
Manipulating 3D Molecules in a Fixed-Dimensional E(3)-Equivariant Latent Space

Zitao Chen^{1,2*} Yijun Jia^{1*†} Zitong Tian^{1,3*} Wei-Ying Ma¹ Yanyan Lan^{1,4,5†}

¹ Institute for AI Industry Research (AIR), Tsinghua University

² Department of Computer Science and Technology, Tsinghua University

³ Qiuzhen College, Tsinghua University

⁴ Beijing Frontier Research Center for Biological Structure, Tsinghua University

⁵ Beijing Academy of Artificial Intelligence

{chen-zt23,tzt23}@mails.tsinghua.edu.cn

{jiayinjun, maweiming, lanyanyan}@air.tsinghua.edu.cn

Abstract

Medicinal chemists often optimize drugs considering their 3D structures and designing structurally distinct molecules that retain key features, such as shapes, pharmacophores, or chemical properties. Previous deep learning approaches address this through supervised tasks like molecule inpainting or property-guided optimization. In this work, we propose a flexible zero-shot molecule manipulation method by navigating in a shared latent space of 3D molecules. We introduce a Variational AutoEncoder (VAE) for 3D molecules, named MolFLAE, which learns a fixed-dimensional, E(3)-equivariant latent space independent of atom counts. MolFLAE encodes 3D molecules using an E(3)-equivariant neural network into fixed number of latent nodes, distinguished by learned embeddings. The latent space is regularized, and molecular structures are reconstructed via a Bayesian Flow Network (BFN) conditioned on the encoder’s latent output. MolFLAE achieves competitive performance on standard unconditional 3D molecule generation benchmarks. Moreover, the latent space of MolFLAE enables zero-shot molecule manipulation, including atom number editing, structure reconstruction, and coordinated latent interpolation for both structure and properties. We further demonstrate our approach on a drug optimization task for the human glucocorticoid receptor, generating molecules with improved hydrophilicity while preserving key interactions, under computational evaluations. These results highlight the flexibility, robustness, and real-world utility of our method, opening new avenues for molecule editing and optimization.³

1 Introduction

Structure-guided molecule optimization is a crucial task in drug discovery. Medicinal chemists edit molecular structures to improve binding affinity, selectivity, and ADMET (absorption, distribution, metabolism, excretion, and toxicity) properties. These modifications can range from subtle changes, such as introducing a chlorine [1] or methyl [2], to more extensive transformations like deconstruction and reconstruction of known ligands [3] or designing chimera molecules that combine beneficial

*Equal contribution.

†Corresponding author.

³The code is available at <https://github.com/MuZhao2333/MolFLAE>

features of different scaffolds [4, 5]. These diversified tasks present exciting opportunities for deep learning models to accelerate real-world drug design.

Previous generative approaches typically decompose 3D molecule editing into a set of narrowly defined subtasks. Notable progress has been made in molecular inpainting [6, 7, 8], property-guided optimization [9, 10], and shape-conditioned regeneration [11]. While effective, these models often rely on task-specific supervision and architectures, limiting their flexibility and generalizability. Moreover, not all molecule editing tasks align well with the supervised learning paradigm. For example, adding substituents may be too trivial to justify training a specialized model, while complex tasks like scaffold hopping by integrating known actives are often data-scarce for supervised approaches. These limitations call for a more flexible framework capable of supporting a broad spectrum of molecule editing tasks in a unified, data-efficient manner.

Previous successes in image editing and style transfer [12, 13, 14] show that latent space navigation allows for powerful, general-purpose manipulations by perturbing latent vectors. However, 3D molecule generation presents unique challenges not encountered in image domains. Molecules consist of variable numbers of atoms, and they exhibit permutation invariance to the atom order and SE(3)-equivariance to the spatial translation and rotation. These characteristics make latent space modeling significantly more challenging, and most existing 3D generative models operate on the product of latent spaces of each atom or functional group [15, 16, 17, 18, 19], resulting in variable dimensional representations. This variability prohibits common operations on vectors, such as interpolation or extrapolation, which are common to image generative models.

To address these challenges, we propose MolFLAE (Molecule Fixed Length AutoEncoder), a Variational AutoEncoder (VAE) for 3D molecules that learns a fixed-dimensional, E(3)-equivariant latent space, independent of atom counts. Our encoder employs an E(3)-equivariant neural network that updates a fixed number of virtual nodes initialized with learnable embeddings, transforming them into fixed-length latent codes for 3D molecules. The latent space is regularized under the standard VAE framework, and a Bayesian Flow Network (BFN) serves as the decoder, reconstructing full molecular structures conditioned on latent codes.

Our autoencoder framework supports a wide range of downstream applications. We first demonstrate that our model can unconditionally generate diverse, valid molecules, achieving competitive performance on standard 3D molecular generation benchmarks. More importantly, our fixed-dimensional latent space enables rich, semantically meaningful manipulations. We show that molecular analogs can be created with varying atom counts, covering simple substitutions to ring contractions. Molecules can also be reconstructed on the shape and orientation of other molecules, yielding chemically plausible outputs. Furthermore, interpolating between two latent codes produces chimera molecules that combine substructures and properties from both parents. Finally, we demonstrate a real-world application of our method in a drug optimization task targeting the human glucocorticoid receptor. We design new molecules that preserve the key binding interactions of known actives while achieving a better balance of potency and hydrophilicity. These results illustrate the flexibility, robustness, and practical utility of our model, highlighting the promise of latent space manipulation as a powerful tool for molecular editing and optimization. Our main contributions are:

- We propose a VAE model that learns a **fixed-dimensional, E(3)-equivariant latent space** for 3D molecular structures;
- The learned latent space enables a **wide range of molecule manipulation tasks**, including analog design, molecule reconstruction, and structure-properties co-interpolation;
- We **introduce quantitative metrics** to evaluate the disentanglement of spatial and semantic latent components, as well as the quality of structural and property interpolation.

2 Related Works

Unconditional Generation for 3D Molecules Unconditional 3D molecule generation has achieved rapid advancements, driven by progress in deep generative models. Early works explored autoregressive models to construct molecules atom-by-atom [20, 21]. More recently, inspired by diffusion-like models [22, 23, 24, 25, 26], models like EDM [15], EquiFM [16], and GeoBFN [17], have significantly improved generation quality. VAEs [27] offer an alternative by decoding latent embeddings into molecules, but modeling 3D molecules is challenging due to variable atom counts and the need

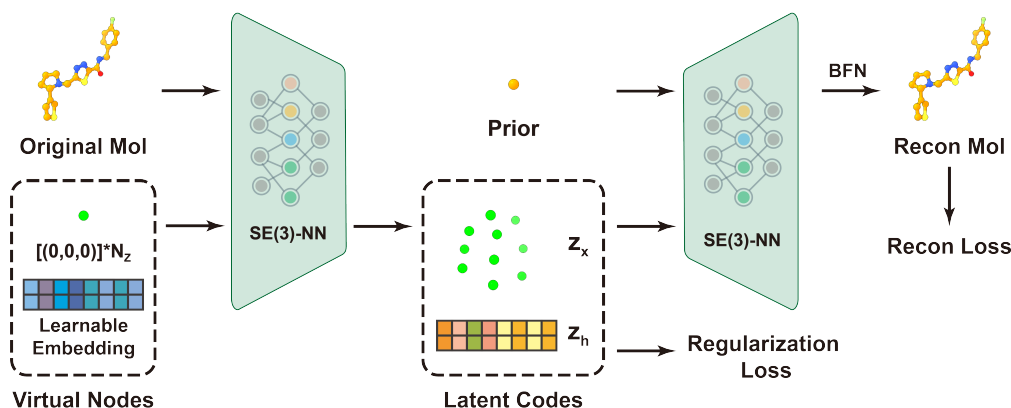


Figure 1: The architecture of MolFLAE. 3D molecules are transformed into latent codes and decoded with BFN. MolFLAE is trained with the recon. loss and the regularization loss of the latent code.

for equivariance of coordinates to rotations and translations. Existing 3D molecular VAEs such as PepGLAD [19] and UniMoMo [28] have made progress by encoding molecular blocks (e.g., amino acid residues) into independent latent nodes. This design ensures E(3)-equivariance but has two key limitations: (1) the number of latent nodes depends on the molecular composition, complicating cross-sample comparisons and interpolation tasks; and (2) spatial relationships between molecular blocks are tightly preserved, restricting the flexibility of the latent space for generative modeling, especially when latent codes are repurposed for tasks like autoregressive modeling by finetuning LLMs.

A key limitation of most unconditional 3D generative models is that their latent spaces vary in length. This variability prohibits operations such as interpolation. Consequently, zero-shot molecule editing becomes non-trivial or even infeasible. To address this, researchers have designed specific models for molecule editing or constructed fixed-dimensional latent spaces, which will be discussed in the following sections.

3D Molecule Editing 3D molecule editing has traditionally been divided into several specialized subtasks. Linker design focuses on connecting fragments to form valid new compounds [6, 7], while scaffold inpainting involves masking molecule cores and regenerating them [8]. These two tasks share similarities and can be unified under a mask-prediction framework. However, other editing tasks are harder to generalize. For example, DecompOpt [29] decomposes ligands into substructures and trains a diffusion model conditioned on these substructures, enabling deconstruction-reconstruction of 3D molecules. Another task is property-guided molecule optimization, where molecules are directly edited to improve specific properties using explicit guidance signals. Approaches such as gradient-based optimization [9] and classifier-free guidance [10] have been explored to this end.

While these task-specific methods have demonstrated strong performance, they often lack flexibility and generality. This raises the question if we can achieve more elegant and general 3D molecule manipulation by navigating in the latent space. This motivation has led researchers to explore unified, fixed-length, and semantically meaningful latent spaces for 3D molecules.

Generate 3D Molecules from Fixed-Dimensional Latent Spaces Fixed-length autoencoders for molecules have been explored through voxel-based models, which discretize 3D space into uniform grids and apply 3D CNNs [30, 31] or neural fields [32]. While straightforward, these methods are not E(3)-equivariant and struggle to disentangle semantic features from orientations, making molecule manipulation in the latent space not flexible. Local-frame-based models [33] represent conformations with SE(3) invariant features, including distances, bond angles, and dihedrals, making outputs orientation-agnostic. UAE-3D [34] has proposed a 3D fixed-length latent space by discarding the inductive bias of geometric equivariance, yet their latent space suffers from similar feature entanglement problem as voxel-based models. A recent work [35] also constructs fixed-length autoencoder, but applies global pooling over atom features, discarding spatial information that is critical for reconstruction and interaction modeling.

In contrast, our method preserves equivariance without requiring data augmentation, contains spatial information, avoids voxelization, and partially disentangles spatial and semantic information which enables unconditional generation and zero-shot molecule editing.

