Prior-Guided Flow Matching for Target-Aware Molecule Design with Learnable Atom Number

Jingyuan Zhou[1], **Hao Qian**[1], **Shikui Tu** [1*], **Lei Xu**[1,2*]

[1]School of Computer Science, Shanghai Jiao Tong University
[2]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ)
{zjoyuan0930, qhonearth, tushikui, leixu}@sjtu.edu.cn

Abstract

Structure-based drug design (SBDD), aiming to generate 3D molecules with high binding affinity toward target proteins, is a vital approach in novel drug discovery. Although recent generative models have shown great potential, they suffer from unstable probability dynamics and mismatch between generated molecule size and the protein pockets geometry, resulting in inconsistent quality and off-target effects. We propose PAFlow, a novel target-aware molecular generation model featuring prior interaction guidance and a learnable atom number predictor. PAFlow adopts the efficient flow matching framework to model the generation process and constructs a new form of conditional flow matching for discrete atom types. A protein–ligand interaction predictor is incorporated to guide the vector field toward higher-affinity regions during generation, while an atom number predictor based on protein pocket information is designed to better align generated molecule size with target geometry. Extensive experiments on the CrossDocked2020 benchmark show that PAFlow achieves a new state-of-the-art in binding affinity (up to -8.31 Avg. Vina Score), simultaneously maintains favorable molecular properties. The code of the paper is provided at https://github.com/CMACH508/PAFlow.

1 Introduction

As a subfield of deep learning in drug discovery, structure-based drug design (SBDD) is considered as a rational and challenging approach for developing novel drugs [1, 2]. It aims to generate drug-like molecules with stable 3D structures and high binding affinities conditioned on the target proteins, which can be formulated as a conditional generation problem. In the past few years, an increasing number of generative models have been applied to SBDD task. One category involves autoregressive models that generate 3D molecules atom by atom [3–5] or fragment by fragment [6, 7]. Another category includes diffusion-based methods [8–11], which gradually denoise standard Gaussian noise to predict full-atom distributions by leveraging both local and global information. Other models, such as Rectified Flow [12] and Bayesian Flow Networks [13], have also been employed in SBDD [14, 15]. Although these methods have demonstrated the potential to generate target-aware molecules, they still face several limitations: (1) autoregressive sampling suffers from unnatural generation orders, leading to unrealistic fragments and error accumulation; (2) the denoising trajectory in diffusion models is highly stochastic, resulting in unstable molecular quality; (3) to the best of our knowledge, all current non-autoregressive methods determine atom numbers in generated molecules by sampling from a predefined distribution, which relies on prior knowledge from reference ligands and often causes mismatches between ligand size and binding site geometry.

*Correspondence authors are Shikui Tu and Lei Xu.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

The recently proposed Flow Matching (FM) framework [16] has shown promising initial results and demonstrates potential to serve as an effective solution for SBDD problems. Specifically, FM is a simulation-free approach for training Continuous Normalizing Flows (CNFs) that exhibits notable generative capabilities. It is compatible with the Gaussian probability paths employed in diffusion models for transitioning between noise and data samples, while also supporting non-Gaussian paths as conditional probability trajectories. During the sampling phase, the adoption of ordinary differential equation (ODE) solvers enables fast and stable generation.

Inspired by the recent advancement of FM, we propose PAFlow, a 3D all-atom **Flow** matching model with **P**rior interaction guidance and a learnable **A**tom number predictor, which is designed to address the above drawbacks of existing methods. To overcome limitations (1) and (2), PAFlow adopts the FM as the framework for molecule generation. Specifically, we employ the established Variance Preserving (VP) probability path [16] for generating continuous atomic coordinates while deriving a new form of Conditional Flow Matching (CFM) for discrete atom type. To further enhance the binding affinity between molecules and target proteins, a protein–ligand interaction predictor is incorporated during generation, which guides the vector field using prior binding knowledge towards poses with tighter interactions. Regarding limitation (3), we develop an atom number predictor that only utilizes protein pocket information rather than reference ligand priors to estimate the appropriate number of atoms. Extensive empirical evaluation on the CrossDocked2020 dataset [17] demonstrates that PAFlow generates molecules with not only significantly improved binding affinity compared to all baseline methods but also maintaining desirable molecular properties.

