ \leq& \ \beta \cdot \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)} \mathrm{KL}[p_\omega(Z_f|X_f,\mathcal{E}_f), r(Z_f)] \\ & - \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)}[\log \ q_\gamma(y|Z_f)].\end{aligned} \tag{12}$$

D. Proof for Parameters Transfer

In this section, we will prove how the proposed two-stage loss function improves the transfer of pre-trained parameters. First, we introduce some additional definitions and lemmas:

Definition 1 (KL Divergence): For two probability distributions P and Q, the KL divergence between them is defined as:

$$D_{\mathrm{KL}}(P\|Q) = \mathbb{E}_{x\sim P}\left[\log \frac{P(x)}{Q(x)}\right] \tag{13}$$

Lemma 2 (Chain Rule of KL Divergence): For three probability distributions $P(X,Y)$, $Q(X,Y)$, and $R(X)$, we have:

$$\begin{aligned}D_{\mathrm{KL}}(P(X,Y)\|Q(X,Y)) =& \ D_{\mathrm{KL}}(P(X)\|R(X)) \\ & + \mathbb{E}_{x\sim P(X)}[D_{\mathrm{KL}}(P(Y|X)\|Q(Y|X))]\end{aligned} \tag{14}$$

Lemma 3 (Non-Negativity of KL Divergence): For any two probability distributions P and Q, we have $D_{\mathrm{KL}}(P\|Q) \geq 0$, with equality holding if and only if $P = Q$ almost everywhere.

The equality holds if and only if $\frac{Q(x)}{P(x)}$ is a constant almost everywhere, i.e., $P = Q$ almost everywhere. The proof of this lemma can be demonstrated using Jensen's inequality, but is omitted here due to space constraints.

Now, we restate and prove the main theorem:

Theorem 2 (Bounding Posterior Distributions via DBP): Let $\mathcal{D}^{pre}, \mathcal{D}^{fine}$ denote the pre-training data and fine-tuning data, respectively, Z_p, Z_f denote the corresponding latent representations, and Y denote the labels for the downstream task. Define:

$$\begin{cases}\mathcal{L}_{pi} = -I(\mathcal{D}^{pre}; Z_p) \\ \mathcal{L}_{fi} = \beta I(\mathcal{D}^{fine}; Z_f) - I(Y; Z_f)\end{cases} \tag{15}$$

We make the following assumptions:

Assumption 1: Let H denote the model hypothesis space, which represents the set of all possible models or hypotheses for the given learning task. Let $P(H)$ and $Q(H)$ denote two different prior probability distributions over this hypothesis space H. These priors represent initial beliefs or assumptions about the likelihood of different hypotheses before observing any data.

Assumption 2: $P(Z_p|\mathcal{D}^{pre}, H) = Q(Z_p|\mathcal{D}^{pre}, H), \forall H \in \mathcal{H}$

Assumption 3: $P(Z_f|\mathcal{D}^{fine}, H) = Q(Z_f|\mathcal{D}^{fine}, H), \forall H \in \mathcal{H}$

Assumption 4: $P(Y|Z_f, H) = Q(Y|Z_f, H), \forall H \in \mathcal{H}$

Rationale: Our assumptions ensure consistency between the true distribution P and the approximate distribution Q during key stages of learning, while allowing flexibility in priors. Assumption 1 introduces different priors, $P(\mathcal{H})$ and $Q(\mathcal{H})$, to capture varying initial beliefs. Assumptions 2 and 3 ensure that, given the same hypothesis H, the latent representations from pre-training (Z_p) and fine-tuning (Z_f) are identical under P and Q. Assumption 4 ensures that, once Z_f and H are fixed, the prediction of Y is the same across both distributions. Together, these assumptions balance flexibility in priors with consistency in learning, enabling rigorous analysis across different distributions.

Then the optimization objectives $\mathcal{L}_{pi}$ and $\mathcal{L}_{fi}$ satisfy:

$$\begin{aligned}& D_{\mathrm{KL}}(P(H|\mathcal{D}^{pre})\|Q(H|\mathcal{D}^{pre})) \\ & \quad \leq D_{\mathrm{KL}}(P(H)\|Q(H)) - \mathcal{L}_{pi}(P) + \mathcal{L}_{pi}(Q) \\ & D_{\mathrm{KL}}(P(H|\mathcal{D}^{fine})\|Q(H|\mathcal{D}^{fine}))\end{aligned}$$

$$\leq D_{\mathrm{KL}}(P(H)\|Q(H)) + \frac{1}{\beta}(\mathcal{L}_{fi}(P) - \mathcal{L}_{fi}(Q)) \quad (16)$$

Theorem 2 provides a theoretical foundation for our Deferred Bottleneck Pretraining (DBP) method by showing how minimizing $\mathcal{L}_{pi}$ and $\mathcal{L}_{fi}$ can control the divergence between the posterior distributions under the pre-training and fine-tuning data, respectively. This theorem supports the core principles of DBP: By minimizing $\mathcal{L}_{pi}(Q)$ during pre-training, the divergence between $P(H|\mathcal{D}^{pre})$ and $Q(H|\mathcal{D}^{pre})$ is reduced, ensuring high mutual information between latent representations and data. This helps the model learn comprehensive representations without prematurely compressing information, which is valuable for downstream tasks. In fine-tuning, minimizing $\mathcal{L}_{fi}(Q)$ reduces the divergence between $P(H|\mathcal{D}^{fine})$ and $Q(H|\mathcal{D}^{fine})$, with DBP delaying compression to this stage. This allows the model to selectively compress and optimize the pre-trained representation based on the downstream task's needs. The core strategy combines minimizing $\mathcal{L}_{fi}(Q)$ and $\mathcal{L}_{fi}(P)$, which maximizes $I_P(Y;Z_f) - \beta I_P(\mathcal{D}^{fine};Z_f)$, balancing mutual information with label prediction and compression, thus reflecting DBP's delayed bottleneck approach guided by downstream supervision.

Proof: First, we prove the first inequality. By Lemma 2, we have:

$$\begin{aligned}
&D_{\mathrm{KL}}(P(H,Z_p,\mathcal{D}^{pre})\|Q(H,Z_p,\mathcal{D}^{pre}))\\
&\overset{(1)}{=} D_{\mathrm{KL}}(P(H)\|Q(H))\\
&\quad + \mathbb{E}_{P(H)}[D_{\mathrm{KL}}(P(Z_p,\mathcal{D}^{pre}|H)\|Q(Z_p,\mathcal{D}^{pre}|H))]\\
&\overset{(2)}{=} D_{\mathrm{KL}}(P(H)\|Q(H))\\
&\quad + \mathbb{E}_{P(H)}[D_{\mathrm{KL}}(P(\mathcal{D}^{pre}|H)\|Q(\mathcal{D}^{pre}|H))]\\
&\quad + \mathbb{E}_{P(H),P(\mathcal{D}^{pre}|H)}[D_{\mathrm{KL}}(P(Z_p|\mathcal{D}^{pre},H)\|Q(Z_p|\mathcal{D}^{pre},H))]\\
&\overset{(3)}{\geq} D_{\mathrm{KL}}(P(H)\|Q(H))\\
&\quad + \mathbb{E}_{P(H)}[D_{\mathrm{KL}}(P(\mathcal{D}^{pre}|H)\|Q(\mathcal{D}^{pre}|H))] \quad (17)
\end{aligned}$$

Explanation: (1) We apply the chain rule of KL divergence (Lemma 2) to decompose the joint distribution. (2) We further decompose the conditional distribution $P(Z_p,\mathcal{D}^{pre}|H)$ using the chain rule. (3) Due to Assumption 2: $P(Z_p|\mathcal{D}^{pre},H) = Q(Z_p|\mathcal{D}^{pre},H)$, the third term becomes zero. We then apply the non-negativity of KL divergence (Lemma 3) to obtain the inequality.

