Controllable Generation of Drug-like Molecules with Multi-modal Variational Flow

Fang Sun
UCLA
fts@cs.ucla.edu

Zhihao Zhan
Mila – Quebec AI Institute
Université de Montréal
zhihao.zhan@mila.quebec

Hongyu Guo
NRC Canada
University of Ottawa
hongyu.guo@uottawa.ca

Ming Zhang
Peking University
mzhang_cs@pku.edu.cn

Jian Tang
Mila – Quebec AI Institute
HEC Montréal
jian.tang@hec.ca

Yizhou Sun
UCLA
yzsun@cs.ucla.edu

Abstract

Designing drug molecules that bind effectively to target proteins while maintaining desired pharmacological properties remains a fundamental challenge in drug discovery. Current approaches struggle to simultaneously control molecular topology and 3D geometry, often requiring expensive retraining for new design objectives. We propose a multi-modal variational flow framework that addresses these limitations by integrating a 2D topology encoder with a 3D geometry generator. Our architecture encodes molecular graphs into a learned latent distribution via junction tree representations, then employs normalizing flows to autoregressively generate atoms in 3D space conditioned on the protein binding site. This design enables zero-shot controllability: by manipulating the latent prior distribution, we can generate molecules with specific substructures or optimized properties without model retraining. Experiments on the CrossDocked benchmark show that our model achieves 31.1% high-affinity rate, substantially outperforming existing methods, while maintaining superior drug-likeness and structural diversity. Our framework opens new possibilities for on-demand molecular design, allowing medicinal chemists to rapidly explore chemical space with precise control over both structural motifs and physicochemical properties.

1 Introduction

In drug discovery, designing molecules that exhibit certain binding properties and functionalities is a core challenge. In this context, drug molecules must bind to a target protein pocket and exhibit suitable biophysical and safety profiles. Molecules are governed by both 2D topological constraints (e.g., ring systems, functional groups) and 3D geometric constraints (e.g., specific atomic coordinates, conformations). Recent data-driven approaches [10, 11, 31, 39] have improved de novo molecule generation but often treat these modalities incompletely, focusing solely on 2D graphs or 3D structures without a comprehensive strategy for controllability.

A pressing issue in molecular design is accounting for two-dimensional (2D) topology and three-dimensional (3D) geometry. **2D Topology** is critical for identifying and enforcing key "pharmacophoric" (molecular features essential for biological activity) patterns [37]. For instance, the difference between the benign serotonin molecule and the hallucinogenic DMT molecule is primarily the presence of extra methyl groups on the amine moiety—these substructural differences in 2D strongly influence bioactivity [9] (Figure 4). **3D Geometry** is equally important because conformational changes can completely alter binding efficacy. For example, cisplatin [20] is a potent anti-cancer drug, whereas its trans isomer is far less active—the difference arises purely from 3D arrangement (Figure 5).

Sun et al., Controllable Generation of Drug-like Molecules with Multi-modal Variational Flow. *Proceedings of the Fourth Learning on Graphs Conference (LoG 2025)*, PMLR 338, Hybrid Event, December 10–12, 2025.

Achieving controllability in advanced drug design is also a pressing issue. It is often necessary to enforce specific substructures known to interact well with a binding site or to steer overall properties like solubility and synthetic accessibility. Achieving this controllability without re-training large generative models is highly desirable in practice [10, 34].

To address the above issues, this paper introduces a new multi-modal generation framework that unifies a 2D topology encoder and a 3D flow-based generator. The method encodes 2D information into a latent Gaussian distribution and exploits a normalizing flow to autoregressively place atoms in 3D, conditioned on the target protein environment. Crucially, the framework supports zero-shot controllability of substructures and numeric properties by manipulating the latent prior distribution. The approach addresses important challenges in drug design, particularly for generating molecules with specific therapeutic properties.

Our contributions can be summarized as follows:

1. We propose a single generative model that simultaneously handles topological and geometric constraints by encoding the 2D structure into a latent prior and learning a 3D flow generator conditioned on the target receptor.
2. Our model achieves higher binding quality than prior methods and can generate molecules containing specific substructures or optimized quantum-chemical properties.
3. We achieve zero-shot controllability by adjusting the latent prior without the need for re-training.

2 Related Work

With the development of geometric deep learning and probabilistic generative models in the past few years, *de novo* molecule generation techniques have evolved drastically, empowering us to sample diverse high-quality molecules with desired properties under various complex scenarios. To this end, there are three major tasks: 2D molecule generation, 3D unbounded molecule generation, and 3D protein-specific molecule generation.

2D molecule generation. This task [11, 19, 22, 31, 32, 38] aims at mining high-quality topological representations and generating valid 2D molecular graphs from scratch. For example, GraphAF [31] uses a flow-based model to generate atoms and bonds in an autoregressive manner. JT-VAE [11] generates molecular graphs with the guidance of a tree-structured scaffold over chemical substructures. To optimize molecules toward desired properties, models like GCPN [38] and GraphAF [31] adopt reinforcement learning to tune the model. For variational auto-encoder (VAE) based models like SD-VAE [5] and JT-VAE [11], each latent encoding of the variational distribution corresponds to a specific group of molecules in the chemical space. Therefore, these VAE-based models can perform zero-shot optimization without retraining the model, as long as they can acquire the optimal latent embedding via linear regression, bayesian optimization, etc.

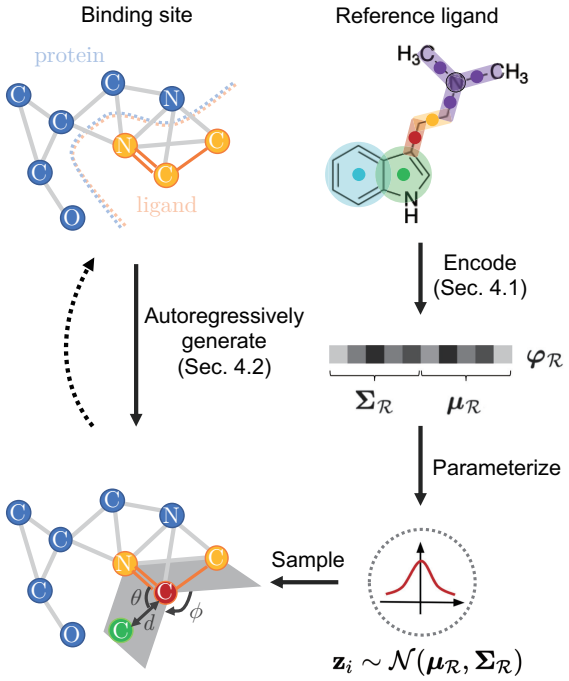

Figure 1: Overview of the variational flow workflow. Our model autoregressively samples from the variational distribution (bottom right) to generate the atoms and bonds of the drug ligand at the binding site using flow transformation (left column). The flow function is parameterized by the 3D binding site geometry. The variational distribution is encoded from the 2D topology of the reference ligand during training (right column). During controllable generation, the reference ligand is not provided. The latent encoding $\varphi_{\mathcal{R}}$ is calculated by mean aggregation of molecules with specified qualities from the training set or selected using Bayesian Optimization, which achieves controllability.

3D Non-Protein Specific Molecule Generation. This task [10, 14, 21] aims to learn the geometric representations of molecules in the 3D space and generate valid molecules with reasonable 3D conformation. For instance, G-SphereNet [21] uses symmetry invariant representations in a spherical coordinate system (SCS) to generate atoms in the 3D space and preserve equivariance. L-Net [14] encodes hierarchical molecular structure with Graph U-Net and directly outputs the topology and geometry of the molecule through a valency rule-based backtracking algorithm. EDM [10] is an equivariant diffusion model that generates 3D molecule geometry via an iterative denoising process. EDM can be configured to perform controllable generation over certain property c by re-training the diffusion model with c's feature vector concatenated to its E(n) equivariant dynamics function. RetMol [34] is a retrieval-based framework for controllable molecule generation that requires no task-specific fine-tuning.

3D Molecule Generation for Target Protein Binding. With the wide availability of large-scale datasets [7, 15] for target protein binding, recent works [17, 18, 25, 29] have been able to generate drug ligands directly based on the 3D geometry of the binding pockets. For example, Pocket2Mol [25] leverages a spatial-autoregressive model; it directly models the *p.d.f.* for atom occurrence in the 3D space as a Gaussian mixture (GMM), and then iteratively places the atoms from the learned distribution until there is no room for new atoms. GraphBP [18], an autoregressive model, retains good model capacity via normalizing flow; variables are randomly sampled from a compact latent space, before they are projected into the chemical space by an arbitrarily complex flow transformation. DiffBP [17] considers the global interaction between the protein pocket and the ligand molecule, and uses a diffusion model to generate ligand molecules non-autoregressively. Despite the promising potential along this line of purely geometric approach, these methods cannot explicitly perceive the topological pharmacophoric patterns within the ligand structure. Nor can they conduct explicit control over specific chemical sub-structures and physio-chemical properties.

3 Preliminaries

This section introduces the problem formulation, the 3D geometry encoding architecture, and the foundations of normalizing flow models that underpin our approach.

3.1 Problem Setup

Our goal is to generate a ligand molecule that binds effectively to a given protein receptor. In this paper, proteins and ligands are represented as graphs. Node features in these graphs include atom type a and position r, while edge feature involves bond type b.

For training, we are given pairs of protein $\mathcal{P}$ and ligand $\mathcal{R}$ in their binding poses. In this paper, we denote the ligand $\mathcal{R}$'s 3D geometry graph and 2D topology graph as $\mathcal{R}_{3D}$ and $\mathcal{R}_{2D}$, respectively. For generation, we are given protein targets $\mathcal{P}$ to generate drug ligands, i.e., $\mathcal{R}_{3D}$, that bind tightly to $\mathcal{P}$. We here consider a protein-ligand pair with M and N atoms respectively. Our model is trained with a set of such binding protein-ligand pairs ($\mathcal{P}$, $\mathcal{R}$).

