3D Equivariant Molecular Graph Pretraining

Rui Jiao\textsuperscript{1,2}, Jiaqi Han\textsuperscript{1,2}, Wenbing Huang\textsuperscript{2,*}, Yu Rong\textsuperscript{3}, Yang Liu\textsuperscript{1,2,*}

\textsuperscript{1} Beijing National Research Center for Information Science and Technology (BNRist), Department of Computer Science and Technology, Tsinghua University
\textsuperscript{2} Institute for AI Industry Research (AIR), Tsinghua University
\textsuperscript{3} Tencent AI Lab
{jiaor21,hanjq21}@mails.tsinghua.edu.cn, hwenbing@126.com, yu.rong@hotmail.com, liuyang2011@tsinghua.edu.cn

Abstract

Pretraining molecular representation models without labels is fundamental to various applications. Conventional methods mainly process 2D molecular graphs and focus solely on 2D tasks, making their pretrained models incapable of characterizing 3D geometry and thus defective for downstream 3D tasks. In this work, we tackle 3D molecular pretraining in a complete and novel sense. In particular, we first propose to adopt an equivariant energy-based model as the backbone for pretraining, which enjoys the merit of fulfilling the symmetry of 3D space. Then we develop a node-level pretraining loss for force prediction, where we further exploit the Riemann-Gaussian distribution to ensure the loss to be E(3)-invariant, enabling more robustness. Moreover, a graph-level noise scale prediction task is also leveraged to further promote the eventual performance. We evaluate our model pretrained from a large-scale 3D dataset GEOM-QM9 on two challenging 3D benchmarks: MD17 and QM9. The experimental results support the better efficacy of our method against current state-of-the-art pretraining approaches, and verify the validity of our design for each proposed component.

1 Introduction

Learning informative molecular representation is a fundamental step for various downstream applications including molecular property prediction \cite{8, 15}, virtual screening \cite{40, 46}, and Molecular Dynamics (MD) simulation \cite{4}. Recent methods exploit Graph Neural Networks (GNNs) \cite{8, 42} for their power in capturing the topology of molecules, which yet is label-hungry and thus powerless for real scenarios when molecular annotations are unavailable. Therefore, the research attention has been paid to the self-supervised pretraining paradigm, to construct the surrogate task by exploring the intrinsic structure within unlabeled molecules. A variety of self-supervised works have been proposed, ranging from generative-based models \cite{16, 11, 12} to contrastive learning \cite{32, 33, 45, 44, 53}.

In many applications, using and analyzing 3D geometry is crucial and even indispensable; for instance, we need to process 3D coordinates for energy prediction in MD simulation or protein-ligand docking. Owing to the fast development in data acquisition, it is now convenient to access large-scale unlabeled molecules with rich 3D conformations \cite{2}. It would be quite exciting if we can develop techniques to obtain pretrained models from these unlabeled molecules for 3D tasks with limited data. Nevertheless, existing self-supervised methods \cite{11, 45, 23} are weak in leveraging the 3D geometry information. First, from the input side, the backbone models they pretrain can only process the input of 2D molecules without the consideration of 3D coordinates. As demonstrated by \cite{27}, certain molecular properties (e.g. potential energy) are closely related to the 3D structure with which they

*Wenbing Huang and Yang Liu are the corresponding authors.

Preprint. Under review.
can be better predicted. Second, for the output side, their pretraining tasks are not 3D-aware, making the knowledge they discover less generalizable in 3D space. Recently, the study by [18] proposes to impose the 3D information for pretraining; however, its goal is still limited to enhancing 2D models for 2D tasks.

In this paper, we investigate 3D molecular pretraining in a complete and novel sense: using 3D backbone models, designing 3D-aware pretraining tasks, and targeting 3D downstream evaluations. However, this is not trivial by any means. The challenges mainly stem from how to maintain the symmetry of our biological world—rotating/translator the 3D conformation of a molecular does not change the law of its behavior. Mathematically, we should make the backbone model E(3)-equivariant, and the pretraining loss E(3)-invariant, where the group E(3) collects the transformations of rotations, reflections, and translations [24]. Unfortunately, typical GNN models [8, 42] and 3D losses based on Euclidean distance [20] do not satisfy such kinds of constraints.

To address the above challenges, this paper makes the following contributions: 1. We propose an energy-based representation model that predicts E(3)-equivariant force for each atom in the input 3D molecule, by leveraging recently-proposed equivariant GNNs [25, 27, 34] as the building block. 2. To pretrain this model, we formulate a physics-inspired node-level force prediction task, which is further translated to a 3D position denoising loss in an equivalent way. More importantly, we develop a novel denoising scheme with the aid of the proposed Riemann-Gaussian distribution, to ensure the E(3)-invariance of the pretraining task. 3. We additionally design a graph-level surrogate task on 3D molecules, in line with the observation from traditional 2D methods [23] that performing node-level and graph-level tasks jointly is able to promote the eventual performance. For this purpose, we teach the model to identify the noise scale of the input tuple consisting of a clean sample and a noisy one.

We pretrain our model on a large-scale dataset with 3D conformations: GEOM-QM9 [2], and then evaluate its performance on the two popular 3D tasks: MD17 [4] and QM9 [22]. Extensive experiments demonstrate that our model outperforms state-of-the-art 2D approaches, even their inputs are augmented with 3D coordinates for fair comparisons. We also inspect how the performance changes if we replace the components of our architecture with other trivial implementations. The results do support the optimal choice of our design.

2 Related Works

Self-supervised molecular pretraining. Self-supervised learning has been well developed in the field of molecular graph representation learning. Many pretraining tasks have been proposed to extract information from large-scale 2D molecule dataset, which can be divided into three categories: contrastive, generative and predictive. Contrastive methods aim to maximize the mutual information between different views of the same graph [32, 38, 45, 44, 33], while generative methods focus on reconstructing the information from different levels of the 2D topological structure of molecules [16, 11]. As for predictive methods, they learn the molecule representation by predicting the pseudo-labels created from the input graphs. For example, GROVER [23] proposes to classify the subgraph structure and predict the existence of specific motifs, which leverages domain knowledge into molecule pretraining. However, all the above methods mainly focus on pretraining the 2D GNN backbones without 3D information. More recently, several methods propose to tackle 3D graphs, including 3D Infomax [31] that maximizes the mutual information between the representations encoded from a 2D and a 3D model, and GraphMVP [18] which proposes both contrastive and generative methods to leverage 3D information into the 2D model. Nevertheless, the motivation of these two methods remains to benefit the 2D models with 3D side information. On the contrary, this paper attempts to pretrain 3D models via 3D objectives with the usage for 3D downstream tasks.

Equivariant graph neural networks. To better analyze the physical and chemical properties of molecules, many researchers regard the molecules as geometric graphs, which additionally assign 3D coordinates on each atom apart from the 2D topological information. Geometric graphs present rotational, translational, and/or mirror symmetry, as the properties are invariant and the dynamic processes are equivariant to the E(3) or SE(3) transformations in 3D space. To introduce this inductive bias, geometrically equivariant graph neural networks have been proposed to model the geometric graphs. According to [10], current 3D GNNs achieve equivariance mainly in three ways: extracting irreducible representations [35, 7], utilizing group regular representations [5, 14] or transforming the 3D vectors into invariant scalars [27, 25, 34, 13]. Previous works showcase the superiority of
Figure 1: Overview of our 3D Molecular Graph Pretraining (3D-MGP). It consists of two tasks: equivariant node-level force prediction and invariant graph-level noise scale prediction. $\text{Rie}_\sigma(\mathbf{X} \mid \mathbf{X})$ refers to the proposed Riemann-Gaussian distribution in Eq. 9.

equivariant models on several 3D molecular tasks \cite{34, 19}, and our goal is to further improve the performance of 3D models via equivariant pretraining on large-scale 3D datasets.

