FlexSBDD: Structure-Based Drug Design with Flexible Protein Modeling

Zaixi Zhang[1,2,3], **Mengdi Wang**[3], **Qi Liu**[1,2]*
1: School of Computer Science and Technology, University of Science and Technology of China
2:State Key Laboratory of Cognitive Intelligence, Hefei, Anhui, China
3:Princeton University
zaixi@mail.ustc.edu.cn, mengdiw@princeton.edu, qiliuql@ustc.edu.cn

Abstract

Structure-based drug design (SBDD), which aims to generate 3D ligand molecules binding to target proteins, is a fundamental task in drug discovery. Existing SBDD methods typically treat protein as rigid and neglect protein structural change when binding with ligand molecules, leading to a big gap with real-world scenarios and inferior generation qualities (e.g., many steric clashes). To bridge the gap, we propose FlexSBDD, a deep generative model capable of accurately modeling the flexible protein-ligand complex structure for ligand molecule generation. FlexSBDD adopts an efficient flow matching framework and leverages E(3)-equivariant network with scalar-vector dual representation to model dynamic structural changes. Moreover, novel data augmentation schemes based on structure relaxation/sidechain repacking are adopted to boost performance. Extensive experiments demonstrate that FlexSBDD achieves state-of-the-art performance in generating high-affinity molecules and effectively modeling the protein's conformation change to increase favorable protein-ligand interactions (e.g., Hydrogen bonds) and decrease steric clashes.

1 Introduction

Deep generative models are profoundly impacting drug discovery, particularly within the challenging subfield of structure-based drug design (SBDD) [1, 54, 36]. SBDD focuses on generating drug-like ligand molecules conditioned on target-binding proteins, necessitating precise modeling of complex geometric structures and detailed protein-ligand interactions. Some early attempts adopt autoregressive models to generate 3D ligand molecules atom-by-atom [53, 57] or fragment-by-fragment [85]. To overcome the limitations of autoregressive methods (e.g., error accumulation), recent works [28, 29] leverage non-autoregressive diffusion-based models [31] to predict the distribution of ligand atom types and positions via denoising and have achieved the state-of-the-art performance.

Despite the remarkable success, most existing SBDD models treat target proteins as rigid and neglect the conformation change. However, according to the "induced fit" theory in biochemistry [43], proteins are flexible structures that undergo structural changes upon ligand binding, leading to enhanced interactions and binding affinity. In structural biology, the protein structure that has a bound small molecule is referred to as ligand-bound or **holo** conformation, and the protein structure without a bound small molecule is called ligand-free or **apo** conformation [16]. For example, Figure. 1 (a)&(b) shows the aligned apo and holo-structures of two proteins, and the extent of structural change is influenced by the specific properties and structures of the proteins and ligands. The neglect of protein flexibility in SBDD leads to several significant drawbacks: (1) The generated protein-ligand complexes are prone to have sub-optimal protein structures and steric clashes as the protein cannot adaptively adjust structures according to different generated molecules [30]. (2) The chemical search

*Qi Liu is the corresponding authors.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-1707

space is overly restricted by existing holo data, limiting the exploration of diverse high-quality drugs. As a result, SBDD models tend to produce molecules that are similar to those fitting the predefined pocket space [53, 62]. (3) A large gap between real-world physical binding process and computational simulations, which may lead to high false positive rates in real-world drug discovery applications i.e., most computationally designed drugs do not have real therapeutic effects [2].

However, several challenges exist for flexible protein modeling in existing SBDD models. Firstly, proteins are macromolecules with thousands of atoms, which brings formidable high degrees of freedom for flexible structural modeling [35]. Secondly, there lack apo-holo structure pair datasets for learning structural changes [16]. Finally, the widely used diffusion-based models are time-consuming for exploring the huge space of flexible protein-ligand complexes [28].

To address the aforementioned challenges, we propose a new method FlexSBDD capable of modeling the flexibility of target protein while generating *de novo* 3D ligand molecules. To reduce the computational complexity, we focus on the key degrees of freedom in protein structure (i.e., C_α coordinate, the orientation of the backbone frame, and the sidechain dihedral angels to determine the full atom structure) inspired by previous works [67, 77, 9]. As for the dataset, we employ the Apobind dataset [3] with additional Apo data generated by OpenMM relaxation [21] and Rosetta repacking [18] as data augmentation. For efficient and stable generation, we adopt a flow-matching framework [49, 5] that defines multimodal conditional flows for different components in protein-ligand complex and use E(3)-equivariant network with scalar-vector dual representation for learning the chemical and geometric information. The input to FlexSBDD is the initialized protein structure (e.g., apo structure). FlexSBDD learns to iteratively update protein-ligand structures and ligand atom types from $time = 0$ to 1 and finally outputs both the generated 3D ligand molecule and the updated protein structure (holo). Extensive evaluations on benchmarks and case studies show the advantage of FlexSBDD in generating structurally valid protein-ligand complexes with high affinity, more favorable non-covalent interactions, and fewer steric clashes. The code of the paper is provided at `https://github.com/zaixizhang/FlexSBDD`. We highlight our main contributions as follows:

- We propose a flow-matching-based generative model FlexSBDD, capable of modeling protein flexibility while generating *de novo* 3D ligand molecules.
- FlexSBDD not only achieves state-of-the-art performance on benchmark datasets (e.g., -9.12 Avg. Vina Dock score), but also learns to adjust the protein structure to increase favorable interactions (e.g., 1.96 Avg. Hydrogen bond acceptors) and decrease steric clashes.
- With a concrete case study on $\text{KRAS}^{\text{G12C}}$, a promising target of solid tumor, we demonstrate FlexSBDD's potential to discover cryptic pockets for drug discovery.

2 Related Works

2.1 Structure-Based Drug Design

Structure-based drug design (SBDD) [87] aims to directly generate 3D ligand molecules inside target protein pockets. LiGAN [62] first uses 3D CNN to encode the protein-ligand structures and generate ligands by atom fitting and bond inference from the predicted atom densities. Several follow-up works have adopted autoregressive models for atom-wise [53, 50, 57, 79, 78] or fragment-wise [27, 60, 85, 78] generation of 3D molecules. For example, Pocket2Mol [57] adopts the geometric vector perceptrons [40] as the context encoder and autoregressively predicts the atom types, atom coordinates, and bond types until the generated molecule is completed. FLAG [85] and DrugGPS [82] predict the next molecular fragment and add it to the partially generated molecule in each round. Recently, powerful diffusion models have started to make a significant impact in SBDD, demonstrating promising results with non-autoregressive sampling [65, 28, 29, 46]. They usually represent the protein-ligand complex as 3D atom point sets, and define diffusion and denoising processes for both continuous atom coordinates and discrete atom types. For instance, TargetDiff [28] proposes a target-aware molecular diffusion process with a SE(3)-equivariant GNN denoiser. While there has been significant progress, existing approaches often neglect the critical aspect of protein flexibility. To address this gap, we explicitly model the conformational change of protein in FlexSBDD.

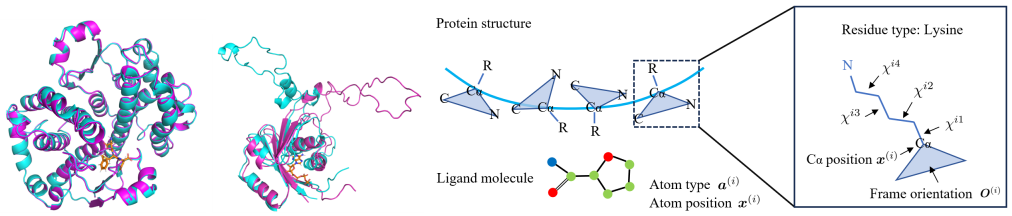


Figure 1: Aligned apo (ligand-free) and holo (ligand-bound) examples of (a) Human Glutathione S-Transferase protein (PDB ID: 10GS) and (b) Human Menkes protein (PDB ID: 2KMX) [3]. Apo structures are colored in "magenta" and holo-structures in "cyan" with the bounded ligand in "orange". (c) Illustration of the protein structure and ligand molecule parameterization.

2.2 Flexible Protein Modeling

Proteins are intrinsically dynamic entities and flexibility is a critical factor affecting protein's function and behavior in biological systems [72, 37]. The dynamic properties are traditionally modeled with physics-based methods such as Molecular Dynamics simulation (MD) [33]. To overcome the computationally demanding drawback of MD, some deep learning-based methods have been recently proposed, e.g., DynamicBind [51], FlexPose [20], SBAlign [68], NeuralPlexer [61], and DiffDock-Pocket [59] consider protein flexibility in protein-ligand docking. However, these methods can hardly extended to the challenging *de novo* ligand generation, leaving it an unsolved problem.

3 Preliminaries

Notations and Problem Formulation: We model protein-ligand complex as $\mathcal{C} = \{\mathcal{P}, \mathcal{G}\}$ [84, 86, 76, 83]. The objective of FlexSBDD is to learn a conditional generative model $p(\{\mathcal{P}', \mathcal{G}\}|\mathcal{P})$ that generates ligand molecule conditioned on the target protein and meanwhile updates the structure of the target protein ($\mathcal{P}'$). Proteins are composed of a sequence of residues (amino acids), each containing 4 backbone atoms (i.e., C_α, N, C, O) and a sidechain that identifies the residue type. In this paper, the residue types are assumed known and the full atom structure of an amino acid can be represented by its C_α coordinates $\boldsymbol{x}_i \in \mathbb{R}^3$, the frame orientation $\boldsymbol{O}^{(i)} \in SO(3)$, and maximally 4 sidechain dihedral angles $\boldsymbol{\chi}^{(i)} = \{\chi^{i1}, \chi^{i2}, \chi^{i3}, \chi^{i4}\} \in [0, 2\pi)^4$, where $i \in \{1, \cdots N_p\}$ and N_p is the number of residues in a protein. The backbone structure of the residue can be determined according to their ideal local coordinates relative to the C_α position $\boldsymbol{x}^{(i)}$ and the orientation $\boldsymbol{O}^{(i)}$ [23]. The sidechain conformations can be derived with the dihedral angles $\boldsymbol{\chi}^{(i)}$ as the bond length/angles are largely fixed [81]. With the above notations, a protein structure with N_p residues can be compactly represented as $\mathcal{P} = \{\boldsymbol{x}^{(i)}, \boldsymbol{O}^{(i)}, \boldsymbol{\chi}^{(i)}\}_{i=1}^{N_p}$ (see the illustration in Figure. 1(c)). Following previous works [74], we treat each amino acid as a node and integrate backbone orientation and sidechain dihedral angles as node features. Such method enjoys the advantage of fewer nodes and less computational cost. The generated ligand molecule can be represented as a set of atoms: $\mathcal{G} = \{\boldsymbol{a}^{(i)}, \boldsymbol{x}^{(i)}\}_{i=1}^{N_l}$, where $\boldsymbol{a}^{(i)} \in \mathbb{R}^{n_a}$ indicate the atom type (n_a is the total number of atom types we consider) and $\boldsymbol{x}^{(i)} \in \mathbb{R}^3$ denotes the atom coordinate. In the constructed protein-ligand complex 3D graph, we use $\boldsymbol{x}^{(i)}$ to denote both C_α coordinates and the ligand atom coordinates for conciseness.