3 Methodology

We encode a variable-size 3D molecule by concatenating it with a set of learnable virtual nodes and updating them together via an E(3)-equivariant neural network to obtain fixed-length embeddings. The regularization loss $\mathcal{L}_{\text{reg}}$ comes from an Multi-Layer Perceptron (MLP) predicting the means and variances to form a variational posterior, from which we sample latent codes that condition a Bayesian Flow Network (BFN) decoder for reconstruction producing the reconstruction loss $\mathcal{L}_{\text{recon}}$. The MolFLAE is trained end-to-end by minimizing

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{reg}}. \quad (1)$$

We recall the classical Variational AutoEncoder [27] (VAE) in Sec. 3.1 that inspires the above loss. Then introduce BFN in Sec. 3.2. And we introduce the encoder and decoder of MolFLAE in Sec. 3.3 Sec. 3.4.

3.1 Variational AutoEncoder

Let $q_{\theta}(\mathbf{z} | \mathbf{x})$ denote the encoder, $p_{\phi}(\mathbf{x} | \mathbf{z})$ the decoder, and $p(\mathbf{z})$ the prior distribution of the latent codes, which is typically chosen as a standard Gaussian. The VAE loss is the negative Evidence Lower Bound (ELBO):

$$\mathcal{L}_{\text{VAE}}(\theta, \phi; \mathbf{x}) = \underbrace{\mathbb{E}_{q_{\theta}(\mathbf{z} | \mathbf{x})} [-\log p_{\phi}(\mathbf{x} | \mathbf{z})]}_{\text{reconstruction loss}} + \underbrace{D_{\text{KL}}(q_{\theta}(\mathbf{z} | \mathbf{x}) \| p(\mathbf{z}))}_{\text{regularization loss}} \quad (2)$$

Our training loss Eq. 1 is inspired by the above VAE loss.

3.2 Bayesian Flow Network

The Bayes Flow Network (BFN) [26] incorporates Bayesian inference to modify the parameters of a collection of independent distributions and using a neural network to integrate the contextual information. Unlike standard diffusion models [22, 23] that primarily handle continuous data via Gaussian noise, BFN extends this paradigm to support both continuous and discrete data types, including categorical variables. This flexibility makes BFN particularly suitable for modeling 3D molecules [17], where the data naturally consists of mixed modalities of continuous coordinates and discrete atom types.

Unlike traditional generative models that operate directly on data, BFN performs inference in the space of distribution parameters. Given a molecule $\mathbf{m}$, the *sender distribution* $p_S(\mathbf{y} | \mathbf{m}; \alpha)$ transforms it into a parameterized noisy distribution by adding noise analogous to the forward process in diffusion models. A brief introduction of BFN can be found in Appendix C.1.

3.3 MolFLAE Encoder

Inspired by the autoencoder in natural languages models [36] who uses of [CLS] tokens for context compression, we append N_Z learnable virtual nodes to the molecule's point cloud and treat the concatenation of their final embeddings as our fixed-length latent codes.

We denote the coordinates and atom type feature of the i -th atom in molecule $\mathcal{M}$ by $\mathbf{x}_M^{(i)} \in \mathbb{R}^3$ and $\mathbf{v}_M^{(i)} \in \mathbb{R}^{D_M}$, respectively. The full molecular input and the learnable virtual nodes are represented as $\mathcal{M} = [\mathbf{x}_M, \mathbf{v}_M]$ and $\mathcal{Z} = [\mathbf{x}_Z, \mathbf{v}_Z]$. The virtual nodes are some artificial atoms with the same size as the real atoms.

We remark that, in order to effectively encode the 3D configuration and chirality of a molecule, the number of virtual nodes N_Z must be at least 4 so that they can form a non-degenerate simplex in 3D space. To ensure sufficient capacity for capturing complex spatial structures, we set $N_Z = 10$.

We employ an E(3)-equivariant neural network ϕ_θ to jointly encode the original molecular point cloud and the appended virtual nodes ($\mathcal{M}, \mathcal{Z}$). After rounds of update, we discard the embeddings of $\mathcal{M}$ and retain only those of $\mathcal{Z}$ as our fixed-length latent representation. In explicit,

$$(-, [\mathbf{z}_x, \mathbf{z}_h]) = \phi_\theta([\mathbf{x}_M, \mathbf{v}_M], [\mathbf{x}_Z, \mathbf{v}_Z]). \quad (3)$$

The latent code is denoted as $[\mathbf{z}_x, \mathbf{z}_h] \in \mathbb{R}^{N_Z \times (3+D_f)}$, where $\mathbf{z}_x$ and $\mathbf{z}_h$ represent the spatial and feature components, D_f is the embedding dimension of features. Note that we only keep the fixed-length part of the ϕ_θ output. So we obtain a fixed-length encoding of the molecules. The network structure and E(3)-equivariance discussion can be found in the Appendix A.

To regularize the latent space, we adopt a VAE formulation. While the initial output $[\mathbf{z}_x, \mathbf{z}_h]$ is deterministic, this can lead to irregular latent geometry and poor interpolatability. To address this, we predict a coordinate-wise Gaussian distribution for each latent dimension:

$$\boldsymbol{\mu}_x = \mathbf{z}_x, \quad [\boldsymbol{\sigma}_x^2, \boldsymbol{\mu}_h, \boldsymbol{\sigma}_h^2] = \text{Linear}(\mathbf{z}_h). \quad (4)$$

The resulting latent posterior is regularized via a KL divergence to a fixed spherical Gaussian prior $\mathcal{N}([0, 0], [\text{var}_x, \text{var}_h]\mathbf{I})$, giving rise to the regularization loss:

$$\mathcal{L}_{\text{reg}} = \text{KL}(\mathcal{N}([\boldsymbol{\mu}_x, \boldsymbol{\mu}_h], [\boldsymbol{\sigma}_x^2, \boldsymbol{\sigma}_h^2]) \parallel \mathcal{N}([0, 0], [\text{var}_x, \text{var}_h]\mathbf{I})), \quad (5)$$

where $\text{var}_x, \text{var}_h$ are two fixed scale parameters. This regularization encourages the latent space to be smooth and continuous, facilitating interpolation between molecules and improving the robustness and diversity of samples generated from the prior distribution. In practice, we project (using the linear layer) the feature embedding $\mathbf{z}_h \in \mathbb{R}^{N_Z \times D_f}$ to $\boldsymbol{\mu}_h \in \mathbb{R}^{N_Z \times D_Z}$. For notational simplicity, we continue to denote the sampled latent code from $\mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\sigma}_h^2\mathbf{I})$ as $\mathbf{z}_h$. The full expression of the regularization loss is provided in Appendix B.

3.4 MolFLAE Decoder

The encoder defines a Gaussian posterior over the latent space. We sample a latent code $(\mathbf{z}_x, \mathbf{z}_h)$ from this distribution and use it as the conditioning input to the BFN decoder. By comparing the coordinates and atom type of reconstructed molecule with the original input, we compute the reconstruction loss:

$$\mathcal{L}_{\text{recon}} = \mathcal{L}_x^n + \mathcal{L}_v^n. \quad (6)$$

In our molecular BFN setup, we must jointly model both continuous and discrete aspects of atomic data. This requires a unified representation that enables neural networks to propagate information across modalities while maintaining compatibility with Bayesian updates. It is enough to define a suitable sender distribution p_S allowing the additivity of precision [26].

We model continuous atomic coordinates using Gaussian distributions. Given ground-truth coordinates $\mathbf{x}_M$ and a noise level $\alpha = \rho^{-1}$, the sender generates a noisy observation by adding isotropic Gaussian noise:

$$p_S(\mathbf{y}^x \mid \mathbf{x}_M; \alpha) = \mathcal{N}(\mathbf{y}^x \mid \mathbf{x}_M, \alpha^{-1}\mathbf{I}). \quad (7)$$

For discrete atom types, we model each atom using a categorical distribution over K classes. This distribution is parameterized by a continuous matrix $\boldsymbol{\theta}^v \in \mathbb{R}^{N_M \times K}$, which is transformed into probabilities via a softmax function. Given the ground-truth atom type matrix $\mathbf{e}_{\mathbf{v}_M} = [\mathbf{v}_M^{(1)}, \dots, \mathbf{v}_M^{(N_M)}]^T \in \mathbb{R}^{N_M \times K}$, where each $\mathbf{v}_M^{(j)}$ is the column one-hot vector representing one of the K atom categories, the sender perturbs it with an artificial Gaussian noise scaled by α' , producing:

$$p_S(\mathbf{y}^v \mid \mathbf{v}_M; \alpha') = \mathcal{N}(\mathbf{y}^v \mid \alpha'(K\mathbf{e}_{\mathbf{v}_M} - \mathbf{1}), \alpha'K\mathbf{I}) \quad (8)$$

For the initial prior $\boldsymbol{\theta}_0$, we follow [26] and adopt standard Gaussian priors for continuous variables and uniform distributions for categorical ones.

Then we can derive the two loss $\mathcal{L}_x^n + \mathcal{L}_v^n$ respectively [37]. The total forward pass can be found in the Algorithm. 2. The inference process is parallel with the general BFN inference but taking two data modalities into consideration. See Appendix C.4

4 Experiments

We train and evaluate MolFLAE on three datasets: QM9 [38], GEOM-Drugs [39] and ZINC-9M (the in-stock subset of ZINC [40] with 9.3M molecules). QM9 contains 134k small molecules with up to 9 heavy atoms, and GEOM-Drugs is a larger-scale dataset featuring 430k drug-like molecules. On both QM9 and GEOM-Drugs experiment, hydrogens are treated explicitly. We use QM9 and GEOM-Drugs to evaluate MolFLAE in unconditional 3D molecule generation task, and demonstrate other applications on the more comprehensive large-scale dataset ZINC-9M, where hydrogens are treated implicitly.

Unconditional Molecule Generation To assess the capability of MolFLAE generate stable, diverse molecules, we first focus on 3D molecule generation task following the setting of prior works [15, 16, 17]. We conduct 10,000 random samplings in the latent space, then decode them into molecules using MolFLAE decoder, subsequently evaluating qualities of these molecules. We sample the atom number from the prior of the training set as previous works like [15]. Table 1 illustrates the benchmark results of unconditional generation with MolFLAE. We also provide results on drug-likeness metrics on GEOM-Drugs in Appendix E, confirming MolFLAE’s outstanding performance in generating structurally reasonable and drug-like molecules compared to previous methods.

Table 1: Performance comparison of different methods on the QM9 and GEOM-Drugs dataset.

