PRE-TRAINING MOLECULAR GRAPH REPRESENTATION WITH 3D GEOMETRY

Anonymous authors
Paper under double-blind review

ABSTRACT

Molecular graph representation learning is a fundamental problem in modern drug and material discovery. Molecular graphs are typically modeled by their 2D topological structures, but it has been recently discovered that 3D geometric information plays a more vital role in predicting molecular functionalities. However, the lack of 3D information in real-world scenarios has significantly impeded the learning of geometric graph representation. To cope with this challenge, we propose the Graph Multi-View Pre-training (GraphMVP) framework where self-supervised learning (SSL) is performed by leveraging the correspondence and consistency between 2D topological structures and 3D geometric views. GraphMVP effectively learns a 2D molecular graph encoder that is enhanced by richer and more discriminative 3D geometry. We further provide theoretical insights to justify the effectiveness of GraphMVP. Finally, comprehensive experiments show that GraphMVP can consistently outperform existing graph SSL methods.

1 INTRODUCTION

In recent years, drug discovery has drawn increasing interest in the machine learning community. Among many challenges therein, how to discriminatively represent a molecule with a vectorized embedding remains a fundamental yet open challenge. The underlying problem can be decomposed into two components: how to design a common latent space for molecule graphs (*i.e.*, designing a suitable encoder) and how to construct an objective function to supervise the training (*i.e.*, defining a learning target). Falling broadly into the second category, our paper studies self-supervised molecular representation learning by leveraging the consistency between 3D geometry and 2D topology.

Motivated by the prominent success of the pretraining-finetuning pipeline (10), unsupervisedly pre-trained graph neural networks for molecules yields promising performance on downstream tasks and becomes increasingly popular (22; 29; 44; 47; 57; 58). The key to pre-training lies in finding an effective proxy task (*i.e.*, training objective) to leverage the power of large unlabeled datasets. Inspired by (31; 41) that molecular properties (13; 29) can be better predicted by 3D geometry due to its encoded energy knowledge, we aim to make use of the 3D geometry of molecules in pre-training. However, the stereochemical structures are often very expensive to obtain, making such 3D geometric information scarce in downstream tasks. To address this problem, we propose the GraphMulti-View Pre-training (GraphMVP) framework, where a 2D molecule encoder is pre-trained with the knowledge of 3D geometry and then fine-tuned on downstream tasks without 3D information.

We attain the aforementioned goal by leveraging two pretext tasks on the 3D and 2D molecular graphs: one contrastive and one generative SSL. Contrastive SSL creates the supervised signal at an **inter-molecule** level: the 3D and 2D graph pairs are positive if they are from the same molecule, and negative otherwise; Then contrastive SSL (50) will align the positive pairs and contrast the negative pairs simultaneously. Generative SSL (19; 27; 48), on the other hand, obtains the supervised signal in an **intra-molecule** way: it learns a 2D/3D representation that can reconstruct its 3D/2D counterpart view for each molecule itself. To cope with the challenge of measuring the quality of reconstruction on molecule 3D and 2D space, we further propose a novel surrogate objective function called variation representation reconstruction (VRR) for the generative SSL task, which can effectively compute such quality in the continuous representation space. The knowledge acquired by these two SSL tasks is complementary, so our GraphMVP framework integrates them to form more discriminative 2D molecular graph representation. Consistent performance improvements empirically validate the effectiveness of GraphMVP.

1.1 PRELIMINARY

2D Molecular Graph represents each molecule as a 2D graph, with atoms as nodes and bonds as edges. We denote it as $g_{2\text{D}} = (X, E)$, where X is the atom attribute matrix and E is the bond attribute matrix. Notice that here E includes both the connectivity and features. Given a 2D molecular graph $g_{2\text{D}}$, its representation h_{2D} can be obtained from a GNN model $h_{2\text{D}} = \text{GNN-2D}(X, E)$.

3D Molecular Graph additionally includes spatial locations of the atoms, which needless to be static since, in real scenarios, atoms are in continual motion on *a potential energy surface* (2). The 3D structures at the local minima on this surface are named *conformer*. As the molecular properties are conformers ensembled (17), GraphMVP enables adopting 3D conformers for learning better representation. Given a conformer $g_{3\text{D}} = (X, R)$, its representation is $h_{3\text{D}} = \text{GNN-3D}(X, R)$, where R is the 3D-coordinate matrix. For notation simplicity, we use $\boldsymbol{x}$ and $\boldsymbol{y}$ afterwards for the 2D and 3D graphs, *i.e.*, $\boldsymbol{x} \triangleq g_{2\text{D}}$ and $\boldsymbol{y} \triangleq g_{3\text{D}}$. Then the latent representations are denoted as $h_{\boldsymbol{x}}$ and $h_{\boldsymbol{y}}$.

2 GRAPHMVP: GRAPH MULTI-VIEW PRE-TRAINING

Our model, termed as Graph Multi-View Pre-training (GraphMVP), is a self-supervised learning approach based on maximizing mutual information (MI) between 3D and 2D views, enabling the learnt representation to capture high-level factors (3; 4; 46) in molecule data. The 3D conformers encode rich information about the molecule energy, which is complementary to the 2D topology. Thus, applying SSL between the 3D and 2D views will provide a better 2D representation.

2.1 MUTUAL INFORMATION AND SELF-SUPERVISED LEARNING

Mutual information (MI) measures the non-linear dependence (4) between two random variables: the larger MI, the stronger dependence between the variables. Therefore for GraphMVP, we can interpret it as maximizing MI between 3D and 2D views: to obtain a more robust 2D/3D representation by sharing more information with its 3D/2D counterparts. We first derive a lower bound for MI (see derivation in Appendix D), and the corresponding objective function $\mathcal{L}_{\text{MI}}$ is

$$I(X;Y) \geq \mathcal{L}_{\text{MI}} = \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\big[\log p(\boldsymbol{y}|\boldsymbol{x}) + \log p(\boldsymbol{x}|\boldsymbol{y})\big]. \tag{1}$$

In GraphMVP, we estimate this lower bound by proposing two modules: one contrastive SSL and one generative SSL. Note that here both $\boldsymbol{x}$ and $\boldsymbol{y}$ are structured data, *i.e.*, 2D and 3D molecular graphs, which brings in extra obstacles in learning. We will discuss how to tackle them below.

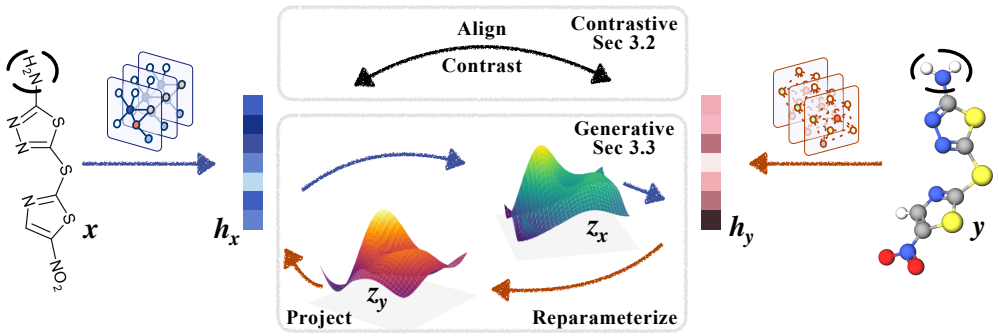

Figure 1: Overview of GraphMVP. The black dashed circles represent subgraph masking.

2.2 CONTRASTIVE SELF-SUPERVISED LEARNING BETWEEN 3D AND 2D VIEWS

Energy-Based Model with Noise Contrastive Estimation (EBM-NCE). If we model the conditional likelihood in Equation (1) with energy-based model (EBM), and then solve it Noise-Contrastive

Estimation (NCE) (15), this will give us the following objective (derivations in Appendix E.2):

$$\begin{aligned}\mathcal{L}_{\text{EBM-NCE}} = &-\frac{1}{2}\mathbb{E}_{p(\boldsymbol{y})}\Big[\mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})}\log\big(1-\sigma(f_x(\boldsymbol{x},\boldsymbol{y}))\big)+\mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{y})}\log\sigma(f_x(\boldsymbol{x},\boldsymbol{y}))\Big]\\ &-\frac{1}{2}\mathbb{E}_{p(\boldsymbol{x})}\Big[\mathbb{E}_{p_n(\boldsymbol{y}|\boldsymbol{x})}\log\big(1-\sigma(f_y(\boldsymbol{y},\boldsymbol{x}))\big)+\mathbb{E}_{p(\boldsymbol{y},\boldsymbol{x})}\log\sigma(f_y(\boldsymbol{y},\boldsymbol{x}))\Big],\end{aligned} \tag{2}$$

where $f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}) = f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}) = \exp(\langle h_{\boldsymbol{x}}, h_{\boldsymbol{y}}\rangle)$, p_n is the noise distribution and σ is the sigmoid function. We take this as the contrastive SSL loss, *i.e.*, $\mathcal{L}_{\text{C}} = \mathcal{L}_{\text{EBM-NCE}}$. We also notice that the final formulation of EBM-NCE shares certain similarities with Jensen-Shannon estimation (JSE) (35). However, the *derivation process and underlying intuition* are different: EBM-NCE models the conditional distribution in MI lower bound (Equation (1)) with EBM, while JSE is a special case of variational estimation of f-divergence. Besides, EBM-NCE shares more flexibilites and provides more sampling options under the maximizing MI framework. We expand the a more comprehensive comparison in Appendix E, plus some potential benefits with EBM-NCE.