On the other hand, we have:

$$\begin{aligned}
&D_{\mathrm{KL}}(P(H,Z_p,\mathcal{D}^{pre})\|Q(H,Z_p,\mathcal{D}^{pre}))\\
&\overset{(1)}{=} \mathbb{E}_{P(H,Z_p,\mathcal{D}^{pre})}\left[\log\frac{P(h,z_p,d^{pre})}{Q(h,z_p,d^{pre})}\right]\\
&\overset{(2)}{=} \mathbb{E}_{P(H,Z_p,\mathcal{D}^{pre})}\left[\log\frac{P(h)P(d^{pre}|h)P(z_p|d^{pre},h)}{Q(h)Q(d^{pre}|h)Q(z_p|d^{pre},h)}\right]\\
&\overset{(3)}{=} \mathbb{E}_{P(H,Z_p,\mathcal{D}^{pre})}\left[\log\frac{P(h)}{Q(h)}\right.\\
&\quad \left.+\log\frac{P(d^{pre}|h)}{Q(d^{pre}|h)}+\log\frac{P(z_p|d^{pre},h)}{Q(z_p|d^{pre},h)}\right]\\
&\overset{(4)}{=} D_{\mathrm{KL}}(P(H)\|Q(H)) + \mathbb{E}_{P(H,\mathcal{D}^{pre})}\left[\log\frac{P(d^{pre}|h)}{Q(d^{pre}|h)}\right]\\
&\quad + \mathbb{E}_{P(H,Z_p,\mathcal{D}^{pre})}\left[\log\frac{P(z_p|d^{pre},h)}{Q(z_p|d^{pre},h)}\right]\\
&\overset{(5)}{=} D_{\mathrm{KL}}(P(H)\|Q(H)) + I_P(\mathcal{D}^{pre};H) - I_Q(\mathcal{D}^{pre};H)\\
&\quad + \mathbb{E}_{P(H,\mathcal{D}^{pre})}[D_{\mathrm{KL}}(P(Z_p|\mathcal{D}^{pre},H)\|Q(Z_p|\mathcal{D}^{pre},H))]\\
&\overset{(6)}{=} D_{\mathrm{KL}}(P(H)\|Q(H)) - \mathcal{L}_{pi}(P) + \mathcal{L}_{pi}(Q) \quad (18)
\end{aligned}$$

Combining the above two inequalities, we obtain:

$$\begin{aligned}
&D_{\mathrm{KL}}(P(H|\mathcal{D}^{pre})\|Q(H|\mathcal{D}^{pre}))\\
&\overset{(7)}{\leq} D_{\mathrm{KL}}(P(H,Z_p,\mathcal{D}^{pre})\|Q(H,Z_p,\mathcal{D}^{pre}))\\
&\overset{(8)}{\leq} D_{\mathrm{KL}}(P(H)\|Q(H)) - \mathcal{L}_{pi}(P) + \mathcal{L}_{pi}(Q) \quad (19)
\end{aligned}$$

Explanation: (1) Definition of KL divergence for joint distributions. (2) Applying the chain rule to decompose joint probabilities. (3) Separating the logarithm of a product into a sum of logarithms. (4) Recognizing the first term as KL divergence of H and separating expectations. (5) Identifying mutual information terms and KL divergence for Z_p. (6) Defining $\mathcal{L}_{pi}$ in terms of mutual information. (7) Applying the data processing inequality for KL divergence. (8) Using the result from (18). where the second equality uses assumption (2), and the last inequality uses Lemma 3.

Next, we prove the second inequality. Similarly, we have:

$$\begin{aligned}
&D_{\mathrm{KL}}(P(H,Z_f,\mathcal{D}^{fine},Y)|Q(H,Z_f,\mathcal{D}^{fine},Y))\\
&\overset{(1)}{=} D_{\mathrm{KL}}(P(H)|Q(H))\\
&\quad + \mathbb{E}P(H)\left[D\mathrm{KL}(P(Z_f,\mathcal{D}^{fine},Y|H)|Q(Z_f,\mathcal{D}^{fine},Y|H))\right]\\
&\overset{(2)}{=} D_{\mathrm{KL}}(P(H)|Q(H))\\
&\quad + \mathbb{E}P(H)\left[D\mathrm{KL}(P(\mathcal{D}^{fine}|H)|Q(\mathcal{D}^{fine}|H))\right]\\
&\quad + \mathbb{E}P(H)\mathbb{E}P(\mathcal{D}^{fine}|H)\left[D_{\mathrm{KL}}(P(Z_f|\mathcal{D}^{fine},H)\right.\\
&\qquad \left.|Q(Z_f|\mathcal{D}^{fine},H))\right]\\
&\quad + \mathbb{E}P(H)\mathbb{E}P(\mathcal{D}^{fine}|H)\mathbb{E}P(Z_f|\mathcal{D}^{fine},H)\\
&\qquad [D\mathrm{KL}(P(Y|Z_f,H)|Q(Y|Z_f,H))]\\
&\overset{(3)}{\geq} D_{\mathrm{KL}}(P(H)|Q(H))\\
&\quad + \mathbb{E}P(H)\left[D\mathrm{KL}(P(\mathcal{D}^{fine}|H)|Q(\mathcal{D}^{fine}|H))\right] \quad (20)
\end{aligned}$$

Explanation: (1) We apply the chain rule of KL divergence to decompose the joint distribution. (2) We further decompose the conditional distribution $P(Z_f,\mathcal{D}^{fine},Y|H)$ using the chain rule multiple times. (3) Due to Assumptions 3 and 4, the third and fourth terms become zero. We then apply the non-negativity of KL divergence to obtain the inequality.

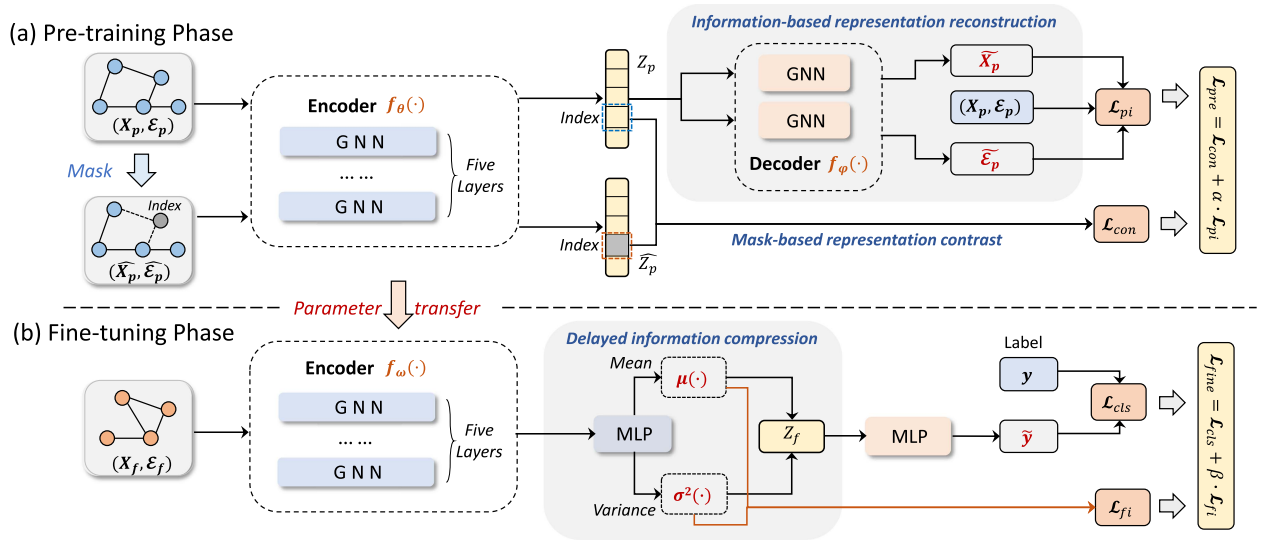

Fig. 2. Architecture of DBP framework. Subfigure (a) corresponds to the generative and contrastive learning based self-supervised pre-training model. The optimization objective of pre-training consists of L_{con} and L_{pi} which are respectively used to extract general knowledge and avoid excessive information compression. Subfigure (b) indicates the information control based fine-tuning model. The optimization objective of fine-tuning, which is composed of L_{cls} and L_{fi}, encourages enhanced information compression to improve classification performance. The two-phase transition is implemented by means of parameter transfer.