Our main contributions can be summarized as follows:

- We propose an SBDD generative model based on the FM framework, where atomic coordinates and atom types are modeled using the existing VP path and a newly developed CFM, respectively.
- A protein-ligand interaction predictor is integrated into the generation process to introduce prior binding knowledge, guiding the vector field toward directions that correspond to higher binding affinity.
- To address the mismatch between molecule size and binding site geometry when sampling from the predefined distributions, we introduce an atom number predictor that estimates the appropriate number of atoms only using the binding site information.
- Extensive experiments on the CrossDocked2020 benchmark demonstrate that PAFlow can generate molecules with **-8.31 Avg. Vina Score**, significantly outperforming other strong baselines by a large margin (-1.24) and establishing a new state-of-the-art, while maintaining favorable molecular properties.

2 Related Works

Structure-Based Drug Design Structure-based drug design is a fundamental task in drug discovery, which aims to generate molecules that specifically bind with high affinity to a given protein pocket [18]. Early efforts such as [19, 20] generate 1D SMILES strings based on protein context, while [21] fit the distribution of target sequence embeddings in latent space to enable target-aware molecular graph generation. Motivated by the advances in 3D and geometric modeling, numerous works have attempted to address the problem directly in the 3D space. [22] voxelizes molecules in atomic density grids to generate 3D molecules in a conditional VAE framework. [3–5] propose to iteratively generate atoms and bonds through autoregressive sampling within target binding site. [6, 7] generate ligand molecules motif by motif utilizing chemical priors of molecular fragments. However, the unnatural generation order during autoregressive sampling leads to unrealistic fragments and severe error accumulation. In recent works, diffusion models have been extensively applied to ligand molecule generation, which denoises atom types and coordinates sampled from prior distributions using SE(3)-equivariant networks [8–11, 23, 24]. [15] also explore the application of Bayesian Flow Networks (BFN) for molecular generation. However, these non-autoregressive methods typically acquire the number of atoms in the generated molecules by sampling from a predefined distribution, which depends on reference ligand information and often results in mismatches between the molecular size and the protein pocket. In this work, we aim to address this issue by introducing a learnable atom number predictor.

Flow Matching Flow Matching has recently attracted considerable attention and has demonstrated its potential in various domains such as image generation [25, 26] and biomolecule design [27, 28]. As a simulation-free approach for training continuous normalizing flows (CNFs), it is compatible with diverse probability paths and achieves better sampling efficiency through ODE solvers. For instance, [16] apply Flow Matching to the Variance Preserving diffusion path and formulate the VP Conditional Flow Matching (CFM), which addresses the unstable probability dynamics in conventional diffusion models. Other works [12, 26, 29] have explored transporting the prior distribution to the data distribution along straight line paths as much as possible. Although such paths offer shorter distances and lower computational cost, they lack the capacity to effectively model complex tasks like SBDD. For instance, [14] adopt this strategy for SBDD but achieve suboptimal performance. Therefore, we adopt the more robust VP path to model continuous atomic coordinates and construct a novel CFM tailored for discrete atomic types.

3 Preliminaries

Problem Definition The SBDD task can be defined as a conditional generative problem, where the goal is to generate ligand molecules that specifically bind to a given protein target. A target protein $\mathcal{P}$ is represented as a set of atoms $\mathcal{P} = \{(\mathbf{x}_P^{(i)}, \mathbf{a}_P^{(i)})\}_{i=1}^{N_P}$, where $\mathbf{x}_P^{(i)} \in \mathbb{R}^3$ is the 3D coordinates of the i-th protein atom, $\mathbf{a}_P^{(i)} \in \mathbb{R}^{N_f}$ is a one-hot vector representing its features (e.g., element type, amino acid type), N_P is the number of atoms contained in the protein and N_f is the feature dimension of a protein atom. Correspondingly, a binding molecule $\mathcal{M} = \{(\mathbf{x}_M^{(i)}, \mathbf{a}_M^{(i)})\}_{i=1}^{N_M}$ comprises N_M atoms with 3D coordinates $\mathbf{x}_M^{(i)} \in \mathbb{R}^3$ and atom types $\mathbf{a}_M^{(i)} \in \mathbb{R}^K$, where K is the molecule atom type dimension. Then molecular representation can be simplified as $\mathbf{m} = [\mathbf{x}_M, \mathbf{a}_M]$, where $\mathbf{x}_M \in \mathbb{R}^{N_M \times 3}$, $\mathbf{a}_M \in \mathbb{R}^{N_M \times K}$ and $[\cdot, \cdot]$ is the concatenation operator. Similarly, the binding pocket is denoted as $\mathbf{p} = [\mathbf{x}_P, \mathbf{a}_P]$, where $\mathbf{x}_P \in \mathbb{R}^{N_P \times 3}$ and $\mathbf{a}_P \in \mathbb{R}^{N_P \times N_f}$.