# Metrics	QM9					GEOM-Drugs	
	Atom Sta (%)	Mol Sta (%)	Valid (%)	VxU (%)	Novelty (%)	Atom Sta (%)	Valid (%)
Data	99.0	95.2	97.7	97.7	-	86.5	99.9
ENF [41]	85.0	4.9	40.2	39.4	-	-	-
G-Schnet [42]	95.7	68.1	85.5	80.3	-	-	-
GDM-AUG [15]	97.6	71.6	90.4	89.5	74.6	77.7	91.8
EDM [15]	98.7	82.0	91.9	90.7	58.0	81.3	92.6
EDM-Bridge [43]	98.8	84.6	92.0	90.7	-	82.4	92.8
GEOLDM [18]	98.9	89.4	93.8	92.7	57.0	84.4	99.3
GEOBFN ₅₀ [17]	98.3	85.1	92.3	90.7	72.9	75.1	91.7
GEOBFN ₁₀₀ [17]	98.6	87.2	93.0	91.5	70.3	78.9	93.1
UniGEM [44]	99.0	89.8	95.0	93.2	-	85.1	98.4
MolFLAE ₅₀	99.3	90.4	95.9	92.1	77.1	86.9	99.2
MolFLAE ₁₀₀	99.4	92.0	96.8	88.9	74.5	86.7	99.7

Compared with several baseline models, MolFLAE achieves competitive performance across atom stability, molecular stability, and validity metrics on both QM9 and GEOM-Drugs dataset, while requiring fewer sampling steps. These results suggest that our latent space is well-structured, supporting efficient and reliable molecular generation.

Generating Analogs with Different Atom Numbers First, we probe the smoothness of MolFLAE decoder to atom numbers by forcing the generation with increased or decreased atom numbers based on the original latent code. We examine the similarities between generated molecules and original molecules with MCS-IoU (Maximum Common Substructure Intersection-over-Union). Generated molecules share similar orientations, shapes and 2D structures with the original input, validating the desired smoothness. Detailed results are presented in Table 2, and three examples are provided in Fig. 2 for better illustration.

Table 2: Evaluating 2D similarities between generated analogs and original molecules.

Atom Number	-2	-1	0	1	2
MCS-IoU similarity	69.79	76.69	84.08	76.05	69.95
Valid(%)	100.0	99.89	99.76	99.89	99.68
Atom Sta(%)	84.58	83.28	82.48	82.38	82.53

Exploring the disentanglement of the latent space via molecule reconstruction The latent space of MolFLAE consists of two parts, the E(3)-equivariant component $\mathbf{z}_x$ and the E(3)-invariant component $\mathbf{z}_h$. Ideally, spatial and semantic features of molecules disentangle spontaneously, with $\mathbf{z}_x$ encoding the shape and orientation and $\mathbf{z}_h$ encodes substructures of molecules.

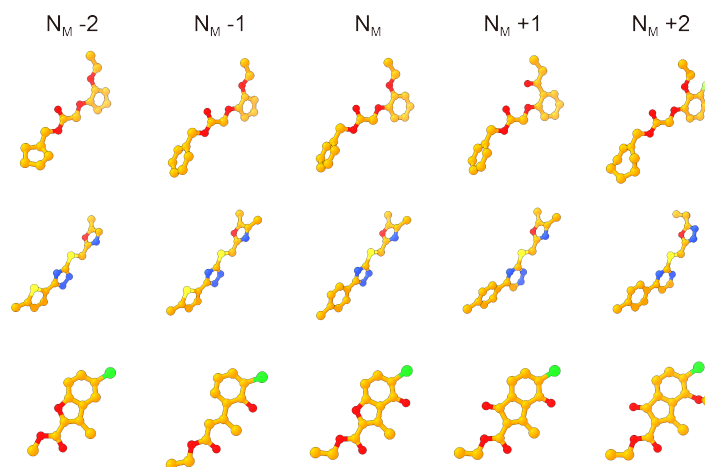


Figure 2: Examples for analog generation with variable atom numbers.

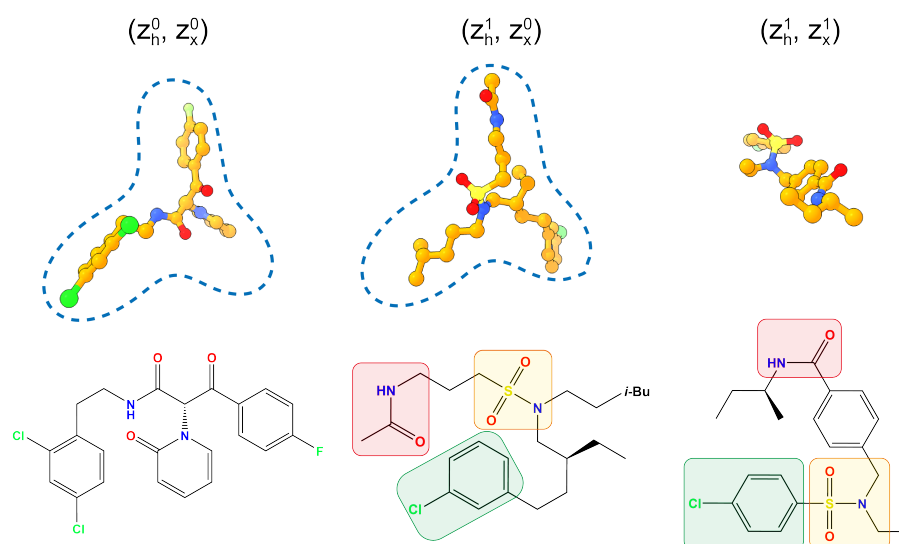


Figure 3: An example for molecule reconstruction with new shape and orientation.

In this section, we explore this disentanglement hypothesis by swapping $\mathbf{z}_x$ and $\mathbf{z}_h$ between molecules and observe decoded molecules. Formally, with two molecules M_0 and M_1 , and their latent $(\mathbf{z}_h^0, \mathbf{z}_x^0)$ and $(\mathbf{z}_h^1, \mathbf{z}_x^1)$, we decode molecules with $(\mathbf{z}_h^0, \mathbf{z}_x^1)$ (named preserving $\mathbf{z}_h$) and $(\mathbf{z}_h^1, \mathbf{z}_x^0)$ (named preserving $\mathbf{z}_x$), respectively.

In experiments, we have observed a partial disentanglement of the MolFLAE latent space. As shown in Fig. 3, following the disentanglement hypothesis, substructures information of M_1 should be able to be extracted by isolating $\mathbf{z}_h^1$. As $\mathbf{z}_h^1$ is not sufficient to reconstruct M_1 , we view it as a deconstructed molecule. Then, we reconstruct these substructures into the shape and orientation of M_0 , by decoding the hybrid latent code $(\mathbf{z}_h^1, \mathbf{z}_x^0)$. The resulted molecule shares a similar shape and orientation with M_0 , indicated by the dash lines in Fig 3. Moreover, it also shares similar substructures as M_1 like **amide**, **chlorobenzyl**, **sulfamide**, indicated by rectangles of corresponding colors.

Quantitatively, we compute the MACCS fingerprint [45] similarity (considering substructure overlapping) and *in situ* shape similarity (considering shape, orientation and relative position) of 1000 hybrid molecules $(\mathbf{z}_h^1, \mathbf{z}_x^0)$ and $(\mathbf{z}_h^0, \mathbf{z}_x^1)$ with the original molecule $(\mathbf{z}_h^0, \mathbf{z}_x^0)$ (Table 3). Under the setting of preserving $\mathbf{z}_x$, the shape similarity is significantly higher than the preserving $\mathbf{z}_h$ (0.394 vs 0.174); indicating $\mathbf{z}_x$ indeed encodes shape and orientation information. Similarly, MACCS similarity is

higher under the preserving $\mathbf{z}_h$ setting than the preserving $\mathbf{z}_x$ setting (0.580 vs 0.421). These results support that the latent space of MolFLAE is partially disentangled, with $\mathbf{z}_h$ representing substructure composition and $\mathbf{z}_x$ representing shape and orientation.

Table 3: Measuring molecule reconstruction similarities under different settings.

Metrics	Preserving $\mathbf{z}_x$				Preserving $\mathbf{z}_h$			
	MACCS Sim $\downarrow$	Shape Sim $\uparrow$	Valid(%)($\uparrow$)	Atom Sta(%)($\uparrow$)	MACCS Sim $\uparrow$	Shape Sim $\downarrow$	Valid(%)($\uparrow$)	Atom Sta(%)($\uparrow$)
MolFLAE	0.421	0.394	100.0	85.20	0.580	0.174	100.0	84.62

Latent Interpolation The fixed-dimensional latent space allows flexible manipulation of molecular representations via vector convex combinations. In MolFLAE, the regularization loss further encourages smooth transitions between latent codes, facilitating continuous transformations between molecules. Fig. 4 presents the interpolation of three pairs of molecules, indicating smooth transformations of shape and orientation.

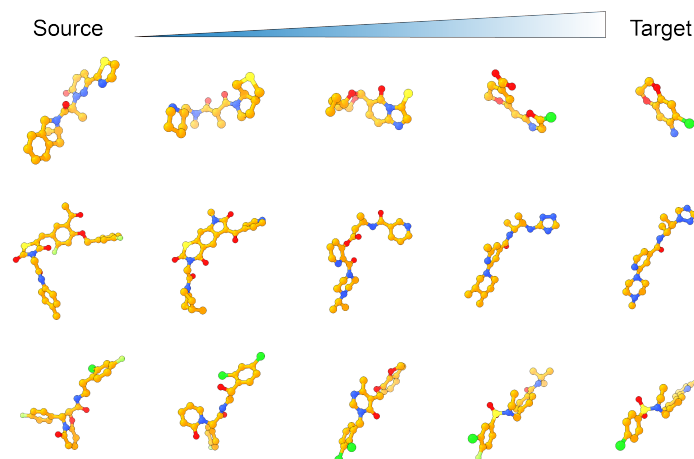


Figure 4: Examples for the latent interpolation between molecules.

To further quantify the quality of interpolation, We evaluate the trend of properties of the intermediate molecules during the transformation, considering both structural and physical properties (detailed descriptions are provided in Appendix D). We compute the Pearson correlation coefficient r between each property value and its corresponding interpolation index (adjusted with the sign of property difference between the source and target molecule). We also report the associated p -values from the Pearson significance test to assess the statistical evidence for linear trends. A property is considered to exhibit significant linear variation along the interpolation trajectory if the null hypothesis of zero correlation is rejected at the 5% significance level, i.e., if $-\log p > -\log(0.05) \approx 1.3$. Detailed results are documented in Table 5 and Table 4. We also report the step-wise molecular validity and atom stability during interpolation in Table 6, which indicates that most intermediate molecules are valid and stable.

Table 4: Monitoring the trend of structural properties along with molecule interpolations.

Interpo Num	Similarity Preference		sp3frac		BertzCT		QED	
	Pearson's r	$-\log p$	Pearson's r	$-\log p$	Pearson's r	$-\log p$	Pearson's r	$-\log p$
8	0.9261	3.3346	0.4314	0.8518	0.6537	1.7888	0.5460	1.2128
10	0.9191	4.1340	0.3982	0.9639	0.6344	2.1138	0.5194	1.4228
12	0.9122	4.8476	0.3809	1.0889	0.6281	2.4728	0.5059	1.6216

Table 5: Monitoring the trend of physical properties along with molecule interpolations.