On the other hand, we have:

$$
\begin{aligned}
&D_{\mathrm{KL}}(P(H, Z_f, \mathcal{D}^{fine}, Y) \| Q(H, Z_f, \mathcal{D}^{fine}, Y)) \\
&\overset{(4)}{=} \mathbb{E}_{P(H, Z_f, \mathcal{D}^{fine}, Y)}\left[\log \frac{P(h, z_f, d^{fine}, y)}{Q(h, z_f, d^{fine}, y)}\right] \\
&\overset{(5)}{=} \mathbb{E}_{P(H, Z_f, \mathcal{D}^{fine}, Y)}\left[\log \frac{P(h)}{Q(h)} + \log \frac{P(d^{fine}|h)}{Q(d^{fine}|h)}\right. \\
&\quad \left. + \log \frac{P(z_f|d^{fine}, h)}{Q(z_f|d^{fine}, h)} + \log \frac{P(y|z_f, h)}{Q(y|z_f, h)}\right] \\
&\overset{(6)}{=} D_{\mathrm{KL}}(P(H) \| Q(H)) + \mathbb{E}_{P(H, \mathcal{D}^{fine})}\left[\log \frac{P(d^{fine}|h)}{Q(d^{fine}|h)}\right] \\
&\quad + \mathbb{E}_{P(H, Z_f, \mathcal{D}^{fine})}\left[\log \frac{P(z_f|d^{fine}, h)}{Q(z_f|d^{fine}, h)}\right] \\
&\quad + \mathbb{E}_{P(H, Z_f, \mathcal{D}^{fine}, Y)}\left[\log \frac{P(y|z_f, h)}{Q(y|z_f, h)}\right] \\
&\overset{(7)}{=} D_{\mathrm{KL}}(P(H) \| Q(H)) + I_P(\mathcal{D}^{fine}; H) - I_Q(\mathcal{D}^{fine}; H) \\
&\quad + I_P(Z_f; \mathcal{D}^{fine}|H) - I_Q(Z_f; \mathcal{D}^{fine}|H) \\
&\quad + I_P(Y; Z_f|H) - I_Q(Y; Z_f|H) \\
&\overset{(8)}{=} D_{\mathrm{KL}}(P(H) \| Q(H)) + \frac{1}{\beta}(\mathcal{L}_{fi}(P) - \mathcal{L}_{fi}(Q))
\end{aligned} \tag{21}
$$

Combining inequalities (20) and (21), we obtain:

$$
\begin{aligned}
&D_{\mathrm{KL}}(P(H|\mathcal{D}^{fine}) \| Q(H|\mathcal{D}^{fine})) \\
&\overset{(9)}{\leq} D_{\mathrm{KL}}(P(H, Z_f, \mathcal{D}^{fine}, Y) \| Q(H, Z_f, \mathcal{D}^{fine}, Y)) \\
&\overset{(10)}{\leq} D_{\mathrm{KL}}(P(H) \| Q(H)) + \frac{1}{\beta}(\mathcal{L}_{fi}(P) - \mathcal{L}_{fi}(Q))
\end{aligned} \tag{22}
$$

Explanation: (4) Definition of KL divergence for joint distributions. (5) Applying the chain rule to decompose joint probabilities. (6) Separating the logarithm of a product into a sum of logarithms and grouping terms. (7) Identifying mutual information terms. (8) Defining $\mathcal{L}_{fi}$ in terms of mutual information, with β as a scaling factor. (9) Applying the data processing inequality for KL divergence. (10) Using the result from (21).

IV. DBP Framework

Given the theoretical analysis in the previous section, we still have to consider how to integrate the proposed DBP strategy into the actual model design, so that the information control objectives can be applied to graph structure data. We will discuss these issues in this section.

A. Solution Overview

The solution overview is illustrated in Fig. 2 which also contains the implementations of the above two optimization information control objectives in the pre-training and fine-tuning phases. The DBP framework contains two parts: i) a self-supervised pre-training model for controlling information and extracting knowledge via generative learning and contrastive learning, and ii) a fine-tuning model with an information control module.

B. Generative and Contrastive Learning Based Self-supervised Pre-training

As shown in Fig. 2(a), We built a self-supervised pre-training model that consists of two components, mask-based representation contrast and information-based representation

reconstruction. The first one is to extract general knowledge from pre-training data, and the second component is used to maintain the mutual information between latent representation and training data. We will describe these two modules in detail in the following subsections.

Mask-based Representation Contrast: We begin by applying a random masking scheme to the original graph $G_p = (X_p, \mathcal{E}_p)$, masking some nodes and their connected edges. We record the indices of these masked nodes and obtain a noisy graph $\widehat{G_p} = (\widehat{X_p}, \widehat{\mathcal{E}_p})$. Both the original graph G_p and the noisy graph $\widehat{G_p}$ are then processed by the encoder f_θ to generate node representations:

$$Z_p = f_\theta(X_p, \mathcal{E}_p), \quad \widehat{Z_p} = f_\theta(\widehat{X_p}, \widehat{\mathcal{E}_p}) \tag{23}$$

For self-supervised learning, **positive samples** are defined as the dot product of representations for a masked node and its connected node in the original graph, while **negative samples** are the dot product of the representations for a masked node and its connected node in the noisy graph. The contrastive self-supervised objective is then formulated as:

$$\begin{aligned}\mathcal{L}_{con} = & \sum_{u \in mask} \sum_{v \in \mathcal{N}(u)} -\ln(\sigma({Z_{p_u}}^T \cdot Z_{p_v})) \\ & - \ln(\sigma(-\widehat{{Z_{p_u}}^T} \cdot \widehat{Z_{p_v}}))\end{aligned} \tag{24}$$

where u represents a masked node, $v \in \mathcal{N}(u)$ denotes a node connected to u, Z_{p_u} and Z_{p_v} are their respective representations in the original graph, while $\widehat{Z_{p_u}}$ and $\widehat{Z_{p_v}}$ are their representations in the noisy graph. The function σ represents the Sigmoid function.

Information-Maintain Representation Reconstruction: We further introduce the information control theory mentioned in Proposition 1 into the node representation learning of the original graph. Here, to obtain the representation Z_p of the original graph by encoder f_θ, we let the conditional probability $p_\theta(Z_p|X_p, \mathcal{E}_p)$ be the variational approximation of the true conditional probability $p(Z_p|X_p, \mathcal{E}_p)$. To calculate $q_\varphi(X_p, \mathcal{E}_p|Z_p)$, we employ a decoder f_φ consisting of two single-layer GNNs to reconstruct node features $\widetilde{X_p}$ and edge features $\widetilde{\mathcal{E}_p}$ of the original graph from Z_p, i.e.,

$$\widetilde{X_p}, \widetilde{\mathcal{E}_p} = f_\varphi(Z_p) \tag{25}$$

The optimization goal of our information reconstruction task can be calculated as the cross-entropy (CE) loss between reconstructed features and original features, and it can be simplified as the upper bound derived in (7).

$$\mathcal{L}_{pi} = -\mathbb{E}_{Z_p \sim p_\theta(Z_p|X_p,\mathcal{E}_p)}[\log\ q_\varphi(X_p, \mathcal{E}_p|Z_p)] \tag{26}$$

Here, $\mathcal{L}_{pi}$ allows the representation to retain more information from the pre-trained dataset by encouraging the reconstructed nodes and features to have more similarity to the original graph. It is not only an approximation of our proposed information control objective, but also indicates that the latent representation encoded by f_θ has the potential to be restored to the original representation.

Pre-training Objective: The above representation contrast and reconstruction components work jointly to extract general knowledge from pre-training data and simultaneously suppress information compression in latent representation learning. This jointly working mechanism determines that the pre-trained model can effectively avoid information forgetting in extracting general knowledge, and information compression is delayed to the fine-tuning phase. Therefore, the overall loss of pre-training should be the sum of the objectives of these two components, i.e.,

$$\begin{aligned}\mathcal{L}_{pre} &= \mathcal{L}_{con} + \alpha \cdot \mathcal{L}_{pi} \\ &= \sum_{u,u' \in mask} -\ln(\sigma({Z_{p_u}}^T \cdot Z_{p_v})) - \ln(\sigma(-\widehat{{Z_{p_{u'}}}^T} \cdot \widehat{Z_{p_{v'}}})) \\ &\quad - \alpha \cdot \mathbb{E}_{Z_p \sim p_\theta(Z_p|X_p,\mathcal{E}_p)}[\log\ q_\varphi(X_p, \mathcal{E}_p|Z_p)]\end{aligned} \tag{27}$$

where α is a hyper-parameter for adjusting the weight of information control.