Preliminaries on Flow Matching In this section, we summarize how the general flow matching method is implemented based on [16]. Let $\mathbb{R}^d$ represent the data space with data points $x = \left(x^1, \ldots, x^d\right) \in \mathbb{R}^d$. q is the data distribution, x_1 denotes a data point from q and x_0 represents a sample from the prior distribution p_0. The time-dependent probability density path is defined as $p_{t\in[0,1]} : \mathbb{R}^d \to \mathbb{R}_{>0}$ and the corresponding vector field is defined as $u_{t\in[0,1]} : \mathbb{R}^d \to \mathbb{R}^d$. The vector field can construct a unique time-dependent flow $\psi_{t\in[0,1]} : \mathbb{R}^d \to \mathbb{R}^d$ defined by the ODE:

$$\frac{\mathrm{d}}{\mathrm{d}t}\psi_t\left(x\right) = u_t\left(\psi_t\left(x\right)\right), \quad \psi_1\left(x\right) = x. \tag{1}$$

FM seeks to regress the target vector field u_t with a neural network $v_\theta(x, t)$:

$$\mathcal{L}_{FM}(\theta) = \mathbb{E}_{t, p_t(x)} \left|\left| v_\theta(x, t) - u_t(x) \right|\right|^2. \tag{2}$$

However, the lack of knowledge about what an appropriate p_t and u_t are makes Eq. 2 infeasible to apply in practice. [16] proposed an alternative that employs a definable conditional probability path $p_t(x|x_1)$ with a conditional vector field $u_t(x|x_1)$. The Conditional Flow Matching (CFM) objective is formulated as follows:

$$\mathcal{L}_{CFM} = \mathbb{E}_{t, p_1(x_1), p_t(x|x_1)} \left|\left| v_\theta(x, t) - u_t(x|x_1) \right|\right|^2. \tag{3}$$

The CFM objective is tractable and shares the same gradient as $\mathcal{L}_{FM}$ [16, 30], thus optimizing the CFM objective is equivalent to optimizing the FM objective. As for inference phase, Eq. 1 can be solved using ODE solvers: $x_1 = \mathrm{ODESolve}(x_0, v_\theta, 1, 0)$.

4 Method

This section elaborates on the implementation details of PAFlow. First, different probability paths are used to model continuous atomic coordinates and discrete atom types within the FM framework while maintaining SE(3)-equivariance during ODE-based sampling. Based on this backbone, a protein–ligand interaction predictor is employed guide the vector field during generation to further enhance molecular quality. Additionally, to address the geometric incompatibility between generated molecules and target proteins, an atom number predictor that relies solely on target protein infor-