3.2 Geometry Graph Encoding

3D-GNNs like SchNet [30] and EGNN [28] preserve SE(3) (*i.e.* roto-translational) equivariance in the 3D space, and have been canonical in encoding 3D molecule geometry. In particular, SchNet solely relies on the relative distance between nodes during message-passing and has been efficient in modeling large bio-molecular systems like protein-ligand interaction. Specifically, geometries $\mathcal{P}$ and $\mathcal{R}_{3D}$ are organized into a radius graph, based on the Euclidean distances between atoms in the 3D space. However, this purely distance-based approach has been inadequate for modeling covalent bonds in molecular structures. Bond lengths are known to be characteristic, e.g., C≡N 1.16 Å, C=C 1.34 Å [16]. Therefore, we explicitly incorporate bond types during massage-passing to better delineate the molecular structure and atomic interactions. We devise **EchNet**, an adapted version of SchNet, to achieve this end:

$$\mathbf{h}_i^{(0)} = \mathrm{Emb}(a_i) \tag{1}$$

$$\mathbf{m}_{ij} = \mathrm{concat}\left\{\mathrm{Erbf}(||r_i - r_j||), \mathrm{Emb}(b_{ij})\right\} \tag{2}$$

$$\mathbf{h}_i^{(l)} = \mathbf{h}_i^{(l-1)} + \sum_{k \in N(i) \setminus j} \mathbf{h}_k^{(l-1)} \odot \Phi^{(l)}(\mathbf{m}_{ki}), \quad l = 1, ..., L \tag{3}$$

where $\mathrm{Erbf}(\cdot)$ is a radial basis function [18], $\mathrm{Emb}(\cdot)$ is the embedding layer, $\mathrm{concat}(\cdot)$ is the concatenation of two vectors, and Φ is the feed-forward neural network. L is the number of convolution layers, b_{ij} is the bond type between atoms i and j (bond type '0' for non-existent bonds), $\mathbf{h}_i^{(l)}$ is the encoding of atom i at the lth convolution layer, and $\mathbf{m}_{ij}$ is the message propagated from atom i to j. The major difference between EchNet and SchNet resides in Equation 2, where an extra bond-type embedding is concatenated along the distance encoding.

3.3 Generative Flow Model

Flow-based deep generative models (*i.e.*, normalizing flows) transform a simple prior distribution into a complex data distribution by applying a sequence of invertible transformation functions. Through sampling from the prior, new data points (i.e., new $\mathcal{R}_{3D}$ graphs) are then generated.

To be specific, given a prior distribution p_Z, a flow model [6, 27, 36] is defined as an invertible parameterized function $f_\theta : \mathbf{z} \in \mathbb{R}^D \rightarrow \mathbf{x} \in \mathbb{R}^D$, where θ represents the parameters of f, and D is the dimension for $\mathbf{z}$ and $\mathbf{x}$. This maps the latent variable $\mathbf{z} \sim p_Z$ to the data variable $\mathbf{x}$, and the log-likelihood of $\mathbf{x}$ is calculated as

$$\log p_X(\mathbf{x}) = \log p_Z\left(f_\theta^{-1}(\mathbf{x})\right) + \log \left| \det \frac{\partial f_\theta^{-1}(\mathbf{x})}{\partial \mathbf{x}} \right|. \tag{4}$$

To effectively solve the above equation, autoregressive flow model [24] formulates a flow function with an autoregressive computation to enable easy Jacobian determinant computation. Specifically, let $\mathbf{x}_i$ be the i-th component of $\mathbf{x}$ and $\mathbf{x}_i$ conditions on $\mathbf{x}_{1\ldots i-1}$. The inverse function f_θ^{-1} is then defined as follows:

$$\mathbf{x}_i = \sigma_i(\mathbf{x}_{1\ldots i-1}) \odot \mathbf{z}_i + \mu_i(\mathbf{x}_{1\ldots i-1}), \quad i = 1 \ldots D, \tag{5}$$

where $\odot$ denotes element-wise multiplication, $\sigma_i(\cdot) \in \mathbb{R}$ and $\mu_i(\cdot) \in \mathbb{R}$ are non-linear functions of $\mathbf{x}_{1\ldots i-1}$. By doing so, we can effectively calculate the following to compute the log-likelihood in Equation 4:

$$\mathbf{z}_i = \frac{\mathbf{x}_i - \mu_i}{\sigma_i}, \quad \det \frac{\partial f_\theta^{-1}(\mathbf{x})}{\partial \mathbf{x}} = \prod_{i=1}^{D} \frac{1}{\sigma_i}. \tag{6}$$

4 The Proposed Method

Our approach aims to generate 3D binding molecules, namely generating $\mathcal{R}_{3D}$ that binds to $\mathcal{P}$, through sampling from a prior distribution with a variational flow model. The flow model here aims at transforming a simple *prior* distribution into the complex distribution of the 3D binding molecules by applying a sequence of invertible transformation functions as discussed in Section 3.3. As illustrated in **Figure 1**, to obtain desirable topological information in the generated 3D molecules ($\mathcal{R}_{3D}$), as the aim of this paper, our method encodes the topology patterns in 2D molecules ($\mathcal{R}_{2D}$) into the *prior* distribution. By doing so, we can purposefully control the *prior* parameterization for different generation tasks, and sampling from such *prior* thus facilitates the generation of 3D molecules with specific sub-structures and physio-chemical properties encoded in the *prior* distribution.

Next, we will introduce our 2D topology prior encoding and the 3D binding ligand generation components in Sections 4.1 and 4.2, respectively.

4.1 2D Topology Prior Encoding

To encode the 2D topology in $\mathcal{R}_{2D}$ into a prior distribution, we adopt the junction tree encoder architecture from JT-VAE [11]. The whole procedure is illustrated in Figure 3 (in the Appendix) and detailed next.

4.1.1 Ligand Scaffold Extraction and Encoding

Following [11], we extract the coarse-grained structural patterns of the ligand scaffold in a fragment-driven approach. First, the ligand molecule $\mathcal{R}_{2D}$ is parsed into a compilation of occluded canonical sub-structures, according to a set of pre-defined vocabulary rules (detailed in Appendix A.1). Each of such sub-structure in $\mathcal{R}_{2D}$ is then pooled into a node, resulting in a junction tree. Next, the

information of this junction tree is aggregated through a Gated Recurrent Unit (GRU) [4] adapted for tree message passing (Section 4.1.2). This results in a root node embedding $\mathbf{h}_{\text{root}}$ representing the whole junction tree and thus the $\mathcal{R}_{2D}$. Finally, this embedding is then passed through a MLP to define the mean and variance of the topology distribution:

$$(\boldsymbol{\mu}_{\mathcal{R}_{2D}}, \boldsymbol{\Sigma}_{\mathcal{R}_{2D}}) = \varphi_{\mathcal{R}_{2D}} = \text{MLP}(\mathbf{h}_{\text{root}}). \tag{7}$$

Since $\boldsymbol{\mu}_{\mathcal{R}_{2D}}$ and $\boldsymbol{\Sigma}_{\mathcal{R}_{2D}}$ are equally-sized dense vectors, the covariate matrix of the resultant Gaussian $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}_{2D}}, \boldsymbol{\Sigma}_{\mathcal{R}_{2D}})$ is diagonal. This allows us to independently sample from the different components of the Gaussian distribution (elaborated in Appendix A.2). We go on to describe how our model conducts tree message passing through GRU in the junction tree to obtain the above $\mathbf{h}_{\text{root}}$.

4.1.2 Junction Tree Message Passing

The tree message passing scheme arbitrarily selects a leaf node as the root (denoted as $\mathbf{h}_{\text{root}}$), and passes messages from child nodes to parent nodes iteratively in a bottom-up approach. We denote the message from node i to j as $\mathbf{m}_{ij}$, which is updated via a GRU adapted for tree propagation:

$$\mathbf{m}_{ij} = \text{GRU}(\mathbf{x}_i, \{\mathbf{m}_{ki}\}_{k \in N(i) \setminus j}). \tag{8}$$

To be more specific, the GRU architecture is formulated as follows:

$$\begin{aligned}
\mathbf{s}_{ij} &= \sum\nolimits_{k \in N(i) \setminus j} \mathbf{m}_{ki}, && (9)\\
\mathbf{z}_{ij} &= \sigma(\mathbf{W}^z \mathbf{x}_i + \mathbf{U}^z \mathbf{s}_{ij} + \mathbf{b}^z), && (10)\\
\mathbf{r}_{ki} &= \sigma(\mathbf{W}^r \mathbf{x}_i + \mathbf{U}^r \mathbf{m}_{ki} + \mathbf{b}^r), && (11)\\
\widetilde{\mathbf{m}}_{ij} &= \tanh(\mathbf{W}\mathbf{x}_i + \mathbf{U} \sum_{k \in N(i) \setminus j} \mathbf{r}_{ki} \odot \mathbf{m}_{ki}), && (12)\\
\mathbf{m}_{ij} &= (1 - \mathbf{z}_{ij}) \odot \mathbf{s}_{ij} + \mathbf{z}_{ij} \odot \widetilde{\mathbf{m}}_{ij}, && (13)
\end{aligned}$$

where $\mathbf{x}_i$ is a one-hot vector, indicating the type of canonical sub-structure of node i. The latent representation of each node $\mathbf{h}_i$ can be derived by aggregating all the inwards messages from its child nodes as follows:

$$\mathbf{h}_i = \mathbf{W}^o \mathbf{x}_i + \sum_{k \in N(i)} \mathbf{U}^o \mathbf{m}_{ki}. \tag{14}$$

4.1.3 Topology Prior

During training, the 2D scaffold encoding $\varphi_{\mathcal{R}_{2D}} = (\boldsymbol{\mu}_{\mathcal{R}_{2D}}, \boldsymbol{\Sigma}_{\mathcal{R}_{2D}})$ is regularized by a KL divergence to form a compact family of diagonal Gaussians around the standard gaussian $\mathcal{N}(\mathbf{0}, \mathcal{I})$. Therefore, **during generation**, we can easily navigate along this family of Gaussians to generate 3D molecules with specific sub-structures and physio-chemical properties encoded in the *prior* distribution. Next, we will discuss how the flow model leverages this prior to generate binding 3D ligands.

4.2 3D Ligand Generation via Variational Flow

In this section, we will discuss how the model generates a 3D ligand $\mathcal{R}$ (to simplify the equations in this section, we omit the subscript), including its atoms, bonds, and geometric structure.