3 Method

In this section, we will first introduce the backbone architecture for 3D molecular representation modeling. We then present the node-level and graph-level pretraining tasks, respectively.

3.1 Energy-based Molecular Representation Modeling

In general, a molecule consisting of $N$ atoms can be modeled as a molecular graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes (atoms) and $\mathcal{E}$ is the set of edges modeling the connectivity, e.g., bond connection or spatial proximity. Each atom is assigned a node feature $h_i$, $1 \leq i \leq N$, representing the properties of the atom, such as atom type. In this work, we elaborate on the 3D information of a molecule, that is, apart from the node attribute $h_i \in \mathbb{R}^m$ for atom $i$, we extra consider the 3D coordinate $x_i \in \mathbb{R}^3$. We denote the configurations of all nodes as $\mathbf{X} \in \mathbb{R}^{3 \times N}$, and similarly the node features as $\mathbf{H} \in \mathbb{R}^{m \times N}$. Our goal is to pretrain a capable GNN model $\varphi$ that can be generally applied to different downstream 3D tasks, which is formally depicted as $\varphi(\mathbf{X}, \mathbf{H}, \mathcal{E})$. We hereafter omit the input of $\mathcal{E}$ for conciseness, unless otherwise specified.

Unlike conventional 2D molecular graph pretraining, we are now provided with vital 3D information, making it possible to leverage the rich geometric context in a unified framework. However, involving 3D conformation is not free of difficulty, and one constraint we should enforce is to make $\varphi$ equivariant for the vector output, and invariant for the scalar output. The notion of equivariance/invariance is defined as below.

**Definition 1.** A GNN model $\varphi$ is call $E(3)$-equivariant, if for any transformation $g \in E(3)$, $\varphi(g \cdot \mathbf{X}, \mathbf{H}) = g \cdot \varphi(\mathbf{X}, \mathbf{H})$; and it is called invariant if $\varphi(g \cdot \mathbf{X}, \mathbf{H}) = \varphi(\mathbf{X}, \mathbf{H})$.

In Definition 1, the group action $g \cdot \mathbf{X}$ is implemented as matrix multiplication $\mathbf{O} \mathbf{X}$ for orthogonal transformation $\mathbf{O} \in \mathbb{R}^{3 \times 3}$ (namely rotations and reflections that satisfy $\mathbf{O}^\top \mathbf{O} = \mathbf{I}$) and addition $\mathbf{X} + \mathbf{t}$ for translation $\mathbf{t} \in \mathbb{R}^3$. Basically, for an equivariant function, the output will translate/rotate/reflect in the same way as the input, while for the invariant case, the output stays unchanged whatever group action we conduct on the input. Equivariance/invariance is so essential that it characterizes the symmetry of the 3D biochemistry: rotating or translating a molecular will never change its potential energy. There are a variety of $E(3)$-equivariant GNNs \cite{10} that can be utilized as our backbone. By choosing an $E(3)$-equivariant GNN, $\varphi$ is instantiated as $\varphi_{\text{EGN}}$. It is necessary to mention that $\varphi_{\text{EGN}}$ is also permutation equivariant regarding the order of the columns in $\mathbf{X}$ and $\mathbf{H}$.

We now introduce our idea of how to tackle 3D molecular pretraining in a more domain-knowledge-reliable manner. As well studied in biochemistry, the interaction between atoms in 3D space is...
captured by the forces and potentials, depending on the positions of the atoms, i.e., the molecular conformation. This intriguing connection inspires us to incorporate the concept of energy and force into our representation model, making room for designing unsupervised pretraining objectives with 3D geometric information. In light of this consideration, we introduce a representation model that jointly takes into account both energy and force. We denote the energy of a molecule as $E \in \mathbb{R}$ and the resultant interaction force exerting on atom $i$ as $f_i \in \mathbb{R}^3$, $1 \leq i \leq N$. We collect all forces as the matrix $F \in \mathbb{R}^{3 \times N}$. Clearly, $E$ is an invariant graph-level scalar, while $F$ is an equivariant node-level vector, in terms of the input transformation.

To derive $E$ and $F$ by the equivariant model $\varphi_{\text{EGN}}$, we fist obtain a node-level invariant representation in the latent space, i.e.,

$$H' = \varphi_{\text{EGN}} (X, H),$$

where $H' \in \mathbb{R}^{k \times N}$ is an invariant representation. Afterwards, we yield the graph-level energy of the molecule via a graph pooling operation:

$$E(X) = \varphi_{\text{Proj}} \left( \sum_{i=1}^{N} h'_i \right),$$

where $h'_i$ is the $i$-th column of $H'$, $\varphi_{\text{Proj}} : \mathbb{R}^k \rightarrow \mathbb{R}$ is the projection head, realized by a Multi-Layer Perceptron (MLP). Essentially, force corresponds to the direction that causes potential energy decreasing, which implies

$$F(X) = -\lim_{\Delta X \rightarrow 0} \frac{\Delta E}{\Delta X} = -\nabla_X E(X),$$

where $\nabla_X$ denotes the gradient w.r.t. $X$. It is easy to verify that $E$ is invariant and $F$ is equivariant\footnote{Indeed, $F$ is orthogonality-equivariant but translation-invariant.}.

We attempt to design the first proxy task by leveraging the predicted force $\hat{F}$ to fit the ground-truth force. However, there is usually no force label provided in the pretraining dataset. Fortunately, we can fulfill this purpose from the lens of node-level denoising—we first add noise to each node’s coordinate and then estimate the virtual force that pulls the noisy coordinate back to the clean one. We will provide the details in § 3.2. Upon the denoising process in the first pretraining task, we further construct a graph-level pretraining objective in an intuitive sense: a desirable model should be able to tell how much noise is added to its input. The detailed strategy is presented in § 3.3.

### 3.2 Node-Level Pretraining: Equivariant Force Prediction

The main idea is to formulate the equivariant force prediction loss over the training molecules $G$:

$$\mathcal{L}_{\text{EFP}} = \mathbb{E}_{G \sim G} \left[ \| F(X) - \nabla_X \log p(X) \|_F \right],$$

where $F$ is produced by the model $\varphi_{\text{EGN}}$ (Eq. (1-3)), $\| \cdot \|_F$ computes the Frobenius norm. We have assumed a Boltzmann energy distribution $\text{(3)}$ for the training data $G$:

$$p(X) = \frac{1}{Z} \exp \left( -\frac{\tilde{E}(X)}{kT} \right),$$

where $\tilde{E}$ is the real energy, $k$ is the Boltzmann constant, $T$ is the temperature, and $Z$ is the normalization term. By taking a logarithm of $p(X)$ and compute the gradient over $X$, yielding

$$\nabla_X \log p(X) \propto -\nabla_X \tilde{E} (X) := \tilde{F}.$$  

It means Eq. (4) is actually measuring the gap between the predicted and real forces, which complies with our initial goal. Nevertheless, we still have no idea of what the exact form of the data density $p(X)$ looks like, hence it is infeasible to directly apply the loss Eq. (4). Fortunately, the work by $\text{[39]}$ draws a promising conclusion that Eq. (4) can be equivalently translated to a denoising problem which is tractable to solve. In a nutshell, we instead sample a noisy sample $\tilde{X}$ from $X$ according to a certain conditional distribution, i.e., $\tilde{X} \sim p(X | X)$. Then we substitute the noisy sample into the model $\varphi_{\text{EGN}}$ and perform the replacement of Eq. (4) by

$$\mathcal{L}_{\text{EFP-DN}} = \mathbb{E}_{G \sim G, \tilde{X} \sim p(X | X)} \left[ \| F(\tilde{X}) - \nabla_X \log p(\tilde{X} | X) \|_F \right].$$

The equivalence between Eq. (4) and Eq. (7) is stated by the following proposition.
Proposition 1. For an arbitrary noise sampling $p(\tilde{X} \mid X)$, $\mathcal{L}_{\text{EFP}} = \mathcal{L}_{\text{EFP-DN}} + C$ where C is a constant independent to $\varphi_{\text{EGN}}$, if certain mild conditions hold.