2.3 Generative Self-Supervised Learning between 3D and 2D Views

Variational Molecule Reconstruction. One alternative solution is to use a variational lower bound to approximate the conditional log-likelihood terms in Equation (1). To this end, we propose a variational generative SSL, equipped with a *crafty* surrogate loss, which we describe follows. Take one direction for illustration, when generating 3D conformers from their corresponding 2D topology, we want to model the conditional likelihood $p(\boldsymbol{y}|\boldsymbol{x})$. By introducing a reparameterized variable $\boldsymbol{z}_{\boldsymbol{x}} = \mu_{\boldsymbol{x}} + \sigma_{\boldsymbol{x}} \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$ and $\mu_{\boldsymbol{x}}$ and $\sigma_{\boldsymbol{x}}$ are two flexible functions on $h_{\boldsymbol{x}}$, we have a lower bound on the conditional likelihood in Equation (1):

$$\log p(\boldsymbol{y}|\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}\big[\log p(\boldsymbol{y}|\boldsymbol{z}_{\boldsymbol{x}})\big] - KL(q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})||p(\boldsymbol{z}_{\boldsymbol{x}})). \tag{3}$$

The expression is similar for $\log p(\boldsymbol{x}|\boldsymbol{y})$. The above objective is composed of a conditional log-likelihood and a KL-divergence. This term has also been recognized as the *reconstruction term*: it is essentially to reconstruct the 3D conformers ($\boldsymbol{y}$) from the sampled 2D molecular graph representation ($\boldsymbol{z}_{\boldsymbol{x}}$). However, performing the reconstruction on the structured data space is not easy: since molecules are discrete, modeling and measuring on the molecule space are difficult.

Variational Representation Reconstruction (VRR). To cope with this challenge, we propose a novel surrogate loss by transferring the reconstruction from data space to representation space. Instead of decoding the latent code $z_{\boldsymbol{x}}$ to data space, we can directly project it to the 3D representation space, denoted as $q_{\boldsymbol{x}}(z_{\boldsymbol{x}})$. Since the representation space is continuous, we may as well model the conditional log-likelihood with Gaussian distribution, resulting in L2 distance for reconstruction, *i.e.*, $\|q_{\boldsymbol{x}}(z_{\boldsymbol{x}}) - \text{SG}(h_{\boldsymbol{y}}(\boldsymbol{y}))\|^2$. Here SG is short for stop-gradient, assuming that $h_{\boldsymbol{y}}$ is a fixed learnt representation function, which has been widely adopted in the SSL literature (8; 14). We term this surrogate loss as VRR, and we take it for the generative SSL loss (derivations are in Appendix F):

$$\begin{aligned}\mathcal{L}_{\text{G}} = \mathcal{L}_{\text{VRR}} = &\frac{1}{2}\Big[\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}\big[\|q_{\boldsymbol{x}}(\boldsymbol{z}_{\boldsymbol{x}}) - \text{SG}(h_{\boldsymbol{y}})\|^2\big] + \mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{y}}|\boldsymbol{y})}\big[\|q_{\boldsymbol{y}}(\boldsymbol{z}_{\boldsymbol{y}}) - \text{SG}(h_{\boldsymbol{x}})\|_2^2\big]\Big]\\ &+\frac{\beta}{2}\cdot\Big[KL(q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})||p(\boldsymbol{z}_{\boldsymbol{x}})) + KL(q(\boldsymbol{z}_{\boldsymbol{y}}|\boldsymbol{y})||p(\boldsymbol{z}_{\boldsymbol{y}}))\Big].\end{aligned} \tag{4}$$

Note that MI is invariant to continuous bijective function (4). Thus, this surrogate loss would be exact if the encoding function h satisfies this condition. However, we find that GNN, though does not meet the condition, can provide robust performance, which empirically justify the effectiveness of VRR.

A Unified View. Here following the definition of VRR, we would like to provide a unified view on the generative SSL. (1) We can do reconstruction to the data space as Equation (3). (2) We can do reconstruction to the representation as VRR Equation (4). (2.a) If we remove the stochasticity in VRR, then it is simply the representation reconstruction (RR), as will be tested in the ablation study Appendix C.3. (2.b) If we make the two views *share the same representation function*, like CNN for multi-view learning on images, then it is reduced to the non-contrastive SSL (8; 14). In other words, *these non-contrastive SSL methods are indeed special cases of VRR.*

2.4 Multi-task Objective Function

At the SSL pre-training stage, we design the above two pretext tasks: one contrastive and one generative. We conjecture then empirically prove that these two tasks are focusing on different

learning aspects, which are concluded into following two points. (1) From the perspective of representation learning, contrastive SSL is learning from **inter-data** and generative SSL is learning by **intra-data**. For contrastive SSL, one key step is to obtain the negative view pairs from inter-data for contrasting; while generative SSL focuses on each data point itself, by reconstructing the key features at the intra-data level. (2) From the perspective of distribution learning, contrastive SSL and generative SSL are learning the data distribution from **local** and **global** manner, respectively. Contrastive SSL learns the distribution locally by contrasting the pairwise distance at the inter-data level. Thus, with sufficient number of data, the local contrastive operation can iteratively recover the data distribution. Generative SSL, on the other hand, learns the global data density function directly.

Therefore, contrastive and generative SSL are essentially conducting representation and distribution learning with different intuitions and disciplines, and we expect that combining these two can lead to better representation. We later carry out an ablation study (Appendix C.3) to verify this empirically. Thus we arrive at minimizing the following complete objective for GraphMVP:

$$\mathcal{L}_{\text{GraphMVP}} = \alpha_1 \cdot \mathcal{L}_{\text{C}} + \alpha_2 \cdot \mathcal{L}_{\text{G}}, \tag{5}$$

where α_1, α_2 are weighting coefficients. A later performed ablation study (Appendix C.3) delivers two important messages: (1) Both individual contrastive and generative SSL on 3D conformers can consistently help improve the 2D representation learning; (2) Combining the two SSL strategies can yield further improvements. Thus, we draw the conclusion that GraphMVP (Equation (5)) is able to obtain an augmented 2D representation by fully utilizing the 3D information.

3 Experiments

Datasets. For pre-training datasets, we take 50k molecules from GEOM (2). As mentioned before, conformer ensemble can better reflect the molecular property, so we take $C = 5$ conformers for each molecule. For downstream tasks, we follow the mainstream research line (22; 57; 58) on exploring 8 molecular property prediction tasks.

Backbone models. We use Graph Isomorphism Network (GIN) (54) for 2D molecular modeling and SchNet (41) for 3D geometric modeling.

Baselines. Due to the rapid growth of this field (30; 32; 51; 53), we are only able to test the most well-acknowledged SSL methods from the accepted works in top machine learning conferences. To be more specific, we carry out comprehensive experiments by considering 7 SSL baselines, which are *all* operated on 2D GNN graph, including EdgePred (16), AttrMask (22), GPT-GNN (23), InfoGraph (44), ContextPred (22), and JOAO (57).

Preliminary results. As observed in Table 1, we can first tell that these downstream tasks are very hard, and there's no overwhelming best SSL model. However, we can still see GraphMVP can obtain a fairly large performance gain w.r.t. the overall performance. This preliminary results help support the effectiveness of GraphMVP, and we will continue exploring further along this direction.

Table 1: Results for eight molecular property prediction tasks (classification). For each downstream task, we report the mean (and standard deviation) ROC-AUC of 3 seeds with scaffold splitting. For GraphMVP , we set $M = 0.15$ and $C = 5$.