Riemanian Flow Matching: Flow Matching (FM) [49, 5, 4], a simulation-free method for learning continuous normalizing flows (CNFs) [15], has shown better performance and efficiency than diffusion-based models on a series of biomolecular tasks [9, 77, 69]. To model the complicated protein-ligand complex structures, we need to apply the general flow matching on the Riemannian manifolds [14]. Let $\mathcal{M}$ be the manifold space with metric g. q is the probability distribution of data $x \in \mathcal{M}$, and p be the prior distribution. The time-dependant probability path on $\mathcal{M}$ is defined as $p_{t\in[0,1]} : \mathcal{M} \to \mathbb{R}_{>0}$ satisfying $p_0 = p$ and $p_1 = q$. $u_t(x) \in \mathcal{T}_x\mathcal{M}$ is the corresponding gradient vector of the path on x at time t. Flow Matching aims to learn a neural network $v_\theta(x, t)$ to approximate the target vector field u_t: $\mathcal{L}_{FM}(\theta) = \mathbb{E}_{t,p_t(x)}\|v_\theta(x, t) - u_t(x)\|_g^2$. However, u_t is intractable in practice and an alternative is to use a conditional density path $p_t(x|x_1)$ with a conditional gradient

https://doi.org/10.52202/079017-1707

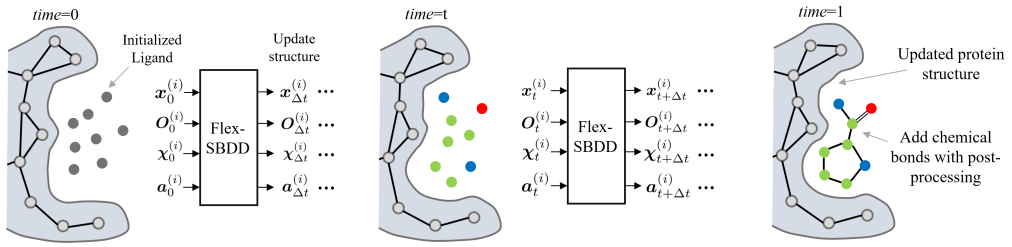


Figure 2: **Overview of FlexSBDD**. The flow matching-based generative process starts from an apo protein structure and the initialized ligand molecule. At each time step, FlexSBDD updates $\mathcal{C}_t$ to $\mathcal{C}_{t+\Delta t}$ and finally obtains the holo protein-ligand structure at $t = 1$. In the illustration, the gray dots indicate protein residues and the other dots indicate ligand atoms with different element types.

field $u_t(x|x_1)$ and use Conditional Flow Matching (CFM) objective as:

$$\mathcal{L}_{CFM}(\theta) = \mathbb{E}_{t,p_1(x_1),p_t(x|x_1)}\|v_\theta(x,t) - u_t(x|x_1)\|_g^2, \tag{1}$$

because $\mathcal{L}_{FM}$ and $\mathcal{L}_{CFM}$ have the same gradients according to previous works [14, 49]. t is sampled from the uniform distribution between 0 and 1. Once the gradient field v_θ is learned, we can integrate ordinary differential equations (ODE): $\frac{d}{dt}\phi_t(x) = v_\theta(\phi_t(x), t)$ with $\phi_0(x) = x, \phi_t(x) = x_t$ and ODE solvers [10] to push data from prior distribution p_0 to the data distribution p_1. Specifically, a data point from the prior distribution p_0 is $\mathcal{C}_0 = \{\mathcal{P}_0, \mathcal{G}_0\}$, where $\mathcal{P}_0$ is the initialized protein structure (e.g., the apo conformation) and $\mathcal{G}_0$ is the initialized ligand molecule. The number of ligand atoms is sampled from the reference dataset distribution; the atom types are initialized with uniform distributions; the atom coordinates are initialized with Gaussian distributions inside the protein pocket following [28]. The target data distribution p_1 is the holo protein-ligand complex $\mathcal{C}_1$ from datasets.

4 FlexSBDD

Figure. 2 shows the overview of FlexSBDD. Given the initialized target protein structure, the goal of FlexSBDD is to generate the binding ligand molecule as well as the adjusted protein structure. In other words, the output is the generated protein-ligand complex. Given the complexity of the protein-ligand system, we first introduce flow matching on the protein backbone (Sec. 4.1), side chain (Sec. 4.2), and ligand atom type (Sec. 4.3) respectively. Then we show E(3)-equivariant network in Sec. 4.4. Finally, we discuss the training and generation procedure of FlexSBDD (Sec. 4.5).

4.1 FlexSBDD on Protein Backbone and Ligand Coordinates

Following previous works [9, 41, 77], The backbone atom positions of each residue in a protein can be modeled as a rigid frame $T = (\boldsymbol{x}, \boldsymbol{O}) \in SE(3)$, consisting of C_α coordinate $\boldsymbol{x} \in \mathbb{R}^3$ (we use $\boldsymbol{x}$ to denote coordinates in the rest of this paper) and the frame orientation matrix $\boldsymbol{O} \in SO(3)$. For simplicity, the following deduction focuses on a single residue/frame and can be generalized to all the residues in the protein. The conditional flow of frame T_t is defined to be along the geodesic path connecting T_0 (apo frame) and T_1 (holo frame): $T_t = \exp_{T_0}(t \log_{T_0}(T_1))$, where $\exp_T$ represents the exponential map and $\log_T$ denotes the logarithmic map at T [77, 9]. Specifically, the conditional flow for the Euclidean coordinate vector $\boldsymbol{x}_t$ and the orientation matrix $\boldsymbol{O}_t$ are defined as:

$$\text{Coordinates}(\mathbb{R}^3)\colon \boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1 \tag{2}$$

$$\text{Orientations}(\text{SO}(3))\colon \boldsymbol{O}_t = \exp_{\boldsymbol{O}_0}(t \log_{\boldsymbol{O}_0}(\boldsymbol{O}_1)). \tag{3}$$

Both $\mathbb{R}^3$ and $\text{SO}(3)$ are simple manifolds and their closed-form geodesics can be derived. The exponential map $\exp_{\boldsymbol{O}_0}$ can be computed using Rodrigues' formula and the logarithmic map $\log_{\boldsymbol{O}_0}$ is similarly easy to compute with its Lie algebra $\mathfrak{so}(3)$ [77]. In protein-ligand complex, the ligand atom coordinates have the same data modality and probability path as C_α. The loss function of FlexSBDD

for protein backbone and ligand coordinates is the summation of the following two terms:

$$\mathcal{L}_{coord}(\theta) = \mathbb{E}_{t,p_1(\boldsymbol{x}_1),p_0(\boldsymbol{x}_0),p_t(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{x}_1)} \frac{1}{N_p+N_l} \sum_{i=1}^{N_p+N_l} \left\| v_\theta^{(i)}(\boldsymbol{x}_t^{(i)}, t) - \boldsymbol{x}_1^{(i)} + \boldsymbol{x}_0^{(i)} \right\|_2^2, \quad (4)$$

$$\mathcal{L}_{ori}(\theta) = \mathbb{E}_{t,p_1(\boldsymbol{O}_1),p_0(\boldsymbol{O}_0),p_t(\boldsymbol{O}_t|\boldsymbol{O}_0,\boldsymbol{O}_1)} \frac{1}{N_p} \sum_{i=1}^{N_p} \left\| v_\theta^{(i)}(\boldsymbol{O}_t^{(i)}, t) - \frac{\log_{\boldsymbol{O}_t^{(i)}}(\boldsymbol{O}_1^{(i)})}{1-t} \right\|_{\mathrm{SO}(3)}^2, \quad (5)$$

where the superscript (i) in $\boldsymbol{x}^{(i)}$ and $\boldsymbol{O}^{(i)}$ is the index of the residues/ligand atoms. For conciseness, we use Equ. 4 to represent the coordinate loss with respect to N_p α-Carbon and N_l ligand atoms.

4.2 FlexSBDD on Sidechain Torsion Angles

In FlexSBDD, we define sidechain torsion angles on the torus space to model the protein sidechain structures. There are maximally four sidechain dihedral angels for each residue i.e., $\boldsymbol{\chi}^{(i)} = \{\chi^{i1}, \chi^{i2}, \chi^{i3}, \chi^{i4}\} \in [0, 2\pi)^4$ and there are totally $4N_p$ torsion angles [81]. Since each torsion angle lies in $[0, 2\pi)$, the $4N_p$ torsion angles of sidechains define a hypertorus $\mathbb{T}^{4N_p}$. The manifold of the hypertorus is parameterized as the quotient space $\mathbb{R}^{4N_p}/2\pi\mathbb{Z}^{4N_p}$, leading to the equivalence relations $\boldsymbol{\chi} = (\chi^{(1)}, \ldots, \chi^{(4N_p)}) \sim (\chi^{(1)} + 2\pi, \ldots, \chi^{(4N_p)}) \sim (\chi^{(1)}, \ldots, \chi^{(4N_p)} + 2\pi)$ [39, 81]. We use the linear interpolation paths and the conditional flow for $\boldsymbol{\chi}$ is defined as:

$$\boldsymbol{\chi}_t = (1-t)\boldsymbol{\chi}_0 + t \cdot \mathrm{reg}(\boldsymbol{\chi}_1 - \boldsymbol{\chi}_0), \quad (6)$$

where $\mathrm{reg}(\cdot)$ means regularizing the torsion angles by $\mathrm{reg}(\boldsymbol{\chi}) = (\boldsymbol{\chi} + \pi) \mod (2\pi) - \pi$. This leads to the closed-from expression of the loss to train the conditional Torus Flow Matching:

$$\mathcal{L}_{sc}(\theta) = \mathbb{E}_{t,p_1(\boldsymbol{\chi}_1),p_0(\boldsymbol{\chi}_0),p_t(\boldsymbol{\chi}_t|\boldsymbol{\chi}_0,\boldsymbol{\chi}_1)} \frac{1}{N_p} \sum_{i=1}^{N_p} \left\| v_\theta^{(i)}(\boldsymbol{\chi}_t^{(i)}, t) - \mathrm{reg}(\boldsymbol{\chi}_1^{(i)} - \boldsymbol{\chi}_0^{(i)}) \right\|_2^2. \quad (7)$$

4.3 FlexSBDD on Ligand Atom Types

The ligand atom types are denoted as $\boldsymbol{a} = \{\boldsymbol{a}^{(i)}\}_{i=1}^{N_p}$ where $\boldsymbol{a}^{(i)}$ is the i-th atom probability vector with n_a dimensions: $\boldsymbol{a}^{(i)} \in \mathbb{R}^{n_a}$. To build a path, we define $\boldsymbol{a}_0$ as a uniform distribution over all atom types and $\boldsymbol{a}_1$ as the one-hot vector indicating the ground truth atom type. The probability path is define as $\boldsymbol{a}_t = t\boldsymbol{a}_1 + (1-t)\boldsymbol{a}_0$, and $u_t(\boldsymbol{a}|\boldsymbol{a}_0, \boldsymbol{a}_1) = \boldsymbol{a}_1 - \boldsymbol{a}_0$. $\boldsymbol{a}_t$ is a probability vector because its summation over all types equals 1. Following [47, 70, 12], we use Cross-Entropy loss $\mathrm{CE}(\cdot,\cdot)$ to directly measure the difference between the ground truth type and the predicted one:

$$\mathcal{L}_{atom}(\theta) = \mathbb{E}_{t\sim\mathcal{U}(0,1),p_1(\boldsymbol{a}_1),p_0(\boldsymbol{a}_0),p_t(\boldsymbol{a}|\boldsymbol{a}_0,\boldsymbol{a}_1)} \frac{1}{N_l} \sum_{n=1}^{N_l} \mathrm{CE}\left(\boldsymbol{a}_t^{(i)} + (1-t)v_\theta^{(i)}(\boldsymbol{a}_t^{(i)}, t), \boldsymbol{a}_1^{(i)}\right), \quad (8)$$

We also note the recent progress of sequential flow matching methods [71, 12], which can be seamlessly integrated into FlexSBDD and are left for future works.