We now discuss the formulation of the conditional probability $p(\tilde{X} \mid X)$. Different from traditional denoising process on images or other Euclidean data [20][29][21], in our case when considering the 3D geometry, $p(\tilde{X} \mid X)$ should be doubly $E(3)$-invariant, namely,

$$p(g_1 \cdot \tilde{X} \mid g_2 \cdot X) = p(\tilde{X} \mid X), \forall g_1, g_2 \in E(3).$$

(8)

This is consistent with our understanding: the behavior of molecules with the same geometry should be independent to different conformations. For example, when we rotate the sample $X$, the property of $p(X \mid X)$ by Eq. (8) ensures the loss in Eq. (7) to be unchanged, which is what we desire; similarly, conducting rotation on $X$ should also obey the same rule. A conventional choice of $p(X \mid X)$ is utilizing the standard Gaussian with noise scale $\sigma$: $p(\tilde{X} \mid X) = \mathcal{N}(X, \sigma I)$. This naive form fails to meet the doubly $E(3)$-invariant property in Eq. (8), which could cause mistaken supervision in Eq. (7).

To show this, we derive $\nabla_X \log p(\tilde{X} \mid X) = -\frac{\tilde{X} - X}{\sigma^2}$ as the force target; if we setting $\tilde{X} = RX$ for some rotation matrix $R$, then probably $\nabla_X \log p(\tilde{X} \mid X) \neq 0$, which, however, does not align the true fact that the force between $X$ and $\tilde{X}$ should be zero since they share the same geometry.

To devise the form with the symmetry in Eq. (8), we instead resort to Riemann-Gaussian distribution [24] defined as follows:

$$p_\sigma(\tilde{X} \mid X) = \text{Rie}_\sigma(\tilde{X} \mid X) := \frac{1}{Z(\sigma)} \exp \left( -\frac{d^2(\tilde{X}, X)}{4\sigma^2} \right),$$

(9)

where $Z(\sigma)$ is the normalization term, and $d$ is the metric that calculates the difference between $\tilde{X}$ and $X$. Riemann-Gaussian is a generalization version of typical Gaussian, by choosing various distances $d$ beyond the Euclidean metric. Here, to pursue the constraint in Eq. (3), we propose to use

$$d(X_1, X_2) = \|Y_1^\top Y_1 - Y_2^\top Y_2\|_F,$$

(10)

where $Y = X - \mu(X)$ re-positions $X$ towards zero mean ($\mu(X)$ denotes the mean of the columns in $X$). One clear benefit is that the distance function $d$ defined in Eq. (10) satisfies the doubly-$E(3)$-invariance constraint in Eq. (8). Note that $d$ is also permutation invariant with regard to the order of the columns of $\tilde{X}$ and $X$. We summary the above discussion as a formal proposition as follows.

Proposition 2. For Riemann-Gaussian $\text{Rie}_\sigma(\tilde{X} \mid X)$ defined in Eq. (9), it is doubly $E(3)$-invariant in accordance to Eq. (8).

The gradient of Riemann-Gaussian is derived as follows.

$$\nabla_{\tilde{X}} \log p_\sigma(\tilde{X} \mid X) = -\frac{1}{\sigma^2} \left[ (\tilde{Y} \tilde{Y}^\top) \tilde{Y} - (\tilde{Y} \tilde{Y}^\top) Y \right],$$

(11)

which is of the complexity $O(N)$, making it computationally efficient for even large-scale molecules. We provide the details of how the gradient is calculated in Appendix.

The last remaining recipe is how to sample $\tilde{X}$ from $X$ according to the Riemann-Gaussian distribution to provide the input to Eq. (7). It is non-straightforward to accomplish this goal, since the normalization term $Z(\sigma)$ of Riemann-Gaussian is unknown. Here we resort to Langevin dynamics [26] which is widely used for approximated sampling when only non-normalized probability density is given. We provide the details in Appendix. Furthermore, to better explore the conformation space, we employ a sampling scheme with multiple levels of noise [30]. Particularly, let $\{ \sigma_l \}_{l=1}^L$ be a series of noises with different scales. The final EFP loss is

$$\mathcal{L}_{\text{EFP-Final}} = \mathbb{E}_{\tilde{g} \sim \mathbb{Z}_l \sim U(1, L), \tilde{X} \sim p_{\sigma_l}(\tilde{X} \mid X)} \left[ \|F(\tilde{X}) - \frac{1}{\alpha} \nabla_{\tilde{X}} \log p_{\sigma_l}(\tilde{X} \mid X) \|^2_\mathcal{F} \right],$$

(12)

where $U(1, L)$ is the discrete uniform distribution, and $\nabla_{\tilde{X}} \log p_{\sigma_l}(\tilde{X} \mid X)$ is provided by Eq. (11). We have also added $\alpha$ as an invariant normalizing coefficient aiming to balance the magnitude induced by taking the inner-product for better numerical stability; its value is given by $\alpha = (\|\tilde{Y} \tilde{Y}^\top\|_\mathcal{F} + \|\tilde{Y} \tilde{Y}^\top\|_\mathcal{F})/2$ in our experiments.
3.3 Graph-Level Pretraining: Invariant Noise-scale Prediction

In the last subsection, we have constructed a node-level pretraining objective for local force prediction. To further discover global patterns within the input data, this subsection presents how to design a graph-level self-supervised task. Previous studies \cite{11, 23} have revealed for 2D molecules that the node- and graph-level tasks are able to promote each other. Here, we investigate this idea on 3D graphs.

Recalling that $\bar{X}$ is distributed by $p_{\sigma_l}(\bar{X} | X)$, it is expected that a well-behaved model should identify how much the perturbed sample is deviated from the original data. Such intuition inspires us to set up a classification problem as noise scale prediction. Specifically, our $\psi$ shares the same EGN backbone as $E$ in Eq. \eqref{eq2}, yielding exactly the same invariant node- and graph-level embeddings. For the input $X$ and $\bar{X}$, we first obtain their graph-level embedding $u$ and $\bar{u}$ via $\varphi_{\text{EGN}}$, respectively. Instead of using the scalar projection head $\varphi_{\text{Proj}}$ for energy computation, we employ a classification head $\varphi_{\text{Scale}}$ that takes as input a concatenation of the graph-level embeddings of the original conformation $u$ and the perturbed conformation $\bar{u}$. The output of $\varphi_{\text{Scale}}$ is the logits $p \in \mathbb{R}^L = \varphi_{\text{Scale}}(u∥\bar{u})$, where $L$ is the number of noise levels. Finally, a cross-entropy loss is computed between the logits and the label, which is the sampled noise level for the current input. The objective of the invariant noise-scale prediction task is thus given by

$$L_{\text{INP}} = \mathbb{E}_{G \sim \mathcal{G}, l \sim U(1, L), \bar{X} \sim p_{\sigma_l}(\bar{X} | X)} [L_{\text{CE}}(\|l\|, p)],$$

(13)

where $L_{\text{CE}}$ is the cross-entropy loss and $\|l\|$ is the one-hot encoding of $l$.