Pre-training	BBBP	Tox21	ToxCast	Sider	ClinTox	MUV	HIV	Bace	Avg
–	65.4(2.4)	74.9(0.8)	61.6(1.2)	58.0(2.4)	58.8(5.5)	71.0(2.5)	75.3(0.5)	72.6(4.9)	67.21
EdgePred	64.5(3.1)	74.5(0.4)	60.8(0.5)	56.7(0.1)	55.8(6.2)	73.3(1.6)	75.1(0.8)	64.6(4.7)	65.64
AttrMask	70.2(0.5)	74.2(0.8)	62.5(0.4)	60.4(0.6)	68.6(9.6)	73.9(1.3)	74.3(1.3)	77.2(1.4)	70.16
GPT-GNN	64.5(1.1)	75.3(0.5)	62.2(0.1)	57.5(4.2)	57.8(3.1)	76.1(2.3)	75.1(0.2)	77.6(0.5)	68.27
InfoGraph	69.2(0.8)	73.0(0.7)	62.0(0.3)	59.2(0.2)	75.1(5.0)	74.0(1.5)	74.5(1.8)	73.9(2.5)	70.10
ContextPred	71.2(0.9)	73.3(0.5)	62.8(0.3)	59.3(1.4)	73.7(4.0)	72.5(2.2)	75.8(1.1)	78.6(1.4)	70.89
JOAO	66.0(0.6)	74.4(0.7)	62.7(0.6)	60.7(1.0)	66.3(3.9)	77.0(2.2)	76.6(0.5)	72.9(2.0)	69.57
GraphMVP	68.5(0.2)	74.5(0.4)	62.7(0.1)	62.3(1.6)	79.0(2.5)	75.0(1.4)	74.8(1.4)	76.8(1.1)	**71.69**

REFERENCES

[1] Michael Arbel, Liang Zhou, and Arthur Gretton. Generalized energy based models. *arXiv preprint arXiv:2003.05033*, 2020. 15, 16

[2] Simon Axelrod and Rafael Gomez-Bombarelli. Geom: Energy-annotated molecular conformations for property prediction and molecular generation. *arXiv preprint arXiv:2006.05531*, 2020. 2, 4, 9, 10

[3] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019. 2

[4] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International Conference on Machine Learning*, pages 531–540. PMLR, 2018. 2, 3, 18

[5] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 8

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020. 8

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on Machine Learning*, pages 1597–1607, 2020.

[8] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 3, 8, 18

[9] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *arXiv preprint arXiv:2004.05718*, 2020. 8

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1, 8

[11] Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. Improved contrastive divergence training of energy based models. *arXiv preprint arXiv:2012.01316*, 2020. 14, 16

[12] Fabian B Fuchs, Daniel E Worrall, Volker Fischer, and Max Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *arXiv preprint arXiv:2006.10503*, 2020. 8, 9, 10

[13] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272. PMLR, 2017. 1, 9

[14] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 3, 18

[15] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010. 3, 14

[16] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems, NeurIPS*, 2017. 4, 8, 10

[17] Paul CD Hawkins. Conformation generation: the state of the art. *Journal of Chemical Information and Modeling*, 57(8):1747–1756, 2017. 2, 9

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 8

[19] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. 1, 18

[20] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 16

[21] Qianjiang Hu, Xiao Wang, Wei Hu, and Guo-Jun Qi. Adco: Adversarial contrast for efficient learning of unsupervised representations from self-trained negative adversaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1074–1083, 2021. 16

[22] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations, ICLR*, 2020. 1, 4, 8, 9, 10

[23] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. Gpt-gnn: Generative pre-training of graph neural networks. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD*, pages 1857–1867, 2020. 4, 8, 10

[24] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, pages 1–11, 2021. 8, 9

[25] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020. 16

[26] Diederik P Kingma and Prafulla Dhariwal. Glow: generative flow with invertible $1\times$ 1 convolutions. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10236–10245, 2018. 17

[27] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1

[28] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European conference on computer vision*, pages 577–593, 2016. 17

[29] Shengchao Liu, Mehmet Furkan Demirel, and Yingyu Liang. N-gram graph: Simple unsupervised representation for graphs, with applications to molecules. *arXiv preprint arXiv:1806.09206*, 2018. 1, 8, 9

[30] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021. 4, 8, 10, 13

[31] Yi Liu, Limei Wang, Meng Liu, Xuan Zhang, Bora Oztekin, and Shuiwang Ji. Spherical message passing for 3d graph networks. *arXiv preprint arXiv:2102.05013*, 2021. 1, 8, 9, 10

[32] Yixin Liu, Shirui Pan, Ming Jin, Chuan Zhou, Feng Xia, and Philip S Yu. Graph self-supervised learning: A survey. *arXiv preprint arXiv:2103.00111*, 2021. 4, 8, 10, 13, 16

[33] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*, 2012. 15

[34] Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. Survae flows: Surjections to bridge the gap between vaes and flows. *Advances in Neural Information Processing Systems*, 33, 2020. 17

[35] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 271–279, 2016. 3, 16

[36] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 8, 13

[37] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019. 16

[38] Zhuoran Qiao, Anders S Christensen, Frederick R Manby, Matthew Welborn, Anima Anandkumar, and Thomas F Miller III. Unite: Unitary n-body tensor equivariant network with applications to quantum chemistry. *arXiv preprint arXiv:2105.14655*, 2021. 10

[39] 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 *Advances in Neural Information Processing Systems, NeurIPS*, 2020. 10

[40] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021. 8, 9, 10

[41] Kristof T Schütt, Pieter-Jan Kindermans, Huziel E Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *arXiv preprint arXiv:1706.08566*, 2017. 1, 4, 8, 9, 10

[42] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021. 14, 15, 16

[43] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 14, 16

[44] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *International Conference on Learning Representations, ICLR*, 2020. 1, 4, 8, 10, 16

[45] Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. *arXiv preprint arXiv:2102.06810*, 2021. 18

[46] Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. *arXiv preprint arXiv:1907.13625*, 2019. 2

[47] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018. 1, 8

[48] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008. 1

[49] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matthew J. Kusner. Unsupervised point cloud pre-training via view-point occlusion, completion. In *ICCV*, 2021. 8

[50] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning, ICML*, 2020. 1, 8, 13

[51] Lirong Wu, Haitao Lin, Zhangyang Gao, Cheng Tan, Stan Li, et al. Self-supervised on graphs: Contrastive, generative, or predictive. *arXiv preprint arXiv:2105.07342*, 2021. 4, 8, 10, 13, 16

[52] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018. 9

[53] Yaochen Xie, Zhao Xu, Jingtun Zhang, Zhengyang Wang, and Shuiwang Ji. Self-supervised learning of graph neural networks: A unified review. *arXiv preprint arXiv:2102.10757*, 2021. 4, 8, 10, 13, 16

[54] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018. 4, 8, 9, 10

[55] Minghao Xu, Hang Wang, Bingbing Ni, Hongyu Guo, and Jian Tang. Self-supervised graph-level representation learning with local and global structure. In *International Conference on Machine Learning, ICML*, 2021. 8, 10

[56] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, et al. Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 59(8):3370–3388, 2019. 8

[57] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In *International Conference on Machine Learning, ICML*, 2021. 1, 4, 8, 9, 10

[58] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems, NeurIPS*, 2020. 1, 4, 8, 9, 10

A SELF-SUPERVISED LEARNING ON MOLECULAR GRAPH

Self-supervised learning (SSL) methods have attracted massive attention recently, trending from vision (6–8; 18; 49), language (5; 10; 36) to graph (22; 29; 44; 47; 57; 58). In general, there are two categories of SSL: contrastive and generative, where they differ on the design of the supervised signals. Contrastive SSL realizes the supervised signals at the **inter-data** level, learning the representation by contrasting with other data points; while generative SSL focuses on reconstructing the original data at the **intra-data** level. Both venues have been explored (30; 32; 51; 53) on the graph applications.

A.1 CONTRASTIVE GRAPH SSL

Contrastive graph SSL first applies transformations to construct different *views* for each graph. Each view incorporates different granularities of information, like node-, subgraph-, and graph-level. It then solves two sub-tasks simultaneously: (1) aligning the representations of views from the same data; (2) contrasting the representations of views from different data, leading to a uniformly distributed latent space (50). The key difference among existing methods is thus the design of view constructions. InfoGraph (44; 47) contrasted the node (local) and graph (global) views. As an extension, GraphLoG (55) learned the context (subgraph or motif) view using clustering and contrasted it with both node and graph views. ContextPred (22) contrasted between node and context views. GraphCL and JOAO (57; 58) made comprehensive comparisons among four graph-level transformations and further learned to select the most effective combinations.