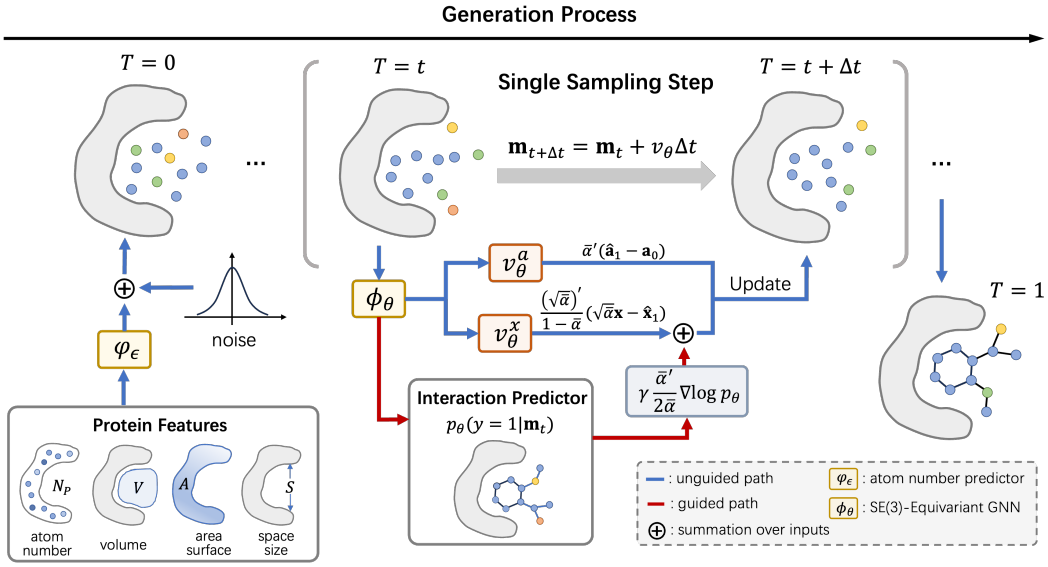


Figure 1: Overview of the PAFlow generation process. The atom predictor first estimates the number of atoms in the generated molecule based on the target protein information, which is then used to initialize the molecule. SE(3)-EGNN is applied to predict vector fields for atomic coordinates and atom types, while a protein-ligand interaction predictor provides prior binding guidance for the coordinate vector field. The final ligand molecule with high binding affinity is obtained through iterative updates. For simplicity, $\bar{\alpha}_{1-t}$ is denoted as $\bar{\alpha}$.

mation—rather than prior knowledge from reference ligands—is proposed. The whole training and sampling procedures are summarized in the Appendix C.

4.1 Flow Matching in Molecule Generation

For PAFlow, a data point from the prior distribution p_0 is $\mathbf{m}_0$ and the target data distribution p_1 is the ligand molecule $\mathbf{m}_1$ that can specifically binds to the target protein. The molecular probability path can be formulated as a product of atom coordinate distribution and atom type distribution following [8]:

$$p_t(\mathbf{m}|\mathbf{m}_1, \mathbf{p}) = p_t(\mathbf{x}|\mathbf{x}_1, \mathbf{p}) \cdot p_t(\mathbf{a}|\mathbf{a}_1, \mathbf{p}). \tag{4}$$

For brevity we use $\mathbf{x}$ in place of $\mathbf{x}_M$. [31] proposes that in the multivariable case the probability path for each variable can be constructed separately, which allows us to model the two variables $\mathbf{x}$ and $\mathbf{a}$—belonging to different data types—using separate probability paths. The continuous atomic coordinates $\mathbf{x}$ are modeled using the Variance Preserving path, with the probability path and target conditional vector field defined as follows [16]:

$$p_t(\mathbf{x}|\mathbf{x}_1, \mathbf{p}) = \mathcal{N}(\mathbf{x}|\sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_1, (1-\bar{\alpha}_{1-t})\mathbf{I}), \qquad u_t^x(\mathbf{x}|\mathbf{x}_1, \mathbf{p}) = \frac{(\sqrt{\bar{\alpha}_{1-t}})'}{1-\bar{\alpha}_{1-t}}(\sqrt{\bar{\alpha}_{1-t}}\mathbf{x} - \mathbf{x}_1), \tag{5}$$

where $\mathcal{N}$ is a Gaussian distribution and the fixed variance schedules are defined as β_t $(t = 0, \Delta t, ..., 1)$, with $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{s=0}^{t}$, $\bar{\beta}_t = 1 - \bar{\alpha}_t$, following [8]. $(\sqrt{\bar{\alpha}_{1-t}})'$ denotes the derivative with respect to time t. For the discrete atomic types $\mathbf{a}$, the probability density path is defined according to [8] as a categorical distribution $\mathcal{C}$:

$$p_t(\mathbf{a}|\mathbf{a}_1, \mathbf{p}) = \mathcal{C}(\mathbf{a}|\mathbf{c}(\mathbf{a}, \mathbf{a}_1)), \qquad where \quad \mathbf{c}(\mathbf{a}, \mathbf{a}_1) = \bar{\alpha}_{1-t}\mathbf{a}_1 + (1-\bar{\alpha}_{1-t})/K \tag{6}$$

In practice $\mathbf{x}$ and $\mathbf{a}$ follow different schedules, but we retain the same notation for conciseness. Since this probability path has not been constructed as CFM in previous work, we derive its conditional vector field as below:

$$u_t^a(\mathbf{c}(\mathbf{a}, \mathbf{a}_1)|\mathbf{a}_1, \mathbf{p}) = \bar{\alpha}'_{1-t}(\mathbf{a}_1 - \mathbf{a}_0). \tag{7}$$