C. Information Controlled Fine-tuning

Similar to recent studies, during fine-tuning, we also employ the same encoder f_ω, whose parameters ω are initialized with the parameter θ of f_θ, as in the pre-training phase. Notice that the labeled map $G_f = (X_f, \mathcal{E}_f)$ in downstream task is encoded to obtain the latent representation Z_f by f_ω which calculates the modeling probability $p_\omega(Z_f|X_f, \mathcal{E}_f)$, i.e.,

$$Z_f = f_\omega(X_f, \mathcal{E}_f). \tag{28}$$

As demonstrated in Fig. 2(b), different from other pre-training and fine-tuning works, to avoid information forgetting, the information compression in pre-training is suppressed, therefore, we should enhance this operation in the fine-tuning phase.

Delayed Information Compression: To achieve delayed information compression in fine-tuning, we add a compression module consisting of two MLPs between f_ω and the classifier to learn the mean μ and variance σ^2 respectively.

$$\{\mu, \sigma^2\} = \text{MLPs}(Z_f) \tag{29}$$

Then we sample the graph representation Z_f from the multivariate normal distribution $\mathcal{N}(Z_f|\mu, \sigma^2)$ with a reparameterization trick to realize back-propagation, i.e.,

$$Z_t = \mu + \sigma^2 \odot \varepsilon \tag{30}$$

where $\varepsilon \sim \mathcal{N}(0, 1)$. We further refer to the information control theory mentioned in Proposition 2 to calculate the Kullback-Leibler (KL) divergence of distribution $\mathcal{N}(Z_f|\mu, \sigma^2)$ and Gaussian prior distribution $r(Z_f)$ to realize information control during fine-tuning. The optimization objective is the term in formula (6) to enhance compression,

$$\mathcal{L}_{fi} = \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)}\,\text{KL}[p_\omega(Z_f|X_f, \mathcal{E}_f), r(Z_f)] \tag{31}$$

Fine-tuning Objective: Furthermore, we first take Z_f as the input of classifier f_γ to compute the variational approximation $q_\gamma(y|Z_f)$, and the classification loss is the first term in (6),

$$\mathcal{L}_{cls} = -\mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)}[\log\ q_\gamma(y|Z_f)]. \tag{32}$$

And based on this, the overall loss of fine-tuning can be defined as,

$$\begin{aligned}\mathcal{L}_{fine} &= \mathcal{L}_{cls} + \beta \cdot \mathcal{L}_{fi} \\ &= -\mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)}[\log\ q_\gamma(y|Z_f)] \\ &\quad + \beta \cdot \mathbb{E}_{Z_f \sim p_\omega(Z_f|X_f,\mathcal{E}_f)}\, \mathrm{KL}[p_\omega(Z_f|X_f,\mathcal{E}_f), r(Z_f)],\end{aligned} \tag{33}$$

where β is a hyper-parameter for adjusting the weight of information control.

V. Experiments

In this section, we compare the performance of our proposed DBP and various state-of-the-art pre-trained baselines on both chemistry and biology domains. Then, we conduct a series of comprehensive model analyses to witness our motivation and the effectiveness of our delayed bottlenecking information control strategy.

A. Experimental Settings

Datasets: Following the setting of [25], we conduct experiments on data from two domains: molecular property prediction in chemistry and biological function prediction in biology. For chemistry domain, we use Zinc-2M - 2 million unlabeled molecules sampled from the ZINC15 database [53] in the pre-training phase and eight binary classification datasets in MoleculeNet [54] in the fine-tuning phase, which are split by the scaffold splitting scheme. For biology domain, we utilize 395K unlabeled protein ego-networks [25] for self-supervised pre-training and predict 40 fine-grained biological functions of 8 species in the fine-tuning phase.

Setups: Following the setting of [25], we employ a five-layer GNN with 300-dimensional hidden units as the encoder and two single-layer GNNs as the decoder in the pre-training phase. We use an Adam optimizer [55] with a learning rate of 1×10^{-3} to pre-train the GNN for 100 epochs. In the fine-tuning phase, an information control module and a linear classifier are appended upon the pre-trained GNN and the information control module consists of two MLPs. We also train the model for 100 epochs using the Adam optimizer with learning rate of 1×10^{-3} and batch size of 32. We utilize a fixed-step-size learning rate scheduler, which multiplies the learning rate by 0.3 every 30 epochs. All the results are the average of five independent runs with the same configuration and different random seeds.

Attribute Masking Scheme: On chemistry domain, before pre-training on large-scale molecular graphs, we randomly mask nodes in 25% of attributes to obtain perturbed graphs. To compare and learn the node representation of the masked position and the node representation of the same position in the original graph, we need to record the index of the mask position for each training data. On biology domain, before pre-training on a large-scale protein self-network graph, we randomly mask the attributes of 25% of its edges to generate a perturbation graph. We also record the index while controlling the mask by setting additional weights for the variables.

GNN Architectures: Our experiments are mainly conducted on GIN, but to verify the effectiveness of our method, we conduct experiments with different GNN architectures in Table 3 in the original manuscript. All GNNs in our experiments (e.g., GCN [56], GraphSAGE [57], GAT [58], GIN [59]) are with 5 layers, 300-dimensional hidden units, and a mean pooling readout function. In addition, two attention heads are employed in each layer of the GAT model.

Baselines: To demonstrate the effectiveness and robustness of DBP, we compare it with state-of-the-art self-supervised pre-training methods on chemistry and biology domains: EdgePred [57], InfoGraph [60], AttrMasking [25], ContextPred [25], GraphPartition [61], GraphCL [26], GraphLoG [27], GraphMAE [62], S2GAE [63], MGSSL [64], MICRO-Graph [65] and GraphFP [66]. EdgePred infers link existence between node pairs. ContextPred explores graph structure distribution by sampling and predicting surrounding structures. AttrMasking masks and predicts node or edge attributes based on the neighborhood to learn their distribution. InfoGraph constructs contrastive loss using node representations of the graph instance and other graphs. GraphPartition is a topology-based method that partitions nodes into subsets to minimize cross-subset edges. GraphCL utilizes node dropping, edge perturbation, attribute masking, and subgraph sampling to construct views for contrastive learning. GraphLoG aligns embeddings of related graphs/subgraphs to construct a locally smooth latent space, and models global structure with a hierarchical model. GraphMAE conducts self-supervised pre-training by masking and reconstructing node features, introducing the masked autoencoding idea from computer vision. S2GAE randomly masks edges and learns to reconstruct them with direction-aware masking strategies and a cross-correlation decoder, demonstrating superior performance on link prediction, node classification, and graph classification tasks. MGSSL introduces motif-based self-supervised learning using BRICS to extract functional groups as motifs for GNN pre-training. MICRO-Graph leverages motif-driven contrastive learning to sample informative subgraphs using EM-clustering, improving GNN performance. GraphFP performs fragment-based pretraining on molecular and fragment graphs, enhancing results on both molecular and biological benchmarks.

B. Performance Comparison

Results on Chemistry Domain: The performance comparison between the proposed DBP and SOTA methods on chemistry domain is shown in Table I. DBP achieves the highest average ROC-AUC score and gain among all self-supervised learning strategies and performs best on six of eight tasks. We believe that such significant improvements can be attributed to the proposed information compression delayed pre-training strategy, which preserves more beneficial information in the pre-training phase and benefits downstream tasks in the fine-tuning phase.