4.4 Model Architecture

FlexSBDD is parameterized with an E(3)-Equivariant Neural Network with scalar-vector dual feature representation to effectively capture the 3D geometric attributes [40, 57]. The scalar features contain basic biochemical knowledge (e.g., residue/atom types), and the vector features contain geometric knowledge of the structure (e.g., direction to the geometric center). The basic building blocks include geometric vector linear (GVL) and geometric vector perceptron (GVP). We also incorporate the geometric vector normalization (GVNorm) and the geometric vector gate (GVGate) for the model's stability and better performance. There are mainly two modules: an encoder that is responsible for encoding the protein-ligand complex 3D graph (see details in Sec. D.2) and a decoder that updates both the coordinates, frame orientation, atom types, and side-chain torsion angles (see details in Sec. D.3). Similar to previous works [57], the update process satisfies the E(3)-equivariance. More model details are in the Appendix. D.5.

https://doi.org/10.52202/079017-1707

4.5 Training and Generation

Training with Data Augementation: For protein-ligand complexes from the training set, we associate them with apo conformations from Apobind [3] to create apo-holo pairs. Additionally, we create synthetic apo conformations as data augmentation. This is done by first removing the ligands from the holo proteins, followed by applying OpenMM [21] relaxation and Rosetta repacking [18] to these proteins. For each holo-structure, we generate a total of 9 additional structures: 3 only with sidechain repacking, 3 with both structure relaxation and repacking, and 3 with additional random perturbations with up to 30 degrees to the sidechain angles. In each training iteration, we randomly sample from the corresponding pool of apo structure $\mathcal{C}_0$ of the holo-structure $\mathcal{C}_1$ and interpolate to obtain $\mathcal{C}_t$. The overall loss function is the weighted summation of the above four loss functions:

$$\mathcal{L} = w_{\text{atom}}\mathcal{L}_{\text{atom}} + w_{\text{coord}}\mathcal{L}_{\text{coord}} + w_{\text{ori}}\mathcal{L}_{\text{ori}} + w_{\text{sc}}\mathcal{L}_{\text{sc}}, \tag{9}$$

where $w_{\text{atom}}, w_{\text{coord}}, w_{\text{ori}}$, and w_{sc} are the loss weights and are set to 2.0, 1.0, 1.0, and 1.0 in the default setting. We adopt Adam [42] optimizer for the optimization and finish training on a Tesla A100 GPU.

Generation: Starting with the apo structure and an initialized ligand molecule, denoted as $\mathcal{C}_0$, the generation process of FlexSBDD is the integration of the ODE $\frac{d\mathcal{C}_t}{dt} = v_\theta(\mathcal{C}_t, t)$ from $t = 0$ to $t = 1$ with an Euler solver [10]. Specifically, for each component in $\mathcal{C}_t$, we have:

$$\boldsymbol{x}_{t+\Delta t}^{(i)} = \boldsymbol{x}_t^{(i)} + v_\theta(\boldsymbol{x}_t^{(i)}, t)\Delta t; \quad \boldsymbol{O}_{t+\Delta t}^{(i)} = \boldsymbol{O}_t^{(i)} \exp\left(v_\theta(\boldsymbol{O}_t^{(i)}, t)\Delta t\right); \tag{10}$$

$$\boldsymbol{\chi}_{t+\Delta t}^{(i)} = \text{reg}\left(\boldsymbol{\chi}_t^{(i)} + v_\theta(\boldsymbol{\chi}_t^{(i)}, t)\Delta t\right); \quad \boldsymbol{a}_{t+\Delta t}^{(i)} = \text{norm}\left(\boldsymbol{a}_t^{(i)} + v_\theta(\boldsymbol{a}_t^{(i)}, t)\Delta t\right); \tag{11}$$

where Δt is the time step. $\text{norm}(\cdot)$ means normalizing the vector to a probability vector such that its summation is 1, and $\text{reg}(\cdot)$ means regularizing the angles by $\text{reg}(\tau) = (\tau + \pi) \mod (2\pi) - \pi$.

5 Experiments

5.1 Experimental Setup

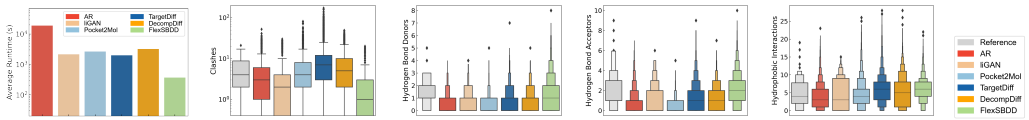
Datasets: Following previous works [65, 57], we use two popular benchmark datasets for experimental evaluations: CrossDocked and Binding MOAD. **Binding MOAD** dataset [34] contains around 41k experimentally determined protein-ligand complexes. We further filter and split the Binding MOAD dataset based on the proteins' enzyme commission number [7], resulting in 40k protein-ligand pairs for training, 100 pairs for validation, and 100 pairs for testing following previous work [65]. **CrossDocked** dataset [25] contains 22.5 million protein-molecule pairs generated through cross-docking. We use the same data preprocessing and splitting as [52], only high-quality docking poses (RMSD between the docked pose and the ground truth < 1Å) are kept and 30% sequence identity dataset split is adopted. This produces $100,000$ protein-ligand pairs for training and 100 proteins for testing. We regard the protein-ligand structures in the datasets as holo-structures. The corresponding apo structures are obtained from Apobind and the generated apo structures as described in Sec. 4.5. We note that it is fair to compare FlexSBDD with other baseline methods as the additional apo structures contain no ligand molecules and cannot be used by baselines for training.

Baseline Methods: We compare FlexSBDD with five representative methods for SBDD. **LiGAN** [62] is a conditional VAE model that represents protein-ligand complex as an atomic density grid. **AR** [52] and **Pocket2Mol** [57] are autoregressive schemes that generate 3D ligand molecules atom-by-atom. **TargetDiff** [28] and **DecompDiff** [29] are state-of-the-art diffusion-based models.

Evaluation: We comprehensively evaluate the generated molecules from three perspectives: **binding affinity and molecular properties**, **molecular structures**, and **protein-ligand interactions**: (1) Following previous work [28, 29], we use AutoDock Vina [22] to calculate and report the mean and median of binding affinity-related metrics, including **Vina Score**, **Vina Min**, **Vina Dock**, and **High Affinity**. Vina Score directly measures the binding affinity based on the generated 3D molecules; Vina Min performs a local structure relaxation before calculation; Vina Dock includes an extra step of re-docking, which serves to reveal the optimal binding affinity achievable; High affinity measures the percentage of generated molecules with higher binding affinity than the reference molecule in the

Methods	Vina Score (↓)		Vina Min (↓)		Vina Dock (↓)		High Affinity (↑)		QED (↑)		SA (↑)		Diversity (↑)	
	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.
Reference	-6.36	-6.46	-6.71	-6.49	-7.45	-7.26	-	-	0.48	0.47	0.73	0.74	-	-
LiGAN	-	-	-	-	-6.33	-6.20	21.1%	11.1%	0.39	0.39	0.59	0.57	0.66	0.67
AR	-5.75	-5.64	-6.18	-5.88	-6.75	-6.62	37.9%	31.0%	0.51	0.50	0.63	0.63	0.70	0.70
Pocket2Mol	-5.14	-4.70	-6.42	-5.82	-7.15	-6.79	48.4%	51.0%	0.56	0.57	**0.74**	**0.75**	0.69	0.71
TargetDiff	-5.47	-6.30	-6.64	-6.86	-7.80	-7.91	58.1%	59.1%	0.48	0.48	0.58	0.58	0.72	0.71
DecompDiff	-5.67	-6.04	-7.04	-6.91	-8.39	-8.43	64.4%	71.0%	0.45	0.43	0.61	0.60	0.68	0.68
FlexSBDD	**-6.64**	**-7.25**	**-8.27**	**-8.46**	**-9.12**	**-9.25**	**78.5%**	**84.2%**	**0.58**	**0.59**	0.69	0.73	**0.76**	**0.75**

Table 1: Overview of properties of the reference dataset and the molecules generated by different methods on the **CrossDocked** dataset. (↑) / (↓) denotes the larger/smaller, the better. The best results are marked with **bold** and the runner-up with underline.



(a) Runtime (b) Clashes (c) HB Donors (d) HB Acceptors (e) Hydrophobic

Figure 3: Computational Efficiency and Interaction Analysis on CrossDocked. (a) The average time required by different methods to generate 100 ligand molecules for a protein target. (b) The number of steric clashes. (c) The number of hydrogen bond donors in the ligand molecules. (d) The number of hydrogen bond acceptors in the ligand molecules. (e) The number of hydrophobic interactions.

test set. As for the other molecular properties, we consider **QED**, **SA**, and **Diversity**. QED measures how likely a molecule is a potential drug candidate; SA (synthesize accessibility) represents the difficulty of drug synthesis; Diversity is computed as the average pairwise dissimilarity between all generated molecules for a binding pocket. (2) In terms of molecular structures, we calculate the Jensen-Shannon divergences (JSD) in bond length/angle distributions between the reference molecules and the generated molecules following [29]. (3) We adopt PoseCheck [30] to evaluate whether methods can establish favorable interaction between protein and ligand. The interactions include **Hydrogen bonds** and **Hydrophobic interactions**. We also conduct **Steric clashes** analysis to examine unphysical structures/interactions.

5.2 Main Results

In Table. 1 and 4, we show the binding affinity and the drug-related properties of the generated molecules on two benchmarks. We can observe that our FlexSBDD outperforms baselines by a large margin in affinity-related metrics. For example, FlexSBDD surpasses the strongest baseline DecompDiff by 0.73 and 0.82 in Avg. and Med. Vina Dock on CrossDokecd and 0.92 and 1.03 respectively on Binding MOAD. These gains indicate the strong capability of FlexSBDD to explore high-affinity drug molecules and adjust protein structures for tight binding. As for molecular properties QED and SA, FlexSBDD also achieves competitive performance with baselines. As discussed in [29], these properties are usually employed as preliminary screening criteria in real drug discovery scenarios as long as they fall into a reasonable range. Finally, the high diversity indicates that FlexSBDD can explore larger chemical space with flexible protein modeling, which is important for early drug discovery. Generation efficiency is also a key factor to consider when sampling a large batch of molecules for screening. A major drawback of widely-used diffusion models is their inference speed, which may require 1000 time steps to produce high-quality samples. In contrast, flow matching methods remove stochasticity from the sampling path and can achieve stable and high-quality generation with much fewer steps (e.g., 20 steps in FlexSBDD). In Figure. 3, we observe that FlexSBDD can generate molecules much more efficiently than autoregressive-based methods such as AR [52] and diffusion-based methods such as TargetDiff [28] and DecompDiff [29].

We further consider steric clashes, hydrogen bonds, and hydrophobic interactions. **Steric Clashes** happens when two neutral atoms come into closer proximity than the combined extent of their van der Waals radii [63], indicating energetically unfavorable and physically unrealistic structures. **Hydrogen bonds (HBs)** [58] and **Hydrophobic interactions** are polar interactions that significantly contribute to the binding affinity between proteins and ligands (More details in Appendix A.1). In Figure. 3, we show the average number of steric clashes, hydrogen bond donors, acceptors, and hydrophobic

https://doi.org/10.52202/079017-1707

interactions in the generated ligands (without redocking). We observe that FlexSBDD can generate ligands introducing fewer clashes and more favorable interactions. For example, the average steric clashes for DecompDiff and FlexSBDD are 6.43 and 1.39 respectively. The average number of HB Acceptors for DecompDiff and FlexSBDD are 1.18 and 1.96 respectively. This could be attributed to the flexible protein adjustment capability of FlexSBDD, which could adaptively adjust protein and ligand conformations to reduce clashes and increase favorable protein-ligand interactions.

In Figure. 4, we show examples of the generated ligand molecules for target proteins. Especially, We colored the original holo protein structure green and the updated structure with FlexSBDD cyan for comparison. Firstly, we observe FlexSBDD can generate ligand molecules with higher affinity and comparable QED and SA compared with reference complexes from datasets and molecules generated by DecompDiff. Moreover, the protein structures of FlexSBDD are adjusted to accommodate the generated ligand molecules. Consistent with the prior knowledge in biology [8], we generally observe that the loop regions in protein structures exhibit greater flexibility, whereas the alpha-helix and beta-sheet regions display more rigidity. To further evaluate the validity of the updated protein structure, we employ self-consistency Template Modeling (scTM) following [48] (more details in Appendix A.2). scTM score ranges from 0 to 1 and a larger scTM score indicates better structural validity. On average, the updated protein structures by FlexSBDD have a scTM score of 0.964, comparable to the score of the original structures from the datasets (0.975). Moreover, to evaluate the validity of sidechain structure, we compute the Mean Absolute Error (MAE) of sidechain angles following previous works [81]. In Table. 6, we observe FlexSBDD achieves better performance in sidechain structure prediction. These results indicate FlexSBDD has learned the protein flexible changes and maintains the structural validity. More results are included in the Appendix B.