The complete derivation is given in Appendix A.2. In fact, the probability paths for atom coordinates and types are identical to those applied in [8]; however, the generation strategies adopted by the two methods are fundamentally different. In [8], generation is performed by iteratively denoising

the Gaussian noise. Instead, our work formulates the generation process as the integration of the ODE $\frac{\mathbf{dm}_t}{\mathbf{d}t} = v_\theta(\mathbf{m}_t, t)$ from $t = 0$ to $t = 1$ using an Euler solver [32] starting from an initialized ligand molecule $\mathbf{m}_0$, where the vector field is approximated with a neural network parameterized by θ. Specifically, for $\mathbf{x}$ and $\mathbf{a}$ we have:

$$\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + v_\theta^x(\mathbf{m}_t, \mathbf{p}, t)\Delta t, \qquad \mathbf{c}(\mathbf{a}_{t+\Delta t}, \mathbf{a}_1) = \mathbf{c}(\mathbf{a}_t, \mathbf{a}_1) + v_\theta^a(\mathbf{m}_t, \mathbf{p}, t)\Delta t, \tag{8}$$

where $\mathbf{x}_0$ is initialized with a standard Gaussian distribution inside the protein pocket and $\mathbf{a}_0$ are initialized with a uniform distribution. The generative process is expected to be invariant to translations and rotations of the protein-ligand complex, an essential inductive bias when generating 3D molecules [33–36]. Given the evidence that an invariant distribution composed with an equivariant invertible function will result in an invariant distribution [33], we have the following proposition (proof in Appendix A.1).

Proposition 1. *Denoting the SE(3)-transformation as T_g and $(v_\theta^x(\mathbf{m}_t, \mathbf{p}, t), v_\theta^a(\mathbf{m}_t, \mathbf{p}, t)) = v_\theta(\mathbf{m}_t, \mathbf{p}, t)$, if we shift the Center of Mass (CoM) of protein atoms to zero and parameterize $v_\theta(\mathbf{m}_t, \mathbf{p}, t)$ with an SE(3)-equivariant network, then the generation process is invariant w.r.t T_g on the protein-ligand complex.*

There are different ways to parameterize v_θ. In this work the neural network is designed to predict $[\mathbf{x}_1, \mathbf{a}_1]$, which are then processed through Eq. 5 and Eq. 7 to obtain v_θ^x and v_θ^a respectively. To guarantee the invariance constraint, the generation process is parameterized using an SE(3)-Equivariant GNN ϕ_θ:

$$[\hat{\mathbf{x}}_1, \hat{\mathbf{a}}_1] = \phi_\theta([\mathbf{x}_t, \mathbf{a}_t], t, \mathbf{p}), \tag{9}$$

where the l-th layer works as:

$$\mathbf{h}_i^{l+1} = \mathbf{h}_i^l + \sum_{j \in \mathcal{V}, i \neq j} f_h(d_{ij}^l, \mathbf{h}_i^l, \mathbf{h}_j^l, \mathbf{e}_{ij}; \theta_h) \tag{10}$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + \sum_{j \in \mathcal{V}, i \neq j} (\mathbf{x}_i^l - \mathbf{x}_j^l) f_x(d_{ij}^l, \mathbf{h}_i^{l+1}, \mathbf{h}_j^{l+1}, \mathbf{e}_{ij}; \theta_x) \cdot \mathbf{l}_{mask}. \tag{11}$$

l denotes the layer index and $\mathcal{V}$ is the set of atoms from both the protein and the ligand molecule. $d_{ij} = ||\mathbf{x}_i - \mathbf{x}_j||$ and $\mathbf{e}_{ij}$ represents the relative distance and option edge features between atom i and atom j. $\mathbf{l}_{mask}$ is a mask applied to the ligand atoms to keep the protein atom coordinates fixed. f_h and f_x are graph attention networks. $\hat{\mathbf{a}}_1$ can be obtained by input $\mathbf{h}^L = [\mathbf{h}_1^L, ..., \mathbf{h}_{N_M-1}^L]_t$ into a multi-layer perceptron and a softmax function.