Results on Biology Domain: The results in Table II show that DBP also achieves the best performance on biology domain, especially achieving a gain of 6.3% compared to the No-pretrain baseline. This illustrates that the proposed strategy is general

TABLE I
ROC-AUC SCORES (%) ON DOWNSTREAM MOLECULAR PROPERTY PREDICTION TASK COMPARED WITH STATE-OF-THE-ART METHODS

Method	BBBP	Tox21	ToxCast	SIDER	ClinTox	MUV	HIV	BACE	AVG.	GAIN
No Pre-training	65.8 ± 4.5	74.0 ± 0.8	63.4 ± 0.6	57.3 ± 1.6	58.0 ± 4.4	71.8 ± 2.5	75.3 ± 1.9	70.1 ± 5.4	67.0	-
EdgePred	67.3 ± 2.4	76.0 ± 0.6	64.1 ± 0.6	60.4 ± 0.7	64.1 ± 3.7	74.1 ± 2.1	76.3 ± 1.0	79.9 ± 0.9	70.3	3.3
InfoGraph	68.2 ± 0.7	75.5 ± 0.6	63.1 ± 0.3	59.4 ± 1.0	70.5 ± 1.8	75.6 ± 1.2	77.6 ± 0.4	78.9 ± 1.1	71.1	4.1
AttrMasking	64.3 ± 2.8	76.7 ± 0.4	64.2 ± 0.5	61.0 ± 0.7	71.8 ± 4.1	74.7 ± 1.4	77.2 ± 1.1	79.3 ± 1.6	71.1	4.1
ContextPred	68.0 ± 2.0	75.7 ± 0.7	63.9 ± 0.6	60.9 ± 0.6	65.9 ± 3.8	75.8 ± 1.7	77.3 ± 1.0	79.6 ± 1.2	70.9	3.9
GraphPartition	70.3 ± 0.7	75.2 ± 0.4	63.2 ± 0.3	61.0 ± 0.8	64.2 ± 0.5	75.4 ± 1.7	77.1 ± 0.7	79.6 ± 1.8	70.8	3.8
GraphCL	69.5 ± 0.5	75.4 ± 0.9	63.8 ± 0.4	60.8 ± 0.7	70.1 ± 1.9	74.5 ± 1.3	77.6 ± 0.9	78.2 ± 1.2	71.3	4.3
GraphLoG	72.5 ± 0.8	75.7 ± 0.5	63.5 ± 0.7	61.2 ± 1.1	76.7 ± 3.3	76.0 ± 1.1	77.8 ± 0.8	83.5 ± 1.2	73.4	6.4
GraphMAE	72.1 ± 0.5	75.2 ± 0.6	64.0 ± 0.2	60.1 ± 1.1	82.1 ± 1.1	76.3 ± 2.4	76.9 ± 1.0	83.1 ± 0.7	73.7	6.7
S2GAE	71.8 ± 0.5	75.5 ± 0.8	63.9 ± 0.3	60.6 ± 0.8	80.7 ± 1.9	76.4 ± 1.6	76.4 ± 1.5	80.5 ± 0.9	73.3	6.3
MGSSL	69.8 ± 2.5	75.5 ± 0.6	64.0 ± 0.5	60.9 ± 0.7	71.0 ± 5.5	74.7 ± 1.9	77.6 ± 0.4	82.5 ± 2.7	71.5	4.5
MICRO-Graph	69.5 ± 2.5	75.3 ± 0.6	63.8 ± 0.5	60.6 ± 0.7	70.2 ± 5.5	74.4 ± 1.9	77.5 ± 1.3	82.3 ± 2.7	71.4	4.4
GraphFP$_{C_P}$	71.5 ± 1.0	75.3 ± 0.5	64.3 ± 0.7	60.7 ± 0.6	52.3 ± 2.4	76.3 ± 1.6	76.7 ± 1.0	78.1 ± 0.6	69.4	2.4
GraphFP$_{C_{PF}}$	71.6 ± 0.9	73.4 ± 0.6	64.0 ± 0.6	61.9 ± 0.7	59.7 ± 3.4	74.6 ± 1.1	77.4 ± 1.0	80.7 ± 1.0	70.4	3.4
DBP	**72.8 ± 0.4**	**77.8 ± 0.4**	**65.5 ± 0.3**	**62.5 ± 0.8**	**82.8 ± 1.3**	**77.3 ± 1.1**	**78.8 ± 1.2**	**83.7 ± 1.0**	**74.9**	**7.9**

Bold indicates the best performance while underline indicates the second best on each dataset.

TABLE II
ROC-AUC SCORES (%) ON DOWNSTREAM BIOLOGICAL FUNCTION PREDICTION TASK COMPARED WITH STATE-OF-THE-ART METHODS

Method	ROC-AUC	GAIN	Method	ROC-AUC	GAIN
No Pre-training	64.8 ± 1.0	-	EdgePred	65.9 ± 0.7	1.1
InfoGraph	65.1 ± 0.5	0.3	AttrMasking	65.7 ± 0.5	0.9
ContextPred	65.8 ± 0.3	1.0	GraphPartition	68.7 ± 0.2	3.9
GraphCL	68.9 ± 0.6	4.1	GraphLoG	69.1 ± 0.7	4.3
DBP	**ROC-AUC: 71.2 ± 0.4**		**GAIN: 6.4**		

Bold indicates the best performance while underline indicates the second best.

TABLE III
ROC-AUC SCORES (%) UNDER VARIOUS GNN ARCHITECTURES. ALL RESULTS ARE REPORTED ON BIOLOGY DOMAIN

Method	GCN	GraphSAGE	GAT	GIN
EdgePred	64.7 ± 1.0	67.4 ± 1.5	67.4 ± 1.3	65.9 ± 1.7
AttrMasking	64.4 ± 1.2	64.3 ± 0.8	67.7 ± 1.2	65.7 ± 1.3
ContextPred	64.6 ± 1.4	66.3 ± 0.7	66.9 ± 2.0	66.0 ± 1.2
GraphCL	67.6 ± 1.3	68.6 ± 0.4	67.2 ± 0.7	67.9 ± 0.9
GraphLoG	68.2 ± 0.6	68.8 ± 0.8	67.5 ± 1.0	69.1 ± 0.7
DBP	**70.5 ± 0.7**	**70.2 ± 0.9**	**68.9 ± 1.3**	**71.2 ± 0.4**

Bold indicates the best performance while underline indicates the second best.

and generalizable. We argue that these properties are mainly affected by reasonable information control, which can learn more transferable prior knowledge and transfer them to the fine-tuning phase, thus benefiting more downstream tasks involving fine-grained classification.

Results w.r.t. Different GNN Architectures: Table III compares the performance of DBP and state-of-the-art pre-training baselines, w.r.t. four different GNN architectures: GCN [56], GraphSAGE [57], GAT [58], and GIN [59]. It can be observed that the proposed DBP consistently yields the best performance among all methods across architectures. This demonstrates that our strategy is pluggable and applicable to various GNN architectures. We deem that this improvement over previous works is mainly from the information compression delayed pre-training strategy in DBP, which is not included in existing methods.

C. Analysis of the Training Process

To gain a deeper understanding of our model, we analyzed various aspects of the training dynamics.

Dynamics of Predictive Information: Fig. 3 illustrates the changes in mutual information between the representation Z and the label Y during the fine-tuning phase for different methods. Different values of β represent the varying intensities of the information control during the fine-tuning phase, where the blue line represents the DBP without information control, and AttrMasking represents the traditional masking pre-training strategy. Across different variants (e.g., GCN, GAT, and GIN) and datasets (e.g., BBBP, ToxCast, SIDER, and ClinTox), these comparisons exhibit similar trends. Overall, the DBP method with information control strategies achieves higher mutual information $I(Y;Z)$ during the fine-tuning phase as compared to the traditional methods and DBP with less information control, which demonstrates the universality and effectiveness of our information control strategy across different datasets and models in enhancing the extraction of predictive relevant information during the fine-tuning stage.

Dynamics of Performance Change: Fig. 4 shows the changes in performance during the fine-tuning phase for different methods, using AUC-ROC as the performance metric. Similar to Fig. 3, different values of β represent the varying intensities of information control during the fine-tuning phase, where the blue line represents the DBP without information control, and AttrMasking represents the traditional masking pre-training strategy. Across different variants and datasets, our strategy improves performance, but the performance ceiling is related to the choice of the intensity of information control β. For instance, on the ClinTox dataset, GCN exhibits the highest performance at $\beta = 0.0005$, while GIN performs better at $\beta = 0.1$. We will continue to analyze the sensitivity to β in Section V-D.