5.3 Sub-structure analysis

We further conduct sub-structure analysis to evaluate whether FlexSBDD can generate valid molecular conformations. In Table. 2 and Table. 5 in Appendix B, we compute different bond distance and bond angle distributions of the generated molecules and compare them against the corresponding reference empirical distributions following [28, 29]. We can observe that our model has a comparable or better performance on all the bond distances and angles, demonstrating the strong capability of FlexSBDD to generate realistic 3D molecules directly.

Bond	liGAN	AR	Pocket2 Mol	Target Diff	Decomp Diff	Flex SBDD
C−C	0.601	0.609	0.496	0.369	**0.359**	0.367
C=C	0.665	0.620	0.561	0.505	0.537	**0.280**
C−N	0.634	0.474	0.416	0.363	0.344	**0.277**
C=N	0.749	0.635	0.629	0.550	0.584	**0.384**
C−O	0.656	0.492	0.454	0.421	0.376	**0.253**
C=O	0.661	0.558	0.516	0.461	0.374	**0.245**
C:C	0.497	0.451	0.416	0.263	0.251	**0.193**
C:N	0.638	0.552	0.487	0.235	0.269	**0.189**

Table 2: Jensen-Shannon divergence between bond distance distributions of reference and generated ligands (the lower, the better). "-", "=", and ":" denote single, double, and aromatic bonds.

5.4 Rediscover Cryptic Pockets with FlexSBDD

The dynamic nature of proteins frequently results in the formation of *cryptic pockets*, which can reveal novel druggable sites not found in static structures and make previously "undruggable" proteins into potential drug targets [55]. To study the capability of FlexSBDD to explore cryptic pockets, we take $KRAS^{G12C}$ for a case study, which is a promising target in the treatment of solid tumors, and over 3 decades of efforts have been devoted to discovering its inhibitors (drug molecules) [17, 32]. The binding mode of ARS-1620 (green, PDB id 5V9U) represents the typical binding pocket exploited by previous research, which limits the exploration of high-affinity inhibitors. Here, we take the protein structure of ARS-1620 as the apo structure and generate ligand molecules. By comparing and filtering the generated molecules according to recent literature [45], we managed to rediscover the cryptic pockets with FlexSBDD. In Figure. 5, the updated structure is colored cyan and the generated ligand molecule is colored orange. We observe that the side chain rotation of residue Histidine-95 (marked with the red rectangle) is consistent with the report in [45] (PDB id 6P8Z), which forms a new subpocket and contributes a lot to binding affinity. This case

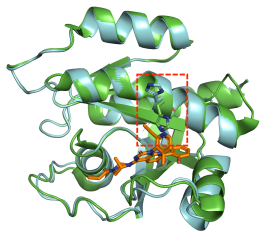


Figure 5: The predicted side chain rotation of residue H95 (marked with the red rectangle) is consistent with experimental observation [45]

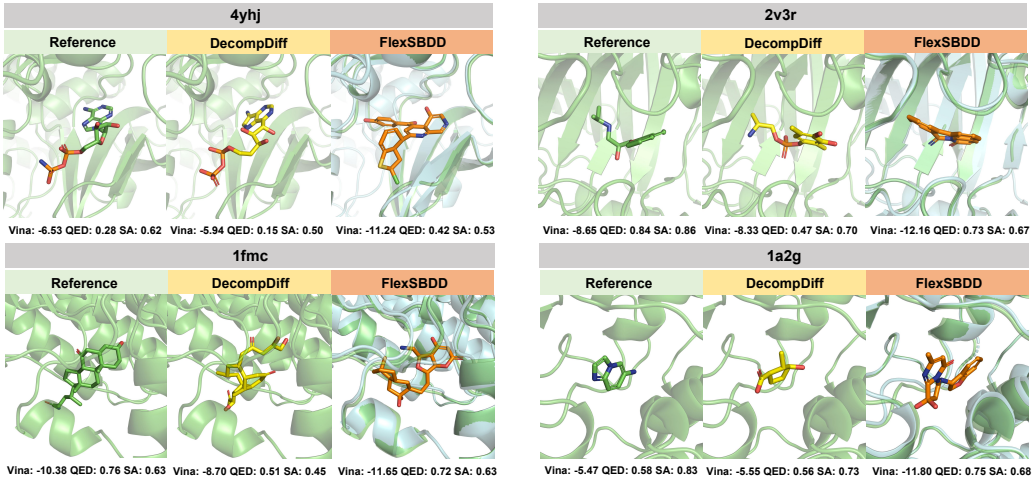


Figure 4: Examples of the generated ligand molecules for target proteins (PDB ID: 4yhj, 2v3r, 1fmc, 1a2g). We colored the original holo-structure from the dataset green and the updated protein structure with FlexSBDD cyan (structures are aligned). The Carbon atoms in Reference, DecompDiff, and FlexSBDD ligands are colored green, yellow, and orange. Vina Score, QED, and SA are reported.

study demonstrates FlexSBDD's capability to accurately model flexible protein structure, update sidechains to reduce steric clashes, and explore cryptic pockets for drug discovery.

5.5 Ablation Studies

We conduct a series of ablation experiments to study the effect of different modules on the generation capability of FlexSBDD: (1) **Exp0**: In model training, we remove the data augmentation mentioned in Sec. 5.1, (2) **Exp1**: we replace the geometric vector modules with EGNN [64] adopted in [65], which only has scalar features without vector features, (3) **Exp2**: we do not update the backbone structure of the protein (i.e., $\boldsymbol{x}, \boldsymbol{O}$) (4) **Exp3**: we do not update the sidechain dihedral angles of the protein (i.e., χ), (5) **Exp4**: we fix the whole protein structure in ligand molecule generation. We retrain all the FlexSBDD variant models for comparison. The results are present in Table. 3.

Methods	Vina Score (↓)		Vina Min (↓)		Vina Dock (↓)		QED (↑)	
	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.
Exp0	-6.12	-6.50	-7.69	-7.83	-8.76	-8.98	0.54	0.56
Exp1	-6.50	-6.85	-7.91	-8.10	-8.95	-9.03	0.55	<u>0.58</u>
Exp2	<u>-6.57</u>	<u>-7.19</u>	<u>-8.14</u>	<u>-8.31</u>	<u>-9.05</u>	<u>-9.20</u>	0.56	0.57
Exp3	-6.45	-7.03	-7.94	-8.07	-8.86	-8.92	0.56	0.55
Exp4	-6.32	-6.88	-7.80	-7.91	-8.82	-8.85	<u>0.57</u>	0.55
FlexSBDD	**-6.64**	**-7.25**	**-8.27**	**-8.46**	**-9.12**	**-9.25**	**0.58**	**0.59**

Table 3: Effect of different modules on the generation performance of FlexSBDD. The best results are marked with **bold** and the runner-up with <u>underline</u>. The original FlexSBDD is incorporated for comparison.

By comparing results from Exp0 and FlexSBDD, we can find that data augmentation indeed helps boost performance by introducing more diverse apo structures. In comparing Exp1 with FlexSBDD, it is obvious that scalar-vector dual feature representation can benefit ligand molecule generation by well capturing geometrical features. When comparing EXP2, 3, and 4 with FlexSBDD, we observe that the modeling of flexible protein structures including backbone and sidechains is important for FlexSBDD. Specifically, we find modeling the flexibility of sidechain angles is more important than backbone structure as the backbone is more rigid. For example, the average Vina Dock drops to -8.86 for Exp3 (FlexSBDD w/o flexible sidechain) while only drops to -9.05 for Exp2 (FlexSBDD w/o flexible backbone). According to [43], the sidechains are critical to "induced fit", where they adjust positions to accommodate the ligand and enhance binding affinity. Overall, the FlexSBDD variants still demonstrate competitive performance, showing the advantage of flow-matching architecture.

5.6 Hyperparameter Analysis

We investigate the influence of two important hyperparameters on the performance of FlexSBDD, the hidden dimension size and the total number of iteration steps T in flow matching. In Figure. 6, we

https://doi.org/10.52202/079017-1707

observe the trend of generating higher-quality molecules with larger hidden dimension sizes and more iteration steps. In the default setting, we set the node scaler feature size to 256 and total iteration steps to 20 to achieve a balance between the computational complexity and the generation quality.

6 Conclusion

In this paper, we propose FlexSBDD, a deep generative model capable of modeling the flexible protein structure for ligand molecule generation. FlexSBDD adopts a flow matching framework for efficient ligand generation and leverages E(3)-equivariant network with scalar-vector dual feature representation to effectively model dynamic structural changes. Extensive experiments show its state-of-the-art performance in generating high-affinity molecules with less steric clashes and more favorable interactions. Potential future works include leveraging FlexSBDD to discover more cryptic pockets and modeling other functional proteins such as antibodies, peptides, and enzymes.

7 Acknowledgements

This research was supported by grants from the National Natural Science Foundation of China (Grant No. 623B2095) and the Fundamental Research Funds for the Central Universities.

References

[1] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 2024.

[2] Yusuf O Adeshina, Eric J Deeds, and John Karanicolas. Machine learning classification can reduce false positives in structure-based virtual screening. *Proceedings of the National Academy of Sciences*, 117(31):18477–18488, 2020.

[3] Rishal Aggarwal, Akash Gupta, and U Priyakumar. Apobind: a dataset of ligand unbound protein conformations for machine learning applications in de novo drug design. *arXiv preprint arXiv:2108.09926*, 2021.

[4] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[5] Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2022.

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[7] Amos Bairoch. The enzyme database in 2000. *Nucleic acids research*, 28(1):304–305, 2000.

[8] Amélie Barozet, Pablo Chacón, and Juan Cortés. Current approaches to flexible loop modeling. *Current research in structural biology*, 3:187–191, 2021.

[9] Avishek Joey Bose, Tara Akhound-Sadegh, Kilian Fatras, Guillaume Huguet, Jarrid Rector-Brooks, Cheng-Hao Liu, Andrei Cristian Nica, Maksym Korablyov, Michael Bronstein, and Alexander Tong. Se (3)-stochastic flow matching for protein backbone generation. *arXiv preprint arXiv:2310.02391*, 2023.

[10] Kathryn Eleda Brenan, Stephen L Campbell, and Linda Ruth Petzold. *Numerical solution of initial-value problems in differential-algebraic equations*. SIAM, 1995.

[11] ID Brown. On the geometry of o–h… o hydrogen bonds. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(1):24–31, 1976.

[12] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *ICML*, 2024.

[13] Deliang Chen, Numan Oezguen, Petri Urvil, Colin Ferguson, Sara M Dann, and Tor C Savidge. Regulation of protein-ligand binding affinity by hydrogen bond pairing. *Science advances*, 2(3):e1501240, 2016.

[14] Ricky TQ Chen and Yaron Lipman. Riemannian flow matching on general geometries. *arXiv preprint arXiv:2302.03660*, 2023.

[15] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[16] Jordan J Clark, Mark L Benson, Richard D Smith, and Heather A Carlson. Inherent versus induced protein flexibility: comparisons within and between apo and holo structures. *PLoS computational biology*, 15(1):e1006705, 2019.

[17] Adrienne D Cox, Stephen W Fesik, Alec C Kimmelman, Ji Luo, and Channing J Der. Drugging the undruggable ras: Mission possible? *Nature reviews Drug discovery*, 13(11):828–851, 2014.