A.2 GENERATIVE GRAPH SSL

Generative graph SSL aims at reconstructing important structures for each graph. By so doing, it consequently learns a representation capable of encoding key ingredients of the data. EdgePred (16) and AttrMask (22) predicted the adjacency matrix and masked tokens (nodes and edges) respectively. GPT-GNN (23) reconstructed the whole graph in an auto-regressive approach.

Recall that all previous methods **merely** focus on the 2D topology. However, for science-centric tasks such as molecular property prediction, 3D geometry should be incorporated as it provides complementary and comprehensive information (31; 41). To mitigate this gap, we propose GraphMVP to leverage the 3D geometry with unsupervised graph pre-training.

B MOLECULAR GRAPH REPRESENTATION

Graph neural network (GNN) has become the mainstream modeling methods for molecular graph representation. Existing methods can generally be split into two venues: 2D GNN and 3D GNN, depending on what levels of information is being considered. 2D GNN focuses on the topological structures of the graph, like the adjacency among nodes, while 3D GNN is able to model the "energy" of molecules by taking account the spatial positions of atoms.

First, we want to highlight that GraphMVP is model-agnostic, *i.e.*, it can be applied to any 2D and 3D GNN representation function, yet the specific 3D and 2D representations are not the main focus of this work. Second, we acknowledge there are a lot of advanced 3D (12; 24; 31; 40) and 2D (9; 29; 56) representation methods. However, considering the *graph SSL literature* and *graph representation liteature* (illustrated below), we adopt GIN (54) and SchNet (41) in current GraphMVP.

B.1 2D MOLECULAR GRAPH NEURAL NETWORK

The 2D representation is taking each molecule as a 2D graph, with atoms as nodes and bonds as edges, *i.e.*, $g_{2D} = (X, E)$. $X \in \mathbb{R}^{n \times d_n}$ is the atom attribute matrix, where n is the number of atoms (nodes) and d_n is the atom attribute dimension. $E \in \mathbb{R}^{m \times d_e}$ is the bond attribute matrix, where m is the number of bonds (edges) and d_m is the bond attribute dimension. Notice that here E also includes the connectivity. Then we will apply one transformation function T_{2D} on the topological graph. Given a 2D graph g_{2D}, its 2D molecular representation is:

$$h_{2D} = \text{GNN-2D}(T_{2D}(g_{2D})) = \text{GNN-2D}(T_{2D}(X, E)). \tag{6}$$

The core operation of 2D GNN is the message passing function (13), which updates the node representation based on adjacency information. We have variants depending on the design of message and aggregation functions, and we pick GIN (54) in this work.

GIN There has been a long research line on 2D graph representation learning (13; 29; 54). Among these, graph isomorphism network (GIN) model (54) has been widely used as the backbone in recent graph self-supervised learning work (22; 57; 58). Thus, we as well adopt GIN as the base model for 2D representation.

Recall each molecule is represented as a molecular graph, *i.e.*, $g_{\text{2D}} = (X, E)$, where X and E are feature matrices for atoms and bonds respectively. Then the message passing function is defined as:

$$z_i^{(k+1)} = \text{MLP}_{\text{atom}}^{(k+1)}\Big(z_i^{(k)} + \sum_{j \in \mathcal{N}(i)} \big(z_j^{(k)} + \text{MLP}_{\text{bond}}^{(k+1)}(E_{ij})\big)\Big), \tag{7}$$

where $z_0 = X$ and $\text{MLP}_{\text{atom}}^{(k+1)}$ and $\text{MLP}_{\text{bond}}^{(k+1)}$ are the $(l+1)$-th MLP layers on the atom- and bond-level respectively. Repeating this for K times, and we can encode K-hop neighborhood information for each center atom in the molecular data, and we take the last layer for each node/atom representation. The graph-level molecular representation is the mean of the node representation:

$$z(\boldsymbol{x}) = \frac{1}{N} \sum_i z_i^{(K)} \tag{8}$$

B.2 3D Molecular Graph Neural Network

Recently, the 3D geometric representation learning has brought breakthrough progress in molecule modeling (12; 24; 31; 40; 41). 3D Molecular Graph additionally includes spatial locations of the atoms, which needless to be static since, in real scenarios, atoms are in continual motion on *a potential energy surface* (2). The 3D structures at the local minima on this surface are named *molecular conformation* or *conformer*. As the molecular properties are a function of the conformer ensembles (17), this reveals another limitation of existing mainstream methods: to predict properties from a single 2D or 3D graph cannot account for this fact (2), while our proposed method can alleviate this issue to a certain extent.

For specific 3D molecular graph, it additionally includes spatial positions of the atoms. We represent each conformer as $g_{\text{3D}} = (X, R)$, where $R \in \mathbb{R}^{n \times 3}$ is the 3D-coordinate matrix, and the corresponding representation is:

$$h_{\text{3D}} = \text{GNN-3D}(T_{\text{3D}}(g_{\text{3D}})) = \text{GNN-3D}(T_{\text{3D}}(X, R)), \tag{9}$$

where R is the 3D-coordinate matrix and T_{3D} is the 3D transformation. Note that further information such as plane and torsion angles can be solved from the positions.

SchNet SchNet (41) is composed of the following key steps:

$$\begin{aligned} z_i^{(0)} &= \text{embedding}(x_i) \\ z_i^{(t+1)} &= \text{MLP}\Big(\sum_{j=1}^{n} f(x_j^{(t-1)}, r_i, r_j)\Big) \\ h_i &= \text{MLP}(z_i^{(K)}), \end{aligned} \tag{10}$$

where K is the number of hidden layers, and

$$f(x_j, r_i, r_j) = x_j \cdot e_k(r_i - r_j) = x_j \cdot \exp(-\gamma \| \|r_i - r_j\|_2 - \mu\|_2^2) \tag{11}$$

is the continuous-filter convolution layer, enabling the modeling of continuous positions of atoms.

We adopt SchNet for the following reasons. (1) SchNet is a very strong geometric representation method after *fair* benchmarking. (2) SchNet can be trained more efficiently, comparing to the other recent 3D models. To support these two points, we make a comparison among the most recent 3D geometric models (12; 31; 40) on QM9 dataset. QM9 (52) is a molecule dataset approximating 12 thermodynamic properties calculated by density functional theory (DFT) algorithm. Notice:

UNiTE (38) is the state-of-the-art 3D GNN, but it requires a commercial software for feature extraction, thus we exclude it for now.

Table 2: Reproduced MAE on QM9. 100k for training, 17,748 for val, 13,083 for test. The last column is the approximated running time.

	alpha	gap	homo	lumo	mu	cv	g298	h298	r2	u298	u0	zpve	time
SE(3)-Trans (12)	0.143	59	36	36	0.052	0.068	68	72	1.969	68	74	5.517	15h
SchNet (41)	0.077	50	32	26	0.030	0.032	15	14	0.122	14	14	1.751	3h
EGNN (40)	0.075	49	29	26	0.030	0.032	11	10	0.076	10	10	1.562	36h
SphereNet (31)	0.054	41	22	19	0.028	0.027	10	8	0.295	8	8	1.401	50h

Table 2 shows that, under a fair comparison (w.r.t. data splitting, seed, cuda version, etc), SchNet can reach pretty comparable performance, yet the efficiency of SchNet is much better. Combining these two points, we adopt SchNet in current version of GraphMVP.

C Experiments

C.1 Experimental Settings

Datasets. We pre-train models on the same dataset then fine-tune on the wide range of downstream tasks. We randomly select 50k qualified molecules from GEOM (2) with both 2D and 3D structures for the pre-training. Clarified in Section 1.1, conformer ensembles can better reflect the molecular property, thus we take C conformers of each molecule. For downstream tasks, we first stick to the same setting of the main graph SSL work (22; 57; 58), exploring 8 binary molecular property prediction tasks, which are all in the low-data regime. Then we explore 6 regression tasks from various low-data domains to be more comprehensive.

2D GNN. We follow the research line of SSL on molecule graph (22; 57; 58), using the same Graph Isomorphism Network (GIN) (54) as the backbone model, with the same feature sets.

3D GNN. We choose SchNet (41) for geometric modeling, since SchNet: (1) is found to be a strong geometric representation learning method with *fair* benchmarking; (2) can be trained more efficiently, comparing to the other recent 3D models. We provide details in Appendix B.2.

C.2 Main Results on Molecular Property Prediction.

We carry out comprehensive comparisons with 10 SSL baselines and random initialization. For pre-training, we apply all SSL methods on the same dataset based on GEOM (2). For fine-tuning, we follow the same setting (22; 57; 58) with 8 low-data molecular property prediction tasks.