Our overall unsupervised training objective, illustrated in Fig.\ref{fig1} is a combination of both node-level equivariant force prediction loss and graph-level invariant noise scale prediction loss, leading to,

$$L = \lambda_1 L_{\text{EFP-Final}} + \lambda_2 L_{\text{INP}},$$

(14)

where $\lambda_1, \lambda_2$ are the balancing coefficients between the two objectives.

4 Experiments

4.1 Experimental Settings

Pretraining dataset We use a large-scale molecular dataset GEOM-QM9 \cite{2} with corresponding 3D conformations as our pretraining dataset. Specifically, we select the conformations with relatively-high Boltzmann weight for each molecule, and filter out the conformations that overlap with the testing molecules in downstream tasks, leading to 600k conformations in total.

Downstream tasks To thoroughly evaluate our proposed pretraining framework, we employ the two widely-adopted 3D molecular property prediction datasets: MD17 \cite{4} and QM9 \cite{22}, as the downstream tasks. In detail, MD17 contains the simulated dynamical trajectories of 8 small organic molecules, with the recorded energy and force at each frame. We select 9,500/500 frames as the training/validation set of each molecule. We jointly optimize the energy and force predictions by firstly obtaining the energy and deriving the force by $F = -\nabla_X E$. QM9 labels 12 chemical properties of small molecules with stable 3D structures. We follow the data split in Cormorant \cite{11} and EGNN \cite{25}, where the sizes of training, validation, and test sets are 100k, 18k, and 13k, respectively.

Baselines The baseline without any pretraining is termed as Base. Several widely-used 2D pretraining tasks are evaluated: AttrPred \cite{11} reconstructs the masked atom types; EdgePred \cite{9} predicts the existence of chemical bonds; GPT-GNN \cite{12} autoregressively generates the 2D graph in a pre-defined order; InfoGraph \cite{32} maximizes the mutual information between node and graph representations; GraphCL \cite{45} applies contrastive learning on graph representations through several augmentations. In addition, we also compare with GraphMVP \cite{13} which proposes contrastive and generative tasks based on 2D- and 3D-GNN models. Notably, different from the original setting in GraphMVP which focuses on the performance of the pretrained 2D GNN, we preserve its 3D model for our 3D tasks in the experiments. We further extend the 2D AttrPred task into 3D flavor, dubbed PosPred, which masks the positions of a random subset of atoms with the center of each input molecule, and then reconstructs the masked positions. For all above model-agnostic methods, we adapt exactly the same 3D backbone as our method, ensuring fairness. Particularly, we leverage
Table 1: MAE (lower is better) on MD17 for energy (top) and force (bottom) prediction tasks. All methods share the same backbone as Base.

|       | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Average |
|-------|---------|---------|---------|--------|-------|-----------|---------|--------|---------|
| Base [25] | 0.2044  | 0.0755  | 0.0532  | 0.0748 | 0.1961| 0.1289    | 0.1036  | 0.1165 | 0.1191  |
| AttrPred [11] | 0.1951  | 0.0709  | 0.0495  | 0.0796 | 0.1239| 0.1531    | 0.0924  | 0.1066 | 0.1089  |
| EdgePred [9]  | 0.2232  | 0.0717  | 0.0503  | 0.0739 | 0.1428| 0.1346    | 0.1018  | 0.0978 | 0.1120  |
| GPT-GNN [12] | 0.1656  | 0.0720  | 0.0488  | 0.0740 | 0.1370| 0.1460    | 0.0929  | 0.1040 | 0.1050  |
| InfoGraph [13] | 0.3320  | 0.0817  | 0.0577  | 0.0859 | 0.1356| 0.2122    | 0.1261  | 0.1392 | 0.1463  |
| GraphCL [45] | 0.1299  | 0.0706  | 0.0492  | 0.0722 | 0.1267| 0.1187    | 0.0901  | 0.1049 | 0.1053  |
| GraphMVP [18] | 0.1575  | 0.0853  | 0.0479  | 0.0726 | 0.2315| 0.1375    | 0.0964  | 0.1041 | 0.1166  |
| PosPred | 0.1470  | 0.0677  | 0.0497  | 0.0748 | 0.1101| 0.1161    | 0.0902  | 0.1007 | 0.0945  |
| 3D-MGP | **0.1118** | **0.0671** | **0.0484** | **0.0693** | **0.1101** | **0.1161** | **0.0902** | **0.1007** | **0.0945** |

|       | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Average |
|-------|---------|---------|---------|--------|-------|-----------|---------|--------|---------|
| Base [25] | 0.3885  | 0.1861  | 0.0599  | 0.1464 | 0.3310| 0.2683    | 0.1563  | 0.1323 | 0.2086  |
| AttrPred [11] | 0.3643  | 0.2277  | 0.0567  | 0.1456 | 0.1773| 0.3890    | 0.1093  | 0.1560 | 0.2032  |
| EdgePred [9]  | 0.4707  | 0.2036  | 0.0743  | 0.1268 | 0.2310| 0.3400    | 0.1854  | 0.1933 | 0.2281  |
| GPT-GNN [12] | 0.4278  | 0.2492  | 0.0703  | 0.1484 | 0.2080| 0.3609    | 0.1541  | 0.2219 | 0.2301  |
| InfoGraph [13] | 0.6578  | 0.2743  | 0.1257  | 0.2467 | 0.2860| 0.5793    | 0.3821  | 0.4238 | 0.3742  |
| GraphCL [45] | 0.2333  | 0.1845  | 0.0503  | 0.0852 | 0.0966| 0.1587    | 0.0725  | 0.1167 | 0.1247  |
| GraphMVP [18] | 0.3198  | 0.2800  | 0.0629  | 0.0788 | 0.2350| 0.2641    | 0.0903  | 0.1339 | 0.1831  |
| PosPred | 0.3050  | 0.2023  | 0.0519  | 0.0937 | 0.0971| 0.2481    | 0.0945  | 0.1270 | 0.1525  |
| 3D-MGP | **0.1560** | **0.1648** | **0.0389** | **0.0737** | **0.0829** | **0.1187** | **0.0619** | **0.0773** | **0.0968** |

EGNN [25] that takes as input both the 2D graphs and 3D coordinates. Detailed hyper-parameters are deferred to Appendix.

4.2 Main Results

Table 1 and 2 document the results of all pretraining methods on MD17 and QM9 respectively, where the underlined numbers indicate the previous SOTAs on that task, and the numbers in bold indicate the current best results. We explain the results by answering the questions as follows.

Q1. How does our 3D-MGP perform in general? It is observed from both Table 1 and Table 2 that 3D-MGP achieves the best performance in most cases, and its general effectiveness is better justified by checking the average MAE of the last column in Table 1. Particularly for force prediction, the superiority of 3D-MGP to other methods is more remarkable (3D-MGP achieves 0.0969, while the second best GraphCL is 0.1247), probably because the design of our node-level force prediction during pretraining is generalizable to the real force distribution after finetuning.

Q2. Are the 3D-aware pretraining tasks always helpful? Compared with Base, 3D-MGP consistently delivers meaningful improvement on MD17, and gains better performance on QM9 except the evaluation on $R^2$. We conjecture that the quantity $R^2$ assessing Electronic spatial extent is hardly recovered by the pretraining dataset, hence incurring negative transfer for all pretraining methods. Interestingly, PosPred usually behaves promisingly on MD17 even its 3D prediction objective is simple.