4.2.1 Autoregressive Generation via Flow

We formulate the procedure of generating a new ligand $\mathcal{R}$ as a Markovian sampling process, where atoms and bonds are autoregressively added according to the intermediary state at the binding site. The generation process at step $i = 4$ is illustrated in Figure 2 and elucidated next.

Firstly, we construct radius graph $\mathcal{G}_i$ based on protein graph $\mathcal{P}$ and ligand sub-graph $\mathcal{R}_{1:i-1}$:

$$\mathcal{G}_i = \tau(\mathcal{P} \cup \mathcal{R}_{1:i-1}), \tag{15}$$

where the radius operator $\tau(\cdot)$ adds edges (of bond order 0) to neighboring atoms within radius τ. In particular, at generation step 1, when no ligand atoms have yet been generated, $\mathcal{G}_i$ is simply $\tau(\mathcal{P})$.

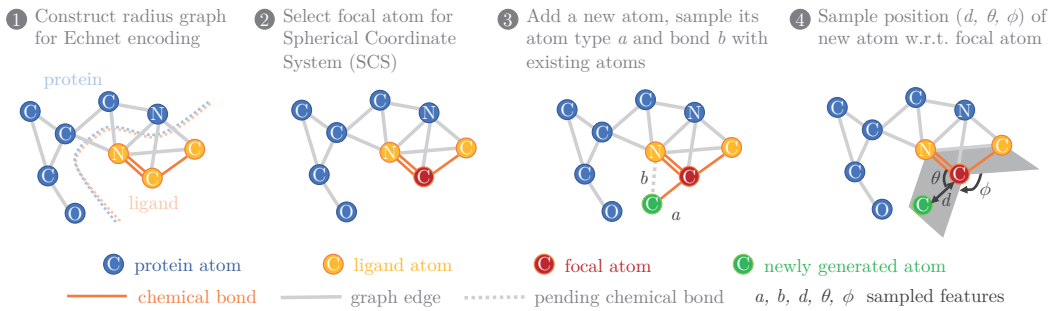

Figure 2: Generation procedure of our model. Atoms are added autoregressively, whose types, bonds and positions are sampled from prior distribution $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})$ and predicted via normalizing flow.

The Echnet encoder, as discussed in Section 3.2, outputs the encoding of each atom in both protein and ligand:

$$\tilde{\mathbf{e}}_{1:M}, \mathbf{e}_{1:i-1} = \text{Echnet}(\mathcal{G}_i). \tag{16}$$

Secondly, we randomly sample a focal atom f_i from all possible candidates. For each atom, its eligibility as a focal atom is determined by a binary focal classifier. Except in the first step, only atoms from the ligand molecule are considered. Based on f_i and two of its nearest neighbors, we construct a spherical coordinate system (SCS), transforming Cartesian coordinates into polar coordinates (d, θ, ϕ). Autoregressive generation under the SCS preserves the equivariance quality of our model. Refer to Appendices G.1 and G.2 for implementation details of focal atom classification and SCS construction. Proof for equivariance can be found in Appendix G.3.

Finally, we add a new atom to the drug ligand via sequential generation of its atom type a_i, bindings with existing atoms $b_{1:i-1,i}$ and SCS coordinates $\mathbf{x}_i^{(\text{pos})} = (d_i, \theta_i, \phi_i)$, in order to better capture the underlying dependencies [18]. This is achieved by sampling the prior random variables $\mathbf{z}_i^{(\text{node})}$, $\mathbf{z}_{1:i-1,i}^{(\text{bond})}$ and $\mathbf{z}_i^{(\text{pos})}$ from the variational distribution $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})$:

$$\left[\mathbf{z}_i^{(\text{node})}; \mathbf{z}_{1:i-1,i}^{(\text{bond})}; \mathbf{z}_i^{(\text{pos})}\right] \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}\right). \tag{17}$$

Recall from Equation 7 that $\mathcal{N}\left(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}\right)$ is parameterized as a diagonal Gaussian, so each component of the random variable can be sampled independently from each other. In particular, $\mathbf{z}^{(\text{bond})}$ is repeatedly sampled for $(i-1)$ times because we need to determine the bond type between the new atom i and each of the previous $(i-1)$ ligand atoms. The priors are then consecutively projected to the 3D geometric space via flow transformation $\mathcal{F}_i$:

$$\mathbf{x}_i^{(\text{node})}, \mathbf{x}_{1:i-1,i}^{(\text{bond})}, \mathbf{x}_i^{(\text{pos})} = \mathcal{F}_i\left(\mathbf{z}_i^{(\text{node})}, \mathbf{z}_{1:i-1,i}^{(\text{bond})}, \mathbf{z}_i^{(\text{pos})}; \mathbf{e}_{1:i-1}\right). \tag{18}$$

4.2.2 Parameterization of the Flow Transformation

Following the paradigm described in Equatioin 5, the above flow transformation $\mathcal{F}_i$ is parameterized with the subsequent steps:

$$\mu_i^{(\text{node})}, \sigma_i^{(\text{node})} = \text{Node-MLP}(\mathbf{e}_{f_i}), \tag{19}$$

$$\mathbf{x}_i^{(\text{node})} = \sigma_i^{(\text{node})} \odot \mathbf{z}_i^{(\text{node})} + \mu_i^{(\text{node})}, \tag{20}$$

$$\mu_{1:i-1,i}^{(\text{bond})}, \sigma_{1:i-1,i}^{(\text{bond})} = \text{Bond-MLP}(\mathbf{e}_{1:i-1}, \mathbf{x}_i^{(\text{node})}), \tag{21}$$

$$\mathbf{x}_{1:i-1,i}^{(\text{bond})} = \sigma_{1:i-1,i}^{(\text{bond})} \odot \mathbf{z}_{1:i-1,i}^{(\text{bond})} + \mu_{1:i-1,i}^{(\text{bond})}, \tag{22}$$

$$\mu_i^{(\text{pos})}, \sigma_i^{(\text{pos})} = \text{Position-MLP}(\mathbf{e}_{f_i}, \mathbf{x}_i^{(\text{node})}, \mathbf{x}_{1:i-1,i}^{(\text{bond})}), \tag{23}$$

$$\mathbf{x}_i^{(\text{pos})} = \sigma_i^{(\text{pos})} \odot \mathbf{z}_i^{(\text{pos})} + \mu_i^{(\text{pos})}, \tag{24}$$

where $\mathbf{e}_{f_i}$ is the encoding of the focal atom f_i, and Node-MLP, Bond-MLP, and Position-MLP are layers of flow MLPs, detailed in Appendix B. $\odot$ denotes element-wise multiplication, and $\mathbf{x}_i^{(\text{node})}, \mathbf{x}_{1:i-1,i}^{(\text{bond})}, \mathbf{x}_i^{(\text{pos})}$ are the vectorized representation of atom type, bond type, and SCS-based

position, and σ, μ are parameters for flow transformation. The sequential dependencies between a, b, d, θ, ϕ are embodied in Equations 21 and 23, where new atom/bond types that have just been generated are immediately used to parameterize σ and μ of the next flow transformation.

Thus, we have rendered all the sampled features $a_i, b_{1:i-1,i}, d_i, \theta_i, \phi_i$ from step i, and successfully generate the new atom and its associated bonds. We go on with this iteration, until the focal classifier reports that no atom is eligible for f_i, and the generation procedure is called to an end. Algorithm 2 from Appendix C explains the generation algorithm in more detail.

4.2.3 The Objective Function

Our model is trained with a variational flow objective that combines flow-based likelihood maximization with a KL regularization term:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{flow}} + \beta \mathcal{L}_{\text{KL}}, \tag{25}$$

where $\mathcal{L}_{\text{flow}}$ is the negative log-likelihood of the flow model for generating 3D molecular structures, $\mathcal{L}_{\text{KL}}$ aligns the learned topology prior $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})$ with a standard Gaussian $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and β is a hyperparameter that balances these two terms. The detailed derivation of these loss components and the training algorithm are provided in Appendix D.

4.3 Controllable generation

During generation, the variational prior in our model provides a flexible interface for controlling certain properties of the generated 3D molecules in a zero-shot manner, without the need to re-train the model. For a certain desired ligand attribute ρ, there are two ways to acquire its corresponding prior:

1) Mean aggregation. This is suitable for qualitative attributes, such as the existence of a certain pharmacophore or sub-structure. We collect a set of such molecules with attribute ρ (denoted as $\{\mathcal{R}_a\}_{a \in I}$, where I is the index set), and carry out mean aggregation over their structural encodings:

$$(\boldsymbol{\mu}_\rho, \boldsymbol{\Sigma}_\rho) = \frac{1}{|I|} \sum_{a \in I} (\boldsymbol{\mu}_{\mathcal{R}_a}, \boldsymbol{\Sigma}_{\mathcal{R}_a}). \tag{26}$$

2) Bayesian Optimization. This approach is ideal for numerical attributes like free energy and enthalpy. We use a sparse gaussian process (SGP) to fit the relationship f between latent encoding and the desired property value

$$f : (\boldsymbol{\mu}_{\mathcal{R}_a}, \boldsymbol{\Sigma}_{\mathcal{R}_a}) \mapsto ||\rho||. \tag{27}$$

upon this relation f, we perform bayesian optimization to find the latent encoding that corresponds to the maximum value of $||\rho||$:

$$(\boldsymbol{\mu}_\rho, \boldsymbol{\Sigma}_\rho) = \underset{\boldsymbol{\mu}, \boldsymbol{\Sigma}}{\operatorname{argmax}}\, f(\boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{28}$$

5 Experiments

We conduct three sets of experiments to verify the effectiveness of our model:

- Basic protein-specific generation. Prior set to $\mathcal{N}(\mathbf{0}, \mathcal{I})$.
- Protein-specific generation with controlled sub-structure. Prior selected using mean aggregation.
- Protein-specific generation with controlled properties. Prior selected using Bayesian Optimization.

5.1 3D Molecular Generation Conditioned on Protein Pocket

Dataset. We use the benchmarking CrossDocked dataset [7], which contains 22.5 million protein-ligand pairs, to evaluate the generation performance of our model. For fair comparison, we follow Pocket2Mol [25] to prepare and split the data.