[18] Rhiju Das and David Baker. Macromolecular modeling with rosetta. *Annu. Rev. Biochem.*, 77:363–382, 2008.

[19] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

[20] Tiejun Dong, Ziduo Yang, Jun Zhou, and Calvin Yu-Chian Chen. Equivariant flexible modeling of the protein–ligand binding pose with geometric deep learning. *Journal of Chemical Theory and Computation*, 19(22):8446–8459, 2023.

[21] Peter Eastman and Vijay Pande. Openmm: A hardware-independent framework for molecular simulations. *Computing in science & engineering*, 12(4):34–39, 2010.

[22] Jerome Eberhardt, Diogo Santos-Martins, Andreas F Tillack, and Stefano Forli. Autodock vina 1.2. 0: New docking methods, expanded force field, and python bindings. *Journal of Chemical Information and Modeling*, 61(8):3891–3898, 2021.

[23] RA Engh and R Huber. Structure quality and target parameters. 2012.

[24] Wei Feng, Lvwei Wang, Zaiyun Lin, Yanhao Zhu, Han Wang, Jianqiang Dong, Rong Bai, Huting Wang, Jielong Zhou, Wei Peng, et al. Generation of 3d molecules in pockets via a language model. *Nature Machine Intelligence*, pages 1–12, 2024.

[25] Paul G Francoeur, Tomohide Masuda, Jocelyn Sunseri, Andrew Jia, Richard B Iovanisci, Ian Snyder, and David R Koes. Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design. *Journal of chemical information and modeling*, 60(9):4200–4215, 2020.

[26] Richard A Friesner, Jay L Banks, Robert B Murphy, Thomas A Halgren, Jasna J Klicic, Daniel T Mainz, Matthew P Repasky, Eric H Knoll, Mee Shelley, Jason K Perry, et al. Glide: a new approach for rapid, accurate docking and scoring. 1. method and assessment of docking accuracy. *Journal of medicinal chemistry*, 47(7):1739–1749, 2004.

[27] Harrison Green, David R Koes, and Jacob D Durrant. Deepfrag: a deep convolutional neural network for fragment-based lead optimization. *Chemical Science*, 12(23):8036–8047, 2021.

[28] Jiaqi Guan, Wesley Wei Qian, Xingang Peng, Yufeng Su, Jian Peng, and Jianzhu Ma. 3d equivariant diffusion for target-aware molecule generation and affinity prediction. *ICLR*, 2023.

[29] Jiaqi Guan, Xiangxin Zhou, Yuwei Yang, Yu Bao, Jian Peng, Jianzhu Ma, Qiang Liu, Liang Wang, and Quanquan Gu. Decompdiff: Diffusion models with decomposed priors for structure-based drug design. *ICML*, 2023.

[30] Charles Harris, Kieran Didi, Arian Jamasb, Chaitanya Joshi, Simon Mathis, Pietro Lio, and Tom Blundell. Posecheck: Generative models for 3d structure-based drug design produce unrealistic poses. In *NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development*, 2023.

[31] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[32] Matthew Holderfield. Efforts to develop kras inhibitors. *Cold Spring Harbor Perspectives in Medicine*, 8(7):a031864, 2018.

[33] Scott A Hollingsworth and Ron O Dror. Molecular dynamics simulation for all. *Neuron*, 99(6):1129–1143, 2018.

 https://doi.org/10.52202/079017-1707

[34] Liegi Hu, Mark L Benson, Richard D Smith, Michael G Lerner, and Heather A Carlson. Binding moad (mother of all databases). *Proteins: Structure, Function, and Bioinformatics*, 60(3):333–340, 2005.

[35] John B Ingraham, Max Baranov, Zak Costello, Karl W Barber, Wujie Wang, Ahmed Ismail, Vincent Frappier, Dana M Lord, Christopher Ng-Thow-Hing, Erik R Van Vlack, et al. Illuminating protein space with a programmable generative model. *Nature*, pages 1–9, 2023.

[36] Clemens Isert, Kenneth Atz, and Gisbert Schneider. Structure-based drug design with geometric deep learning. *Current Opinion in Structural Biology*, 79:102548, 2023.

[37] Giacomo Janson, Gilberto Valdes-Garcia, Lim Heo, and Michael Feig. Direct generation of protein conformational ensembles via machine learning. *Nature Communications*, 14(1):774, 2023.

[38] Yan-Bin Jia. Quaternions and rotations. *Com S*, 477(577):15, 2008.

[39] Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. Torsional diffusion for molecular conformer generation. *NeurIPS*, 2022.

[40] Bowen Jing, Stephan Eismann, Pratham N Soni, and Ron O Dror. Equivariant graph neural networks for 3d macromolecular structure. *ICML*, 2021.

[41] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[42] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[43] Daniel E Koshland Jr. The key–lock theory and the induced fit theory. *Angewandte Chemie International Edition in English*, 33(23-24):2375–2378, 1995.

[44] Rohith Krishna, Jue Wang, Woody Ahern, Pascal Sturmfels, Preetham Venkatesh, Indrek Kalvet, Gyu Rie Lee, Felix S Morey-Burrows, Ivan Anishchenko, Ian R Humphreys, et al. Generalized biomolecular modeling and design with rosettafold all-atom. *Science*, 384(6693):eadl2528, 2024.

[45] Brian A Lanman, Jennifer R Allen, John G Allen, Albert K Amegadzie, Kate S Ashton, Shon K Booker, Jian Jeffrey Chen, Ning Chen, Michael J Frohn, Guy Goodman, et al. Discovery of a covalent inhibitor of krasg12c (amg 510) for the treatment of solid tumors, 2019.

[46] Haitao Lin, Yufei Huang, Haotian Zhang, Lirong Wu, Siyuan Li, Zhiyuan Chen, and Stan Z Li. Functional-group-based diffusion for pocket-specific molecule generation and elaboration. *NeurIPS*, 2023.

[47] Haitao Lin, Odin Zhang, Huifeng Zhao, Lirong Wu, Dejun Jiang, Zicheng Liu, Yufei Huang, and Stan Z Li. Ppflow: Target-aware peptide design with torsional flow matching. *ICML*, 2024.

[48] Yeqing Lin and Mohammed AlQuraishi. Generating novel, designable, and diverse protein structures by equivariantly diffusing oriented residue clouds. *ICML*, 2023.

[49] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[50] Meng Liu, Youzhi Luo, Kanji Uchino, Koji Maruhashi, and Shuiwang Ji. Generating 3d molecules for target protein binding. *ICML*, 2022.

[51] Wei Lu, Ji-Xian Zhang, Weifeng Huang, Ziqiao Zhang, Xiangyu Jia, Zhenyu Wang, Leilei Shi, Chengtao Li, Peter Wolynes, and Shuangjia Zheng. Dynamicbind: Predicting ligand-specific protein-ligand complex structure with a deep equivariant generative model. 2023.

[52] Shitong Luo, Jiaqi Guan, Jianzhu Ma, and Jian Peng. A 3d generative model for structure-based drug design. *NeurIPS*, 34:6229–6239, 2021.

[53] Youzhi Luo and Shuiwang Ji. An autoregressive flow model for 3d molecular geometry generation from scratch. In *ICLR*, 2021.

[54] Dominic D Martinelli. Generative machine learning for de novo drug discovery: A systematic review. *Computers in Biology and Medicine*, 145:105403, 2022.

[55] Artur Meller, Soumendranath Bhakat, Shahlo Solieva, and Gregory R Bowman. Accelerating cryptic pocket discovery using alphafold. *Journal of Chemical Theory and Computation*, 19(14):4355–4363, 2023.

[56] Emily E Meyer, Kenneth J Rosenberg, and Jacob Israelachvili. Recent progress in understanding hydrophobic interactions. *Proceedings of the National Academy of Sciences*, 103(43):15739–15746, 2006.

[57] Xingang Peng, Shitong Luo, Jiaqi Guan, Qi Xie, Jian Peng, and Jianzhu Ma. Pocket2mol: Efficient molecular sampling based on 3d protein pockets. *ICML*, 2022.

[58] George C Pimentel and AL McClellan. Hydrogen bonding. *Annual Review of Physical Chemistry*, 22(1):347–385, 1971.

[59] Michael Plainer, Marcella Toth, Simon Dobers, Hannes Stark, Gabriele Corso, Céline Marquet, and Regina Barzilay. Diffdock-pocket: Diffusion for pocket-level docking with sidechain flexibility. In *NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development*, 2023.

[60] Alexander Powers, Helen Yu, Patricia Suriana, and Ron Dror. Fragment-based ligand generation guided by geometric deep learning on protein-ligand structure. *bioRxiv*, 2022.

[61] Zhuoran Qiao, Weili Nie, Arash Vahdat, Thomas F Miller III, and Animashree Anandkumar. State-specific protein–ligand complex structure prediction with a multiscale deep generative model. *Nature Machine Intelligence*, pages 1–14, 2024.

[62] Matthew Ragoza, Tomohide Masuda, and David Ryan Koes. Generating 3d molecules conditional on receptor binding sites with deep generative models. *Chemical science*, 13(9):2701–2713, 2022.

[63] Srinivas Ramachandran, Pradeep Kota, Feng Ding, and Nikolay V Dokholyan. Automated minimization of steric clashes in protein structures. *Proteins: Structure, Function, and Bioinformatics*, 79(1):261–270, 2011.

[64] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks, 2021.

[65] Arne Schneuing, Yuanqi Du, Charles Harris, Arian Jamasb, Ilia Igashov, Weitao Du, Tom Blundell, Pietro Lió, Carla Gomes, Max Welling, et al. Structure-based drug design with equivariant diffusion models. *arXiv preprint arXiv:2210.13695*, 2022.

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

[67] Chence Shi, Chuanrui Wang, Jiarui Lu, Bozitao Zhong, and Jian Tang. Protein sequence and structure co-design with equivariant translation. In *The Eleventh International Conference on Learning Representations*, 2022.

[68] Vignesh Ram Somnath, Matteo Pariset, Ya-Ping Hsieh, Maria Rodriguez Martinez, Andreas Krause, and Charlotte Bunne. Aligned diffusion schrödinger bridges. In *Uncertainty in Artificial Intelligence*, pages 1985–1995. PMLR, 2023.

[69] 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. *Advances in Neural Information Processing Systems*, 36, 2024.

[70] Hannes Stark, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. Harmonic self-conditioned flow matching for multi-ligand docking and binding site design. *arXiv preprint arXiv:2310.05764*, 2023.

[71] Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to dna sequence design. *arXiv preprint arXiv:2402.05841*, 2024.

[72] Sandhya P Tiwari and Nathalie Reuter. Conservation of intrinsic dynamics in proteins—what have computational models taught us? *Current Opinion in Structural Biology*, 50:75–81, 2018.

[73] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

 https://doi.org/10.52202/079017-1707

[74] Limei Wang, Haoran Liu, Yi Liu, Jerry Kurtin, and Shuiwang Ji. Learning hierarchical protein representations via complete 3d graph networks. In *The Eleventh International Conference on Learning Representations*, 2022.

[75] Ruidong Wu, Fan Ding, Rui Wang, Rui Shen, Xiwen Zhang, Shitong Luo, Chenpeng Su, Zuofan Wu, Qi Xie, Bonnie Berger, et al. High-resolution de novo structure prediction from primary sequence. *BioRxiv*, pages 2022–07, 2022.

[76] Jiaxian Yan, Zaixi Zhang, Jintao Zhu, Kai Zhang, Jianfeng Pei, and Qi Liu. Deltadock: A unified framework for accurate, efficient, and physically reliable molecular docking. *Advances in Neural Information Processing Systems*, 2024.

[77] Jason Yim, Andrew Campbell, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Regina Barzilay, Tommi Jaakkola, et al. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023.

[78] Odin Zhang, Tianyue Wang, Gaoqi Weng, Dejun Jiang, Ning Wang, Xiaorui Wang, Huifeng Zhao, Jialu Wu, Ercheng Wang, Guangyong Chen, et al. Learning on topological surface and geometric structure for 3d molecular generation. *Nature Computational Science*, 3(10):849–859, 2023.