Interpo Num	Labute ASA		TPSA		LogP		MR	
	Pearson's r	-log p	Pearson's r	-log p	Pearson's r	-log p	Pearson's r	-log p
8	0.9067	4.6366	0.5711	1.3783	0.5400	1.2363	0.8467	3.5188
10	0.9041	5.7687	0.5620	1.6533	0.5216	1.4533	0.8388	4.3350
12	0.8939	6.8567	0.5425	1.8572	0.4925	1.6259	0.8255	5.1216

Table 6: Step-wise Validity and Atom Stability during interpolation.

Interpo Num	Metrics	Step											
		1	2	3	4	5	6	7	8	9	10	11	12
8	Valid(%)	99.90	99.90	99.90	100.0	99.90	100.0	100.0	100.0				
	Atom Sta(%)	82.23	82.95	84.17	84.57	84.21	83.74	83.10	82.63				
10	Valid(%)	100.0	99.90	99.90	99.80	99.80	100.0	100.0	100.0	99.80	99.90		
	Atom Sta(%)	82.23	82.75	83.53	84.07	84.36	84.35	84.16	84.17	82.78	82.67		
12	Valid(%)	99.90	100.0	99.90	100.0	100.0	100.0	100.0	100.0	99.90	99.90	100.0	99.80
	Atom Sta(%)	82.39	82.50	83.76	83.67	83.92	84.65	84.20	84.50	84.27	83.00	82.84	82.57

Applying MolFLAE to optimize molecules targeting the human glucocorticoid receptor (hGR)

To assess the real-world utility of our method, we applied our method to optimize drug candidates for the hGR, which requires balancing hydrophobic-interaction-centric binding with aqueous solubility. The hGR is a key target for anti-inflammatory, and our optimization starts from two known actives. AZD2906 is a potent hGR modulator but is poorly soluble [46], while BI-653048 is more soluble but less potent [47]. To showcase the performance of our model, We computationally evaluate the potency and hydrophilicity with Glide docking and QikProp CLogPo/w from the Schrodinger Suite, where lower docking scores indicate better potency, and lower CLogPo/w values indicate better hydrophilicity. These computational metrics align with real-world properties of AZD2906 and BI-653048. AZD2906 has a docking score of -13.16 and a high CLogPo/w of 5.61, whereas BI-653048 shows a better CLogPo/w of 3.90 but a weaker docking score of -10.62 (Fig. 5B and C). These results facilitate the computational evaluation of MolFLAE generated molecules.

To explore trade-offs, we blended these two molecules in latent space using 90% AZD2906 and 10% BI-653048, generating 100 candidates. The top 10 molecules outperformed BI-653048 in docking score, and 8 also improved hydrophilicity (CLogPo/w) compared to AZD2906. These candidates preserved AZD2906's binding shape while introducing polar groups for better solubility. Sample 34, for instance, it retained key pharmacophores of AZD2906 and BI-653048 for interacting with the receptor (indicated by colored rectangles for each pharmacophore), achieving a balanced property with its docking score of -11.15 and CLogPo/w of 3.75 (Fig. 5F). Moreover, its docking pose closely matched both its generated conformation (RMSD = 1.35 Å) and AZD2906's crystal structure (Fig. 5D and E), representing the advantage of explicitly modeling of 3D coordinates by MolFLAE. These results highlight our method's potential for meaningful molecular optimization and drug design.

5 Conclusion and Future Works

In this work, we present MolFLAE, a flexible VAE framework for manipulating 3D molecules within a fixed-dimensional, E(3)-equivariant latent space. Our method demonstrates strong performance across multiple tasks, including unconditional generation, analog design, substructure reconstruction, and latent interpolation. We further validate the real-world utility of MolFLAE through a case study on generating drug-like molecules targeting the hGR, balancing potency and solubility.

Beyond the reported experiments, MolFLAE naturally extends to a wider range of tasks. For example, molecule inpainting can be achieved by encoding discontinuous fragments and decoding to larger atom sets. Structural superposition can be achieved efficiently via the weighted Kabsch algorithm on latent nodes, avoiding the high complexity of atom-wise bipartite matching. These applications are exemplified in Fig 6, and a deeper exploration is left for future work due to space constraints.

While MolFLAE demonstrates strong performance across multiple tasks, there remains room for improvement in the disentanglement and interpretability of its latent space. We hypothesize that better disentanglement can be achieved by enforcing the invariance of semantic latent $\mathbf{z}_h$ to molecular

conformational changes or other non-rigid perturbations. Future works may explore incorporating self-contrastive objectives to better capture chemical semantics with latent representations.

In summary, our results highlight the versatility of MolFLAE and its promise as a general-purpose framework for 3D molecule generation and editing. This work opens new directions for exploring the broader applications of fixed-dimensional, E(3)-equivariant latent spaces in molecular modeling.

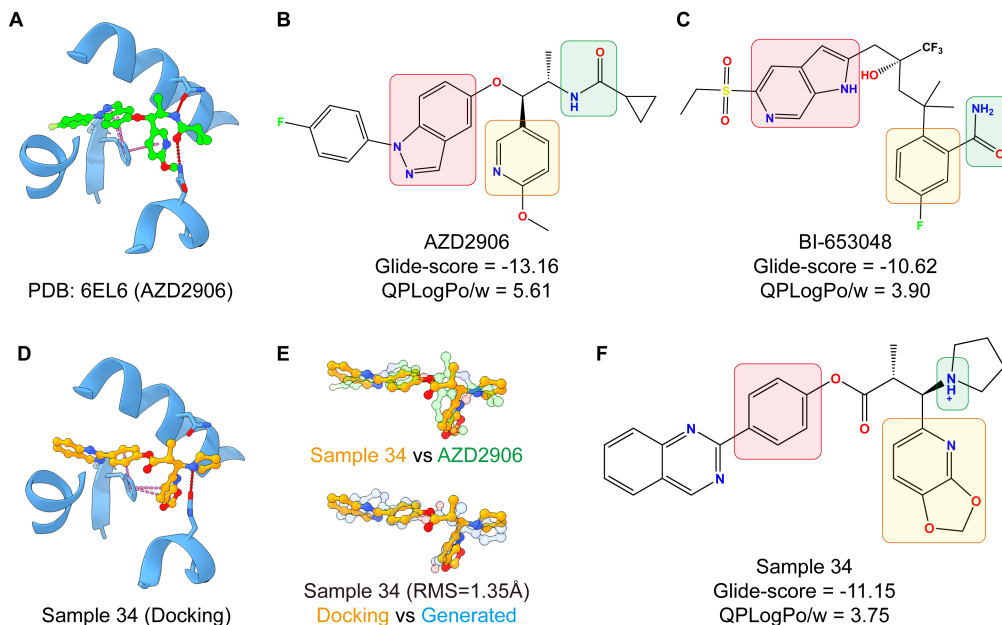


Figure 5: Applying MolFLAE to optimizing AZD2096 targeting the hGR. **A**, the crystal structure of AZD2096 in complex with hGR. **B**, **C**, and **F**, 2D structures of AZD2096, BI-653048, and sample 34, with their docking score and CLogPo/w. **D**, the docking pose of sample 34. **E**, comparing the docking pose of sample 34 with AZD2096 and its generated pose before docking.

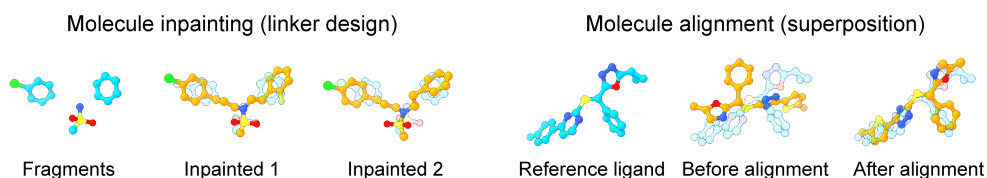


Figure 6: Exemplifying the application of MolFLAE to molecule inpainting and superposition.

Acknowledgements

This work is supported by Beijing Academy of Artificial Intelligence and Beijing Frontier Research Center for Biological Structure Fundings.

References

- [1] Debora Chiodi and Yoshihiro Ishihara. “magic chloro”: Profound effects of the chlorine atom in drug discovery. *Journal of Medicinal Chemistry*, 66(8):5305–5331, 2023. PMID: 37014977.
- [2] Heike Schönherr and Tim Cernak. Profound methyl effects in drug discovery and a call for new c-h methylation reactions. *Angewandte Chemie International Edition*, 52(47):12256–12267, 2013.
- [3] J. Henry Blackwell, Iacovos N. Michaelides, and Floriane Gibault. A perspective on the strategic application of deconstruction–reconstruction in drug discovery. *Journal of Medicinal Chemistry*, 0(0):null, 0. PMID: 40324045.
- [4] Tingting Chen, Jiafu Leng, Jun Tan, Yongjun Zhao, Shanshan Xie, Shifang Zhao, Xiangyu Yan, Liqiao Zhu, Jun Luo, Lingyi Kong, and Yong Yin. Discovery of novel potent covalent glutathione peroxidase 4 inhibitors as highly selective ferroptosis inducers for the treatment of triple-negative breast cancer. *Journal of Medicinal Chemistry*, 66(14):10036–10059, 2023. PMID: 37452764.
- [5] Kunyu Shi, Jifa Zhang, Enda Zhou, Jiaying Wang, and Yuxi Wang. Small-molecule receptor-interacting protein 1 (rip1) inhibitors as therapeutic agents for multifaceted diseases: Current medicinal chemistry insights and emerging opportunities. *Journal of Medicinal Chemistry*, 65(22):14971–14999, 2022. PMID: 36346971.
- [6] Ilia Igashov, Hannes Stärk, Clément Vignac, Victor Garcia Satorras, Pascal Frossard, Max Welling, Michael Bronstein, and Bruno Correia. Equivariant 3d-conditional diffusion models for molecular linker design, 2022.
- [7] Jiaqi Guan, Xingang Peng, PeiQi Jiang, Yunan Luo, Jian Peng, and Jianzhu Ma. Linkernet: Fragment poses and linker co-design with 3d equivariant diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [8] Shicheng Chen, Odin Zhang, Chenran Jiang, Huifeng Zhao, Xujun Zhang, Mengting Chen, Yun Liu, Qun Su, Zhenxing Wu, Xinyue Wang, Wanglin Qu, Yuanyi Ye, Xin Chai, Ning Wang, Tianyue Wang, Yuan An, Guanlin Wu, Qianqian Yang, Jiean Chen, Wei Xie, Haitao Lin, Dan Li, Chang-Yu Hsieh, Yong Huang, Yu Kang, Tingjun Hou, and Peichen Pan. Deep lead optimization enveloped in protein pocket and its application in designing potent and selective ligands targeting LTK protein. *Nat. Mac. Intell.*, 7(3):448–458, 2025.
- [9] Keyue Qiu, Yuxuan Song, Jie Yu, Hongbo Ma, Ziyao Cao, Zhilong Zhang, Yushuai Wu, Mingyue Zheng, Hao Zhou, and Wei-Ying Ma. Empower structure-based molecule optimization with gradient guidance, 2025.
- [10] Alex Morehead and Jianlin Cheng. Geometry-complete diffusion for 3d molecule generation and optimization. *ArXiv*, 2023.
- [11] Keir Adams, Kento Abeywardane, Jenna Fromer, and Connor W. Coley. ShEPhERD: Diffusing shape, electrostatics, and pharmacophores for bioisosteric drug design. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [12] Peiye Zhuang, Oluwasanmi O Koyejo, and Alex Schwing. Enjoy your editing: Controllable {gan}s for image editing via latent space navigation. In *International Conference on Learning Representations*, 20221.
- [13] Qiucheng Wu, Yujian Liu, Handong Zhao, Ajinkya Kale, Trung Bui, Tong Yu, Zhe Lin, Yang Zhang, and Shiyu Chang. Uncovering the disentanglement capability in text-to-image diffusion models. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1900–1910, 2023.