Baselines. Due to the rapid growth of graph SSL (30; 32; 51; 53), we are only able to benchmark the most well-acknowledged, peer-reviewed baselines: EdgePred (16), InfoGraph (44), GPT-GNN (23), AttrMask & ContextPred(22), GraphLoG(55), G-{Contextual, Motif}(39), GraphCL(58), JOAO(57).

Our method. GraphMVP has two key factors: i) masking ratio (M) and ii) number of conformers for each molecule (C). We set $M = 0.15$ and $C = 5$ by default. For EBM-NCE loss, we adopt the empirical distribution for noise distribution.

C.3 Ablation Study: The Effect of Objective Function

Table 3: Ablation on the objective function.

GraphMVP Loss	Contrastive	Generative	Avg
Random			67.21
InfoNCE only	✓		68.85
EBM-NCE only	✓		70.15
VRR only		✓	69.29
RR only		✓	68.89
InfoNCE + VRR	✓	✓	70.67
EBM-NCE + VRR	✓	✓	71.69
InfoNCE + RR	✓	✓	70.60
EBM-NCE + RR	✓	✓	70.94

In Section 2, we introduce a new contrastive learning objective family called EBM-NCE, and we take either InfoNCE and EBM-NCE as a contrastive loss. For the generative SSL task, we propose a novel objective function called variational representation reconstruction (VRR) in Equation (4). As discussed in Section 2.3, stochasticity is important for GraphMVP since it can capture the conformer distribution for each 2D molecular graph. To verify this, we add an ablation study on *representation reconstruction (RR)* by removing stochasticity in VRR. Thus, here we deploy an ablation study to explore the effect for each individual objective function (InfoNCE, EBM-NCE, VRR and RR), followed by the pairwise combinations between them.

The results in Table 3 give certain constructive insights as follows: (1) Each individual SSL objective function (middle block) can lead to better performance. This strengthens the claim that adding 3D information is helpful for 2D representation learning. (2) According to the combination of those SSL objective functions (bottom block), adding both contrastive and generative SSL can consistently improve the performance. This verifies our claim that conducting SSL at both the inter-data and intra-data level is beneficial. (3) We can see VRR is consistently better than RR on all settings, which verify that stochasticity is an important factor in modeling 3D conformers for molecules.

D MAXIMIZE MUTUAL INFORMATION

In what follows, we will use X and Y to denote the data space for $2D$ graph and $3D$ graph respectively. Then the latent representations are denoted as $h_{\boldsymbol{x}}$ and $h_{\boldsymbol{y}}$.

D.1 FORMULATION

The standard formulation for mutual information (MI) is

$$I(X;Y) = \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\big[\log \frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}\big]. \tag{12}$$

Another well-explained MI inspired from wikipedia is given in Figure 2.

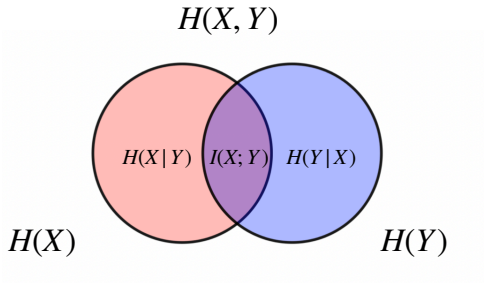

Figure 2: Venn diagram of mutual information. Inspired by wikipedia.

Mutual information (MI) between random variables measures the corresponding non-linear dependence. As can be seen in the first equation in Equation (12), the larger the divergence between the joint ($p(\boldsymbol{x},\boldsymbol{y})$ and the product of the marginals $p(\boldsymbol{x})p(\boldsymbol{y})$, the stronger the dependence between X and Y.

Thus, following this logic, maximizing MI between 3D and 2D views can force the 3D/2D representation to capture higher-level factors, *e.g.*, the occurrence of important substructure that is semantically vital for downstream tasks. Or equivalently, maximizing MI can decrease the uncertainty in 2D representation given 3D geometric information.

D.2 A LOWER BOUND TO MI

To solve MI, we first extract a lower bound:

$$\begin{aligned}
I(X;Y) &= \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big[\log \frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}\Big] \\
&\geq \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big[\log \frac{p(\boldsymbol{x},\boldsymbol{y})}{\sqrt{p(\boldsymbol{x})p(\boldsymbol{y})}}\Big] \\
&= \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big[\log \frac{(p(\boldsymbol{x},\boldsymbol{y}))^2}{p(\boldsymbol{x})p(\boldsymbol{y})}\Big] \\
&= \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big[\log p(\boldsymbol{x}|\boldsymbol{y})\Big] + \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big[\log p(\boldsymbol{y}|\boldsymbol{x})\Big] \\
&= -\frac{1}{2}[H(Y|X) + H(X|Y)].
\end{aligned} \tag{13}$$

Thus, *maximizing* MI is equivalent to *minimizing* the following objective function:

$$\mathcal{L}_{\text{MI}} = \frac{1}{2}[H(Y|X) + H(X|Y)] \tag{14}$$

In the following sections, we will describe two self-supervised learning methods for solving MI. Notice that the methods are very general, and can be applied to various applications. Here we apply it mainly for making 3D geometry useful for 2D representation learning on molecules.

E CONTRASTIVE SELF-SUPERVISED LEARNING

The essence of contrastive self-supervised learning is to align positive view pairs and contrast negative view pairs, such that the obtained representation space is well distributed (50). We display the pipeline in Figure 3. Along the research line in graph SSL (30; 32; 51; 53), InfoNCE and EBM-NCE are the two most-widely used, as discussed below.

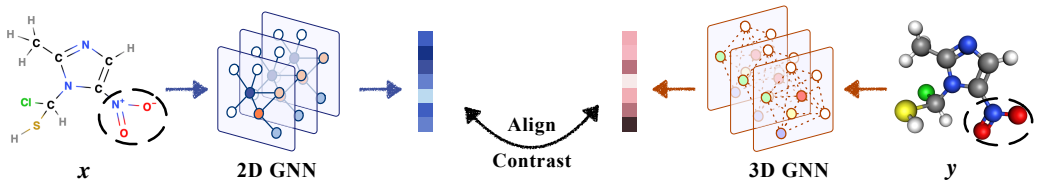

Figure 3: Contrastive SSL in GraphMVP. The black dashed circles represent subgraph masking.

E.1 INFONCE

InfoNCE (36) is first proposed to approximate MI Equation (12):

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{2}\mathbb{E}\left[\log \frac{\exp(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))}{\exp(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y})) + \sum_j \exp(f_{\boldsymbol{x}}(\boldsymbol{x}^j,\boldsymbol{y})} + \log \frac{\exp(f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}))}{\exp(f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x})) + \sum_j \exp f_{\boldsymbol{y}}(\boldsymbol{y}^j,\boldsymbol{x})}\right], \tag{15}$$

where $\boldsymbol{x}^j, \boldsymbol{y}^j$ are randomly sampled 3D and 2D views regarding to the anchored pair $(\boldsymbol{x}, \boldsymbol{y})$. $f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}), f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x})$ are scoring functions for the two corresponding views, whose formulation can be quite flexible. Here we use $f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}) = f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}) = \exp(\langle h_{\boldsymbol{x}}, h_{\boldsymbol{y}}\rangle)$.