[79] Odin Zhang, Jintu Zhang, Jieyu Jin, Xujun Zhang, RenLing Hu, Chao Shen, Hanqun Cao, Hongyan Du, Yu Kang, Yafeng Deng, et al. Resgen is a pocket-aware 3d molecular generation model based on parallel multiscale modelling. *Nature Machine Intelligence*, 5(9):1020–1030, 2023.

[80] Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 57(4):702–710, 2004.

[81] Yangtian Zhang, Zuobai Zhang, Bozitao Zhong, Sanchit Misra, and Jian Tang. Diffpack: A torsional diffusion model for autoregressive protein side-chain packing. *arXiv preprint arXiv:2306.01794*, 2023.

[82] Zaixi Zhang and Qi Liu. Learning subpocket prototypes for generalizable structure-based drug design. *ICML*, 2023.

[83] Zaixi Zhang, Qi Liu, Chee-Kong Lee, Chang-Yu Hsieh, and Enhong Chen. An equivariant generative framework for molecular graph-structure co-design. *Chemical Science*, 14(31):8380–8392, 2023.

[84] Zaixi Zhang, Zepu Lu, Hao Zhongkai, Marinka Zitnik, and Qi Liu. Full-atom protein pocket design via iterative refinement. *Advances in Neural Information Processing Systems*, 36:16816–16836, 2023.

[85] Zaixi Zhang, Yaosen Min, Shuxin Zheng, and Qi Liu. Molecule generation for target protein binding with structural motifs. In *The Eleventh International Conference on Learning Representations*, 2023.

[86] Zaixi Zhang, Wanxiang Shen, Qi Liu, and Marinka Zitnik. Pocketgen: Generating full-atom ligand-binding protein pockets. *bioRxiv*, pages 2024–02, 2024.

[87] Zaixi Zhang, Jiaxian Yan, Qi Liu, Enhong Chen, and Marinka Zitnik. A systematic survey in geometric deep learning for structure-based drug design. *arXiv preprint arXiv:2306.11768*, 2023.

A Data Analysis

A.1 Protein-ligand Interaction Analysis

We consider steric clashes, hydrogen bonds, and hydrophobic interactions in protein-ligand interaction analysis with PoseCheck [30]. **Steric Clashes** happens when two neutral atoms come into closer proximity than the combined extent of their van der Waals radii [63], indicating energetically unfavorable and physically unrealistic structures. In PoseCheck, a clash is counted when the pairwise distance between a protein and ligand atom falls below the sum of their van der Waals radii, allowing a clash tolerance of 0.5 Å. **Hydrogen bonds (HBs)** represent a form of molecular interaction where a hydrogen atom, covalently linked to an element of high electronegativity like nitrogen, oxygen, or fluorine, engages with another electronegative atom [58]. These bonds are crucial in numerous protein-ligand interactions [13] and necessitate precise geometric alignments to form [11]. HBs are directional, bestowing distinct roles on the atoms involved: the hydrogen covalently bonded to the electronegative atom acts as a "donor", while the atom that receives the HB is known as an "acceptor". **Hydrophobic interactions** are a type of non-covalent bonding that occurs among hydrophobic molecules or moieties within an aqueous setting. Driven by water's propensity to hydrogen bond with itself, these interactions result in the segregation of non-polar entities, compelling them to cluster together away from the water-rich environment [56]. This behavior significantly contributes to the binding affinity between proteins and ligands.

A.2 Protein Sturcture Analysis

Following [48], scTM takes a generated structure and feeds it into ProteinMPNN [19], a state-of-the-art structure-conditioned sequence generation method. With a sampling temperature of 0.1, we generate eight sequences per input structure and then use OmegaFold [75] to predict the structure of each putative sequence. We follow [48] that substitutes AlphaFold2 with OmegaFold for better sequence prediction performance. Finally, scTM is measured by computing the TM-score [80], a metric of structural congruence of the OmegaFold-predicted structure and the original updated structure by our FlexSBDD. scTM scores range from 0 to 1, with higher numbers corresponding to the increased likelihood that an input structure is designable (higher structural validity).

B More Results and Analysis

Table. 5, 7 and Figure. 6 show the additional results on substructure analysis, and hyperparameters analysis.

B.1 Benchmark Results on CrossDocked

In Table. 4, we show the additional results on the Binding MOAD dataset.

Methods	Vina Score (↓)		Vina Min (↓)		Vina Dock (↓)		High Affinity (↑)		QED (↑)		SA (↑)		Diversity (↑)	
	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.
Reference	-6.68	-6.66	-7.62	-7.47	-8.21	-8.16	-	-	0.60	0.58	0.33	0.34	-	-
LiGAN	-	-	-	-	-7.09	-6.96	18.9%	16.2%	0.37	0.38	0.37	0.40	0.68	0.69
AR	-6.19	-6.04	-6.86	-6.80	-7.65	-7.61	32.6%	31.9%	0.42	0.40	0.35	0.36	0.68	0.69
Pocket2Mol	-6.02	-5.97	-6.71	-6.80	-7.69	-7.74	36.9%	37.1%	0.60	0.59	0.34	0.35	0.70	0.72
TargetDiff	-6.13	-6.20	-6.85	-6.92	-7.95	-7.94	40.3%	39.7%	0.50	0.48	0.29	0.33	0.71	0.70
DecompDiff	-6.37	-6.41	-7.52	-7.31	-8.46	-8.51	56.4%	58.3%	0.60	0.61	0.32	0.34	0.68	0.66
FlexSBDD	**-7.04**	**-7.20**	**-8.36**	**-8.73**	**-9.38**	**-9.54**	**74.5%**	**76.9%**	**0.63**	**0.64**	**0.42**	**0.41**	**0.73**	**0.74**

Table 4: Overview of properties of the reference dataset and the molecules generated by different methods on **Binding MOAD** dataset. (↑) / (↓) denotes the larger/smaller, the better. The best results are marked with **bold** and the runner-up with underline.

B.2 Bond Angle Distributions

In Table. 5, we show the Jensen-Shannon divergence between bond angle distributions of the reference molecules and the generated molecules.

https://doi.org/10.52202/079017-1707

Bond	liGAN	AR	Pocket2 Mol	Target Diff	Decomp Diff	Flex SBDD
CCC	0.598	0.340	0.323	0.328	<u>0.314</u>	**0.285**
CCO	0.637	0.442	0.401	0.385	<u>0.324</u>	**0.316**
CNC	0.604	0.419	<u>0.237</u>	0.367	0.297	**0.226**
OPO	0.512	0.367	0.274	0.303	<u>0.217</u>	**0.210**
NCC	0.621	0.392	0.351	0.354	<u>0.294</u>	**0.283**
CC=O	0.636	0.476	0.353	0.356	**0.259**	<u>0.270</u>
COC	0.606	0.459	**0.317**	0.389	0.339	<u>0.320</u>

Table 5: Jensen-Shannon divergence between bond angle distributions of the reference molecules and the generated molecules, and lower values indicate better performances. We highlight the best two results with **bold text** and <u>underlined text</u>, respectively.

B.3 Validity of Side Chain Prediction

In FlexSBDD, the sidechain torsion angles are predicted and the sidechain conformations are derived based on the dihedral angles and the ideal bond length/angles. To evaluate the validity of sidechain structure, we compute the Mean Absolute Error (MAE) of sidechain angles (degrees) following previous works [81] in Table. 6. We also compare the results with NeuralPlexer [61], the state-of-the-art protein-ligand complex structure prediction. In the table, we report the average MAE and can observe that FlexSBDD achieves better performance in generating valid sidechain structures.

Method	χ_1	χ_2	χ_3	χ_4
NeuralPlexer	15.40	18.77	44.83	50.24
FlexSBDD	**12.95**	**17.80**	**32.18**	**46.71**

Table 6: The MAE of NeuralPlexer and FlexSBDD on sidechain torsion angles.

B.4 More Results on Abation Studies

In Table. 7, we show the average number of interactions in the ablation studies.

Methods	Steric Clashes (↓)	HB Donors (↑)	HB Acceptors (↑)	Hydrophobic (↑)
Exp0	1.56	<u>1.38</u>	1.74	4.79
Exp1	1.77	1.35	1.77	5.40
Exp2	<u>1.45</u>	1.23	<u>1.85</u>	<u>5.75</u>
Exp3	1.92	1.11	1.62	5.23
Exp4	1.90	1.09	1.58	5.29
FlexSBDD	**1.39**	**1.40**	**1.96**	**6.12**

Table 7: Effect of different modules on the generation performance of FlexSBDD. We show the average number of interactions here. We highlight the best two results with **bold text** and <u>underlined text</u>, respectively.

B.5 Evaluation of Binding Affinity with GlideSP

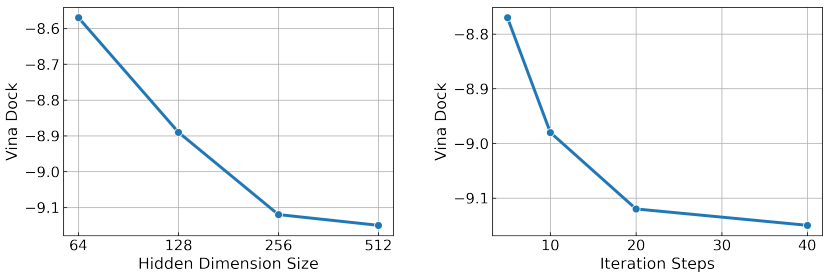
Besides Vina Scores, we also try to leverage other docking methods to evaluate the binding affinity [24]. In Table. 8, we further leverage Glide [26] to evaluate the generated ligand molecules, which demonstrates superior ability in filtering active compounds. Specifically, we calculate the min-in-place GlideSP score following [24], where the ligand structure undergoes force-field-based energy minimization within the receptor's field before scoring. In the table below, we observe that FlexSBDD can also achieve the best score on Glide, demonstrating its strong performance in generating protein-binding molecules.

Table 8: Comparison of Average Min-in-Place GlideSP Scores on CrossDocked.

Methods	Avg. min-in-place GlideSP score (↓)
Reference	-6.32
LiGAN	-6.14
AR	-6.20
Pocket2Mol	-6.71
TargetDiff	-6.86
DecompDiff	-7.09
FlexSBDD	**-7.55**

B.6 Hyperparameter Analysis

In Figure. 6, we show hyperparameter analysis for the hidden dimension size and the total steps of flow matching.



(a) (b)

Figure 6: Hyperparameters Analysis with respect to (a) the hidden dimension size (the node scaler features) and (b) the total iteration steps of flow matching. When varying the dimension size of the node scaler features, the other features are scaled proportionally.

C More Details of FlexSBDD Training and Generation

C.1 Hyperparameters settings

To construct the protein-ligand KNN graph, we set k as 8 (each node is connected to its nearest 8 neighbors). In the default setting, we use a hidden size of 256, 128, 128, and 64 for the scalar features of nodes, scalar features of edges, vector features of nodes, and vector features of edges, respectively. The Encoder and Decoder have 6 layers respectively with the number of attention heads set as 4. The number of integration steps in flow matching is 20 for FlexSBDD. The hyperparameters for the loss function: $w_{\text{atom}}, w_{\text{coord}}, w_{\text{ori}}$, and w_{sc} are selected based on grid search ($\{0.5, 1.0, 2.0, 3.0\}$). $w_{\text{atom}}, w_{\text{coord}}, w_{\text{ori}}$, and w_{sc} are set to 2.0, 1.0, 1.0, and 1.0 in the default setting. To train FlexSBDD, we use the Adam [42] as our optimizer with a learning rate of 0.001, $betas = (0.95, 0.999)$, and batch size 4 for 500k iterations. It takes around 36 hours on n one NVIDIA GeForce GTX A100 GPU to complete the training.