- [14] Yong-Hyun Park, Mingi Kwon, Jaewoong Choi, Junghyo Jo, and Youngjung Uh. Understanding the latent space of diffusion models through the lens of riemannian geometry. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [15] Emiel Hoogetboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 8867–8887. PMLR, 2022.
- [16] Yuxuan Song, Jingjing Gong, Minkai Xu, Ziyao Cao, Yanyan Lan, Stefano Ermon, Hao Zhou, and Wei-Ying Ma. Equivariant flow matching with hybrid probability transport for 3d molecule generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [17] Yuxuan Song, Jingjing Gong, Hao Zhou, Mingyue Zheng, Jingjing Liu, and Wei-Ying Ma. Unified generative modeling of 3d molecules with bayesian flow networks. In *The Twelfth International Conference on Learning Representations*, 2024.
- [18] Minkai Xu, Alexander S. Powers, Ron O. Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3d molecule generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 38592–38610. PMLR, 2023.
- [19] Xiangzhe Kong, Yinjun Jia, Wenbing Huang, and Yang Liu. Full-atom peptide design with geometric latent diffusion. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.
- [20] Youzhi Luo and Shuiwang Ji. An autoregressive flow model for 3d molecular geometry generation from scratch. In *International Conference on Learning Representations*, 2022.
- [21] Niklas W. A. Gebauer, Michael Gastegger, and Kristof Schütt. Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7564–7576, 2019.
- [22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. 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.
- [23] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.
- [24] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.
- [25] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.
- [26] Alex Graves, Rupesh Kumar Srivastava, Timothy Atkinson, and Faustino Gomez. Bayesian flow networks, 2023.
- [27] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.
- [28] Xiangzhe Kong, Zishen Zhang, Ziting Zhang, Rui Jiao, Jianzhu Ma, Wenbing Huang, Kai Liu, and Yang Liu. Unimomo: Unified generative modeling of 3d molecules for de novo binder design. In *Forty-second International Conference on Machine Learning*, 2025.

- [29] Xiangxin Zhou, Xiwei Cheng, Yuwei Yang, Yu Bao, Liang Wang, and Quanquan Gu. Decompt: Controllable and decomposed diffusion models for structure-based molecular optimization. In *The Twelfth International Conference on Learning Representations*, 2024.
- [30] Tomohide Masuda, Matthew Ragoza, and David Ryan Koes. Generating 3d molecular structures conditional on a receptor binding site with deep generative models. *CoRR*, abs/2010.14442, 2020.
- [31] Pedro O. Pinheiro, Joshua Rackers, Joseph Kleinhenz, Michael Maser, Omar Mahmood, Andrew Martin Watkins, Stephen Ra, Vishnu Sresht, and Saeed Saremi. 3d molecule generation by denoising voxel grids. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [32] Matthieu Kirchmeyer, Pedro O. Pinheiro, and Saeed Saremi. Score-based 3d molecule generation with neural fields. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.
- [33] Robin Winter, Frank No'e, and Djork-Arné Clevert. Auto-encoding molecular conformations. *ArXiv*, abs/2101.01618, 2021.
- [34] Yanchen Luo, Zhiyuan Liu, Yi Zhao, Sihang Li, Kenji Kawaguchi, Tat-Seng Chua, and Xiang Wang. Towards unified latent space for 3d molecular latent diffusion modeling, 2025.
- [35] Tianxiao Li, Martin Renqiang Min, Hongyu Guo, and Mark Gerstein. 3d autoencoding diffusion model for molecule interpolation and manipulation, 2024.
- [36] Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*, 2024.
- [37] Yanru Qu, Keyue Qiu, Yuxuan Song, Jingjing Gong, Jiawei Han, Mingyue Zheng, Hao Zhou, and Wei-Ying Ma. MolCRAFT: Structure-based drug design in continuous parameter space. In *Forty-first International Conference on Machine Learning*, 2024.
- [38] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 08 2014.
- [39] Simon Axelrod and Rafael Gómez-Bombarelli. GEOM: energy-annotated molecular conformations for property prediction and molecular generation. *CoRR*, abs/2006.05531, 2020.
- [40] John J. Irwin, Khanh G. Tang, Jennifer Young, Chinzorig Dandarchuluun, Benjamin R. Wong, Munkhzul Khurelbaatar, Yurii S. Moroz, John Mayfield, and Roger A. Sayle. Zinc20—a free ultralarge-scale chemical database for ligand discovery. *Journal of Chemical Information and Modeling*, 60(12):6065–6073, 2020. PMID: 33118813.
- [41] Victor Garcia Satorras, Emiel Hooeboom, Fabian Fuchs, Ingmar Posner, and Max Welling. E(n) equivariant normalizing flows. In *NeurIPS*, pages 4181–4192, 2021.
- [42] Niklas W. A. Gebauer, Michael Gastegger, and Kristof T. Schütt. Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. *CoRR*, abs/1906.00957, 2019.
- [43] Lemeng Wu, Chengyue Gong, Xingchao Liu, Mao Ye, and Qiang Liu. Diffusion-based molecule generation with informative prior bridges. In *NeurIPS*, 2022.
- [44] Shikun Feng, Yuyan Ni, Lu yan, Zhi-Ming Ma, Wei-Ying Ma, and Yanyan Lan. UniGEM: A unified approach to generation and property prediction for molecules. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [45] Joseph L. Durant, Burton A. Leland, Douglas R. Henry, and James G. Nourse. Reoptimization of mdl keys for use in drug discovery. *Journal of Chemical Information and Computer Sciences*, 42(6):1273–1280, 2002. PMID: 12444722.

- [46] Martin Hemmerling, Karl Edman, Matti Lepistö, Anders Eriksson, Svetlana Ivanova, Jan Dahmén, Hartmut Rehwinkel, Markus Berger, Ramon Hendrickx, Matthew Dearman, Tina Jellesmark Jensen, Lisa Wissler, and Thomas Hansson. Discovery of indazole ethers as novel, potent, non-steroidal glucocorticoid receptor modulators. *Bioorganic & Medicinal Chemistry Letters*, 26(23):5741–5748, 2016.
- [47] Christian Harcken, Doris Riether, Pingrong Liu, Hossein Razavi, Usha Patel, Thomas Lee, Todd Bosanac, Yancey Ward, Mark Ralph, Zhidong Chen, Donald Souza, Richard M. Nelson, Alison Kukulka, Tazmeen N. Fadra-Khan, Ljiljana Zuvela-Jelaska, Mita Patel, David S. Thomson, and Gerald H. Nabozny. Optimization of drug-like properties of nonsteroidal glucocorticoid mimetics and identification of a clinical candidate. *ACS Medicinal Chemistry Letters*, 5(12):1318–1323, 2014.
- [48] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. In *International Conference on Learning Representations*, 2022.
- [49] Kaiwen Xue, Yuhao Zhou, Shen Nie, Xu Min, Xiaolu Zhang, JUN ZHOU, and Chongxuan Li. Unifying bayesian flow networks and diffusion models through stochastic differential equations. In *Forty-first International Conference on Machine Learning*, 2024.
- [50] Steven H. Bertz. The first general index of molecular complexity. *Journal of the American Chemical Society*, 103(12):3599–3601, 1981.
- [51] G. Richard Bickerton, Gaia V. Paolini, Jeremy Besnard, Sorel Muresan, and Andrew L. Hopkins. Quantifying the chemical beauty of drugs. *Nature Chemistry*, 4(2):90–98, February 2012.
- [52] Paul Labute. A widely applicable set of descriptors. *Journal of Molecular Graphics and Modelling*, 18(4):464–477, 2000.
- [53] Peter Ertl, Bernhard Rohde, and Paul Selzer. Fast calculation of molecular polar surface area as a sum of fragment-based contributions and its application to the prediction of drug transport properties. *Journal of Medicinal Chemistry*, 43(20):3714–3717, 2000. PMID: 11020286.
- [54] Scott A. Wildman and Gordon M. Crippen. Prediction of physicochemical parameters by atomic contributions. *Journal of Chemical Information and Computer Sciences*, 39(5):868–873, 1999.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [NA].
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes]" is generally preferable to "[No]", it is perfectly acceptable to answer "[No]" provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No]" or "[NA]" is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading “NeurIPS Paper Checklist”,**
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers.**

1. Claims

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

Answer: [Yes]

Justification: We do claim our contributions and scope in the abstract and the last part of introduction.

Guidelines:

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

2. Limitations

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

Answer: [Yes]

Justification: We discuss the limitations of our work in the Discussion and Conclusion part.

Guidelines:

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

3. Theory assumptions and proofs

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

Answer: [NA]

Justification: Our work includes no theoretical analysis.

Guidelines:

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

4. Experimental result reproducibility

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

Answer: [Yes]

Justification: We give the experiment details in Appendix. A B C and hyperparameters in Appendix. G.

Guidelines:

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

5. Open access to data and code

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

Answer: [No]

Justification: We will release our codes to reproduce the main experiment soon.

Guidelines:

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

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

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

Answer: [\[Yes\]](#)

Justification: We give the experiment details in Appendix. A B C and hyperparameters in Appendix. G.

Guidelines:

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

7. Experiment statistical significance

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

Answer: [\[No\]](#)

Justification: Our experiments follow standard deep learning practice and do not report statistical error bars or significance tests.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

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

Answer: [\[Yes\]](#)

Justification: We provide our computer resources in Appendix. G.

Guidelines:

- The answer NA means that the paper does not include experiments.

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

9. Code of ethics

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

Answer: [Yes]

Justification: The paper does conform, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

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

10. Broader impacts

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

Answer: [Yes]

Justification: We have discussed the potential impact of our work in Appendix H.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

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

Answer: [Yes]

Justification: We have discussed the Safeguards of our work in Appendix H.