Derivation of InfoNCE

$$\begin{aligned}
I(X;Y) - \log(K) &= \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\big[\log \frac{1}{K}\frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}\big] \\
&= \sum_{\boldsymbol{x}^i,\boldsymbol{y}^i}\big[\log \frac{1}{K}\frac{p(\boldsymbol{x}^i,\boldsymbol{y}^i)}{p(\boldsymbol{x}^i)p(\boldsymbol{y}^i)}\big] \\
&\geq -\sum_{\boldsymbol{x}^i,\boldsymbol{y}^i}\big[\log\big(1 + (K-1)\frac{p(\boldsymbol{x}^i)p(\boldsymbol{y}^i)}{p(\boldsymbol{x}^i,\boldsymbol{y}^i)}\big)\big] \\
&= -\sum_{\boldsymbol{x}^i,\boldsymbol{y}^i}\big[\log \frac{\frac{p(\boldsymbol{x}^i,\boldsymbol{y}^i)}{p(\boldsymbol{x}^i)p(\boldsymbol{y}^i)} + (K-1)}{\frac{p(\boldsymbol{x}^i,\boldsymbol{y}^i)}{p(\boldsymbol{x}^i)p(\boldsymbol{y}^i)}}\big] \\
&\approx -\sum_{\boldsymbol{x}^i,\boldsymbol{y}^i}\big[\log \frac{\frac{p(\boldsymbol{x}^i,\boldsymbol{y}^i)}{p(\boldsymbol{x}^i)p(\boldsymbol{y}^i)} + (K-1)\mathbb{E}_{\boldsymbol{x}^j\neq\boldsymbol{x}^i}\frac{p(\boldsymbol{x}^j,\boldsymbol{y}^i)}{p(\boldsymbol{x}^j)p(\boldsymbol{y}^i)}}{\frac{p(\boldsymbol{x}^i,\boldsymbol{y}^i)}{p(\boldsymbol{x}^i)p(\boldsymbol{y}^i)}}\big] \quad \text{// ①} \\
&= \sum_{\boldsymbol{x}^i,\boldsymbol{y}^i}\big[\log \frac{\exp(f_{\boldsymbol{x}}(\boldsymbol{x}^i,\boldsymbol{y}^i))}{\exp(f_{\boldsymbol{x}}(\boldsymbol{x}^i,\boldsymbol{y}^i)) + \sum_{j=1}^K f_{\boldsymbol{x}}(\boldsymbol{x}^j,\boldsymbol{y}^i)}\big],
\end{aligned} \tag{16}$$

where we set $f_{\boldsymbol{x}}(\boldsymbol{x}^i,\boldsymbol{y}^i) = \log \frac{p(\boldsymbol{x}^i,\boldsymbol{y}^i)}{p(\boldsymbol{x}^i)p(\boldsymbol{y}^i)}$.

Notice that in ①, we are using data $x \in X$ as the anchor points. If we use the $y \in Y$ as the anchor points and follow the similar steps, we can obtain

$$I(X;Y) - \log(K) \geq \sum_{\boldsymbol{y}^i,\boldsymbol{x}^i}\big[\log \frac{\exp(f_{\boldsymbol{y}}(\boldsymbol{y}^i,\boldsymbol{x}^i))}{\exp f_{\boldsymbol{y}}(\boldsymbol{y}^i,\boldsymbol{x}^i) + \sum_{j=1}^K \exp(f_{\boldsymbol{y}}(\boldsymbol{y}^j,\boldsymbol{x}^i))}\big]. \tag{17}$$

Thus, by add both together, we can have the objective function as Equation (15).

E.2 EBM-NCE

We here provide an alternative approach to maximizing MI using energy-based model (EBM). To our best knowledge, we are the **first** to give the rigorous proof of using EBM to maximize the MI.

E.2.1 ENERGY-BASED MODEL (EBM)

Energy-based model (EBM) is a powerful tool for modeling the data distribution. The classic formulation is:

$$p(\boldsymbol{x}) = \frac{\exp(-E(\boldsymbol{x}))}{A}, \tag{18}$$

where the bottleneck is the intractable partition function $A = \int_{\boldsymbol{x}} \exp(-E(\boldsymbol{x}))d\boldsymbol{x}$. Recently, there have been quite a lot progress along this direction (11; 15; 42; 43). Noise Contrastive Estimation (NCE) (15) is one of the powerful tools here, as we will introduce later.

E.2.2 EBM FOR MI

Recall that our objective function is Equation (14): $\mathcal{L}_{\text{MI}} = \frac{1}{2}[H(Y|X) + H(X|Y)]$. Then we model the conditional likelihood with energy-based model (EBM). This gives us

$$\mathcal{L}_{\text{EBM}} = -\frac{1}{2}\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big[\log\frac{\exp(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))}{A_{\boldsymbol{x}|\boldsymbol{y}}} + \log\frac{\exp(f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}))}{A_{\boldsymbol{y}|\boldsymbol{x}}}\Big], \tag{19}$$

where $f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}) = -E(\boldsymbol{x}|\boldsymbol{y})$ and $f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}) = -E(\boldsymbol{y}|\boldsymbol{x})$ are the negative energy functions, and $A_{\boldsymbol{x}|\boldsymbol{y}}$ and $A_{\boldsymbol{y}|\boldsymbol{x}}$ are the corresponding partition functions.

Under the EBM framework, if we solve Equation (19) with Noise Contrastive Estimation (NCE) (15), the final EBM-NCE objective is

$$\begin{aligned}\mathcal{L}_{\text{EBM-NCE}} = &-\frac{1}{2}\mathbb{E}_{p_{\text{data}}(y)}\Big[\mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})}[\log\big(1-\sigma(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))\big)] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})}[\log\sigma(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))]\Big] \\ &-\frac{1}{2}\mathbb{E}_{p_{\text{data}}(x)}\Big[\mathbb{E}_{p_n(\boldsymbol{y}|\boldsymbol{x})}[\log\big(1-\sigma(f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}))\big)] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{y}|\boldsymbol{x})}[\log\sigma(f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}))]\Big].\end{aligned} \tag{20}$$

Next we will give the detailed derivations.

E.2.3 DERIVATION OF CONDITIONAL EBM WITH NCE

WLOG, let's consider the $p_\theta(\boldsymbol{x}|\boldsymbol{y})$ first, and by EBM it is as follows:

$$p_\theta(\boldsymbol{x}|\boldsymbol{y}) = \frac{\exp(-E(\boldsymbol{x}|\boldsymbol{y}))}{\int\exp(-E(\tilde{\boldsymbol{x}}|\boldsymbol{y}))d\tilde{\boldsymbol{x}}} = \frac{\exp(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))}{\int\exp(f_{\boldsymbol{x}}(\tilde{\boldsymbol{x}}|\boldsymbol{y}))d\tilde{\boldsymbol{x}}} = \frac{\exp(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))}{A_{\boldsymbol{x}|\boldsymbol{y}}}. \tag{21}$$

$$p(y|x) = \frac{\exp(-E(y|x))}{\int\exp(-E(\tilde{y}|x))d\tilde{y}} = \frac{\exp(f_y(y,x))}{A_{y|x}} \tag{22}$$

Then we solve this using NCE. NCE handles the intractability issue by transforming it as a binary classification task. We take the partition function $A_{\boldsymbol{x}|\boldsymbol{y}}$ as a parameter, and introduce a noise distribution p_n. Based on this, we introduce a mixture model: $\boldsymbol{z} = 0$ if the conditional $\boldsymbol{x}|\boldsymbol{y}$ is from $p_n(\boldsymbol{x}|\boldsymbol{y})$, and $\boldsymbol{z} = 1$ if $\boldsymbol{x}|\boldsymbol{y}$ is from $p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})$. So the joint distribution is:

$$p_{n,\text{data}}(\boldsymbol{x}|\boldsymbol{y}) = p(\boldsymbol{z}=1)p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y}) + p(\boldsymbol{z}=0)p_n(\boldsymbol{x}|\boldsymbol{y})$$

The posterior of $p(\boldsymbol{z}=0|\boldsymbol{x},\boldsymbol{y})$ is

$$p_{n,\text{data}}(\boldsymbol{z}=0|\boldsymbol{x},\boldsymbol{y}) = \frac{p(\boldsymbol{z}=0)p_n(\boldsymbol{x}|\boldsymbol{y})}{p(z=0)p_n(\boldsymbol{x}|\boldsymbol{y}) + p(\boldsymbol{z}=1)p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} = \frac{\nu\cdot p_n(\boldsymbol{x}|\boldsymbol{y})}{\nu\cdot p_n(\boldsymbol{x}|\boldsymbol{y}) + p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})},$$

where $\nu = \frac{p(\boldsymbol{z}=0)}{p(\boldsymbol{z}=1)}$.