Q3. How do the traditional 2D methods perform on 3D tasks? Most 2D methods struggle especially for force prediction on MD17 and property prediction on QM9. For example, the average MAE of force prediction by InfoGraph is 0.3742, which is much worse than the Base method trained from scratch (0.2086). GraphCL, as the current state of the art, is able to enhance Base most of the time, but the enhancement is inferior to our 3D-MGP. Although GraphMVP has taken the 3D information into account, it almost performs worse than 3D-MGP, since GraphMVP mainly focuses on 2D tasks by only utilizing the 3D geometry as side information.

4.3 Ablation Studies

Contribution of each component As shown in Table 3, we conduct ablation studies on MD17 to show how each component in our model contributes. First, we inspect the contributions of the node-level task (i.e. EFP) and the graph-level task (i.e. INP). It is shown that both EFP and INP improve the performance individually, and their combination leads to more precise predictions. Second, to evaluate the importance of the proposed Riemann-Gaussian distribution, we relax the distribution in
Table 2: MAE (lower is better) on QM9 for property prediction tasks. All methods share the same backbone as Base.

| Method          | α   | ∆α | ϵHOMO | ϵLUMO | µ   | Cν | G   | H   | R²  | U   | U₀  | ZPVE |
|-----------------|-----|----|-------|-------|-----|----|-----|-----|-----|-----|-----|------|
| Base [25]       | 0.070 | 49.9 | 28.0  | 24.3  | 0.031 | 0.031 | 10.1 | 10.9 | **0.067** | 9.7 | 9.3 | 1.51  |
| AttrPred [11]   | 0.072 | 50.0 | 31.3  | 27.8  | 0.020 | 0.062 | 11.2 | 11.4 | 0.423 | 10.8 | 10.7 | 1.90  |
| EdgePred [9]    | 0.086 | 58.2 | 37.4  | 31.9  | 0.039 | 0.038 | 14.5 | 14.8 | 0.112 | 14.2 | 14.7 | 1.81  |
| GPF-GNN [13]    | 0.103 | 54.1 | 35.7  | 28.8  | 0.039 | 0.032 | 12.2 | 14.8 | 0.158 | 24.8 | 12.0 | 1.75  |
| InfoGraph [32]  | 0.099 | 72.2 | 48.1  | 38.1  | 0.041 | 0.030 | 16.5 | 14.5 | 0.114 | 14.9 | 16.4 | 1.69  |
| GraphCL [45]    | 0.066 | 45.5 | 26.8  | 22.9  | 0.027 | 0.028 | 10.2 | 9.6  | 0.095 | 9.7  | 9.6  | 1.42  |
| GraphMVP [18]   | 0.070 | 46.9 | 28.5  | 26.3  | 0.031 | 0.033 | 11.2 | 10.4 | 0.082 | 10.3 | 10.2 | 1.63  |
| PosPred         | 0.067 | 40.6 | 25.1  | 20.9  | 0.024 | 0.035 | 10.9 | 10.2 | 0.115 | 10.3 | 10.2 | 1.46  |
| 3D-MGP          | **0.057** | 37.1 | 21.3  | 18.2  | **0.020** | **0.026** | **9.3** | **8.7** | **0.092** | **8.6** | **8.6** | **1.38** |

Table 3: Ablation studies on MD17 tasks.

| Method          | EFP | INP | Riemann Distance | Energy-based | Energy Avg. | Force Avg. |
|-----------------|-----|-----|------------------|--------------|-------------|------------|
| Base            | ✓   | ✓   | ✓                | ✓            | 0.1191      | 0.2086     |
| Ours            | ✓   | ✓   | ✓                | ✓            | 0.0876      | 0.0968     |
| EFP only        | ✓   | ✓   | ✓                | ✓            | 0.0905      | 0.1193     |
| INP only        | ✓   | ✓   | ✓                | ✓            | 0.0974      | 0.1350     |
| Gaussian Distance | ✓   | ✓   | ✓                | ✓            | 0.0912      | 0.1060     |
| Direct Force    | ✓   | ✓   | ✓                | ✓            | 0.0914      | 0.1267     |

Eq. (9) as the Gaussian distribution $p(\tilde{X} | X) = \mathcal{N}(X, \sigma I)$, violating the doubly $E(3)$-invariance in Eq. (8). The results in Table 3 suggest that such relaxation causes certain performance detriment. While deriving Riemann-Gaussian distribution is motivated from the theoretical aspect, its benefit is also supported empirically. Besides, we analyze the necessity of the energy-based modeling proposed in § 3.1. Instead of deriving the force as the gradient of the energy model, it is also possible to straightforwardly apply the equivariant output from EGNN as the predicted force signal in the EFP loss in Eq. 7, which is called Direct Force. The last column in Table 3 reports that this variant suffers higher MAEs. From an algorithmic point of view, the energy-based strategy is able to better capture the global patterns and therefore lead to preferable performance, by first pooling the embeddings of all nodes as the energy and then computing the gradient of energy as force.

Table 4: MAE on MD17 with different backbones.

| Force          | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Avg. |
|----------------|---------|---------|---------|--------|-------|-----------|--------|--------|------|
| SchNet [27] w/ Pretraining | 0.4131 | 0.2374 | 0.0982 | 0.1781 | 0.1835 | 0.3145 | 0.1810 | 0.2438 | 0.2312 |
| ET [34] w/ Pretraining | 0.3186 | 0.1654 | 0.0607 | 0.1038 | 0.1229 | 0.1973 | 0.0860 | 0.1211 | **0.1470** |

The performance with different backbone model. We further apply our method to another two 3D backbones, SchNet [27] and Equivariant Transformer (ET) [34] to evaluate the generalization of the self-supervised tasks. Table 4 collects the results for force prediction on MD17. The compelling improvement verifies that our pretraining method generalizes well to a broad family of 3D backbones consistently.

4.4 Visualization

To probe the representation space of different pretrained models, we visualize the local energy landscape around a given conformation. To do so, we first fix the pretrained representation model and finetune an energy projection head on MD17 to fit ground-truth energy labels, in order to project the pretrained representations onto the energy surface. Note that there is initially no energy projection head for other methods, and we manually add an MLP on top of their backbone models similar to Eq. 2. After training the energy head, we select a random aspirin conformation $X$ from MD17 and
randomly generate two directions $D_1, D_2 \in \mathbb{R}^{3 \times N}$ according to Gaussian distribution. We construct a 2-dimension conformation plane as $\{\tilde{X}(i,j) | \tilde{X}(i,j) = X + iD_1 + jD_2\}$. For each point by varying the values of $i$ and $j$, we calculate its output energy by $E_{i,j} = E(\varphi_{\text{EGN}}(\tilde{X}(i,j)))$, where $E, \varphi_{\text{EGN}}$ denote the energy projection head and the pretrained model, respectively. Fig. 2 plots the energy landscape $(i,j,E_{i,j})$ for several compared approaches and our 3D-MGP. We interestingly find that the landscape by our method converges towards the original conformation smoothly and decreasingly, which implies the observed conformation corresponds to a metastable state with locally-lowest energy on the projected conformation plane. However, the 2D-based pretrained models such as EdgePred, AttrPred, and GraphCL deliver rugged landscapes, probably because their knowledge acquired from the pretraining process does not comply with the underlying energy distribution. The Base method outputs a flat surface, as it is less knowledgeable by solely learning from the small data.