Setup. Following GraphBP [18] and Pocket2Mol, we randomly sample 100 molecules for every protein pocket in the generation stage. The quality of generated molecules is evaluated by the following key metrics: **High Affinity (HA)** estimates the percentage of generated molecules that have higher *CNNAfinity* calculated by the *Gnina* program [23]; **Lipinski** estimates the mean number

of Lipinski rules followed by the generated molecules; **Novelty** is calculated as $1 -$ the average of maximum Tanimoto similarities to training set molecules among the generated molecules; and **Diversity** is calculated as $1 -$ the average Tanimoto similarities of generated molecules for every protein pockets. Additional metrics are detailed in Appendix E. We choose GraphBP and Pocket2Mol as our baselines, which represent the state-of-art models for binding molecule generation. For GraphBP and our model, we trained them on the dataset for 40 epochs with the same hyperparameters. For Pocket2Mol [25], we obtain the pre-trained model from their authors and then compute the scores using *Gnina*.

Method	HA↑	Lipinski↑	Novelty↑	Diversity↑
GraphBP	0.134	4.909	0.569	0.835
Pocket2Mol	0.272	4.920	0.624	0.688
Ours (w/o 2D encoder)	0.263	4.080	0.605	0.807
Ours	**0.311**	**4.968**	**0.737**	**0.930**

Table 1: Performance of different methods on 3D molecular generation based on protein pockets. Best results are in **bold**.

Results. The comparison results are presented in Table 1. We can see that our model outperforms the two strong baselines in terms of HA, Lipinski, Novelty, and Diversity. This shows that our model learns good molecular representations by combining 2D topology with 3D geometry, and is able to generate robustly good molecules under a different prior. The great novelty and diversity of our molecules can be attributed to our variational training strategy. The distribution shift between training and generation allows our model to explore larger and more complex molecules in the chemical space, and can hopefully generate novel drug-like molecules that have never been discovered before.

Ablation study. A critical question about the variational flow architecture is: Will the KL loss term (equation 39) collapse the variational distribution to a standard gaussian, and degrade our model to a purely flow-based model? This is not happening, because our model can learn potentially better molecular representations from $\varphi_{\mathcal{R}_{2\mathrm{D}}}$. To empirically prove this claim and mimic the degradation scenario, we perform an ablation study by masking the 2D encoder $\varphi_{\mathcal{R}_{2\mathrm{D}}}$ and substituting the prior as the standard gaussian $\mathcal{N}(\mathbf{0}, \mathcal{I})$ during training. From the 3rd row of Table 1, we can see that the ablated 'Ours (w/o 2D encoder)' version performs much worse. In particular, the HA value for the ablated variant drops drastically from 31.1% to 26.3%. This shows that the variational-flow framework is very effective in balancing the 2D/3D dual data sources and preventing model degradation.

5.2 Sub-structure Analysis

Setup. As it is pointed out in Pocket2Mol [25] that conventional metrics could not reflect the geometry of sampled molecules, we conduct additional sub-structure analysis. Following Pocket2Mol, We compare our model with previous works by the KL divergence between the distributions of generated bond lengths and dihedral angles and the corresponding distributions of the test set.

Sub-Structure	GraphBP	Pocket2Mol	Ours (w/o 2D encoder)	Ours
CC	0.27	2.18	1.05	**0.22**
C=O	0.83	3.78	0.73	**0.67**
CN	**0.70**	1.78	1.27	0.77
CCCC	2.15	2.10	2.03	**2.00**
CCCO	2.37	2.27	2.17	**2.17**
CC=CC	2.20	2.85	2.70	**2.04**

Table 2: The KL divergence of the bond distances (upper part) and dihedral angles (lower part) with the test set. The best results are in **bold**.

Results. The results are presented in Table 2. In comparison to GraphBP and Pocket2Mol, our model yields the best results on dihedral angles, which indicates that it is more capable of modeling complex dependencies. It also achieves competitive results on bond distances. Ablation study shows 2D encoder helps our model better capture the geometry of sub-structures.

Rate of specified sub-structure(%)	Test Set	GraphBP	Pocket2Mol	Ours w/ latent ρ	Ours w/o latent ρ
alkenyl	76	41.8	90.7	**93.0**	66.2
imine	47	5.5	51.7	**83.5**	49.7
thiophene-like	10	2.1	2.7	**45.0**	5.4
pyran-like	13	4.7	14.3	**86.5**	17.1

Table 3: Controllable generation for specified sub-structures. Best results bolded; closest result to test set underlined.

5.3 Controllable Generation for Specified Chemical Sub-structures

Our pretrained framework could be used to encourage desired sub-structures during generation. We carry out case studies on generation of molecules containing the following motifs: alkenyl (carbon-carbon double bonds), imine (C=N functional groups), thiophene-like structures (5-membered aromatic rings containing sulfur), and pyran-like structures (6-membered rings containing oxygen). For each motif, we use mean aggregation to calculate latent distribution $\mathcal{N}(\boldsymbol{\mu}_\rho, \boldsymbol{\Sigma}_\rho)$ from 500 randomly sampled reference ligand molecules in the training set that contain the motif as a sub-structure. Finally, we calculate the rate of the generated molecules that contain the specified sub-structures on the test set, which is compared with the results of directly sampling from prior distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

The experimental results are summarized in Table 3. With the prior distributions collected from molecules that contain specified sub-structures, our model is more likely to generate ligand molecules with those sub-structures. When directly sampled from prior distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, our model generates specified sub-structures at a rate which is the closest to the appearing rate of these structures in the test set, in comparison with GraphBP and Pocket2Mol.

5.4 Controllable Molecular Generation for Specified Molecular Properties

Our framework can also be explicitly controlled to generate drug-like molecules with desired properties. To support this claim, we perform case studies to optimize the quantum-mechanical (QM) properties of the generated drug ligands right at the binding site. Since *ab-initio* QM property calculations are very computationally expensive, we use the DimeNet [12] model for property prediction, which has been pretrained on the QM-9 [26] dataset. We select 5 important physio-chemical properties for prediction: highest occupied molecular orbital energy ($\epsilon_{\text{HOMO}}/\text{eV}$), internal energy at 0K (U_0/eV), internal energy at 298.15 K (U/eV), enthalpy at 298.15K (H/eV), and free energy at 298.15K (G/eV).

Average property value	Test Set	GraphBP	Pocket2Mol	Ours w/ latent ρ	Ours w/o latent ρ
$\epsilon_{\text{HOMO}}/\text{eV}$	-6.64	-7.03	-6.78	**-6.60**	-6.68
U_0/eV	-174.35	-148.29	-141.00	**-108.76**	-202.17
U/eV	-175.43	-149.03	-141.55	**-127.07**	-202.31
H/eV	-176.83	-150.28	-142.63	**-123.40**	-208.27
G/eV	-160.83	-137.24	-128.75	**-106.63**	-183.01

Table 4: Controllable generation for specified molecular properties. The highest values are in bold.

We retrieve these properties of our CrossDocked training-set molecules from DimeNet, and their latent encodings from the tree encoder. As described in Section 4.3, we use SGP to fit the property-encoding relationship, and then find the optimal latent encoding $\mathcal{N}(\boldsymbol{\mu}_\rho, \boldsymbol{\Sigma}_\rho)$ with the **highest** energy through bayesian optimization, which is further used for controllable property generation. Experiment results are presented in Table 4. Our model achieves consistently higher energy than all the baselines, which clearly shows that the latent φ_ρ is effective in curating desired properties of the generated molecules.

6 Conclusion

This work demonstrates that the long-standing dichotomy between 2D molecular graphs and 3D conformations can be resolved through variational learning. Our experiments reveal an unexpected finding: the KL regularization term, rather than collapsing the latent space, creates a structured manifold where chemical properties vary smoothly—enabling precise navigation for molecular design. The success of zero-shot controllability across diverse objectives, from thiophene-like rings to quantum mechanical properties, suggests that our learned representations capture fundamental structure-property relationships. Beyond the immediate applications in drug discovery, this framework points toward a new paradigm where generative models can serve as interactive design tools, allowing chemists to explore hypotheses about molecular function through direct manipulation of learned chemical spaces.

Acknowledgement

This work was partially supported by NSF 2211557, NSF 2119643, NSF 2312501; the NAIRR Pilot Program (NSF 2202693, NSF 2312501); the SRC JUMP 2.0 Center; Amazon Research Awards; Snapchat Gifts; Mila – Quebec Artificial Intelligence Institute; the Canadian Institute for Advanced Research (CIFAR); C-CAS (Center for Computer Assisted Synthesis, NSF CHE–2202693); and CSIRO (the Commonwealth Scientific and Industrial Research Organisation, NSF 2303037).

References

[1] Maximilian Balandat et al. "BoTorch: A framework for efficient Monte-Carlo Bayesian optimization". In: *Advances in neural information processing systems* 33 (2020), pp. 21524–21538. 13

[2] G. R. Bickerton et al. "Quantifying the chemical beauty of drugs". In: *Nature chemistry* 4.2 (2012), pp. 90–98. 17

[3] CEOI. "Sightseeing trip". In: *http://poj.org/problem?id=1734* (1999). 20

[4] Junyoung Chung et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling". In: *arXiv preprint arXiv:1412.3555* (2014). 5

[5] Hanjun Dai et al. "Syntax-directed variational autoencoder for structured data". In: *arXiv preprint arXiv:1802.08786* (2018). 2

[6] Laurent Dinh, David Krueger, and Yoshua Bengio. "Nice: Non-linear independent components estimation". In: *arXiv preprint arXiv:1410.8516* (2014). 4

[7] Paul G Francoeur et al. "Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design". In: *Journal of chemical information and modeling* 60.9 (2020), pp. 4200–4215. 3, 7

[8] A. K. Ghose, V. N. Viswanadhan, and J. J. Wendoloski. "A Knowledge-Based Approach in Designing Combinatorial or Medicinal Chemistry Libraries for Drug Discovery. 1. A Qualitative and Quantitative Characterization of Known Drug Databases." In: *Journal of Combinatorial Chemistry* (1999). 17

[9] Juan S Gomez-Jeria and Andres Robles-Navarro. "A Note on the Docking of some Hallucinogens to the 5-HT2A Receptor". In: *Journal of Computational Methods in Molecular Design* 5.1 (2015), pp. 45–57. 1