Similarly, we can have the joint distribution under EBM framework as:

$$p_{n,\theta}(\boldsymbol{x}) = p(z=0)p_n(\boldsymbol{x}|\boldsymbol{y}) + p(z=1)p_\theta(\boldsymbol{x}|\boldsymbol{y})$$

And the corresponding posterior is:

$$p_{n,\theta}(\boldsymbol{z}=0|\boldsymbol{x},\boldsymbol{y}) = \frac{p(\boldsymbol{z}=0)p_n(\boldsymbol{x}|\boldsymbol{y})}{p(\boldsymbol{z}=0)p_n(\boldsymbol{x}|\boldsymbol{y}) + p(\boldsymbol{z}=1)p_\theta(\boldsymbol{x}|\boldsymbol{y})} = \frac{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y})}{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y}) + p_\theta(\boldsymbol{x}|\boldsymbol{y})}$$

We indirectly match $p_\theta(\boldsymbol{x}|\boldsymbol{y})$ to $p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})$ by fitting $p_{n,\theta}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})$ to $p_{n,\text{data}}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})$ by minimizing their KL-divergence:

$$\begin{aligned}
&\min_\theta D_{\text{KL}}(p_{n,\text{data}}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})||p_{n,\theta}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})) \\
&= \mathbb{E}_{p_{n,\text{data}}(\boldsymbol{x},\boldsymbol{z}|\boldsymbol{y})}[\log p_{n,\theta}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})] \\
&= \int \sum_{\boldsymbol{z}} p_{n,\text{data}}(\boldsymbol{x},\boldsymbol{z}|\boldsymbol{y}) \cdot \log p_{n,\theta}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y}) d\boldsymbol{x} \\
&= \int \Big\{ p(\boldsymbol{z}=0) p_{n,\text{data}}(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{z}=0) \log p_{n,\theta}(\boldsymbol{z}=0|\boldsymbol{x},\boldsymbol{y}) \\
&\qquad + p(\boldsymbol{z}=1) p_{n,\text{data}}(\boldsymbol{x}|\boldsymbol{z}=1,\boldsymbol{y}) \log p_{n,\theta}(\boldsymbol{z}=1|\boldsymbol{x},\boldsymbol{y}) \Big\} d\boldsymbol{x} \\
&= \nu \cdot \mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \Big[\log p_{n,\theta}(\boldsymbol{z}=0|\boldsymbol{x},\boldsymbol{y}) \Big] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \Big[\log p_{n,\theta}(\boldsymbol{z}=1|\boldsymbol{x},\boldsymbol{y}) \Big] \\
&= \nu \cdot \mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y})}{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y}) + p_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{y})}{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y}) + p_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big].
\end{aligned} \tag{23}$$

This optimal distribution is an estimation to the actual distribution (or data distribution), *i.e.*, $p_\theta(\boldsymbol{x}|\boldsymbol{y}) \approx p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})$. We can follow the similar steps for $p_\theta(\boldsymbol{y}|\boldsymbol{x}) \approx p_{\text{data}}(\boldsymbol{y}|\boldsymbol{x})$. Thus following Equation (23), the objective function is to maximize

$$\nu \cdot \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})} \mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y})}{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y}) + p_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})} \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{y})}{\nu \cdot p_n(\boldsymbol{x}|\boldsymbol{y}) + p_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big]. \tag{24}$$

The we will adopt three strategies to approximate Equation (24):

1. **Self-normalization.** When the EBM is very expressive, *i.e.*, using deep neural network for modeling, we can assume it is able to approximate the normalized density directly (33; 42). In other words, we can set the partition function $A = 1$. This is a self-normalized EBM-NCE, with normalizing constant close to 1, *i.e.*, $p(\boldsymbol{x}) = \exp(-E(\boldsymbol{x})) = \exp(f(\boldsymbol{x}))$ in Equation (18).
2. **Exponential tilting term.** Exponential tilting term (1) is another useful trick. It models the distribution as $\tilde{p}_\theta(\boldsymbol{x}) = q(\boldsymbol{x}) \exp(-E_\theta(\boldsymbol{x}))$, where $q(\boldsymbol{x})$ is the reference distribution. If we use the same reference distribution as the noise distribution, the tilted probability is $\tilde{p}_\theta(\boldsymbol{x}) = p_n(\boldsymbol{x}) \exp(-E_\theta(\boldsymbol{x}))$ in Equation (18).
3. **Sampling.** For many cases, we only need to sample 1 negative points for each data, *i.e.*, $\nu = 1$.

Following these three disciplines, the objective function to optimize $p_\theta(\boldsymbol{x}|\boldsymbol{y})$ becomes

$$\begin{aligned}
&\mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{p_n(\boldsymbol{x}|\boldsymbol{y})}{p_n(\boldsymbol{x}|\boldsymbol{y}) + \tilde{p}_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{\tilde{p}_\theta(\boldsymbol{x}|\boldsymbol{y})}{p_n(\boldsymbol{x}|\boldsymbol{y}) + \tilde{p}_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big] \\
=&\mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{1}{1 + p_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{y})}{1 + p_\theta(\boldsymbol{x}|\boldsymbol{y})} \Big] \\
=&\mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{\exp(-f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))}{\exp(-f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y})) + 1} \Big] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \frac{1}{\exp(-f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y})) + 1} \Big] \\
=&\mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \big(1 - \sigma(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))\big) \Big] + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \Big[\log \sigma(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y})) \Big].
\end{aligned} \tag{25}$$

Thus, the final EBM-NCE contrastive SSL objective is

$$\begin{aligned}
\mathcal{L}_{\text{EBM-NCE}} = &-\frac{1}{2} \mathbb{E}_{p_{\text{data}}(\boldsymbol{y})} \Big[\mathbb{E}_{p_n(\boldsymbol{x}|\boldsymbol{y})} \log \big(1 - \sigma(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y}))\big) + \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}|\boldsymbol{y})} \log \sigma(f_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{y})) \Big] \\
&- \frac{1}{2} \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})} \Big[\mathbb{E}_{p_n(\boldsymbol{y}|\boldsymbol{x})} \log \big(1 - \sigma(f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x}))\big) + \mathbb{E}_{p_{\text{data}}(\boldsymbol{y},\boldsymbol{x})} \log \sigma(f_{\boldsymbol{y}}(\boldsymbol{y},\boldsymbol{x})) \Big].
\end{aligned} \tag{26}$$

E.3 EBM-NCE V.S. JSE AND INFONCE

We acknowledge that there are many other contrastive objectives (37) that can be used to maximize MI. However, in the research line of graph SSL, as summarized in several recent survey papers (32; 51; 53), the two most used ones are InfoNCE and Jensen-Shannon Estimator (JSE) (20; 35).

We conclude that JSE is very similar to EBM-NCE, while the underlying perspectives are totally different, as explained below.

1. **Derivation and Intuition.** Derivation process and underlying intuition are different. JSE (35) starts from f-divergence, then with variational estimation and Fenchel duality on function f. Our proposed EBM-NCE is more straightforward: it models the conditional distribution in the MI lower bound Equation (14) with EBM, and solves it using NCE.
2. **Flexibility.** Modeling the conditional distribution with EBM provides a broader family of algorithms. NCE is just one solution to it, and recent progress on score matching (42; 43) and contrastive divergence (11), though no longer contrastive SSL, adds on more promising directions. Thus, EBM can provide a potential unified framework for structuring our understanding of self-supervised learning.
3. **Noise distribution.** Starting from (20), all the following works on graph SSL (32; 44; 51; 53) have been adopting the empirical distribution for noise distribution. However, this is not the case in EBM-NCE. Classic EBM-NCE uses fixed distribution, while more recent work (1) extends it with adaptively learnable noise distribution. With this discipline, more advanced sampling strategies (w.r.t. the noise distribution) can be proposed, *e.g.*, adversarial negative sampling in (21).

In the above we conclude three key differences between EBM-NCE and JSE, plus the solid and straightforward derivations on EBM-NCE, we would like to share this general contrastive SSL framework to the community.

According to the empirical results Appendix C.3, we observe that EBM-NCE is better than InfoNCE. This can be explained using the claim from (25), where the main technical contribution is to construct many positives and many negatives per anchor point. The binary cross-entropy in EBM-NCE is able to realize this to some extent: make all the positive pairs positive and all the negative pairs negative, where the softmax-based cross-entropy fails to capture this, as in InfoNCE.

To conclude, we are introduce using EBM in modeling MI, which opens many potential venues. As for contrastive SSL, EBM-NCE provides a better perspective than JSE, and is better than InfoNCE on graph-level self-supervised learning.

F GENERATIVE SELF-SUPERVISED LEARNING

Generative SSL is another classic track for unsupervised pre-training (26–28), though the main focus is on distribution learning. In GraphMVP, we start with VAE for the following reasons:

1. One of the biggest attributes of our problem is that the mapping between two views are stochastic: multiple 3D conformers can correspond to the same 2D topology. Thus, we expect a stochastic model (34) like VAE, instead of the deterministic ones.
2. For pre-training and fine-tuning, we need to learn an explicit and powerful representation function that can be used for downstream tasks.
3. The decoder for structured data like graph are often complicated, *e.g.*., the auto-regressive generation. This makes them suboptimal.

To cope with these challenges, in GraphMVP, we start with VAE-like generation model, and later propose a *light-weighted* and *smart* surrogate loss as objective function. Notice that for notation simplicity, for this section, we use $h_{\boldsymbol{y}}$ and $h_{\boldsymbol{x}}$ to delegate the 3D and 2D GNN respectively.