Guidelines:

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

12. Licenses for existing assets

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

Answer: [\[Yes\]](#)

Justification: All assets are properly cited in the main text. Their licenses and terms of use are respected in accordance with the original distribution terms.

Guidelines:

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

13. New assets

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

Answer: [\[Yes\]](#)

Justification: We will provide a new validation set splitted from ZINC-9M, along with our code of latent molecule manipulation experiments.

Guidelines:

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

14. Crowdsourcing and research with human subjects

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

Answer: [NA]

Justification: We do not do such research.

Guidelines:

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

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

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

Answer: [NA]

Justification: We do not do such experiments.

Guidelines:

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

16. Declaration of LLM usage

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

Answer: [NA]

Justification: We do not use LLM on such important parts.

Guidelines:

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

A Neural Network Details

We adopt the network structure of MolCRAFT [37] as our E(3)-Equivariant graph neural network (GNN) backbone. Originally designed to model interactions between ligand and protein pocket atoms, the network distinguishes between *update nodes* $\mathbf{u}$ (whose coordinates are updated) and *condition nodes* $\mathbf{c}$ (which provide contextual information and the coordinates are not updated).

A.1 Neural Network Architecture

To align with our autoencoder framework, we adapt this formulation by assigning roles to nodes based on the encoding or decoding stage. In the encoder, the update nodes correspond to the virtual nodes, while the condition nodes are the atoms of the input ground truth molecule. Conversely, in the decoder, the update nodes are the molecular atoms being generated, and the condition nodes are the latent codes produced by the encoder.

To be convenient, we concatenate the spatial part and feature part of the update nodes $[\mathbf{x}_{\text{update}}, \mathbf{v}_{\text{update}}]$ and condition nodes $[\mathbf{x}_{\text{condition}}, \mathbf{v}_{\text{condition}}]$ with writing $\mathbf{x}^\ell = [\mathbf{x}_{\text{update}}^\ell, \mathbf{x}_{\text{condition}}^\ell]$ and $\mathbf{h}^\ell = [\mathbf{v}_{\text{update}}^\ell, \mathbf{v}_{\text{condition}}^\ell]$, where the superscript represents the ℓ -th layer of ϕ , $0 \leq \ell \leq L$. The Initial hidden embedding $\mathbf{h}^0$ is obtained by an MLP embedding layer that encodes the atom feature $[\mathbf{h}]$. No embedding layer for the atom spatial coordinates. The construction of ϕ_θ is alternately updating the atom feature embeddings $\mathbf{h}$ and coordinates $\mathbf{x}$ as

$$\mathbf{h}_i^{\ell+1} = \mathbf{h}_i^\ell + \sum_{\substack{j \in \mathcal{V}(i) \\ j \neq i}} f_h(d_{ij}^\ell, \mathbf{h}_i^\ell, \mathbf{h}_j^\ell, \mathbf{e}_{ij}; \theta_h), \quad (9)$$

$$\mathbf{x}_i^{\ell+1} = \mathbf{x}_i^\ell + \sum_{\substack{j \in \mathcal{V}(i) \\ j \neq i}} (\mathbf{x}_i^\ell - \mathbf{x}_j^\ell) f_x(d_{ij}^\ell, \mathbf{h}_i^{\ell+1}, \mathbf{h}_j^{\ell+1}, \mathbf{e}_{ij}; \theta_x) \mathbf{1}_{\text{update}}. \quad (10)$$

Here, $\mathcal{V}(i)$ is the neighbors of i , who could have information communication to i . We choose the k nearest nodes from i . $d_{ij}^\ell = \|\mathbf{x}_i^\ell - \mathbf{x}_j^\ell\|_2$ denotes the Euclidean distance between atoms i and j at layer ℓ , and $\mathbf{e}_{ij}$ encodes whether the pair (i, j) belongs to the update nodes, condition nodes, or the connection between them. The indicator $\mathbf{1}_{\text{update}}$ ensures that coordinate updates are only applied to update nodes, keeping the positions of condition nodes fixed.

A.2 E(3)-Equivariance Discussion

In molecular modeling, it is essential that the learned distribution over update nodes be invariant to translations, reflections, and rotations of the condition nodes. This E(3)-invariance reflects a fundamental inductive bias in molecular systems [15, 48]. Since SE(3) is a subgroup of E(3), any E(3)-equivariant model is also SE(3)-equivariant. While some prior works (e.g., [37, 7]) adopt the term SE(3), others (e.g., [15]) use E(3), which more accurately describes the symmetry of their networks. We follow the latter to avoid ambiguity.

The full Euclidean group in $\mathbb{R}^3$, denoted E(3), consists of rigid-body transformations of the form $T(x) = Rx + t$, where $R \in \mathbb{R}^{3 \times 3}$ is a orthogonal matrix and $t \in \mathbb{R}^3$ is a translation vector.

If we pre-align the condition nodes by centering them at their center of mass (i.e., eliminating the translational degree of freedom), then the resulting likelihood becomes E(3)-invariant under the following condition:

Proposition A.1 (Proposition 4.1 in [37]). *Let $T \in \text{E}(3)$ denote a rigid transformation. If the condition nodes are centered at zero and the parameterization $\Phi(\theta, \mathbf{c}, t)$ is E(3)-equivariant, then the likelihood is invariant under T :*

$$p_\phi(T(\mathbf{u}) \mid T(\mathbf{c})) = p_\phi(\mathbf{u} \mid \mathbf{c}).$$

This property ensures that the decoder's predictions respect the underlying geometric symmetries of molecular structures, which is crucial for both sample quality and spatial information learning of latent codes.

A.3 Encoder Network

In the encoder network, the update nodes are the virtual nodes while the condition nodes are the input ground truth molecule.

Before passing into the network, we apply a linear layer to embed the one-hot atom features $\mathbf{v}_M \in \mathbb{R}^{N_M \times D_M}$ into a continuous feature space $\mathbb{R}^{N_M \times D_f}$, where D_f denotes the embedding dimension. The virtual node features are initialized as learnable parameters in the same embedded space $\mathbb{R}^{N_Z \times D_f}$ and are only defined at the embedding level. Their initial spatial positions are set to zero.

After L layers of message passing, the output of the Network ϕ_θ is given by

$$\phi_\theta([\mathbf{x}_M, \mathbf{v}_M], [\mathbf{x}_Z, \mathbf{v}_Z]) = ([\tilde{\mathbf{x}}_M, \tilde{\mathbf{v}}_M], [\tilde{\mathbf{x}}_Z, \tilde{\mathbf{v}}_Z]) = (_, [\mathbf{z}_x, \mathbf{z}_h]), \quad (11)$$

where only the virtual node outputs $[\mathbf{z}_x, \mathbf{z}_h]$ are retained as the final latent code. Since the coordinate updates are designed to be E(3)-equivariant at each layer, the entire encoder ϕ_θ is equivariant by construction.

Importantly, we do not apply a softmax projection to convert feature embeddings into one-hot vectors. Instead, we preserve the continuous representations to retain richer information for downstream generation.

Then we apply a VAE layer to furtherly encode the latent code to a Gaussian distribution. See Appendix B.

A.4 Decoder Network

The decoder follows a mirrored architecture, where the update nodes correspond to the generated molecule, and the conditioning nodes are the latent virtual codes. Similar to the encoder, we discard the outputs corresponding to the latent nodes and retain only the decoded molecule representations for final output. The atom number N_M is to be known beforehand. We can sample it from the training set prior when doing unconditional generation [15] or edit it in generating analogs with different atom numbers.

B VAE Details

We adopt the regularization loss to regularize the latent space. Given the deterministic initial output $[\mathbf{z}_x, \mathbf{z}_h]$, we predict a coordinate-wise Gaussian distribution for each latent dimension:

$$\boldsymbol{\mu}_x = \mathbf{z}_x, \quad [\boldsymbol{\sigma}_x^2, \boldsymbol{\mu}_h, \boldsymbol{\sigma}_h^2] = \text{Linear}(\mathbf{z}_h). \quad (12)$$

where we assume isotropic variance for 3D coordinates (i.e., each atom shares a scalar variance across x, y, z), while the feature dimensions $\mathbf{z}_h$ are assigned independent variances per entry. This design ensures that the latent distribution preserves equivariance in the spatial domain while maintaining expressiveness in the feature domain.

In practice, we project (using the linear layer) the feature embedding $\mathbf{z}_h \in \mathbb{R}^{N_Z \times D_f}$ to $\boldsymbol{\mu}_h \in \mathbb{R}^{N_Z \times D_Z}$. For notational simplicity, we continue to denote the sampled latent code from $\mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\sigma}_h^2 \mathbf{I})$ as $\mathbf{z}_h$.

The resulting latent posterior is regularized via a KL divergence to a fixed spherical Gaussian prior $\mathcal{N}([0, 0], [\text{var}_x, \text{var}_h] \mathbf{I})$, giving rise to the regularization loss:

$$\mathcal{L}_{\text{reg}} = \text{KL}(\mathcal{N}([\boldsymbol{\mu}_x, \boldsymbol{\mu}_h], [\boldsymbol{\sigma}_x^2, \boldsymbol{\sigma}_h^2]) \parallel \mathcal{N}([0, 0], [\text{var}_x, \text{var}_h] \mathbf{I})), \quad (13)$$

where $\text{var}_x, \text{var}_h$ are two fixed scale parameters. Since the KL-divergence between two Gaussian distribution $P_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, $i = 1, 2$ is

$$D_{KL}(P_1 \parallel P_2) = \log \frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}. \quad (14)$$

Hence the regularization loss is $\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{KL}}^{(h)} + \mathcal{L}_{\text{KL}}^{(x)}$, where

$$\mathcal{L}_{\text{KL}}^{(h)} = \sum_{i=1}^{N_Z \times D_Z} \frac{1}{2} \left(\frac{\mu_i^2 + \sigma_i^2}{\text{var}_h} - \log \sigma_i^2 - 1 \right), \quad (15)$$

$$\mathcal{L}_{\text{KL}}^{(x)} = \sum_{i=1}^{N_Z \times 3} \frac{1}{2} \left(\frac{\mu_i^2 + \sigma_i^2}{\text{var}_x} - \log \sigma_i^2 - 1 \right). \quad (16)$$

This regularization encourages the latent space to be smooth and continuous, facilitating interpolation between molecules and improving the robustness and diversity of samples generated from the prior distribution.

C BFN details

C.1 A Brief Introduction to Bayes Flow Network

BFN can be interpreted as a communication protocol between a sender and a receiver. The sender observes the ground-truth molecule $\mathbf{m}$ and deliberately adds noise to obtain a corrupted version $\mathbf{y}$, which is transmitted to the receiver. Given the known precision level (e.g., α from a predefined schedule $\beta(t)$), the receiver performs Bayesian inference and leverages a neural network to incorporate contextual information, producing an estimate of $\mathbf{m}$. The communication cost at each timestep is defined as the KL divergence between the sender's noising distribution and the corrupted output of the receiver's updated belief. By minimizing this divergence across timesteps, the receiver learns to approximate the posterior and generate realistic samples from prior noise using the same learned inference process.