5 Discussion

Conclusion. In this work, we proposed a general self-supervised pretraining framework for 3D tasks on molecules. It consists of a node-level Equivariant Force Prediction (EFP) and a graph-level Invariant Noise-scale Prediction (INP) task to jointly extract the geometric information from large-scale 3D molecular dataset. Experiments on MD17 and QM9 showcase the superiority of the proposed method to conventional 2D-based counterparts. Necessary ablations, visualizations, and analyses are also provided to support the validity of our design in terms of each component and the generalization ability of our method on different backbones.

Limitations and Future work. Our proposed general and effective 3D pretraining method encourages future research including but not limited to the following directions: 1. The extension of E(3)-equivariance models. For example, chirality, i.e., reflection asymmetry, plays an important role in molecule properties. It is promising to extend Eq. (5) into a doubly SE(3)-invariant flavor to distinguish the chiral molecules in 3D space. 2. More applications in other domains. It is expected that our work will facilitate more following research to devise representation learning on various kinds of 3D structures besides the small molecules discussed in this paper, such as point clouds [37] and proteins [36, 41].
References

[1] Brandon Anderson, Truong Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. *Advances in neural information processing systems*, 32, 2019.

[2] Simon Axelrod and Rafael Gomez-Bombarelli. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Scientific Data*, 9(1):1–14, 2022.

[3] Ludwig Boltzmann. Studien über das gleichgewicht der lebenden kraft. *Wissenschaftliche Abhandlungen*, 1:49–96, 1868.

[4] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017.

[5] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[6] Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3165–3176. PMLR, 2020.

[7] Fabian Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. Se(3)-transformers: 3d rotation-translation equivariant attention networks. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[8] 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.

[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[10] Jiaqi Han, Yu Rong, Tingyang Xu, and Wenbing Huang. Geometrically equivariant graph neural networks: A survey. *arXiv preprint arXiv:2202.07230*, 2022.

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

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

[13] Wenbing Huang, Jiaqi Han, Yu Rong, Tingyang Xu, Fuchun Sun, and Junzhou Huang. Equivariant graph mechanics networks with constraints. In *International Conference on Learning Representations*, 2022.

[14] Michael J Hutchinson, Charline Le Lan, Sheheryar Zaidi, Emilien Dupont, Yee Whye Teh, and Hyunjik Kim. Lietransformer: Equivariant self-attention for lie groups. In *International Conference on Machine Learning*, pages 4533–4543. PMLR, 2021.

[15] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30(8):595–608, 2016.
[16] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[17] Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. In *International Conference on Learning Representations*, 2019.

[18] Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. Pre-training molecular graph representation with 3d geometry. *CoRR*, abs/2110.07728, 2021.

[19] 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.

[20] Shitong Luo and Wei Hu. Differentiable manifold reconstruction for point cloud denoising. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1330–1338, 2020.

[21] Shitong Luo, Chence Shi, Minkai Xu, and Jian Tang. Predicting molecular conformation via dynamic graph score matching. *Advances in Neural Information Processing Systems*, 34, 2021.

[22] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

[23] 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 Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020*, NeurIPS 2020, December 6-12, 2020, virtual, 2020.

[24] Salem Said, Lionel Bombrun, Yannick Berthoumieu, and Jonathan H Manton. Riemannian gaussian distributions on the space of symmetric positive definite matrices. *IEEE Transactions on Information Theory*, 63(4):2153–2170, 2017.

[25] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International Conference on Machine Learning*, pages 9323–9332. PMLR, 2021.

[26] Tamar Schlick. *Molecular modeling and simulation: an interdisciplinary guide*, volume 2. Springer, 2010.

[27] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems*, 30, 2017.

[28] K. T. Schütt, P. Kessel, M. Gastegger, K. A. Nicoli, A. Tkatchenko, and K.-R. Müller. Schnetpack: A deep learning toolbox for atomistic systems. *Journal of Chemical Theory and Computation*, 15(1):448–455, 2019.

[29] Chence Shi, Shitong Luo, Minkai Xu, and Jian Tang. Learning gradient fields for molecular conformation generation. In *International Conference on Machine Learning*, pages 9558–9568. PMLR, 2021.

[30] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

[31] Hannes Stärk, Dominique Beaini, Gabriele Corso, Prudencio Tossou, Christian Dallago, Stephan Günnemann, and Pietro Liò. 3d infomax improves gnns for molecular property prediction. *CoRR*, abs/2110.04126, 2021.

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

[33] Mengying Sun, Jing Xing, Huijun Wang, Bin Chen, and Jiayu Zhou. Mocl: Contrastive learning on molecular graphs with multi-level domain knowledge. *CoRR*, abs/2106.04509, 2021.
[34] Philipp Thölke and Gianni De Fabritiis. Equivariant transformers for neural network based molecular potentials. In International Conference on Learning Representations, 2021.

[35] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. arXiv preprint arXiv:1802.08219, 2018.

[36] Raphael John Lamarre Townshend, Martin Vögele, Patricia Adriana Suriana, Alexander Derry, Alexander Powers, Yianni Laloudakis, Siddhika Balachandar, Bowen Jing, Brandon M Anderson, Stephan Eismann, et al. Atom3d: Tasks on molecules in three dimensions. In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1), 2021.

[37] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In Proceedings of the IEEE/CVF international conference on computer vision, pages 1588–1597, 2019.

[38] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019.

[39] Pascal Vincent. A connection between score matching and denoising autoencoders. Neural computation, 23(7):1661–1674, 2011.

[40] Izhar Wallach, Michael Dzamba, and Abraham Heifets. Atomnet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery. arXiv preprint arXiv:1510.02855, 2015.

[41] Renxiao Wang, Xueliang Fang, Yipin Lu, Chao-Yie Yang, and Shaomeng Wang. The pdbbind database: methodologies and updates. Journal of medicinal chemistry, 48(12):4111–4119, 2005.

[42] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? arXiv preprint arXiv:1810.00826, 2018.

[43] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? Advances in Neural Information Processing Systems, 34, 2021.

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

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

[46] Liangzhen Zheng, Jingrong Fan, and Yuguang Mu. Onionnet: a multiple-layer intermolecular-contact-based convolutional neural network for protein–ligand binding affinity prediction. ACS omega, 4(14):15956–15965, 2019.

A Theoretical Analysis

A.1 Proof of Proposition 1

Proposition 1. For an arbitrary noise sampling \( p(\tilde{X} \mid X) \), \( \mathcal{L}_{\text{EFP}} = \mathcal{L}_{\text{EFP-DN}} + C \) where \( C \) is a constant independent to \( \varphi_{\text{EGN}} \), if certain mild conditions hold.

Proposition 1 is equivalent to Eq. (11) in [39]. See Appendix of [39] for detailed proofs.
A.2 Calculation of the Gradients of Riemann-Gaussian Distribution

According to Eq. (9) and Eq. (10), we can derive the formulation of the proposed Riemann-Gaussian distribution as

\[ p_\sigma(\tilde{X} \mid X) = \frac{1}{Z(\sigma)} \exp \left( -\frac{d^2(\tilde{X}, X)}{4\sigma^2} \right) \]

and its partial derivative to \( \tilde{X} \) is

\[ \nabla_{\tilde{X}} p_\sigma(\tilde{X} \mid X) = \nabla_Y p_\sigma(\tilde{X} \mid X) - \mu(\nabla_Y p_\sigma(\tilde{X} \mid X)) \]

where, \( \mu(\cdot) \) computes the mean of the columns, and \( \mu(\tilde{Y}\tilde{Y}^\top) = \mu(YY^\top) = 0 \) since the columns of both \( \tilde{Y} \) and \( Y \) have zero mean. Note that \( \tilde{Y}\tilde{Y}^\top, YY^\top \in \mathbb{R}^{3 \times 3} \) can be calculated in \( O(N) \).