Dynamics of Generalization Gap: Fig. 6 demonstrates the changes in the generalization gap during the fine-tuning process for different methods. Similar to Fig. 4, different values of β represent the varying intensities of the information control during the fine-tuning phase, where the blue line represents the DBP without information control, and AttrMasking represents the

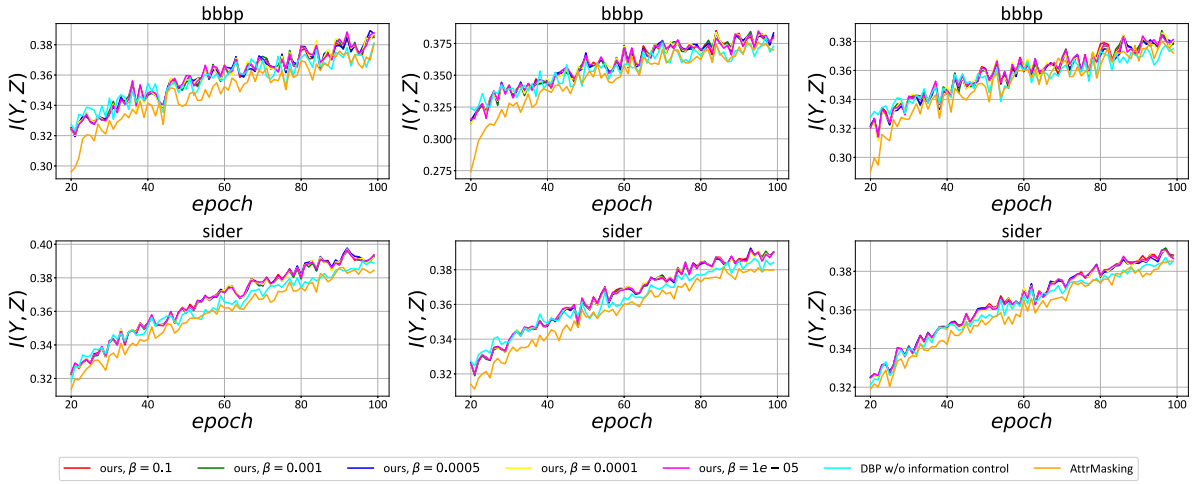

Fig. 3. Dynamics of the mutual information $I(Y, Z)$ between the target labels Y and the learned representations Z across training epochs for different variants on two molecular property prediction datasets (BBBP and SIDER).

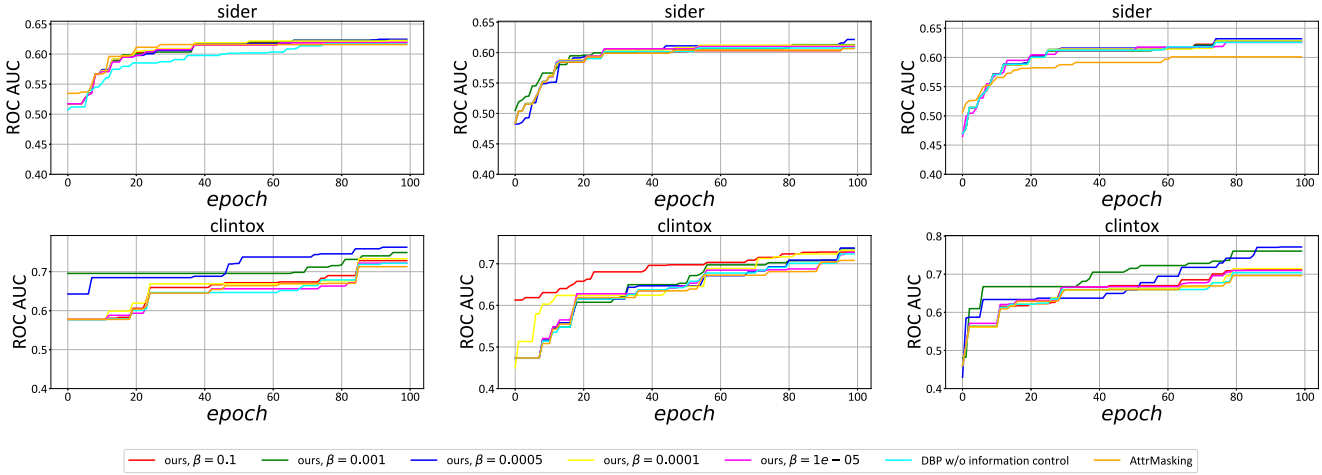

Fig. 4. ROC-AUC curves across training epochs for different variants of the proposed DBP method on SIDER and ClinTox.

traditional masking pre-training strategy. Tests were conducted across different variants and datasets. Overall, our information control strategy achieves a lower generalization error during the fine-tuning phase compared to traditional methods and DBP with less information control. This relates to the conclusions drawn from Fig. 3, as our information control strategy transfers more usable predictive relevant information, leading to the model learning more useful representations and thereby resulting in a smaller generalization error.

D. Further Analysis

Effect of Information Control Objective Function: We attempt to analyze the effect of individually applying the information control objective function in the pre-training or fine-tuning phase on chemistry domain, with the same experimental setups as in Section V-A. As shown in Fig. 5(a), we have an interesting observation: When the pre-trained information control objective function is individually applied, its performance improvement on downstream tasks is limited, but applying the information control objective function independently in the fine-tuning phase can further improve the model performance. The reason might be that the information control objective in the fine-tuning phase is more conducive to improving classification performance, while the additional information retained by the information control module in the pre-training phase needs to be further compressed and selected before being applied to downstream tasks.

Analysis on Information Control Hyperparameters: The hyperparameters α and β that adjust the strength of information control are crucial to DBP, so we further investigate the impact of different hyperparameter combinations on model performance. Experimental results on three downstream chemistry datasets are shown in Fig. 5(b). Interestingly, we observed that the performance of the model on downstream tasks degrades when the hyperparameters are too large or too small. We argue that excessively suppressing information compression in the pre-training

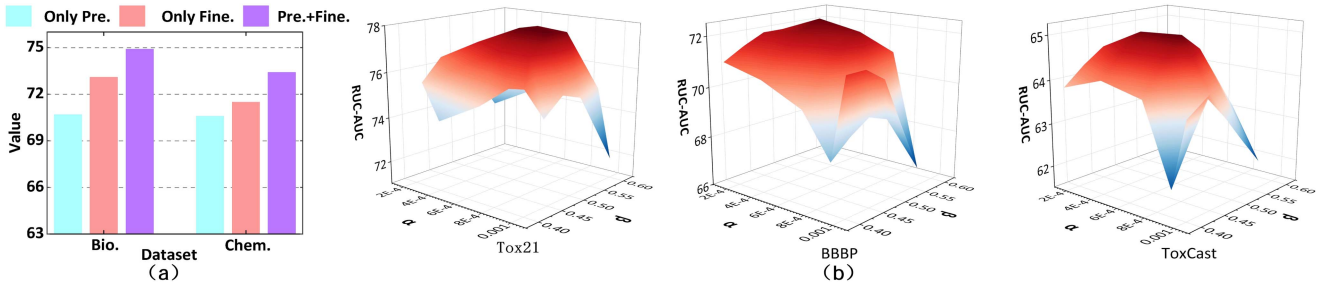

Fig. 5. Hyperparameter sensitivity analysis and ablation study with respect to DBP. Subfigure (a) shows our ablation experiments on the information control modules during the pre-training and fine-tuning stages. Subfigure (b) illustrates our analysis experiments on the relationship between the information control hyperparameters α and β and model performance across three datasets during the pre-training and fine-tuning stages.

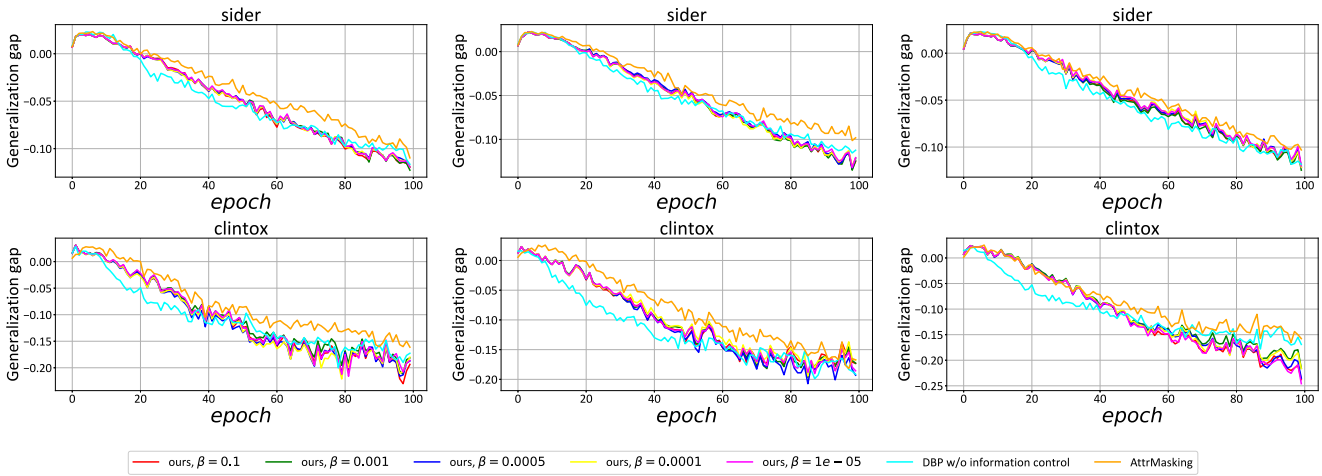

Fig. 6. Generalization gap across training epochs for different variants of the proposed DBP method on four molecular property prediction datasets (BBBP, ToxCast, SIDER, and ClinTox).

phase will interfere with general knowledge extraction, while excessively enhancing information compression in the fine-tuning phase will make the latent representation less informative.