[10] Emiel Hoogeboom et al. "Equivariant diffusion for molecule generation in 3d". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 8867–8887. 1–3

[11] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. "Junction tree variational autoencoder for molecular graph generation". In: *International conference on machine learning*. PMLR. 2018, pp. 2323–2332. 1, 2, 4

[12] Johannes Klicpera et al. "Fast and uncertainty-aware directional message passing for non-equilibrium molecules". In: *arXiv preprint arXiv:2011.14115* (2020). 9

[13] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. "Numba: A llvm-based python jit compiler". In: *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*. 2015, pp. 1–6. 20

[14] Yibo Li, Jianfeng Pei, and Luhua Lai. "Learning to design drug-like molecules in three-dimensional space using deep generative models". In: *arXiv preprint arXiv:2104.08474* (2021). 3

[15] Yibo Li, Jianfeng Pei, and Luhua Lai. "Structure-based de novo drug design using 3D deep generative models". In: *Chemical science* 12.41 (2021), pp. 13664–13675. 3

[16] David R Lide. "Characteristic bond lengths in free molecules". In: *CRC Handbook of Chemistry and Physics. CRC Press/Taylor and Francis,* (2012). 3

[17] Haitao Lin et al. "DiffBP: Generative Diffusion of 3D Molecules for Target Protein Binding". In: *arXiv preprint arXiv:2211.11214* (2022). 3

[18] Meng Liu et al. "Generating 3D Molecules for Target Protein Binding". In: *arXiv preprint arXiv:2204.09410* (2022). 3, 4, 6, 7

[19] Qi Liu et al. "Constrained graph variational autoencoders for molecule design". In: *Advances in neural information processing systems* 31 (2018). 2

[20] Patrick J Loehrer and LAWRENCE H EINHORN. "Cisplatin". In: *Annals of internal medicine* 100.5 (1984), pp. 704–713. 1

[21] Youzhi Luo and Shuiwang Ji. "An autoregressive flow model for 3d molecular geometry generation from scratch". In: *International Conference on Learning Representations (ICLR)*. 2022. 3

[22] Kaushalya Madhawa et al. "Graphnvp: An invertible flow model for generating molecular graphs". In: *arXiv preprint arXiv:1905.11600* (2019). 2

[23] A. T. McNutt et al. "Gnina 1.0: Molecular Docking with Deep Learning." In: *Journal of cheminformatics* (2021). 7

[24] George Papamakarios, Theo Pavlakou, and Iain Murray. "Masked autoregressive flow for density estimation". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, pp. 2335–2344. 4

[25] Xingang Peng et al. "Pocket2Mol: Efficient Molecular Sampling Based on 3D Protein Pockets". In: *arXiv preprint arXiv:2205.07249* (2022). 3, 7, 8

[26] Raghunathan Ramakrishnan et al. "Quantum chemistry structures and properties of 134 kilo molecules". In: *Scientific data* 1.1 (2014), p. 140022. 9

[27] Danilo Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows". In: *International Conference on Machine Learning*. 2015, pp. 1530–1538. 4

[28] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. "E (n) equivariant graph neural networks". In: *International conference on machine learning*. PMLR. 2021, pp. 9323–9332. 3

[29] Arne Schneuing et al. "Structure-based Drug Design with Equivariant Diffusion Models". In: *arXiv preprint arXiv:2210.13695* (2022). 3

[30] K. T. Schutt et al. "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions". In: *arXiv preprint arXiv:1706.08566* (2017). 3

[31] Chence Shi et al. "Graphaf: a flow-based autoregressive model for molecular graph generation". In: *arXiv preprint arXiv:2001.09382* (2020). 1, 2

[32] Martin Simonovsky and Nikos Komodakis. "Graphvae: Towards generation of small graphs using variational autoencoders". In: *International conference on artificial neural networks*. Springer. 2018, pp. 412–422. 2

[33] Robert Tarjan. "Depth-first search and linear graph algorithms". In: *SIAM journal on computing* 1.2 (1972), pp. 146–160. 20

[34] Zichao Wang et al. "Retrieval-based Controllable Molecule Generation". In: *arXiv preprint arXiv:2208.11126* (2022). 2, 3

[35] Eric W Weisstein. "Floyd-warshall algorithm". In: *https://mathworld. wolfram. com/* (2008). 20

[36] Lilian Weng. "Flow-based deep generative models". In: *lilianweng. github. io/lil-log* (2018). 4

[37] C. G. Wermuth et al. "Glossary of terms used in medicinal chemistry (IUPAC Recommendations 1998)". In: *Pure and Applied Chemistry* 70.5 (1998), pp. 1129–1143. DOI: doi:10.1351/pac199870051129. URL: https://doi.org/10.1351/pac199870051129. 1

[38] Jiaxuan You et al. "Graph convolutional policy network for goal-directed molecular graph generation". In: *Advances in neural information processing systems* 31 (2018). 2

[39] Jinhua Zhu et al. "Direct molecular conformation generation". In: *arXiv preprint arXiv:2202.01356* (2022). 1

A Ligand scaffolds encoding

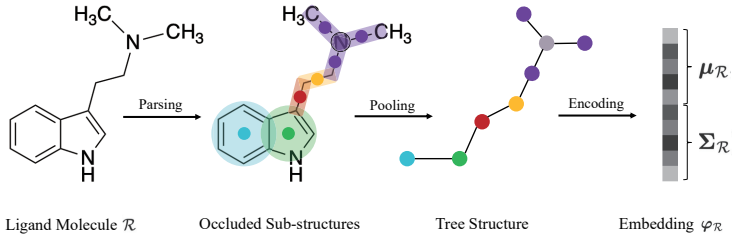

Figure 3: Ligand molecule $\mathcal{R}_{2D}$ (e.g. DMT) is first parsed into a compilation of canonical sub-structures, then pooled into a junction tree structure, and finally encoded into $\varphi_{\mathcal{R}_{2D}} = (\boldsymbol{\mu}_{\mathcal{R}_{2D}}, \boldsymbol{\Sigma}_{\mathcal{R}_{2D}})$.

A.1 Parsing and pooling

There are 3 types of canonical sub-structures, as exemplified by the DMT molecule in Figure 3:

1. Rings, e.g. the blue, green nodes;
2. Non-ring covalent atom pairs, e.g. the red, yellow and purple nodes;
3. Pivot atoms that are connected to 3 or more items, e.g. the gray node.

The rules for identifying sub-structures are self-contained, yielding a relatively sparse and stable set of vocabulary. A total of 427 canonical sub-structures are identified from the 100,000 reference ligands in the CrossDocked dataset. Once the ligand molecule is parsed into a compilation of sub-structures, the molecular graph can be pooled into a junction tree in a straightforward manner, where each sub-structure corresponds to a tree node, and any two intersecting sub-structures yield an edge between their corresponding nodes.

A.2 Latent encoding

Finally, the structural encoding of the whole molecule is obtained by feeding $\mathbf{h}_{\text{root}}$ through a MLP:

$$(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}) = \varphi_{\mathcal{R}} = \text{MLP}(\mathbf{h}_{\text{root}}). \tag{29}$$

Two equally-shaped (34×1) dense vectors $\boldsymbol{\mu}_{\mathcal{R}}$ and $\boldsymbol{\Sigma}_{\mathcal{R}}$ comprise the latent encoding and parameterize the mean and variance of an amortized diagonal Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})$. Their different channels serve different purposes during sampling. A de-quantized one-hot vector for atom type (27 possible choices) is sampled from the first 27 channels. A de-quantized one-hot vector for bond type (single, double, triple, or none) is sampled from the next 4 channels. The latent values for (d, θ, ϕ) are directly sampled from the last three channels because they are continuous by nature. The gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})$ is parameterized to have a diagonal co-variate matrix, so each channel of this gaussian is independent from each other.

All parameters in this section are **trainable**. Our earlier attempts with a pre-trained version of GRU would result in serious degradation of the quality of generated molecules. Thus, end-2-end training of these parameters is ideal for achieving better model performance.

B Implementation Details

Network Architecture. We stack 6 layers of the Echnet and 20 layers of the tree GRU. We use 6 variational flow layers for generation.

Training Details. We train our model for 40 epochs on the full training set with batch size 4. We use Adam optimizer while setting learning rate as 1e-4 and weight decay as 1e-6. For the β-annealing which is applied to the whole training process, we pick the minimum β as 1e-4 and maximum β as 0.015.

Generation Details. We sample 100 molecules for each pocket in the test set. Molecules that have less than 15 atoms are excluded and re-sampled, while molecules that have more than 50 atoms are truncated at the 50-th atom. Additionally, to help our model generate ligand molecules with good geometric properties, we propose to limit the sample space by five validity constraints during generation:

1. A bond always exists between the newly generated atom and the focal atom;
2. At most one other atom could be connected to the newly generated atom with a bond;
3. The newly generated atom could only have bonds with atoms that are predicted positive by the focal classifier;
4. The element of generated atom always lies in C, N, O, P, S, Cl;
5. The length of all generated bonds should be less than 10 Å;
6. When all the generated molecules have a diameter $>$ 10 Å, all unfinished molecules should be dropped for a new round of generation.

These constraints can be flexibly applied, without the need to re-train the model from scratch. Therefore, they can be duly employed to our generation process to achieve uniformly good results on different benchmarks.

Sub-structure Analysis. To approximate the distribution for visualization and calculation of KL divergence, we set 0.01 Å per bin for distance and 1 degree per bin for bond angles and dihedral angles. In total, we have 2,000 bins for distances ranging from 0 to 20 Å, 180 bins for bond angles and 360 bins for dihedral angles. To avoid 0 during calculating KL divergence, we replace every 0-count with 1 / number-of-bins. For fair comparison, we use every model to sample 100 ligand molecules for each protein in the test set.

Controllable Generation for Specified Sub-structures. When we sample from the training set, molecules with more than 16 atoms are dropped because they are likely to be less representative owing to having multiple sub-structures.