A.3 Proof of Proposition 2

**Proposition 2.** For Riemann-Gaussian Rie_\sigma(\tilde{X} \mid X) defined in Eq. (9), it is doubly E(3)-invariant in accordance to Eq. (8).

To prove Proposition 2, we first introduce the following lemma.

**Lemma 1.** Let \( \tilde{Y} = X - \mu(X) \). Then \( \mathcal{F}(X) = Y^\top Y \) is E(3)-invariant, i.e., \( Y_1^\top Y_1 = Y_2^\top Y_2 \) if \( X_1 = g \cdot X_2, \forall g \in E(3) \).

**Proof.** Consider \( X_1 = g \cdot X_2 = OX_2 + t \), where \( O \in \mathbb{R}^{3 \times 3} \) is an orthogonal matrix which represents rotation/reflection and \( t \in \mathbb{R}^3 \) is the translation vector. For \( Y_1, Y_2 \), we have

\[ Y_1 = X_1 - \mu(X_1) = OX_2 + t - \mu(OX_2 + t) = OX_2 - (O\mu(X_2) = OY_2. \]

That is, the translation transformation is reduced by the zero mean operation. Moreover, for the inner-product computation, we have

\[ \mathcal{F}(X_1) = Y_1^\top Y_1 = Y_2^\top O^\top O Y_2 = Y_2^\top Y_2 = \mathcal{F}(X_2). \]

\[ \square \]

Based on the above lemma, we can directly prove Proposition 2 as follows.
Proof.

\[
p_\sigma(g_1 \cdot \mathbf{X} \mid g_2 \cdot \mathbf{X}) = \frac{1}{Z(\sigma)} \exp \left( -\frac{d^2(g_1 \cdot \mathbf{X}, g_2 \cdot \mathbf{X})}{4\sigma^2} \right)
\]

\[
= \frac{1}{Z(\sigma)} \exp \left( -\frac{\|\mathcal{F}(g_1 \cdot \mathbf{X}) - \mathcal{F}(g_2 \cdot \mathbf{X})\|^2_F}{4\sigma^2} \right)
\]

\[
= \frac{1}{Z(\sigma)} \exp \left( -\frac{\|\mathcal{F}(\tilde{X}) - \mathcal{F}(\mathbf{X})\|^2_F}{4\sigma^2} \right)
\]

\[
= p_\sigma(\tilde{X} \mid \mathbf{X}),
\]

which proves the doubly E(3)-invariance of the Riemann-Gaussian distribution.

\[ \Box \]

A.4 Sampling from Riemann-Gaussian Distribution

We adopt Langevin Dynamics \cite{26} to approximately sample noisy coordinates \( \tilde{X} \) from \( \mathbf{X} \) according to the proposed Riemann-Gaussian distribution. The detailed algorithm is shown as follows.

\begin{algorithm}[H]
\caption{Riemann-Gaussian Sampling via Langevin Dynamics}
\begin{algorithmic}
\State \textbf{Input:} Original coordinates \( \mathbf{X} \), Noise scale \( \sigma \), Maximum iteration step \( T \).
\State Initialize \( \mathbf{X}_0 \leftarrow \mathbf{X} \)
\For{step \( t = 1 \) to \( T \)}
\State Derive \( s_t \leftarrow \nabla_{\mathbf{X}_{t-1}} \log \text{Rie}_\sigma(\mathbf{X}_{t-1} | \mathbf{X}) \) according to Eq. (11);
\State Sample \( \epsilon_t \sim \mathcal{N}(0, 1) \);
\State Calculate normalization term \( \beta_t = (\|\mathbf{Y}_{t-1}^{\top} \mathbf{Y}_{t-1}\|_F + \|\mathbf{Y}_{t-1} \mathbf{Y}^{\top}\|_F)/2 \);
\State \( \mathbf{X}_t = \mathbf{X}_{t-1} + \alpha_t s_t / \beta_t + \sqrt{2\alpha_t} \epsilon_t \);
\EndFor
\State \textbf{Output:} Sampled coordinates \( \mathbf{X}_T \).
\end{algorithmic}
\end{algorithm}

For implementation, we find that the model performance is not sensitive to \( T \), and choosing \( T = 1 \) is already able to yield favorable results. See Appendix B.2 for more details.

B Experimental Details

B.1 Hyper-parameter Settings

We apply the EGNN model with 7 layers, 128 hidden dimensions for all methods during the pretraining and finetuning procedure. The atom coordinates are fixed among layers. We represent a molecule as a fully connected graph, and take atom types as the node feature. The edge feature \( e_{ij} \in \{1\text{-hop, 2-hop, 3-hop, others}\} \) is determined by the minimum hops between atom \( i, j \) and models the bond, angle, torsion and non-bond interactions in molecules similar to previous works \cite{17, 21, 43}. For all pretraining methods, we train the model with epoch 300, batch size 64×4 GPUs, Adam optimizer, and cosine decay with initial learning rate \( 1 \times 10^{-3} \). For our 3D-MGP, we select \( \sigma_1 = 10, \sigma_L = 0.01, L = 50 \), uniformly decreasing in log scale, similar to previous generation methods \cite{29, 21} and \( \lambda_1 = 1.0, \lambda_2 = 0.2 \) to balance the EFP and INP tasks. For AttrPred, PosPred, GraphCL and GraphMVP which require mask operations, we set the mask ratio as 0.15. For PosPred, the coordinates are updated only in the last layer. For GraphMVP, we adopt a GIN model with 5 layers, 128 hidden dimensions as the 2D-GNN side. For QM9, we finetune the model on each individual task with epoch 1,000, batch size 96, Adam optimizer with weight decay \( 1 \times 10^{-16} \), and cosine decay with initial learning rate \( 5 \times 10^{-4} \). For MD17, we jointly learn the energy and forces of each molecule trajectory with epoch 1,500, batch size 100, Adam optimizer with weight decay \( 1 \times 10^{-16} \), and ReduceLROnPlateau scheduler with patience 30, decay factor 0.9, and initial learning rate within \( \{1 \times 10^{-3}, 5 \times 10^{-4}\} \). We set \( \lambda_{\text{energy}} = 0.2, \lambda_{\text{force}} = 0.8 \) to balance the loss of energy and forces. All training procedures are conducted on NVIDIA Tesla V100 GPUs.
B.2 Complete Results for Ablation Studies

Contribution of each component We implement 4 variants of the original 3D-MGP to explore the contribution of each component. **EFP only** and **INP only** adjust the weight to $\lambda_1 = 1, \lambda_2 = 0$ and $\lambda_1 = 0, \lambda_2 = 1$. **Gaussian Distance** changes the force target in Eq. (11) into $\nabla \tilde{X} \log p(\tilde{X} \mid X) = -\tilde{X} - X / \sigma^2$, and **Direct Force** updates the coordinates in the last layer of EGNN and predicts the atom forces from the difference of the output coordinates and the input one, i.e., $F = 1 / \sigma (\varphi_{\text{EGN}}(X, H) - X)$, where $1 / \sigma$ is a normalization term to balance different noise scale. Detailed results for MD17 energy and force prediction tasks are shown in Table 5, where the original model achieves better averaged MAEs than the variants. The results indicate the effectiveness of each component in our proposed 3D-MGP.