Comparison with Existing Work: As graph pre-training research advances, some studies have noticed the negative impact of differences between pre-training and fine-tuning tasks, although they do not deeply analyze from an information and neural network training behavior perspective. For example, L2P-GNN[67] approaches from a meta-learning perspective, simulating the fine-tuning process during pre-training to quickly adapt to new downstream tasks. GPPT [68] using ideas from natural language processing's prompt learning, proposes a pre-training method focused on node classification, transforming downstream node classification tasks into edge prediction tasks similar to pre-training goals, bridging the gap between pre-training and fine-tuning objectives.

However, these methods primarily aim to reduce target differences during the fine-tuning stage, without analyzing the entire information transfer process across both stages. In contrast, our method analyzes the fundamental knowledge transfer from pre-training datasets to fine-tune downstream tasks from an information compression perspective. This framework could also be combined with downstream fine-tuning process variants like prompt learning and meta-learning in the future.

Complexity Analysis: The DBP framework encompasses pre-training and fine-tuning phases. Pre-training involves self-supervised and reconstruction tasks, both with a computational complexity of $O(V + E)$. Fine-tuning introduces an information compression module and a classifier, each with a complexity of $O(V)$. Overall, DBP maintains a complexity of $O(V + E)$, similar to traditional GNNs.

VI. Conclusion

In this paper, we reexamine the pre-training process within the traditional pre-training and fine-tuning framework from the perspective of Information Bottleneck, and confirm that the forgetting phenomenon in the pre-training phase exactly has detrimental effects on downstream tasks. Then, we propose a DBP framework that maintains as much as possible mutual information during the pre-training phase by suppressing the compression operation and delays the compression operation to fine-tuning phase. To achieve this, we design two information control objectives that can be directly optimized and further integrate them into the model design. Extensive experiments demonstrate the effectiveness and generalization of DBP.

REFERENCES

[1] S. Bhagat, G. Cormode, and S. Muthukrishnan, "Node classification in social networks," in *Social Network Data Analytics*, Berlin, Germany: Springer, 2011, pp. 115–148.

[2] J. He, H. Liu, Y. Zheng, S. Tang, W. He, and X. Du, "Bi-labeled LDA: Inferring interest tags for non-famous users in social network," *Data Sci. Eng.*, vol. 5, pp. 27–47, 2020.

[3] Y. Wang, P. Li, C. Bai, and J. Leskovec, "TEDIC: Neural modeling of behavioral patterns in dynamic social interaction networks," in *Proc. World Wide Web Conf.*, 2021, pp. 693–705.

[4] F. Zhang, J. Zhai, B. He, S. Zhang, and W. Chen, "Understanding co-running behaviors on integrated CPU/GPU architectures," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 3, pp. 905–918, Mar. 2017.

[5] S. Zhang, H. Yin, T. Chen, Q. V. N. Hung, Z. Huang, and L. Cui, "GCN-based user representation learning for unifying robust recommendation and fraudster detection," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 689–698.

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

[7] S. Liu, M. F. Demirel, and Y. Liang, "N-gram graph: Simple unsupervised representation for graphs, with applications to molecules," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 8466–8478.

[8] Z. Wu et al., "Chemistry-intuitive explanation of graph neural networks for molecular property prediction with substructure masking," *Nature Commun.*, vol. 14, no. 1, 2023, Art. no. 2585.

[9] P. Reiser et al., "Graph neural networks for materials science and chemistry," *Commun. Mater.*, vol. 3, no. 1, 2022, Art. no. 93.

[10] Z. Li, K. Meidani, P. Yadav, and A. BaratiFarimani, "Graph neural networks accelerated molecular dynamics," *J. Chem. Phys.*, vol. 156, no. 14, 2022, Art. no. 144103.

[11] H. Wang et al., "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 968–977.

[12] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 974–983.

[13] L. Yang et al., "DGRec: Graph neural network for recommendation with diversified embedding generation," in *Proc. Sixteenth ACM Int. Conf. Web Search Data Mining*, 2023, pp. 661–669.

[14] J. Chang et al., "Sequential recommendation with graph neural networks," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 378–387.

[15] Y. Hao et al., "Multi-dimensional graph neural network for sequential recommendation," *Pattern Recognit.*, vol. 139, 2023, Art. no. 109504.

[16] C. Gao, X. Wang, X. He, and Y. Li, "Graph neural networks for recommender system," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, 2022, pp. 1623–1625.

[17] C. Shi, M. Xu, H. Guo, M. Zhang, and J. Tang, "A graph to graphs framework for retrosynthesis prediction," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 8818–8827.

[18] C. Shi, M. Xu, Z. Zhu, W. Zhang, M. Zhang, and J. Tang, "GraphAF: A flow-based autoregressive model for molecular graph generation," in *Proc. Int. Conf. Learn. Representations*, 2019.

[19] J. You, B. Liu, Z. Ying, V. Pande, and J. Leskovec, "Graph convolutional policy network for goal-directed molecular graph generation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 6412–6422.

[20] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 494–514, Feb. 2022.

[21] P. Wang, C. Ge, Z. Zhou, X. Wang, Y. Li, and Y. Wang, "Joint gated co-attention based multi-modal networks for subregion house price prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 02, pp. 1667–1680, Feb. 2023.

[22] Z. Zhao, P. Wang, H. Wen, Y. Zhang, Z. Zhou, and Y. Wang, "A twist for graph classification: Optimizing causal information flow in graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2024, pp. 17042–17050.

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

[24] Z. Hou et al., "GraphMAE: Self-supervised masked graph autoencoders," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2022, pp. 594–604.

[25] W. Hu et al., "Strategies for pre-training graph neural networks," in *Proc. Int. Conf. Learn. Representations*, 2020.

[26] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 5812–5823.

[27] M. Xu, H. Wang, B. Ni, H. Guo, and J. Tang, "Self-supervised graph-level representation learning with local and global structure," in *Proc. Int. Conf. Mach. Learn.*, 2021.

[28] F.-Y. Sun, J. Hoffman, V. Verma, and J. Tang, "InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," in *Proc. Int. Conf. Learn. Representations*, 2020.

[29] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning automated," in *Proc. Int. Conf. Mach. Learn.*, 2021.

[30] L. Wu, H. Lin, C. Tan, Z. Gao, and S. Z. Li, "Self-supervised learning on graphs: Contrastive, generative, or predictive," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 4216–4235, Apr. 2023.

[31] M. Tran, S. J. Wagner, M. Boxberg, and T. Peng, "S5CL: Unifying fully-supervised, self-supervised, and semi-supervised learning through hierarchical contrastive learning," in *Proc. 25th Int. Conf. Med. Image Comput. Comput. Assist. Intervention*, Singapore, 2022, pp. 99–108.

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

[33] V. Verma et al., "Graphmix: Improved training of GNNs for semi-supervised learning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 10024–10032.

[34] Z. Liu, X. Yu, Y. Fang, and X. Zhang, "GraphPrompt: Unifying pre-training and downstream tasks for graph neural networks," in *Proc. ACM Web Conf.*, 2023, pp. 417–428.

[35] C. Li and Z. Qiu, "Targeted bert pre-training and fine-tuning approach for entity relation extraction," in *Proc.7th Int. Conf. Pioneering Comput. Scientists Engineers Educators Data Sci.*, Taiyuan, China, 2021, pp. 116–125.