Controllable Generation for Specified Properties. We use DimeNet++ pretrained on QM-9 dataset (preset as part of the DGL library) for generation. We implement Bayesian optimization using BoTorch [1], a framework for efficient Monte-Carlo Bayesian Optimization. Since SGP has $\mathcal{O}(n^3)$ time complexity and $\mathcal{O}(n^2)$ space complexity, we can only afford to perform BO on a 1/10 training dataset with 10000 reference ligand molecules. We use an upper confidence bound (UCB) with $\beta = 0.1$. In the acquisition function, we use 512 raw samples for initializations and set 10 re-starts to get the top 1 encoding with the highest property score. During the inference stage, we notice that DimeNet can produce unrealistic property scores close to infinity, so we regularize the prediction result using the 3σ principle, where the mean and variance statistics is gleaned from the QM-9 dataset labels. This regularization effectively covers about 90% of all the outputs.

C Algorithms for training and generation

The pseudo codes of training and generation algorithms are in Algorithms 1 and 2.

Algorithm 1 Training algorithm of our model

Input: η learning rate, B batch size, T maximum epoch number, Variational annealing hyperparameters $\beta_{\min}, \beta_{\max}$, use $\mathrm{Prod}(\cdot)$ as the product of elements across dimensions of a tensor

Initial: Parameters θ of model (Echnet, junction tree encoder, focal classifier, and Node/Edge/Position-MLP)

for $t = 1, .., T$ **do**
  $\beta = \beta_{\min} + (\beta_{\max} - \beta_{\min}) \sin^2\left(\pi \frac{t}{T}\right)$ ▷ β-annealing acc. to epoch number
  for $b = 1, ..., B$ **do**
    Sample a receptor-ligand pair from dataset, with receptor size M and ligand size N
    Protein receptor $\mathcal{P} = (\tilde{V}, \tilde{E})$, where $\tilde{V} = \{(\tilde{a}_i, \tilde{r}_i)\}_{i=1}^{M}$, and $\tilde{E} = \{\tilde{b}_{ij}\}_{i,j=1}^{M}$
    Drug ligand $\mathcal{R} = (V, E)$, where $V = \{(a_i, r_i)\}_{i=1}^{N}$, and $E = \{b_{ij}\}_{i,j=1}^{N}$
    Re-order $\mathcal{R}$ with ring-first graph traversal
    $(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}) =$ JT-Encoder$(\mathcal{R})$, where prior $Z \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})$ ▷ **2D Global**
    for $i = 1, ..., N$ **do** ▷ **3D Autoregressive**
      Construct sub-graph $\mathcal{G}_i := \tau(\mathcal{P} \cup \mathcal{R}_{1:i-1})$
      if i=1 **then**
        $f_i :=$ Nearest receptor atom
        $\hat{\boldsymbol{y}} =$ one-hot$_M(f_i)$
        Predict focal score $\boldsymbol{y}$ for all receptor atoms $1 : M$
      else
        $f_i := i - 1$
        $\hat{\boldsymbol{y}} =$ one-hot$_{i-1}(f_i)$
        Predict focal score $\boldsymbol{y}$ for all known ligand atoms $1 : i - 1$
      end if
      $\tilde{\mathbf{e}}_{1:M}, \mathbf{e}_{1:i-1} = \mathrm{Echnet}(\mathcal{G}_i)$ ▷ Encode 3D conformation
      $\mathbf{x}_i^{(\mathrm{node})} = a_i + \mathbf{u},\ \mathbf{u} \sim \mathcal{U}[0,1)^{d^{(\mathrm{node})}}$ ▷ Atom type dequantization
      $\mu_i^{(\mathrm{node})}, \sigma_i^{(\mathrm{node})} =$ Node-MLP$(\mathbf{e}_{f_i})$
      $\mathbf{z}_i^{(\mathrm{node})} = \left(\mathbf{x}_i^{(\mathrm{node})} - \mu_i^{(\mathrm{node})}\right) \odot \frac{1}{\sigma_i^{(\mathrm{node})}}$
      $\mathbf{x}_{1:i-1,i}^{(\mathrm{bond})} = b_{1:i-1,i} + \mathbf{u},\ \mathbf{u} \sim \mathcal{U}[0,1)^{(i-1)\times d^{(\mathrm{bond})}}$ ▷ Bond type dequantization
      $\mu_{1:i-1,i}^{(\mathrm{bond})}, \sigma_{1:i-1,i}^{(\mathrm{bond})} =$ Bond-MLP$(\mathbf{e}_{1:i-1}, \mathbf{x}_i^{(\mathrm{node})})$
      $\mathbf{z}_{1:i-1,i}^{(\mathrm{bond})} = \left(\mathbf{x}_{1:i-1,i}^{(\mathrm{bond})} - \mu_{1:i-1,i}^{(\mathrm{bond})}\right) \odot \frac{1}{\sigma_{1:i-1,i}^{(\mathrm{bond})}}$
      $\mathbf{x}_i^{(\mathrm{pos})} = \mathrm{Spherize}(\boldsymbol{r}_i; f_i, c_i, e_i)$ ▷ Spherize atom position to f_i
      $\mu_i^{(\mathrm{pos})}, \sigma_i^{(\mathrm{pos})} =$ Position-MLP$(\mathbf{e}_{f_i}, \mathbf{x}_i^{(\mathrm{node})}, \mathbf{x}_{1:i-1,i}^{(\mathrm{bond})})$
      $\mathbf{z}_i^{(\mathrm{pos})} = \left(\mathbf{x}_i^{(\mathrm{pos})} - \mu_i^{(\mathrm{pos})}\right) \odot \frac{1}{\sigma_i^{(\mathrm{pos})}}$
      $\mathcal{L}_i^{(\mathrm{node})} = -\log(\mathrm{Prod}(\mathcal{N}(\mathbf{z}_i^{(\mathrm{node})} | \boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}))) - \log(\mathrm{Prod}(\frac{1}{\sigma_i^{(\mathrm{node})}}))$
      $\mathcal{L}_{1:i-1,i}^{(\mathrm{bond})} = -\log(\mathrm{Prod}(\mathcal{N}(\mathbf{z}_{1:i-1,i}^{(\mathrm{bond})} | \boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}))) - \log(\mathrm{Prod}(\frac{1}{\sigma_{1:i-1,i}^{(\mathrm{bond})}}))$
      $\mathcal{L}_i^{(\mathrm{pos})} = -\log(\mathrm{Prod}(\mathcal{N}(\mathbf{z}_i^{(\mathrm{pos})} | \boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}))) - \log(\mathrm{Prod}(\frac{1}{\sigma_i^{(\mathrm{pos})}}))$
      $\mathcal{L}_{\mathrm{flow}}^{i,b} = \mathcal{L}_i^{(\mathrm{node})} + \mathcal{L}_{1:i-1,i}^{(\mathrm{bond})} + \mathcal{L}_i^{(\mathrm{pos})}$ ▷ Step-wise loss term for normalizing flow
      $\mathcal{L}_{\mathrm{focal}}^{i,b} = \mathrm{BCELoss}(\boldsymbol{y}, \hat{\boldsymbol{y}})$ ▷ Step-wise loss term for focal classifier
    end for
    $\mathcal{L}_{\mathrm{KL}}^{b} = D_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}) || \mathcal{N}(\mathbf{0}, \boldsymbol{I}))$ ▷ Global loss term for variational distribution Z
    $\mathcal{L}_{\mathrm{total}}^{b} = \frac{1}{N} \sum_{i=1}^{N} \left(\mathcal{L}_{\mathrm{flow}}^{i,b} + \mathcal{L}_{\mathrm{focal}}^{i,b}\right) + \beta \mathcal{L}_{\mathrm{KL}}^{b}$
  end for
  $\theta \leftarrow \mathrm{ADAM}(\sum_{b=1}^{B} \mathcal{L}_{\mathrm{total}}^{b}, \theta, \eta)$ ▷ Parameter update
end for

Algorithm 2 Generation algorithm of our model

Input: T number of protein receptors, B number of drug ligands to generate for each receptor, N maximum number of atoms in the generated ligand. Optional parameters $(\boldsymbol{\mu}_\rho, \boldsymbol{\Sigma}_\rho)$ as the cue to certain desired property ρ, $(\mathbf{0}, \boldsymbol{I})$ by default.

Initial: Trained model (Echnet, junction tree encoder, focal classifier, and Node/Edge/Position-MLP)

for $t = 1, .., T$ **do**
  Sample a protein receptor from dataset, with receptor size M
  Protein receptor $\mathcal{P} = (\tilde{V}, \tilde{E})$, where $\tilde{V} = \{(\tilde{a}_i, \tilde{r}_i)\}_{i=1}^{M}$, and $\tilde{E} = \{\tilde{b}_{ij}\}_{i,j=1}^{M}$
  $\mathrm{LigGen}_t \leftarrow [\,]$
  for $b = 1, ..., B$ **do**
    Drug ligand representation $\mathcal{R} := (V, E)$, initialized as empty
    for $i = 1, ..., N$ **do**
      Construct sub-graph $\mathcal{G}_i := \tau(\mathcal{P} \cup \mathcal{R}_{1:i-1})$
      Predict focal score, sample focal atom f_i from eligible atoms
      if none eligible for f_i **then** ▷ Signal for generation complete
        break inner loop
      end if
      $\tilde{\mathbf{e}}_{1:M}, \mathbf{e}_{1:i-1} = \mathrm{Echnet}(\mathcal{G}_i)$ ▷ Encode 3D conformation
      Sample $\left[\mathbf{z}_i^{(\text{node})}; \mathbf{z}_{1:i-1,i}^{(\text{bond})}; \mathbf{z}_i^{(\text{pos})}\right] \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}\right)$ ▷ Sample from latent space
      $\mu_i^{(\text{node})}, \sigma_i^{(\text{node})} = \text{Node-MLP}(\mathbf{e}_{f_i})$
      $\mathbf{x}_i^{(\text{node})} = \sigma_i^{(\text{node})} \odot \mathbf{z}_i^{(\text{node})} + \mu_i^{(\text{node})}$ ▷ Atom type generation
      $\mu_{1:i-1,i}^{(\text{bond})}, \sigma_{1:i-1,i}^{(\text{bond})} = \text{Bond-MLP}(\mathbf{e}_{1:i-1}, \mathbf{x}_i^{(\text{node})})$
      $\mathbf{x}_{1:i-1,i}^{(\text{bond})} = \sigma_{1:i-1,i}^{(\text{bond})} \odot \mathbf{z}_{1:i-1,i}^{(\text{bond})} + \mu_{1:i-1,i}^{(\text{bond})}$ ▷ Bond type generation
      $\mu_i^{(\text{pos})}, \sigma_i^{(\text{pos})} = \text{Position-MLP}(\mathbf{e}_{f_i}, \mathbf{x}_i^{(\text{node})}, \mathbf{x}_{1:i-1,i}^{(\text{bond})})$
      $\mathbf{x}_i^{(\text{pos})} = \sigma_i^{(\text{pos})} \odot \mathbf{z}_i^{(\text{pos})} + \mu_i^{(\text{pos})}$ ▷ Atom position generation
      Derive $(a_i, \boldsymbol{r}_i)$ from $(\mathbf{x}_i^{(\text{node})}, \mathbf{x}_i^{(\text{pos})})$; $b_{1:i-1,i}$ from $\mathbf{x}_{1:i-1,i}^{(\text{bond})}$
      $V.\mathrm{append}(\{(a_i, r_i)\}); E.\mathrm{append}(\{b_{1:i-1,i}\})$ ▷ Autoregressive ligand generation
    end for
    $\mathrm{LigGen}_t.\mathrm{append}(\mathcal{R})$
  end for
end for
return $[\mathrm{LigGen}_1, \mathrm{LigGen}_2, ..., \mathrm{LigGen}_T]$