F.1 VARIATIONAL MOLECULE RECONSTRUCTION

As shown in Equation (14), our main motivation is to model the conditional likelihood:

$$\mathcal{L}_{\text{MI}} = -\frac{1}{2}\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{x}|\boldsymbol{y}) + \log p(\boldsymbol{y}|\boldsymbol{x})]$$

By introducing a reparameterized variable $\boldsymbol{z}_{\boldsymbol{x}} = \mu_{\boldsymbol{x}} + \sigma_{\boldsymbol{x}} \odot \epsilon$, where $\mu_{\boldsymbol{x}}$ and $\sigma_{\boldsymbol{x}}$ are two flexible functions on $h_{\boldsymbol{x}}$, $\epsilon \sim \mathcal{N}(0, I)$ and $\odot$ is the element-wise production, we have a lower bound on the conditional likelihood:

$$\log p(\boldsymbol{y}|\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}\big[\log p(\boldsymbol{y}|\boldsymbol{z}_{\boldsymbol{x}})\big] - KL(q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})||p(\boldsymbol{z}_{\boldsymbol{x}})). \tag{27}$$

Similarly, we have

$$\log p(\boldsymbol{x}|\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{y}}|\boldsymbol{y})}\big[\log p(\boldsymbol{x}|\boldsymbol{z}_{\boldsymbol{y}})\big] - KL(q(\boldsymbol{z}_{\boldsymbol{y}}|\boldsymbol{y})||p(\boldsymbol{z}_{\boldsymbol{y}})), \tag{28}$$

where $\boldsymbol{z}_{\boldsymbol{y}} = \mu_{\boldsymbol{y}} + \sigma_{\boldsymbol{y}} \odot \epsilon$. Here $\mu_{\boldsymbol{y}}$ and $\sigma_{\boldsymbol{y}}$ are flexible functions on $h_{\boldsymbol{y}}$, and $\epsilon \sim \mathcal{N}(0, I)$. For implementation, we take multi-layer perceptrons (MLPs) for $\mu_{\boldsymbol{x}}, \mu_{\boldsymbol{y}}, \sigma_{\boldsymbol{x}}, \sigma_{\boldsymbol{y}}$.

Both the above objectives are composed of a conditional log-likelihood and a KL-divergence. The conditional log-likelihood has also been recognized as the *reconstruction term*: it is essentially to reconstruct the 3D conformers ($\boldsymbol{y}$) from the sampled 2D molecular graph representation ($\boldsymbol{z}_{\boldsymbol{x}}$). However, performing the graph reconstruction on the data space is not easy: since molecules are discrete, modeling and measuring are not trivial.

F.2 VARIATIONAL REPRESENTATION RECONSTRUCTION

To cope with data reconstruction issue, we propose a novel generative loss termed variation representation reconstruction (VRR). The pipeline is in Figure 4.

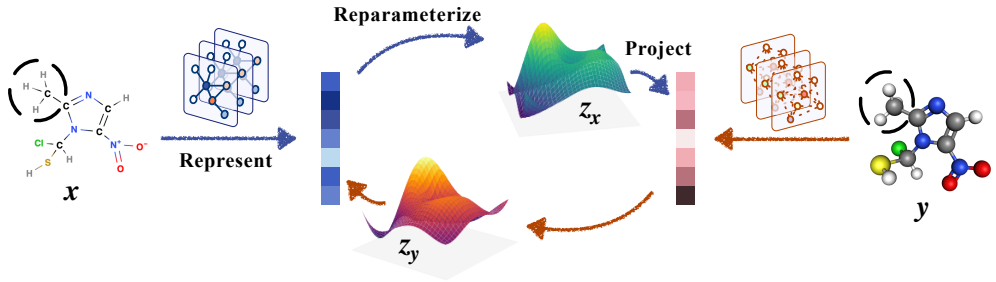

Figure 4: VRR SSL in GraphMVP. The black dashed circles represent subgraph masking.

Our proposed solution is very straightforward. Recall that MI is invariant to continuous bijective function (4). So suppose we have a representation function $h_{\boldsymbol{y}}$ satisfying this condition, and this can guide us a surrogate loss by transferring the reconstruction from data space to the continuous representation space:

$$\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}[\log p(\boldsymbol{y}|\boldsymbol{z}_{\boldsymbol{x}})] = -\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}[\|h_{\boldsymbol{y}}(g_x(\boldsymbol{z}_{\boldsymbol{x}})) - h_{\boldsymbol{y}}(\boldsymbol{y})\|_2^2] + C,$$

where g_x is the decoder and C is a constant, and this introduces to using the mean-squared error (MSE) for **reconstruction on the representation space**.

Then for the reconstruction, current formula has two steps: i) the latent code $z_{\boldsymbol{x}}$ is first mapped to molecule space, and ii) it is mapped to the representation space. We can approximate these two mappings with one projection step, by directly projecting the latent code $z_{\boldsymbol{x}}$ to the 3D representation space, *i.e.*, $q_{\boldsymbol{x}}(z_{\boldsymbol{x}}) \approx h_{\boldsymbol{y}}(g_{\boldsymbol{x}}(z_{\boldsymbol{x}}))$. This gives us a variation representation reconstruction (VRR) SSL objective as below:

$$\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}[\log p(\boldsymbol{y}|\boldsymbol{z}_{\boldsymbol{x}})] = -\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}[\|q_x(\boldsymbol{z}_{\boldsymbol{x}}) - h_{\boldsymbol{y}}(\boldsymbol{y})\|_2^2] + C.$$

β-VAE We consider introducing a β variable (19) to control the disentanglement of the latent representation. To be more specific, we would have

$$\log p(\boldsymbol{y}|\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}\big[\log p(\boldsymbol{y}|\boldsymbol{z}_{\boldsymbol{x}})\big] - \beta \cdot KL(q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})||p(\boldsymbol{z}_{\boldsymbol{x}})). \tag{29}$$

Stop-gradient For the optimization on variational representation reconstruction, related work have found that adding the stop-gradient operator (SG) as a regularizer can make the training more stable without collapse both empirically (8; 14) and theoretically (45). Here, we may as well utilize this SG operation in the objective function:

$$\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}[\log p(\boldsymbol{y}|\boldsymbol{z}_{\boldsymbol{x}})] = -\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}[\|q_x(\boldsymbol{z}_{\boldsymbol{x}}) - \text{SG}(h_{\boldsymbol{y}}(\boldsymbol{y}))\|_2^2] + C. \tag{30}$$

Objective function for VRR Thus, combining both two regularizers mentioned above, the final objective function for VRR is:

$$\begin{aligned}\mathcal{L}_{\text{VRR}} = \frac{1}{2}\Big[\mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})}\big[\|q_x(\boldsymbol{z}_{\boldsymbol{x}}) - \text{SG}(h_{\boldsymbol{y}})\|^2\big] + \mathbb{E}_{q(\boldsymbol{z}_{\boldsymbol{y}}|\boldsymbol{y})}\big[\|q_y(\boldsymbol{z}_{\boldsymbol{y}}) - \text{SG}(h_{\boldsymbol{x}})\|_2^2\big]\Big] \\ + \frac{\beta}{2} \cdot \Big[KL(q(\boldsymbol{z}_{\boldsymbol{x}}|\boldsymbol{x})||p(\boldsymbol{z}_{\boldsymbol{x}})) + KL(q(\boldsymbol{z}_{\boldsymbol{y}}|\boldsymbol{y})||p(\boldsymbol{z}_{\boldsymbol{y}}))\Big].\end{aligned} \tag{31}$$

Note that MI is invariant to continuous bijective function (4), thus this surrogate loss would be exact if the encoding function $h_{\boldsymbol{y}}$ and $h_{\boldsymbol{x}}$ satisfy this condition. However, we find GNN (both GIN and SchNet) can, though do not meet the condition, provide quite robust performance empirically, which justify the effectiveness of VRR.

F.3 Variational Representation Reconstruction and Non-Contrastive SSL

By introducing VRR, we provide another perspective to understand the generative SSL, including the recently-proposed non-contrastive SSL (8; 14).

We provide a unified structure on the intra-data generative SSL:

- Reconstruction to the data space, like Equations (3), (27) and (28).
- Reconstruction to the representation space, *i.e.*, VRR in Equation (31).
  - If we **remove the stochasticity**, then it is simply the representation reconstruction (RR), as we tested in the ablation study Appendix C.3.
  - If we **remove the stochasticity** and assume two views are **sharing the same representation function**, like CNN for multi-view learning on images, then it is reduced to the BYOL (14) and SimSiam (8). In other words, these recently-proposed non-contrastive SSL methods are indeed special cases of VRR.