C.1.1 Training of BFN

Concretely, at each communication step t_i , the sender perturbs $\mathbf{m}$ using the *Sender distribution* (adding noise distribution) $p_S(\mathbf{y}_i | \mathbf{m}; \alpha_i)$ to produce a noisy latent $\mathbf{y}_i$, analogous to the forward process in diffusion models. The receiver then reconstructs (using Bayes updates and Neural Networks) $\hat{\mathbf{m}}$ via the *Output distribution*:

$$p_O(\hat{\mathbf{m}} | \mathbf{y}_i; \theta) = \Phi(\theta_{i-1}, \mathbf{z}, t_i), \quad (17)$$

where Φ is a neural network which is expected to reconstruct the sample $\hat{\mathbf{m}}$ given the Bayes-updated parameters θ_{i-1} , conditioning code $\mathbf{z}$ and time t_i . Then re-applies the same noising process to obtain the *receiver distribution* $p_R(\mathbf{y}_i | \theta_{i-1}, \mathbf{z}; t_i) = \mathbb{E}_{\hat{\mathbf{m}} \sim p_O} p_S(\mathbf{y}_i | \hat{\mathbf{m}}; \alpha_i)$. The training objective is to minimize the KL divergence $\text{KL}(p_S \| p_R)$ at each step, encouraging consistency between sender and receiver.

In practice, we use Gaussian for Continuous data and categorical distribution for discrete data. Therefore Bayesian updates has closed form. The details can be found in the Appendix C.3. *Bayesian update distribution* p_U stems from the Bayesian update function h ,

$$p_U(\theta_i | \theta_{i-1}, \mathbf{m}, \mathbf{z}; \alpha_i) = \mathbb{E}_{\mathbf{y}_i \sim p_S} \delta(\theta_i - h(\theta_{i-1}, \mathbf{y}_i, \alpha_i)), \quad (18)$$

where $\delta(\cdot)$ is Dirac delta distribution. This expectation eliminates the randomness of the sent sample from the sender.

According to the nice additive property of accuracy [26], the best prediction of $\mathbf{m}$ up to time t_i is the *Bayesian flow distribution* p_F which could be obtained by adding all the precision parameters:

$$p_F(\theta_i | \mathbf{m}, \mathbf{z}; t_i) = p_U(\theta_i | \theta_0, \mathbf{m}, \mathbf{z}; \beta(t_i)), \text{ where } \beta(t_i) = \sum_{j \leq i} \alpha_j. \quad (19)$$

Therefore, the training objective for n steps is to minimize:

$$L^n(\mathbf{m}, \mathbf{z}) = \mathbb{E}_{i \sim U(1, n)} \mathbb{E}_{\mathbf{y}_i \sim p_S, \theta_{i-1} \sim p_F} D_{\text{KL}}(p_S \| p_R). \quad (20)$$

C.1.2 Inference of BFN

During inference, the sender is no longer available to provide noisy samples to help the receiver improve its belief. However, as training minimizes $D_{\text{KL}}(p_S \| p_R)$. Thus, we can reuse the same communication mechanism by iteratively applying the learned receiver distribution p_R to generate samples.

Given prior parameters θ_0 , accuracies $\alpha_1, \dots, \alpha_n$ and corresponding times $t_i = i/n$, the n -step sampling procedure recursively generates $\theta_1, \dots, \theta_n$ by sampling $\mathbf{x}'$ from $p_O(\cdot | \theta_{i-1}, \mathbf{z}, t_{i-1})$, $\mathbf{y}$ from $\mathbf{y} \sim p_R(\cdot | \theta_{i-1}, \mathbf{z}, t_{i-1}, \alpha_i)$, then setting $\theta_i = h(\theta_{i-1}, \mathbf{z}, \mathbf{y})$, and pass the result to the neural network. The final sample is drawn from $p_O(\cdot | \theta_n, \mathbf{z}, 1)$.

This recursive procedure enables BFN to generate molecule from a simple prior, guided solely by the learned receiver network and the latent codes. see Algorithm. 1

However, explicitly sampling $\mathbf{x}'$ from $p_O(\cdot | \theta_{i-1}, \mathbf{z}, t_{i-1})$, particularly for discrete data could introduce unnecessary noise and impair the stability of the generation process. Instead, following [37, 49], we directly operate in the parameter space, avoiding noise injection from sampling and enabling a more deterministic and efficient inference. See Algorithm. 3

Algorithm 1: Inference of General Bayes Flow Networks

Input: Initial prior parameter θ_0 , noise schedule $\{\alpha_i\}_{i=1}^n$, timestep grid $\{t_i = i/n\}_{i=1}^n$, conditioning latent code $\mathbf{z}$

Output: Final molecular sample $\mathbf{x}_{\text{final}}$

for $i = 1$ **to** n **do**

Sample $\mathbf{x}' \sim p_O(\cdot | \theta_{i-1}, \mathbf{z}, t_{i-1})$
 Sample $\mathbf{y} \sim p_R(\cdot | \theta_{i-1}, \mathbf{z}, t_{i-1}, \alpha_i)$
 Update latent state: $\theta_i \leftarrow h(\theta_{i-1}, \mathbf{z}, \mathbf{y})$
 Apply the network: $\theta_i \leftarrow \Phi(\theta_i, \mathbf{z}, t_i)$

Sample final output: $\mathbf{x}_{\text{final}} \sim p_O(\cdot | \theta_n, \mathbf{z}, 1)$

C.2 MolFLAE Reconstruction Loss

BFN can be trained by minimizing the KL-divergence between noisy sample distributions. BFN allows training in discrete time and continuous time, and for efficiency we adopt the n -step discrete loss.

Given the ground truth molecule $\mathbf{m} = [\mathbf{x}, \mathbf{v}]$ and its latent code z , we can have the reconstruction loss

$$\mathcal{L}_{\text{recon}} = \mathcal{L}_x^n + \mathcal{L}_v^n. \quad (21)$$

The above two summands are coordinates loss and atom type loss:

- Since the atom coordinates and the noise are Gaussian, the loss can be written analytically as follows:

$$\begin{aligned} \mathcal{L}_x^n &= D_{\text{KL}}(\mathcal{N}(\mathbf{x}, \alpha_i^{-1} \mathbf{I}) \| \mathcal{N}(\hat{\mathbf{x}}(\theta_{i-1}, \mathbf{z}, t), \alpha_i^{-1} \mathbf{I})) \\ &= \frac{\alpha_i}{2} \|\mathbf{x} - \hat{\mathbf{x}}(\theta_{i-1}, \mathbf{z}, t)\|^2 \end{aligned} \quad (22)$$

- The atom type loss can also be derived by taking KL-divergence between Gaussians [26], assuming D_M is the number of atom types, N_M is the number of the atom :

$$\begin{aligned} \mathcal{L}_v^n &= \ln \mathcal{N}(\mathbf{y}^v | \alpha_i(D_M \mathbf{e}_v - \mathbf{1}), \alpha_i D_M \mathbf{I}) \\ &\quad - \sum_{d=1}^{N_M} \ln \left(\sum_{k=1}^{D_M} p_O(k | \theta; t) \mathcal{N}(\cdot^{(d)} | \alpha_i(D_M \mathbf{e}_k - \mathbf{1}), \alpha_i D_M \mathbf{I}) \right) \end{aligned} \quad (23)$$

Together with two training losses, we can summarize the forward pass. In practice, we use reconstruction and regularization loss weights to get the final loss, see Table. 10

Algorithm 2: Forward Pass of MolFLAE

Input: Molecule $\mathcal{M} = (\mathbf{x}_M, \mathbf{v}_M)$, Number of Virtual nodes N_Z

Output: Reconstruction loss $\mathcal{L}_{\text{recon}}$, Regularization $\mathcal{L}_{\text{reg}}$

Introduce N_Z virtual nodes:

$$\mathcal{Z} = [\mathbf{x}_Z, \mathbf{v}_Z]$$

E(3)-equivariant encoding:

$$(_, [\mathbf{z}_x, \mathbf{z}_h]) \leftarrow \phi_\theta((\mathcal{M}, \mathcal{Z}))$$

VAE parameterisation:

$$\boldsymbol{\mu}_x \leftarrow \mathbf{z}_x, \quad [\boldsymbol{\sigma}_x^2, \boldsymbol{\mu}_h, \boldsymbol{\sigma}_h^2] \leftarrow \text{Linear}(\mathbf{z}_h)$$

$$\text{Get regularization loss } \mathcal{L}_{\text{reg}} \quad (\text{Eq. 5})$$

Sample latent code:

$$\mathbf{z} \leftarrow [\boldsymbol{\mu}_x, \boldsymbol{\mu}_h] + [\boldsymbol{\sigma}_x, \boldsymbol{\sigma}_h] \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

BFN decoding (Alg. 3):

$$\hat{\mathbf{m}} \leftarrow \text{BFNDecode}(\mathbf{z})$$

$$\text{Get reconstruction loss } \mathcal{L}_{\text{recon}} \quad (\text{Eq. 6})$$

return $\mathcal{L}_{\text{recon}}, \mathcal{L}_{\text{reg}}$

C.3 Bayes Updating Function for Molecular Data

Continuous coordinates The receiver observes the noisy input $\mathbf{y}^x$ and the corresponding noise level α . Starting from a prior parameterized by $\boldsymbol{\theta}_{i-1}^x = \{\boldsymbol{\mu}_{i-1}, \rho_{i-1}\}$. According to Bayes' rule, it updates its belief by the bayes updating function $h(\boldsymbol{\theta}_{i-1}^x, \mathbf{y}^x, \alpha_i) = (\boldsymbol{\mu}_i, \rho_i)$:

$$\rho_i = \rho_{i-1} + \alpha_i \quad (24)$$

$$\boldsymbol{\mu}_i = \frac{\rho_{i-1}\boldsymbol{\mu}_{i-1} + \alpha_i\mathbf{y}^x}{\rho_i} \quad (25)$$

Discrete atom types Upon receiving the noisy signal $\mathbf{y}^v$ and the noise factor α'_i , the receiver updates its belief by applying the Bayes formula with the previous parameters $\boldsymbol{\theta}_{i-1}^v$. The Bayes updating function is:

$$h(\boldsymbol{\theta}_{i-1}^v, \mathbf{y}^v, \alpha'_i) := \frac{e^{\mathbf{y}^v} \odot \boldsymbol{\theta}_{i-1}^v}{\sum_{k=1}^K e^{\mathbf{y}_k^v} (\boldsymbol{\theta}_{i-1}^v)_k} \quad (26)$$

where $\odot$ denotes element-wise multiplication.

C.4 MolFLAE Decoding Process

In the inference phase of BFN, it is in principle possible to draw samples from the receiver distribution to perform Bayesian updates. However, explicitly sampling such intermediate variables—particularly for discrete data—can introduce unnecessary stochasticity and impair the stability of the generation process. Instead, following [26, 49], we directly operate in the parameter space, avoiding noise injection from sampling and enabling a more deterministic and efficient inference.