D Objective Function Details

Our variational flow model uses the following training objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{flow}} + \beta \mathcal{L}_{\text{KL}}, \tag{30}$$

The flow loss $\mathcal{L}_{\text{flow}}$ maximizes the likelihood of generating correct 3D molecular structures. Through the change of variables formula, the training process inverts the generation process:

$$\mathbf{z}_i^{(\text{node})}, \mathbf{z}_{1:i-1,i}^{(\text{bond})}, \mathbf{z}_i^{(\text{pos})} = \mathcal{F}_i^{-1}\left(\mathbf{x}_i^{(\text{node})}, \mathbf{x}_{1:i-1,i}^{(\text{bond})}, \mathbf{x}_i^{(\text{pos})}; \mathbf{e}_{1:i-1}\right), \tag{31}$$

where the latent variables are computed as:

$$\mathbf{z}_i^{(\text{node})} = \left(\mathbf{x}_i^{(\text{node})} - \mu_i^{(\text{node})}\right) \odot \frac{1}{\sigma_i^{(\text{node})}} \tag{32}$$

$$\mathbf{z}_{1:i-1,i}^{(\text{bond})} = \left(\mathbf{x}_{1:i-1,i}^{(\text{bond})} - \mu_{1:i-1,i}^{(\text{bond})}\right) \odot \frac{1}{\sigma_{1:i-1,i}^{(\text{bond})}} \tag{33}$$

$$\mathbf{z}_i^{(\text{pos})} = \left(\mathbf{x}_i^{(\text{pos})} - \mu_i^{(\text{pos})}\right) \odot \frac{1}{\sigma_i^{(\text{pos})}}. \tag{34}$$

The flow loss components are:

$$\mathcal{L}_i^{(\text{node})} = -\log(\text{Prod}(\mathcal{N}(\mathbf{z}_i^{(\text{node})}|\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}))) - \log(\text{Prod}(\frac{1}{\sigma_i^{(\text{node})}})) \tag{35}$$

$$\mathcal{L}_{1:i-1,i}^{(\text{bond})} = -\log(\text{Prod}(\mathcal{N}(\mathbf{z}_{1:i-1,i}^{(\text{bond})}|\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}))) - \log(\text{Prod}(\frac{1}{\sigma_{1:i-1,i}^{(\text{bond})}})) \tag{36}$$

$$\mathcal{L}_i^{(\text{pos})} = -\log(\text{Prod}(\mathcal{N}(\mathbf{z}_i^{(\text{pos})}|\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}}))) - \log(\text{Prod}(\frac{1}{\sigma_i^{(\text{pos})}})) \tag{37}$$

$$\mathcal{L}_{\text{flow}}^{i,b} = \mathcal{L}_i^{(\text{node})} + \mathcal{L}_{1:i-1,i}^{(\text{bond})} + \mathcal{L}_i^{(\text{pos})}. \tag{38}$$

The KL regularization term aligns the learned 2D topology prior with a standard Gaussian:

$$\mathcal{L}_{\text{KL}}^b = D_{\text{KL}}(\mathcal{N}(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})||\mathcal{N}(\mathbf{0}, \boldsymbol{I})). \tag{39}$$

Additionally, we include a focal classifier loss:

$$\mathcal{L}_{\text{focal}}^{i,b} = \begin{cases} \frac{1}{M}\sum_{j=1}^{M} \text{BCELoss}(\tilde{y}_j, \hat{\tilde{y}}_j), & \text{if } i = 1; \\ \frac{1}{i-1}\sum_{j=1}^{i-1} \text{BCELoss}(y_j, \hat{y}_j), & \text{if } i > 1, \end{cases} \tag{40}$$

where y_j is the predicted focal score and $\hat{y}_j$ is the ground-truth label from the ring-first expert trajectory (Appendix H).

The final per-batch loss is:

$$\mathcal{L}_{\text{total}}^b = \frac{1}{N}\sum_{i=1}^{N}(\mathcal{L}_{\text{flow}}^{i,b} + \mathcal{L}_{\text{focal}}^{i,b}) + \beta \mathcal{L}_{\text{KL}}^b. \tag{41}$$

E Additional Evaluation Metrics

In addition to the key metrics reported in the main paper, we also evaluated the following metrics:

1. **Synthetic Accessibility (SA)**, which represents the easiness of drug synthesis;
2. **Quantitative Estimation of Drug-likeness (QED)**, a measure of drug-likeness based on desirability [2];
3. **LogP** denotes the octanol-water partition coefficient. Good drug candidates have LogP between -0.4 and 5.6 [8];
4. **Time** estimates the time(s) spent on generating 100 molecules for a pocket.

Our model attained reasonably good performance on these drug properties even without explicit guidance from the variational prior. Notably, our model is much more efficient than Pocket2Mol by 2 orders of magnitude in generation time.

F Chemical structure illustration

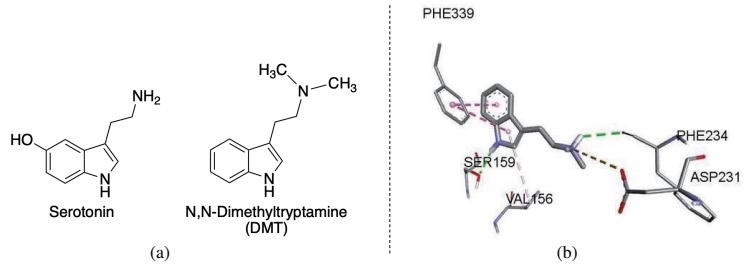

Figure 4: (a) Comparison between Serotonin and DMT structure; (b) Binding pose of DMT with 5-$\mathrm{HT}_{2\mathrm{A}}$, pay special attention to the interaction between NHMe_2 and Asp-231.

Figure 5: Cisplatin (left) versus its trans-isomer (right) illustrating how identical atomic composition with different spatial arrangements leads to dramatically different biological activity. Cisplatin is an effective anti-cancer drug that binds to DNA, while the trans-isomer shows minimal anti-tumor activity.

G Focal atom and the spherical coordinate system

G.1 Focal atom selection

During the **training/generation** process, focal scores are evaluated at each step. In the first step, there are no known ligand atoms, so the focal score is evaluated among the receptor molecules. In the following steps, focal atoms are only evaluated among the known ligand atoms.

For the **generation** process, atoms become *eligible* candidates of the focal atom, as long as their focal scores exceed a given threshold (set to 0.5 in our implementation). The focal atom of that step is then randomly sampled from these eligible candidates. In the case when no atoms are eligible for the focal atom, the generation process is called to an end.

For the **training** process, we use a teacher-forcing strategy. That is to say, we rely on an expert trajectory (curated from the ring-first traversal algorithm, described in Appendix H) to select our focal atom. In the first step, the focal atom is set to be the receptor atom that is nearest to the drug ligand. In the i-th step ($i > 1$), the focal atom is set to be the $(i-1)$-th atom in the expert trajectory. The ground truth label for focal score is set to either 1 (focal) or 0 (non-focal), and we use a mean-reduced BCELoss to evaluate the discrepancy between the predicted focal score and the ground-truth label.

G.2 Construction of the spherical coordinate system (SCS)

We construct a spherical coordinate system (SCS) around the focal atom f and its nearest 2 neighboring atoms c and e. Formally, given the Cartesian coordinates of reference atoms $(\boldsymbol{r}_f, \boldsymbol{r}_c, \boldsymbol{r}_e)$, we want to express the position of an arbitrary atom i in spherical coordinates (d_i, θ_i, ϕ_i). Suppose $\boldsymbol{r}_i$ is the Cartesian coordinates of atom i, $\boldsymbol{n}_1$ is the normal vector of plane $(\boldsymbol{r}_f, \boldsymbol{r}_c, \boldsymbol{r}_i)$, and $\boldsymbol{n}_2$ is the normal vector of plane $(\boldsymbol{r}_f, \boldsymbol{r}_c, \boldsymbol{r}_e)$, then

$$\begin{cases} d_i = \|\boldsymbol{r}_i - \boldsymbol{r}_f\|_2 , & d_i \geq 0 \\ \theta_i = \langle \boldsymbol{r}_i - \boldsymbol{r}_f, \boldsymbol{r}_c - \boldsymbol{r}_f \rangle , & \theta_i \in [0, \pi] \\ \phi_i = \langle \boldsymbol{n}_1, \boldsymbol{n}_2 \rangle , & \phi_i \in [-\pi, \pi] \end{cases} \tag{42}$$