C.2 Pseudo Algorithms

We show the pseudo codes of FlexSBDD training and generation in Algorithm 1 and 2.

D More Details of Neural Network Architecture

D.1 Overview

A series of previous works on biochemistry tasks [40, 57, 79, 20] have shown that representing the nodes and edges in 3D graphs with scalar-vector dual features can greatly improve performance. In

https://doi.org/10.52202/079017-1707

Algorithm 1: Training algorithm of FlexSBDD

Input: Holo data distribution p_1, FlexSBDD model v_θ, Apobind and generated apo structures
while *Training* **do**
  $\mathcal{C}_1 \sim p_1; t \sim \mathcal{U}[0,1]$;
  Sample $\mathcal{C}_0$ from the corresponding apo structure pool (Apobind and data augmentation)
  $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1; \quad \boldsymbol{O}_t = \exp_{\boldsymbol{O}_0}(t \log_{\boldsymbol{O}_0}(\boldsymbol{O}_1))$
  $\boldsymbol{\chi}_t = (1-t)\boldsymbol{\chi}_0 + t \cdot \text{reg}(\boldsymbol{\chi}_1 - \boldsymbol{\chi}_0); \quad \boldsymbol{a}_t = t\boldsymbol{a}_1 + (1-t)\boldsymbol{a}_0; //$ Interpolation
  $\mathcal{L} \leftarrow w_{\text{atom}}\mathcal{L}_{\text{atom}} + w_{\text{coord}}\mathcal{L}_{\text{coord}} + w_{\text{ori}}\mathcal{L}_{\text{ori}} + w_{\text{sc}}\mathcal{L}_{\text{sc}} //$ calculate loss according to Equ. 9;
  $\theta \leftarrow \mathbf{Update}(\theta, \nabla_\theta \mathcal{L}_{FM})$;
return v_θ

Algorithm 2: Generation algorithm of FlexSBDD

Input: Total number of integration steps T, and trained model v_θ
$steps \leftarrow 0, t \leftarrow 0, \Delta t \leftarrow 1/T$;
Initialize $\mathcal{C}_0$ with the apo structure and sampled ligand atoms;
while $steps \leq T-1$ **do**
  $\boldsymbol{x}_{t+\Delta t}^{(i)} = \boldsymbol{x}_t^{(i)} + v_\theta(\boldsymbol{x}_t^{(i)}, t)\Delta t; \quad \boldsymbol{O}_{t+\Delta t}^{(i)} = \boldsymbol{O}_t^{(i)} \exp\left(v_\theta(\boldsymbol{O}_t^{(i)}, t)\Delta t\right);$
  $\boldsymbol{\chi}_{t+\Delta t}^{(i)} = \text{reg}(\boldsymbol{\chi}_t^{(i)} + v_\theta(\boldsymbol{\chi}_t^{(i)}, t)\Delta t); \quad \boldsymbol{a}_{t+\Delta t}^{(i)} = \text{norm}\left(\boldsymbol{a}_t^{(i)} + v_\theta(\boldsymbol{a}_t^{(i)}, t)\Delta t\right);$
  $t \leftarrow t + \Delta t$;
return $\mathcal{C}_1$

FlexSBDD, all nodes and edges in the target protein $\mathcal{P}$ and the generated molecules $\mathcal{G}$ are assigned with both scalar and vector features to better capture the 3D geometric information. The scalar features contain basic biochemical knowledge (e.g., residue/atom types), and the vector features contain geometric knowledge of the structure (e.g., direction to the geometric center). In the rest of this paper, we use "$\cdot$" and "$\rightarrow$" overheads to explicitly indicate scalar features and vector features (e.g., $\dot{\mathbf{v}}$ and $\vec{\mathbf{v}}$).

We adopt the geometric vector linear (GVL) and the geometric vector perceptron (GVP) as the main building blocks to enhance the information flows between the scalar features and the vector features and achieve E(3)-equivariance [40]. The details of GVP and GVL are shown in Appendix D.5. Briefly, they propagate the vector features into the scalar features by row-wise norm and propagate the scalar features to the vector features through gating. The GVP further applies extra non-linear transformations to both the scalar and vector features, following the output of GVL:

$$\begin{aligned} (\dot{\mathbf{v}}', \vec{\mathbf{v}}') &\leftarrow \text{GVL}(\dot{\mathbf{v}}, \vec{\mathbf{v}}), \\ (\dot{\mathbf{v}}', \vec{\mathbf{v}}') &\leftarrow \text{NonLinearTransform}(\dot{\mathbf{v}}', \vec{\mathbf{v}}'), \end{aligned} \tag{12}$$

where $(\dot{\mathbf{v}}, \vec{\mathbf{v}})$ could be any pair of scalar-vector features. Incorporating vector features is essential for our model's ability to directly and precisely update the positions of atoms and model the conformation change of the flexible protein. In our model, we also incorporate the geometric vector normalization (GVNorm) and the geometric vector gate (GVGate) for model's stability and better performance. Specifically, GVNorm combines the layer normalization [6] with the vector normalization [40]; GVGate performs skip connection and fuses features from different blocks.

D.2 Encoder

In FlexSBDD, we represent the protein pocket-ligand complex as a k-nearest neighbor (KNN) graph in which nodes represent protein residues or ligand atoms and each node is connected to its k-nearest neighbors. The input scalar features of residues are onehot embeddings of residue types and the input scalar features of ligand atoms are initialized with a uniform distribution over all atom types. The input vector features of residues are computed based on the coordinates of backbone and sidechain atoms, while the vector features of ligand atoms are the Euclidean vectors pointing to the geometric center of the ligand molecule (see Appendix D.4). The scalar edge features are the radial basis function (RBF) distance encodings [66] and the vector edge features are the relative coordinates. In

the flow matching model, to further incorporate time step information, we embed time with sinusoidal embedding [73] and concatenate it with the input scalar node features following [28].

Generally, the encoder of FlexSBDD follows the message-passing paradigm to update the features. Denote the feature of node i at the l-th layer as $(\dot{\mathbf{v}}_i^{(l)}, \vec{\mathbf{v}}_i^{(l)})$ and the edge feature between node i and j as $(\dot{\mathbf{e}}_{ij}^{(l)}, \vec{\mathbf{e}}_{ij}^{(l)})$ (we use subscript here for the index of nodes instead of time step in the main paper). Each layer consists of a message-passing module M_l and an update module U_l:

$$\begin{aligned} \mathbf{M}_v^{(l)}, \mathbf{M}_e^{(l)} &= M_l(\dot{\mathbf{v}}_j^{(l-1)}, \vec{\mathbf{v}}_j^{(l-1)}, \dot{\mathbf{e}}_{ij}^{(l-1)}, \vec{\mathbf{e}}_{ij}^{(l-1)}), \\ (\dot{\mathbf{v}}_i^{(l)}, \vec{\mathbf{v}}_i^{(l)}), (\dot{\mathbf{e}}_{ij}^{(l)}, \vec{\mathbf{e}}_{ij}^{(l)}) &= U_l(\dot{\mathbf{v}}_i^{(l-1)}, \vec{\mathbf{v}}_i^{(l-1)}, \mathbf{M}_v^{(l)}, \mathbf{M}_e^{(l)}), \end{aligned} \tag{13}$$

where we use $\mathbf{M}_v^{(l)} = (\dot{\mathbf{m}}_i^{(l)}, \vec{\mathbf{m}}_i^{(l)})$ and $\mathbf{M}_e^{(l)} = (\dot{\mathbf{m}}_{ij}^{(l)}, \vec{\mathbf{m}}_{ij}^{(l)})$ to denote the calculated messages for node i and the edge between node i and j. The message-passing module is based on an attention mechanism [73]. The query $(\dot{\mathbf{q}}_i, \vec{\mathbf{q}}_i)$, key $(\dot{\mathbf{k}}_j, \vec{\mathbf{k}}_j)$, value $(\dot{\mathbf{u}}_j, \vec{\mathbf{u}}_j)$, and edge bias $(\dot{\mathbf{b}}_{ij}, \vec{\mathbf{b}}_{ij})$ are first calculated with GVLs (the layer superscripts are omitted here for simplicity):

$$\begin{aligned} \dot{\mathbf{q}}_i, \vec{\mathbf{q}}_i &= \text{GVL}(\dot{\mathbf{v}}_i^{(l-1)}, \vec{\mathbf{v}}_i^{(l-1)}), \\ \dot{\mathbf{k}}_j, \vec{\mathbf{k}}_j, \dot{\mathbf{u}}_j, \vec{\mathbf{u}}_j &= \text{GVL}(\dot{\mathbf{v}}_j^{(l-1)}, \vec{\mathbf{v}}_j^{(l-1)}), \\ \dot{\mathbf{b}}_{ij}, \vec{\mathbf{b}}_{ij} &= \text{GVL}(\dot{\mathbf{e}}_{ij}^{(l-1)}, \vec{\mathbf{e}}_{ij}^{(l-1)}), \end{aligned} \tag{14}$$

Then the attention weights for the scalar are computed as:

$$\begin{aligned} \dot{\mathbf{a}}_{ij} &= \dot{\mathbf{q}}_i \odot \dot{\mathbf{k}}_j \odot \dot{\mathbf{b}}_{ij}, \quad \dot{\mathbf{a}}_{ij} \in \mathbb{R}^{h^s} \\ \hat{a}_{ij} &= \text{softmax}_j \frac{1}{\sqrt{h^s}} \dot{\mathbf{a}}_{ij} \mathbf{1}, \end{aligned} \tag{15}$$

Similarly for the vector features:

$$\begin{aligned} \vec{\mathbf{a}}_{ij} &= \vec{\mathbf{q}}_i \odot \vec{\mathbf{k}}_j \odot \vec{\mathbf{b}}_{ij}, \quad \vec{\mathbf{a}}_{ij} \in \mathbb{R}^{h^v \times 3}, \\ \hat{\hat{a}}_{ij} &= \text{softmax}_j \frac{1}{\sqrt{3h^v}} \mathbf{1}^\top \vec{\mathbf{a}}_{ij} \mathbf{1}, \end{aligned} \tag{16}$$

where $\odot$ denotes the Hadamard product, $\mathbf{1}$ is the vector with all entries as 1, h^s and h^v denote the hidden dimension size of the scalar and vector features respectively. softmax_j means perform softmax over the j index. $\hat{a}_{ij}$ and $\hat{\hat{a}}_{ij}$ are the attention weights (scalar and vector channel) between node i and j. The message is obtained as:

$$\begin{aligned} (\dot{\mathbf{m}}_i^{(l)}, \vec{\mathbf{m}}_i^{(l)}) &= \text{GVL}(\sum_j \hat{a}_{ij} \dot{\mathbf{u}}_j, \sum_j \hat{\hat{a}}_{ij} \vec{\mathbf{u}}_j), \\ (\dot{\mathbf{m}}_{ij}^{(l)}, \vec{\mathbf{m}}_{ij}^{(l)}) &= \text{GVL}\left(\mathbf{a}_{ij}, \vec{\mathbf{a}}_{ij}\right), \end{aligned} \tag{17}$$

where the message features with respect to node i are obtained by applying GVL to the weighted summation of the neighboring value features. Finally, the update module U_l is formulated as:

$$\begin{aligned} \dot{\mathbf{v}}_i^{(l)}, \vec{\mathbf{v}}_i^{(l)} &= \text{GVNorm}(\text{GVGate}(\dot{\mathbf{v}}_i^{(l-1)}, \vec{\mathbf{v}}_i^{(l-1)}, \dot{\mathbf{m}}_i^{(l)}, \vec{\mathbf{m}}_i^{(l)})), \\ \dot{\mathbf{e}}_{ij}^{(l)}, \vec{\mathbf{e}}_{ij}^{(l)} &= \text{GVNorm}(\text{GVGate}(\dot{\mathbf{e}}_{ij}^{(l-1)}, \vec{\mathbf{e}}_{ij}^{(l-1)}, \dot{\mathbf{m}}_{ij}^{(l)}, \vec{\mathbf{m}}_{ij}^{(l)})), \end{aligned} \tag{18}$$

where the GVGate fuses the information from the $(l-1)$-th layer and the calculated messages from the l-th layer. GVNorm is appended to normalize the features.