| Energy | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Average |
|--------|---------|---------|---------|--------|-------|-----------|---------|--------|---------|
| Base   | 0.2044  | 0.0755  | 0.0532  | 0.0748 | 0.1961| 0.1289    | 0.1036  | 0.1165 | 0.1191  |
| Ours   | 0.1118  | 0.0671  | 0.0484  | 0.0693 | 0.1107| 0.1058    | 0.0874  | 0.1001 | 0.0876  |
| EFP only | 0.1180  | 0.0695  | 0.0485  | 0.0713 | 0.1156| 0.1111    | 0.0903  | 0.0999 | 0.0905  |
| INP only | 0.1593  | 0.0684  | 0.0494  | 0.0750 | 0.1217| 0.1199    | 0.0892  | 0.0962 | 0.0974  |

| Gaussian Distance | 0.1214 | 0.0759 | 0.0499 | 0.0716 | 0.1176 | 0.1050 | 0.0895 | 0.0984 | 0.0912 |
| Direct Force      | 0.1179 | 0.0661 | 0.0498 | 0.0725 | 0.1161| 0.1163    | 0.0917  | 0.1006 | 0.0914  |

The performance with different backbone models We apply 3D-MGP on SchNet and Equivariant Transformer (ET) with 7 layers, 128 hidden dimensions. The edge feature is concatenated with the RBF distance expansions. As shown in Table 6, 3D-MGP performs consistently well on a broad family of 3D backbones.

| Energy | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Avg. |
|--------|---------|---------|---------|--------|-------|-----------|---------|--------|------|
| SchNet | 0.1502  | 0.0725  | 0.0505  | 0.0780 | 0.1103| 0.1220    | 0.0963  | 0.1078 | 0.0985 |
| w/ Pretraining | 0.1384 | 0.0680  | 0.0489  | 0.0721 | 0.1104| 0.1093    | 0.0908  | 0.0996 | 0.0922 |
| ET     | 0.1050  | 0.0711  | 0.0495  | 0.0710 | 0.1120| 0.1026    | 0.0904  | 0.1010 | 0.0878 |
| w/ Pretraining | 0.1038 | 0.0682  | 0.0485  | 0.0707 | 0.1105| 0.1023    | 0.0906  | 0.1003 | 0.0869 |

Influence of sampling steps We conduct EFP pretraining task with sampling steps $T = 1, 5, 10$ in Algorithm 1 and compare the finetuning performance on MD17. As shown in Table 7, different sampling steps achieve similar results, which implies that our method is not sensitive to $T$. Hence, we select $T = 1$ for simplicity and efficiency in the remaining experiments.

| Energy | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Avg. |
|--------|---------|---------|---------|--------|-------|-----------|---------|--------|------|
| SchNet | 0.4131  | 0.2374  | 0.0982  | 0.1781 | 0.1835| 0.3145    | 0.1810  | 0.2438 | 0.2312 |
| w/ Pretraining | 0.3186 | 0.1654  | 0.0607  | 0.1038 | 0.1229| 0.1973    | 0.0860  | 0.1211 | 0.1470 |
| ET     | 0.1216  | 0.1479  | 0.0492  | 0.0695 | 0.0390| 0.0655    | 0.0393  | 0.0484 | 0.0726 |
| w/ Pretraining | 0.1124 | 0.1417  | 0.0445  | 0.0618 | 0.0352| 0.0586    | 0.0385  | 0.0477 | 0.0676 |

Balance between node- and graph-level tasks We attempt different loss weights for pretraining and evaluate the model performance on MD17. As shown in Table 8, we find that: 1. Combination
Table 7: MAE on MD17 with EFP pretraining on different sampling steps.

| Energy | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Average |
|--------|---------|---------|---------|--------|-------|-----------|---------|--------|---------|
| Base   | 0.2044  | 0.0755  | 0.0532  | 0.0748 | 0.1961| 0.1289    | 0.1036  | 0.1165 | 0.1191  |
| $T = 1$| 0.1180  | 0.0695  | 0.0485  | 0.0713 | 0.1156| 0.1111    | 0.0903  | 0.0999 | 0.0905  |
| $T = 5$| 0.1119  | 0.0731  | 0.0493  | 0.0731 | 0.1111| 0.1052    | 0.0926  | 0.1025 | 0.0899  |
| $T = 10$| 0.1194 | 0.0733  | 0.0499  | 0.0711 | 0.1120| 0.1062    | 0.0919  | 0.1005 | 0.0905  |

Table 8: MAE on MD17 energy (top) and forces (bottom) tasks with different loss weights. $\lambda_1$, $\lambda_2$ stand for the loss weights for EFP and INP tasks, respectively.

| $\lambda_1$ | $\lambda_2$ | Aspirin | Benzene | Ethanol | Malon. | Naph. | Salicylic | Toluene | Uracil | Average |
|-------------|-------------|---------|---------|---------|--------|-------|-----------|---------|--------|---------|
| 1.0         | 0.2         | 0.1118  | 0.0671  | 0.0484  | 0.0693 | 0.1107| 0.1058    | 0.0874  | 0.1001 | 0.0876  |
| 0.2         | 1.0         | 0.1083  | 0.0723  | 0.0496  | 0.0717 | 0.1131| 0.1015    | 0.0883  | 0.0971 | 0.0877  |
| 0.5         | 0.5         | 0.1088  | 0.0696  | 0.0486  | 0.0710 | 0.1153| 0.1041    | 0.0933  | 0.1002 | 0.0889  |
| 1.0         | 0.0         | 0.1180  | 0.0695  | 0.0485  | 0.0713 | 0.1156| 0.1111    | 0.0903  | 0.0999 | 0.0905  |
| 0.0         | 1.0         | 0.1593  | 0.0684  | 0.0494  | 0.0750 | 0.1217| 0.1199    | 0.0892  | 0.0962 | 0.0974  |

C More Visualizations

We further apply the visualization method on QM9 by finetuning an energy head on $U$ (Internal Energy at 298K) prediction task. Different from MD17, which provides a dynamic trajectory for one given molecule, QM9 provides only one metastable conformer for each molecule. As shown in Fig. 3, the visualization results indicate that: 1. 2D generative methods, like AttrPred or EdgePred, create irregular energy surface w.r.t. the changes of 3D coordinates. 2. GraphCL and GraphMVP generate upper convex surface, which contradicts with the fact that metastable conformers stay in the local minima in the energy landscape, probably because these methods have never been exposed to the perturbed unstable conformers during the backbone pretraining and energy head finetuning stages. 3. Our method still delivers a smooth conical surface converging towards the metastable point, which is consist with the phenomenon on MD17.

D Code and Related Assets

Our code is available at [https://anonymous.4open.science/r/3D-MGP-4931](https://anonymous.4open.science/r/3D-MGP-4931). The implementation and hyper-parameter settings are partially based on ConfGF [29], a generative model via denoising score matching. The implementation of baseline methods refers to GraphMVP [13]. The implementation of backbone equivariant geometric networks refers to the official codes of EGNN [25], SchNetPack [28], TorchMD-Net [34] and Pytorch Geometric [5]. Detailed URLs and licenses are listed in Table 9.
Figure 3: Energy landscape visualization on QM9.

Table 9: List of open-source assets used.

| Asset            | URL                                      | license   |
|------------------|------------------------------------------|-----------|
| ConfGF [29]      | https://github.com/DeepGraphLearning/ConfGF | MIT license |
| GraphMVP [18]    | https://github.com/chao1224/GraphMVP      | N/A       |
| EGNN [25]        | https://github.com/vgsatorras/egnn        | MIT license |
| SchNetPack [28]  | https://github.com/atomistic-machine-learning/schnetpack | MIT license |
| TorchMD-Net [34] | https://github.com/torchmd/torchmd-net    | MIT license |
| Pytorch Geometric [5] | https://github.com/pyg-team/pytorch_geometric | MIT license |