Conversely, we can also render the Cartesian coordinates of i from its spherical coordinates:

$$(x_i, y_i, z_i) = \boldsymbol{r}_i = \boldsymbol{r}_f + \frac{d_i(\boldsymbol{r}_c - \boldsymbol{r}_f)\cos\theta_i}{\|\boldsymbol{r}_c - \boldsymbol{r}_f\|_2^2} + \frac{d_i(\boldsymbol{r}_{e,\phi_i} - \boldsymbol{r}_{e,cf})\sin\theta_i}{\|\boldsymbol{r}_{e,\phi_i} - \boldsymbol{r}_{e,cf}\|_2^2}, \tag{43}$$

where $\boldsymbol{r}_{e,cf}$ is the coordinate of the projection of e on the line $(\boldsymbol{r}_f, \boldsymbol{r}_c)$, and $\boldsymbol{r}_{e,\phi_i}$ is the coordinate of e after rotating the plane $(\boldsymbol{r}_f, \boldsymbol{r}_c, \boldsymbol{r}_e)$ along the line $(\boldsymbol{r}_f, \boldsymbol{r}_c)$ by the torsion angle ϕ_i. We define this operation as $h : (d_i, \theta_i, \phi_i) \mapsto (x_i, y_i, z_i)$. Note that this transformation between the Cartesian and spherical coordinates is SE(3)-equivariant. Namely, for any orthogonal matrix $Q \in \mathbb{R}^{3\times 3}$ and translation vector $\boldsymbol{b} \in \mathbb{R}^3$:

$$h\left(d_i, \theta_i, \phi_i; Q\boldsymbol{r}_f + \boldsymbol{b}, Q\boldsymbol{r}_c + \boldsymbol{b}, Q\boldsymbol{r}_e + \boldsymbol{b}\right) \tag{44}$$

$$= Q\boldsymbol{r}_f + \boldsymbol{b} + \frac{d_i \cos\theta_i \left(Q\boldsymbol{r}_c - Q\boldsymbol{r}_f\right)}{\|Q\boldsymbol{r}_c - Q\boldsymbol{r}_f\|_2^2} + \frac{d_i \sin\theta_i \left(Q\boldsymbol{r}_{e,\phi_i} - Q\boldsymbol{r}_{e,cf}\right)}{\|Q\boldsymbol{r}_{e,\phi_i} - Q\boldsymbol{r}_{e,cf}\|_2^2} \tag{45}$$

$$= Q\boldsymbol{r}_f + \boldsymbol{b} + Q\frac{d_i \cos\theta_i \left(\boldsymbol{r}_c - \boldsymbol{r}_f\right)}{\|\boldsymbol{r}_c - \boldsymbol{r}_f\|_2^2} + Q\frac{d_i \sin\theta_i \left(\boldsymbol{r}_{e,\phi_i} - \boldsymbol{r}_{e,cf}\right)}{\|\boldsymbol{r}_{e,\phi_i} - \boldsymbol{r}_{e,cf}\|_2^2} \tag{46}$$

$$= Q\left[\boldsymbol{r}_f + \frac{d_i \cos\theta_i \left(\boldsymbol{r}_c - \boldsymbol{r}_f\right)}{\|\boldsymbol{r}_c - \boldsymbol{r}_f\|_2^2} + \frac{d_i \sin\theta_i \left(\boldsymbol{r}_{e,\phi_i} - \boldsymbol{r}_{e,cf}\right)}{\|\boldsymbol{r}_{e,\phi_i} - \boldsymbol{r}_{e,cf}\|_2^2}\right] + \boldsymbol{b} \tag{47}$$

$$= Qh\left(d_i, \theta_i, \phi_i; \boldsymbol{r}_f, \boldsymbol{r}_c, \boldsymbol{r}_e\right) + \boldsymbol{b}. \tag{48}$$

G.3 Equivariance of the generation process

The intuition behind preserving equivariance during the generation process is: when the rest part of a molecule moves in space, the newly generated part of that molecule should move accordingly. Formally, we have the following theorem:

Theorem 1 (SE(3)-equivariant generation). At the i-th ($i = 1, ..., N$) generation step, the generation probability is equivariant to any orthogonal matrix $Q \in \mathbb{R}^{3\times 3}$ and translation vector $\boldsymbol{b} \in \mathbb{R}^3$:

$$p\left(Q\boldsymbol{r}_i + \boldsymbol{b} \mid A_i, B_i, R_i Q^T + \mathbf{1}\boldsymbol{b}^T\right) = p\left(\boldsymbol{r}_i \mid A_i, B_i, R_i\right), \tag{49}$$

where A_i is the types of all known atoms, B_i is the types of all known bonds, and R_i is the Cartesian coordinates of all known atoms.

1° To prove the above theorem, we first show that the spherical coordinates (d_i, θ_i, ϕ_i) are SE(3)-invariant to flow transformation. By combining Equations 16 and 18, we can derive the following clean-cut equation:

$$(d_i, \theta_i, \phi_i) = g(\mathbf{z}_i^a, \mathbf{z}_{1:i-1,i}^b, z_i^d, z_i^\theta, z_i^\phi; A_i, B_i, R_i), \tag{50}$$

where g is an invertible flow transformation, parameterized by A_i, B_i, R_i through EchNet. Since EchNet perceives 3D geometry only through relative distances, g is thus SE(3)-invariant:

$$g(\mathbf{z}_i^a, \mathbf{z}_{1:i-1,i}^b, z_i^d, z_i^\theta, z_i^\phi; A_i, B_i, R_iQ^T + \mathbf{1}\boldsymbol{b}^T) = g(\mathbf{z}_i^a, \mathbf{z}_{1:i-1,i}^b, z_i^d, z_i^\theta, z_i^\phi; A_i, B_i, R_i). \tag{51}$$

2° We further show that Cartesian coordinates are SE(3)-equivariant to flow transformation under the same set of reference atoms. For Equation 51, we substitute its LHS into Equation 44, and its RHS into Equation 48:

$$h\left(g(\mathbf{z}_i^a, \mathbf{z}_{1:i-1,i}^b, z_i^d, z_i^\theta, z_i^\phi; A_i, B_i, R_iQ^T + \mathbf{1}\boldsymbol{b}^T); Q\boldsymbol{r}_f + \boldsymbol{b}, Q\boldsymbol{r}_c + \boldsymbol{b}, Q\boldsymbol{r}_e + \boldsymbol{b}\right) \tag{52}$$

$$= Qh\left(g(\mathbf{z}_i^a, \mathbf{z}_{1:i-1,i}^b, z_i^d, z_i^\theta, z_i^\phi; A_i, B_i, R_i); \boldsymbol{r}_f, \boldsymbol{r}_c, \boldsymbol{r}_e\right) + \boldsymbol{b}. \tag{53}$$

We define a short-hand composite function $g^r := h \circ g$, and the resultant equation unequivocally shows that Cartesian coordinates are SE(3)-equivariant to g^r under the same f, c, e:

$$g^r(\mathbf{z}_i^a, \mathbf{z}_{1:i-1,i}^b, z_i^d, z_i^\theta, z_i^\phi; A_i, B_i, R_iQ^T + \mathbf{1}\boldsymbol{b}^T) = Qg^r(\mathbf{z}_i^a, \mathbf{z}_{1:i-1,i}^b, z_i^d, z_i^\theta, z_i^\phi; A_i, B_i, R_i) + \boldsymbol{b} = Q\boldsymbol{r}_i + \boldsymbol{b}. \tag{54}$$

3° Finally, since both sides of Equation 54 share the same underlying distribution $[\mathbf{z}_i^a; \mathbf{z}_{1:i-1,i}^b; z_i^d; z_i^\theta; z_i^\phi] \sim \mathcal{N}(\boldsymbol{\mu}_\mathcal{R}, \boldsymbol{\Sigma}_\mathcal{R})$, we eventually come to the SE(3)-equivariance of our generation process, formulated as Equation 49. **Theorem 1** is thus proved.

H Ring-first traversal algorithm

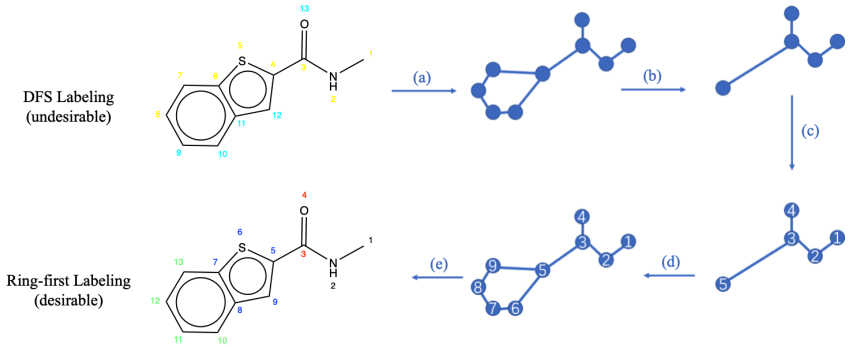

Figure 6: Overview of the ring-first traversal algorithm, from step (a) to (e). The desirable ring-first labeling is demonstrated in the down-left, and the undesirable DFS labeling is in the up-left. Consecutive labels are marked with the same color.

For our auto-regressive model, we curate an expert trajectory for training (Appendix G.1), using the ring-first traversal algorithm. Chemical structures are typically composed of rings and functional groups, and the ring-first traversal algorithm tries to label atoms from the same ring/functional group consecutively to preserve the local semantics of the chemical structure **during training**. For example, in the left side of Figure 6, the ring-first labeling is a desirable delineation of the different rings/functional groups existing in the molecule (benzene, thiophene, and carbonyl), but the DFS labeling mixes them up.

We propose a minimal-ring Floyd algorithm to implement the ring-first strategy. As shown in Figure 6, it consists of 5 steps:

(a) Find the minimal ring via Floyd's algorthm [3, 35], and substitute it into a single point;
(b) Repeat step (a), until there is no ring in the graph, *i.e.* the graph has become a tree;
(c) Apply depth-first search (DFS) [33] to label the tree nodes;
(d) If a tree node has once been rings, expand it back into a ring. Label the nodes in the same ring consecutively. Offset the labeling of successive nodes accordingly;
(e) Repeat step (d), until the original molecular graph structure is restored. The resultant labeling is the desirable ring-first labeling.

This ring-first traversal algorithm has a time complexity of $\mathcal{O}(n^5)$, where n is the number of nodes in the molecular graph (excluding hydrogen atoms). To accelerate the algorithm, we refactor our code with Numba [13] to allow just-in-time compilation and achieve a marked $\sim 50\times$ acceleration. We have thus been able to improve our training procedure using ring-first traversal, without losing efficiency.