D.3 Decoder

The node/edge features of the decoder are initialized from the output of the encoder and are updated the same as the encoder. The decoder of FlexSBDD further updates the protein C_α and ligand atom coordinates as follows:

$$\begin{aligned} \dot{\boldsymbol{f}}_i^{(l)}, \vec{\boldsymbol{f}}_i^{(l)} &= \text{GVL}(\text{GVP}(\sum_j \hat{a}_{ij} \dot{\mathbf{u}}_j, \sum_j \hat{\hat{a}}_{ij} \vec{\mathbf{u}}_j)), \\ \vec{\boldsymbol{r}}_i^{(l)} &= \sum_j \frac{\boldsymbol{x}_i^{(l-1)} - \boldsymbol{x}_j^{(l-1)}}{\|\boldsymbol{x}_i^{(l-1)} - \boldsymbol{x}_j^{(l-1)}\|_2} \text{MLP}(\text{concat}(\dot{\mathbf{a}}_{ij}, \|\vec{\mathbf{a}}_{ij}\|_2^{(r)})), \\ \boldsymbol{x}_i^{(l)} &= \boldsymbol{x}_i^{(l-1)} + \vec{\boldsymbol{f}}_i^{(l)} + \vec{\boldsymbol{r}}_i^{(l)}, \end{aligned} \tag{19}$$

https://doi.org/10.52202/079017-1707

where $\boldsymbol{x}_i^{(l)}$ is the node i's coordinate at the l-th layer and $\|\cdot\|_2^{(r)}$ denotes the row-wise L2 norm. The attention weights $(\hat{\dot{a}}_{ij}, \hat{\vec{a}}_{ij}, \dot{\mathbf{a}}_{ij}, \vec{\mathbf{a}}_{ij})$ and features $(\dot{\mathbf{u}}_j, \vec{\mathbf{u}}_j)$ are calculated similarly to Equations 14 and 15. Finally, we apply MLPs on last layer representations $(\dot{\mathbf{v}}_i^{(L)}, \vec{\mathbf{v}}_i^{(L)})$ (totally L layers) that capture the chemical and geometric attributes for ligand atom type $\boldsymbol{a}^{(i)}$, residue sidechain dihedral angles $\boldsymbol{\chi}^{(i)}$, and the residue backbone orientation $\boldsymbol{O}^{(i)}$ prediction. For the efficient encapsulation of three-dimensional rotations for $\boldsymbol{O}^{(i)}$, we predict a unit quaternion vector [38]. The quaternion can be easily transformed into a rotation matrix and is a more concise representation of a rotation in 3D. The protein-ligand structure is then adjusted based on the updated coordinates, orientation, and the sidechain dihedral angles. Similar to previous works [57], the update process satisfies the E(3)-equivariance.

D.4 Feature Initialization

Protein node vector features Following [20], the vector feature for protein nodes in FlexSBDD consists of three parts with a total dimension of $[N_p, 24, 3]$ (N_p is the total number of protein nodes).

(1) Euclidean vectors between the C, CA, N, CB (CA for GLY) atoms for a given residue (shape: $[N_p, 16, 3]$). It encompasses various combinations like C to C, C to CA, C to N, C to CB, CA to C, and so forth.

(2) Euclidean vectors between atom j and atom k in for all side-chain dihedral angles (shape: $[N_p, 4, 3]$). For instance, in an amino acid like Arginine (ARG), the side-chain angles (denoted as $\chi_1, \chi_2, \chi_3, \chi_4$) are defined by specific sequences of four atoms (i-j-k-l) according to Rosetta: χ_1: N-CA-CB-CG, χ_2: CA-CB-CG-CD, χ_3: CB-CG-CD-NE, χ_4: CG-CD-NE-CZ. Then the Euclidean vectors can be obtained by combining vectors of CA to CB, CB to CG, CG to CD, and CD to NE. For residues with less than 4 sidechain angles, the corresponding vectors are assigned 0.

(3) Euclidean vectors between CA and atom k in all side-chain dihedral angles (shape: $[N_p, 4, 3]$). For example, the vectors for ARG can be obtained by combining Euclidean vectors of CA to CB, CA to CG, CA to CD, and CA to NE. For residues with less than 4 sidechain angles, the corresponding vectors are assigned 0.

Ligand node vector features The vector features for ligand nodes are initialized as the Euclidean vectors between ligand atoms and the geometric center of the ligand molecule (shape: $[N_l, 1, 3]$). N_l is the number of ligand atoms.

D.5 Geometric Vector Modules

In Algorithm 3 and 4, we show the Geometric vector linear (GVL) and the Geometric vector perception (GVP) modules in FlexSBDD. In Algorithm 5 and 5, we show the GVNorm and GVGate modules for the stability and better performance of FlexSBDD.

Algorithm 3: Geometric Vector Linear (GVL)

Input: Scalar and vector features $(\mathbf{v}, \vec{\mathbf{v}})$
Output: Updated scalar and vector features $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$
Function `GVL(`$\mathbf{v}, \vec{\mathbf{v}}$`):`
  $\vec{\mathbf{v}}' \leftarrow \text{LinearNoBias}(\vec{\mathbf{v}})$;
  $\mathbf{v}' \leftarrow \|\vec{\mathbf{v}}'\|_2$;
  $\mathbf{v}'' \leftarrow \text{concat}(\mathbf{v}, \mathbf{v}')$;
  $\mathbf{v}^u \leftarrow \text{Linear}(\mathbf{v}'')$;
  $\vec{\mathbf{v}}'' \leftarrow \text{LinearNoBias}(\vec{\mathbf{v}}')$;
  $\vec{\mathbf{v}}^u \leftarrow \text{sigmoid}(\mathbf{v}^u) \odot \vec{\mathbf{v}}''$;
  return $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$;

Algorithm 4: Geometric Vector Perceptron (GVP)

Input: Scalar and vector features $(\mathbf{v}, \vec{\mathbf{v}})$
Output: Nonlinear transformed scalar and vector features $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$
Function `GVP(`$\mathbf{v}, \vec{\mathbf{v}}$`):`

$\mathbf{v}', \vec{\mathbf{v}}' \leftarrow \mathrm{GVL}(\mathbf{v}, \vec{\mathbf{v}})$;
$\mathbf{v}^u \leftarrow \mathrm{leaky\_relu}(\mathbf{v}')$;
$\vec{\mathbf{v}}'' \leftarrow \mathrm{LinearNoBias}(\vec{\mathbf{v}}')$;
$\mathbf{v}_{dot} \leftarrow (\vec{\mathbf{v}}' \odot \vec{\mathbf{v}}'')\mathbf{1}$;
$\mathbf{v}_{mask} \leftarrow 1$ if $\mathbf{v}_{dot} \geq 0$, else 0;
$\vec{\mathbf{v}}_{act} \leftarrow \mathbf{v}_{dot} \oslash \|\vec{\mathbf{v}}''\|_2^2 \odot \vec{\mathbf{v}}$; `// ` $\oslash$ `is element-wise division`
$\vec{\mathbf{v}}^u \leftarrow \alpha\vec{\mathbf{v}}' + (1-\alpha)\left(\mathbf{v}_{mask} \odot \vec{\mathbf{v}}' + (1 - \mathbf{v}_{mask}) \odot (\vec{\mathbf{v}}' - \vec{\mathbf{v}}_{act})\right)$; `// ` $\alpha = 0.01$
return $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$;

Algorithm 5: Geometric Vector Normalization (GVNorm)

Input: Scalar and vector features $(\mathbf{v}, \vec{\mathbf{v}})$
Output: Normalized scalar and vector features $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$
Function `GVNorm(`$\mathbf{v}, \vec{\mathbf{v}}$`):`

$\mathbf{v}^u \leftarrow \mathrm{LayerNorm}(\mathbf{v})$;
$\vec{\mathbf{v}}' \leftarrow \vec{\mathbf{v}} / \sqrt{\frac{1}{h'}\langle \vec{\mathbf{v}}, \vec{\mathbf{v}} \rangle_F}$; `//` $\vec{\mathbf{v}}' \in \mathbb{R}^{h' \times 3}$
$\vec{\mathbf{v}}^u \leftarrow \gamma\vec{\mathbf{v}}' + \beta$; `//` $\gamma \in \mathbb{R}^1, \beta \in \mathbb{R}^1$ `are trainable parameters`
return $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$;

Algorithm 6: Geometric vector gate (GVGate)

Input: Scalar and vector features $(\mathbf{v}, \vec{\mathbf{v}})$
Output: Updated scalar and vector features $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$
Function `GVGate(`$\mathbf{v}_p, \mathbf{v}_q, \vec{\mathbf{v}}_p, \vec{\mathbf{v}}_q$`):`

$\mathbf{v}_c \leftarrow \mathrm{concat}\left(\mathbf{v}_p, \mathbf{v}_q, \mathbf{v}_p - \mathbf{v}_q\right)$;
$\vec{\mathbf{v}}_c \leftarrow \mathrm{concat}\left(\vec{\mathbf{v}}_p, \vec{\mathbf{v}}_q, \vec{\mathbf{v}}_p - \vec{\mathbf{v}}_q\right)$;
$\mathbf{v}_g, \vec{\mathbf{v}}_g \leftarrow \mathrm{GVL}\left(\mathbf{v}_c, \vec{\mathbf{v}}_c\right)$;
$\mathbf{g}_s \leftarrow \mathrm{sigmoid}\left(\mathbf{v}_g\right)$;
$\mathbf{g}_v \leftarrow \mathrm{sigmoid}\left(\|\vec{\mathbf{v}}_g\|_2\right)$;
$\mathbf{v}^u \leftarrow \mathbf{g}_s \odot \mathbf{v}_p + (1 - \mathbf{g}_s) \odot \mathbf{v}_q$;
$\vec{\mathbf{v}}^u \leftarrow \mathbf{g}_v \odot \vec{\mathbf{v}}_p + (1 - \mathbf{g}_v) \odot \vec{\mathbf{v}}_q$;
return $(\mathbf{v}^u, \vec{\mathbf{v}}^u)$

E Limitations and Broader Impact

One limitation of FlexSBDD is that it only considers small molecule design. Recently, other drug modalities such as antibodies, peptides, and nucleic acids have played critical roles in drug discoveries and bio-engineering. We would like to build a generalized version of FlexSBDD for other drug modalities. Another limitation is the limited dataset size, which restricts the scaling of the proposed models. In the future, we may benefit from the generated protein-ligand interaction data from generative AI models e.g., AlphaFold 3 [1] and RoseTTAFold All-Atom [44].

As for the broader impacts, there are many potential applications of our work, e.g., discovering cryptic pockets and generating drugs to cure various diseases. We acknowledge the necessity for regulatory oversight of our Structure-Based Drug Design (SBDD) technique to prevent the creation of harmful molecules. Overall, we believe the positive influence of our work outweighs the potential negative impacts.

https://doi.org/10.52202/079017-1707

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: In the abstract and introduction, we clearly state the contributions of our paper.

   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: In Appendix.E, we clearly describe the limitations of the work and the potential ways to reduce the limitations in future works.

   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]

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: The code will be included in `https://github.com/zaixizhang/FlexSBDD`.

   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?

https://doi.org/10.52202/079017-1707

Answer: [Yes]

Justification: We clearly discussed the training datasets and other details.

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: The paper specify all the training and test details.

   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: We followed previous works and did not report the error bars.

   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: The paper provides sufficient information on the computer resources.

   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 research in this paper conforms 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: The paper discussed potential societal impacts.

    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.

https://doi.org/10.52202/079017-1707

- 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: [NA]

    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: The existing assets are properly cited and credited.

    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: [NA]

    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]

    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]

    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.

https://doi.org/10.52202/079017-1707