[36] D. Wan and M. Bansal, "FactPEGASUS: Factuality-aware pre-training and fine-tuning for abstractive summarization," 2022, *arXiv:2205.07830*.

[37] M. C. Anderson and S. B. Floresco, "Prefrontal-hippocampal interactions supporting the extinction of emotional memories: The retrieval stopping model," *Neuropsychopharmacology*, vol. 47, no. 1, pp. 180–195, 2022.

[38] T. Gruber, L. Luncz, J. Mörchen, C. Schuppli, R. L. Kendal, and K. Hockings, "Cultural change in animals: A flexible behavioural adaptation to human disturbance," *Palgrave Commun.*, vol. 5, no. 1, 2019, Art. no. 64.

[39] T. Kitazono, S. Hara-Kuge, O. Matsuda, A. Inoue, M. Fujiwara, and T. Ishihara, "Multiple signaling pathways coordinately regulate forgetting of olfactory adaptation through control of sensory responses in caenorhabditis elegans," *J. Neurosci.*, vol. 37, no. 42, pp. 10240–10251, 2017.

[40] L. Gravitz, "The importance of forgetting," *Nature*, vol. 19, pp. 1–20, 2019.

[41] A. Achille, M. Rovere, and S. Soatto, "Critical learning periods in deep networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[42] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information (2017)," 2017, *arXiv: 1703.00810*.

[43] J. Li et al., "Enhanced spiking neural network with forgetting phenomenon based on electronic synaptic devices," *Neurocomputing*, vol. 408, pp. 21–30, 2020.

[44] J. Peng et al., "Learning by active forgetting for neural networks," 2021, *arXiv:2111.10831*.

[45] A. Cossu, T. Tuytelaars, A. Carta, L. Passaro, V. Lomonaco, and D. Bacciu, "Continual pre-training mitigates forgetting in language and vision," 2022, *arXiv:2205.09357*.

[46] Z. Feng et al., "Codebert: A pre-trained model for programming and natural languages," 2020, *arXiv: 2002.08155*.

[47] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. Int. Conf. Learn. Representations*, 2019.

[48] S. Liu, H. Wang, W. Liu, J. Lasenby, H. Guo, and J. Tang, "Pre-training molecular graph representation with 3D geometry," 2021, *arXiv:2110.07728*.

[49] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," in *Proc. Int. Conf. Learn. Representations*, 2017.

[50] T. Wu, H. Ren, P. Li, and J. Leskovec, "Graph information bottleneck," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020.

[51] J. Yu, T. Xu, Y. Rong, Y. Bian, J. Huang, and R. He, "Graph information bottleneck for subgraph recognition," in *Proc. Int. Conf. Learn. Representations*, 2020.

[52] Z. Peng et al., "Graph representation learning via graphical mutual information maximization," in *Proc. World Wide Web Conf.*, 2020, pp. 259–270.
[53] T. Sterling and J. J. Irwin, "ZINC 15–ligand discovery for everyone," *J. Chem. Inf. Model.*, vol. 55, pp. 324–337, 2015.
[54] Z. Wu et al., "MoleculeNet: A benchmark for molecular machine learning," *Chem. Sci.*, vol. 9, pp. 513–530, 2018.
[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.
[56] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.
[57] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017.
[58] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[59] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *Proc. Int. Conf. Learn. Representations*, 2019.
[60] N. Liu, S. Jian, D. Li, and H. Xu, "Unsupervised hierarchical graph pooling via substructure-sensitive mutual information maximization," in *Proc. Proc. Conf. Inf. Knowl. Manage.*, 2022, pp. 1299–1308.
[61] Y. You, T. Chen, Z. Wang, and Y. Shen, "When does self-supervision help graph convolutional networks?," in *Proc. Int. Conf. Mach. Learn.*, 2020.
[62] Z. Hou et al., "GraphMAE: Self-supervised masked graph autoencoders," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 594–604.
[63] Q. Tan et al., "S2gae: Self-supervised graph autoencoders are generalizable learners with graph masking," in *Proc. 16th ACM Int. Conf. Web Search Data Mining*, 2023, pp. 787–795.
[64] Z. Zhang, Q. Liu, H. Wang, C. Lu, and C.-K. Lee, "Motif-based graph self-supervised learning for molecular property prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 15870–15882.
[65] S. Zhang, Z. Hu, A. Subramonian, and Y. Sun, "Motif-driven contrastive learning of graph representations," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 8, pp. 4063–4075, Aug. 2024.
[66] K.-D. Luong and A. K. Singh, "Fragment-based pretraining and finetuning on molecular graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 17584–17601.
[67] Y. Lu, X. Jiang, Y. Fang, and C. Shi, "Learning to pre-train graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4276–4284.
[68] M. Sun, K. Zhou, X. He, Y. Wang, and X. Wang, "GPPT: Graph pre-training and prompt tuning to generalize graph neural networks," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 1717–1727.

Zhe Zhao received the bachelor's degree in computer science from Anhui University in 2021. He is currently working toward the doctoral degree with the School of Data Science, University of Science and Technology of China and Computer Science School, City University of Hong Kong. His research interests mainly include data mining, machine learning, and multi-objective optimization.

Pengkun Wang (Member, IEEE) received the PhD degree from the University of Science and Technology of China in 2023. He is now an associate researcher with the Suzhou Institute for Advanced Research, University of Science and Technology of China (USTC). He has published more than 30 papers on top conferences and journals such as *IEEE Transactions on Knowledge and Data Engineering*, AAAI, ICLR, KDD, and WWW. His mainly research interests include imbalanced/long-tailed learning, interpretable data augmentation, multi-objective optimization, and open environment data mining. He also works on applying generalized deep learning to real-world tasks such as urban computing and AI4Science.

Xu Wang received the PhD degree from the University of Science and Technology of China in 2023. He is now an associate researcher with Suzhou Institute for Advanced Research, University of Science and Technology of China (USTC). He has published more than 10 papers on top conferences and journals such as KDD, *IEEE Transactions on Vehicular Technology*, and WSDM. His mainly research interests include spatiotemporal data mining, time series analysis, and AI for science.

Haibin Wen is currently working toward the undergraduate degree with Shaoguan University. His research interests include encompassing high-performance computing and neural networks.

Xiaolong Xie received the graduate degree and an undergraduate degree in 2021 from Nanchang University, where he is currently working toward the graduate degree in software engineering. He works in the Emergency Rescue VR Laboratory, focusing on neural radiation fields and 3D reconstruction. His contributions to various projects have earned him accolades from supervisors and peers.

Zhengyang Zhou (Member, IEEE) received the PhD degree from the University of Science and Technology of China in 2023. He is now an associate researcher with the Suzhou Institute for Advanced Research, University of Science and Technology of China (USTC). He has published more than 30 papers on top conferences and journals such as NeurIPS, ICLR, KDD, *IEEE Transactions on Knowledge and Data Engineering*, WWW, AAAI, and ICDE. His mainly research interests include spatiotemporal data mining, human-centered urban computing, and deep learning generalization with model behavior analysis. He is now especially interested in improving generalization capacity of neural networks for streaming and spatiotemporal data.

Qingfu Zhang (Fellow, IEEE) received the BS degree in mathematics from Shanxi University, Taiyuan, China, in 1984, the MS degree in applied mathematics, and the PhD degree in information engineering from Xidian University, Xi'an, China, in 1991 and 1994, respectively. He is a chair professor of computational intelligence with the Department of Computer Science, City University of Hong Kong, Hong Kong. His main research interests include evolutionary computation, optimization, metaheuristics, and their applications. He was awarded the 2010 *IEEE Transactions on Evolutionary Computation* Outstanding Paper Award. He is an associate editor of the *IEEE Transactions on Evolutionary Computation* and the *IEEE Transactions on Cyberntics*.

Yang Wang (Senior Member, IEEE) received the PhD degree from the University of Science and Technology of China in 2007. He is now an associate professor with the School of Computer Science and Technology, School of Software Engineering, and School of Data Science in USTC. Since then, he keeps working with USTC till now as a postdoc and an associate professor successively. Meanwhile, he also serves as the vice dean with the School of Software Engineering, USTC. His research interest mainly includes wireless (sensor) networks, distribute systems, data mining, and machine learning, and he is also interested in all kinds of applications of AI and data mining technologies, especially in urban computing and AI4Science.