To implement this approach, we define $\gamma(t) := \frac{\beta(t)}{1-\beta(t)}$. Let $\hat{\mathbf{m}} = [\hat{\mathbf{x}}, \hat{\mathbf{v}}]$ denote the neural network's output at a given step, where $\hat{\mathbf{v}}$ represents the continuous (pre-softmax) logits for atom types. Instead of sampling noisy observations explicitly, we directly use $\hat{\mathbf{m}}$ to update the parameters for the next iteration, thereby bypassing the stochastic sampling step in the standard Bayesian update $\boldsymbol{\theta}_i = h(\boldsymbol{\theta}_{i-1}, \mathbf{y}, \alpha)$.

Under this formulation, the Bayes Flow parameter updates simplify to:

$$p_F(\boldsymbol{\mu} \mid \hat{\mathbf{x}}, \mathbf{z}; t) = \mathcal{N}(\boldsymbol{\mu} \mid \gamma(t)\hat{\mathbf{x}}, \gamma(t)(1-\gamma(t))\mathbf{I}) \quad (27)$$

$$p_F(\boldsymbol{\theta}^v \mid \hat{\mathbf{v}}, \mathbf{z}; t) = \mathbb{E}_{\mathcal{N}(\mathbf{y}^v \mid \beta(t)(K\mathbf{e}_v - \mathbf{1}), \beta(t)K\mathbf{I})} [\delta(\boldsymbol{\theta}^v - \text{softmax}(\mathbf{y}^v))] \quad (28)$$

Here, the continuous coordinates are updated using a closed-form Gaussian posterior, while the discrete atom types are updated via an expected categorical distribution induced by a softmax over noisy logits. In practice, this expectation is approximated with a single Monte Carlo sample.

Throughout the generation process, updates are performed entirely in the parameter space, avoiding noisy sampling steps—except for the final decoding stage, where an actual molecular structure is drawn from the output distribution.

Algorithm 3: Decoder: sampling Molecules conditioned on latent code

Input: Network Φ , latent code $\mathbf{z} \in \mathbb{R}^{N_Z(3+D_Z)}$, total steps N , number of atoms N_M , number of types D_M , noise levels σ_1, β_1

Output: Sampled molecule $[\hat{\mathbf{x}}, \hat{\mathbf{v}}]$

/ Define update function*

**/*

Function update($\hat{\mathbf{x}} \in \mathbb{R}^{N_M \times 3}$, $\hat{\mathbf{v}} \in \mathbb{R}^{N_M \times D_M}$, $\beta(t)$, $\beta'(t)$, $t \in \mathbb{R}^+$):

$\gamma \leftarrow \frac{\beta(t)}{1-\beta(t)}$

$\boldsymbol{\mu} \sim \mathcal{N}(\gamma \hat{\mathbf{x}}, \gamma(1-\gamma)\mathbf{I})$

$\mathbf{y}^v \sim \mathcal{N}(\mathbf{y}^v \mid \beta'(t)(D_M \mathbf{e}_{\hat{\mathbf{v}}} - 1), \beta'(t)D_M \mathbf{I})$

$\boldsymbol{\theta}^v \leftarrow [\text{softmax}((\mathbf{y}^v)^{(d)})]_{d=1}^{N_M}$

return $\boldsymbol{\mu}, \boldsymbol{\theta}^v$

/ Initialize parameters*

**/*

$\boldsymbol{\mu} \leftarrow 0, \boldsymbol{\rho} \leftarrow 1, \boldsymbol{\theta}^v \leftarrow \left[\frac{1}{D_M} \right]_{N_M \times D_M}$

for $i = 1$ **to** N **do**

$t \leftarrow \frac{i-1}{n}$

Sample $\hat{\mathbf{x}}, \hat{\mathbf{v}} \sim p_O(\boldsymbol{\mu}, \boldsymbol{\theta}^v, \mathbf{z}, t)$

Update latent: $\boldsymbol{\mu}, \boldsymbol{\theta}^v \leftarrow \text{update}(\hat{\mathbf{x}}, \hat{\mathbf{v}}, \sigma_1, \beta_1, t)$

/ Final sampling*

**/*

Sample $\hat{\mathbf{x}}, \hat{\mathbf{v}} \sim p_O(\boldsymbol{\mu}, \boldsymbol{\theta}^v, \mathbf{z}, 1)$

return $[\hat{\mathbf{x}}, \hat{\mathbf{v}}]$

D Molecules Property Metrics

Following the setup of EDM[15], in our unconditional generation experiments, we employed the following metrics to evaluate the quality of generated molecules:

Atom Stability: Proportion of atoms with valid bond counts.

Molecular Stability: Proportion of molecules where all atoms are stable.

Validity: Proportion of molecules with RDKit-parsable SMILES.

Novelty: Proportion of molecules whose SMILES are not in the training set.

Uniqueness: Proportion of unique molecules in the generated set.

In our interpolation experiments, to fully assess the chemical and physical properties of intermediary molecules generated during interpolation, we employed the following metrics:

Similarity Preference: Defined as follows:

$$\text{Similarity Preference} = \frac{S_t - S_s}{S_t + S_s}$$

where S_t and S_s denote the Tanimoto similarity (calculated using MACCS fingerprints [45]) to the target and source molecules.

sp3frac: Represents the proportion of sp³-hybridized carbon atoms in a molecule relative to the total number of carbon atoms.

BertzCT [50]: A topological index based on a molecule's structure, considering factors like atomic connectivity, ring size, and number.

QED [51]: Evaluates the similarity of a molecule to known drug molecules by considering multiple physicochemical properties.

Labute ASA [52]: Measures the surface area of each atom in a molecule in contact with the solvent, reflecting the molecule’s solvent interaction ability.

TPSA [53]: A value calculated based on a molecule’s topological structure and atomic polarity, used to predict solubility and biological membrane penetration.

logP [54]: The logarithm of the partition coefficient of an organic compound between octanol and water, indicating the hydrophobicity of a molecule.

MR [54]: Measures a molecule’s ability to refract light, related to factors like polarizability, molecular weight, and density.

E Results on drug-likeness metrics on GEOM-Drugs

To provide a more comprehensive evaluation, we conduct experiments on the GEOM-Drugs dataset with additional drug likeness metrics, where MolFLAE showed significant improvements over existing baselines. The results are summarized in Table 7. It should be noted that we were unable to locate open resources for GeoBFN’s checkpoints on GEOM-Drugs, so experiments on GeoBFN were not conducted.

In details, we sample 10,000 molecules from each model with their default settings, obtaining atom positions and types, and then inferred bond types using OpenBabel. We then fix the bond order using Schrodinger due to some bugs in OpenBabel. The final molecules are then be evaluated using RDKit for the following metrics: QED, SA, Lipinski, and Strain Energy.

Table 7: Comparison of models on QED, SA, Lipinski and Strain Energy on GEOM-Drugs dataset.

# Metrics	QED ($\uparrow$)	SA ($\uparrow$)	Lipinski ($\uparrow$)	Strain Energy ($\downarrow$)
Data	0.64	0.84	4.80	80.19
EDM	0.36	0.59	4.26	705.2
GeoLDM	0.40	0.63	4.31	446.1
UniGEM	0.36	0.63	4.24	490.4
MolFLAE	0.60	0.75	4.75	84.66

F Ablation Study

We conduct ablation experiments on three key design choices of MolFLAE: the presence of the regularization loss, the feature embedding dimension D_Z , and the number of virtual nodes N_Z . All models are evaluated using 100-step decoding. Unless otherwise noted, all hyperparameters follow the configuration in Table 10.

Table 8 summarizes the results on QM9. Each row varies only one component while keeping all other settings fixed.

We do ablation on Regularization loss, embedding dimension D_Z and length of latent codes N_Z . Our decoder uses 100 steps for sampling. And the MolFLAE means the same hyperparameters as in Table 10. The following three experiments keeps the same settings except for the marked one. Removing the regularization loss leads to a drop in molecular stability and novelty, suggesting that

Table 8: Ablation study, performance comparison of different model config on QM9.

Model Config (steps=100)	Atom Sta (%)	Mol Sta (%)	Valid (%)	VxU (%)	Novelty (%)
MolFLAE	99.39	92.01	96.81	88.94	74.49
w/o Regularization Loss	98.73	85.01	97.82	86.43	66.04
$D_Z = 16$	98.90	87.05	93.09	94.19	70.32
$N_Z = 5$	99.21	89.24	96.60	73.21	61.54

latent smoothness is crucial for robust generation. Reducing the latent dimensionality ($D_Z = 16$)

or the number of latent nodes ($N_Z = 5$) also impacts overall performance, particularly in terms of novelty and reconstruction fidelity.

Moreover, as our iterative decoding process is a little complex, we tried to simplify it to a one-step decoding variant. However, this approach failed to generate any valid molecules.

G Hyper-parameter Settings

Hyperparameters for training on QM9, GEOM-DRUG and ZINC-9M are listed in Table 10. We followed prior works in the choice of network structure, and carried out ablation study to determine the number of latent codes introduced.

Table 9: Training costs.

Dataset	GPUs	Time	Max Epoch
GEOM-DRUG	4 Nvidia A100s(80G)	6 days	15
QM9	4 Nvidia A100s(80G)	16h	250
ZINC-9M	8 Nvidia A800s(80G)	3 days	25

Table 10: Hyperparameters for training.

Parameter	Value or description
Train/Val/Test Splitting	6921421/996/remaining data for GEOM-DRUG 9322660/932/remaining data for ZINC-9M 100000/17748/remaining data for QM9
Batch size	100 for GEOM-DRUG, 200 for ZINC-9M, 400 for QM9
Optimizer	Adam
β_1	0.95
β_2	0.99
Lr	0.005
Weight decay	0
Learning rate decay policy	ReduceLROnPlateau
Learning rate factor	0.4 for GEOM-DRUG, 0.6 for QM9 and ZINC-9M
Patience	3 for GEOM-DRUG, 10 for QM9 and ZINC-9M
Min learning rate	1.00E-06
Embedding dimension D_f	128
Head number	16
Layer number	9
k (knn)	32
Activation function	ReLU
N_Z	10
D_Z	32
var_x	100
var_h	1
Reconstruction loss weight	1
Regularization loss weight	0.1

H Broader Impacts and Safety Discussion

Our work develops an autoencoder model for molecular design, which has potential positive societal impact in areas such as drug discovery, materials science, and green chemistry by enabling the

efficient generation of candidate molecules with desired properties. However, we acknowledge that it may also be misused, for instance to generate harmful or toxic compounds. While our model is trained and evaluated on general-purpose datasets without any bias toward hazardous compounds, we emphasize that any downstream deployment should include domain-specific safeguards, such as toxicity filters and expert oversight.