4.2 Prior-Guided Generation

Inspired by previous works [23, 10, 24] that utilize prior knowledge to guide the denoising process of diffusion models, we leverage predicted protein-ligand interactions, i.e. binding affinity, as guidance for the vector field, thereby further improving the quality of generated molecules. The ground truth of binding affinity is denoted as y. The binding affinity of ligands in the training set is normalized to $[0, 1]$, thus $y = 1$ indicates a strong interaction between the molecule and the target protein. To predict the binding affinity $\hat{y}$ between generated molecule and target protein, the final atom hidden embedding $\mathbf{h}^L$ containing useful global information is used following [8, 23] and Eq. 9 can be rewritten as:

$$[\hat{\mathbf{x}}_1, \hat{\mathbf{a}}_1, \hat{y}] = \phi_\theta([\mathbf{x}_t, \mathbf{a}_t], t, \mathbf{p}), \qquad \hat{y} = \frac{1}{N_M} \sum_{i=0}^{N_M-1} \sigma(\mathrm{MLP}(\mathbf{h}^L)), \tag{12}$$

where σ is a sigmoid function. According to *Lemma 1* from [37], we derive the predictor guidance generation process as follows:

$$\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + (v_\theta^x(\mathbf{m}_t, \mathbf{p}, t) + \gamma \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}} \nabla \log p_\theta(y = 1|\mathbf{m}_t))\Delta t, \tag{13}$$

$$\mathbf{c}(\mathbf{a}_{t+\Delta t}, \mathbf{a}_1) = \mathbf{c}(\mathbf{a}_t, \mathbf{a}_1) + v_\theta^a(\mathbf{m}_t, \mathbf{p}, t)\Delta t. \tag{14}$$

Detailed derivation can be found in Appendix A.3. A scaling factor γ is added to control the gradient strength and the likelihood function can be defined as:

$$p_\theta(y = 1|\mathbf{m}_t) \propto \exp(-\ell(\hat{y}, y = 1)), \tag{15}$$

where $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is the mean square error (MSE) loss function to evaluate the deviation between the predicted score $\hat{y}$ and $y = 1$.

Explicitly guiding discrete atom types through gradients in molecular generation is challenging. However, in this work, at sampling time step t, the sampled atom type $\mathbf{a}_{t+\Delta t}$, is obtained from $\mathbf{x}_t$ and a_t via the SE(3)-Equivariant GNN. As a result, the optimized $\mathbf{x}_t$ naturally influences $\mathbf{a}_{t+\Delta t}$, thereby providing the implicit guidance for atom types. This guidance strategy is consistent with the approach employed in [24].

4.3 Learnable Atom Number Predictor

To generate molecules that better match the size of the protein pocket and avoid relying on prior knowledge from reference ligands, an atom number predictor φ_ϵ with neural network is trained using only target protein information. [38] suggests that there is a certain correlation between the volume of the binding pocket V and the ligand size. To provide the network with comprehensive information about the binding pocket and improve prediction accuracy, the surface area of the binding site A, the number of atoms within the pocket N_P, and the space size S—which is used by existing models to determine atom numbers—are also utilized as inputs. Besides, a dataset consisting of binding site information–ligand atom number pairs is constructed for training. To enhance training stability and accelerate convergence, normalized ligand atom numbers are used as training labels, then the network outputs must be denormalized to obtain the final predicted number of atoms:

$$\hat{N}_M = \hat{n}_M(N_M^{max} - N_M^{min}) + N_M^{min}, \qquad \hat{n}_M = \varphi_\epsilon([N_P, V, A, S]) + \tau \tag{16}$$

[39] propose that introducing noise in generative models can enhance both the diversity and quality of discrete outputs. We observe the improved molecular generation performance when a small Gaussian noise term $\tau \sim N(0, \delta^2)$ is added to the predictor's output, which resembles the reparameterization trick used in variational autoencoders. Neural networks inherently involve uncertainty in their predictions, and the injection of Gaussian noise may reflect this uncertainty. As a form of regularization, this strategy helps mitigate overconfidence in potentially inaccurate point estimates and encourages exploration of a broader solution space. We further prove the effectiveness of noise injection in a mathematical form in Appendix A.4.

5 Experiments

5.1 Experimental Setup

Dataset We train and evaluate PAFlow on the CrossDocked2020 dataset [17]. Following the same strategy as in [3, 8], binding poses with RMSD greater than 1 Å and protein pairs with sequence identity over 30% are excluded. Similar to [23], binding affinities in the CrossDocked2020 dataset are normalized to the range $[0, 1]$ for training the protein–ligand interaction predictor, where higher values indicate better binding affinity. We then randomly sample 100,000 complexes for training and select 100 complexes with distinct target proteins for testing. We further process the data and construct a dataset in the form of $\mathcal{D} = \{(N_M, N_P, V, A, S)\}$, where V and A are computed by the PyKVFinder [40] package. The dataset consists of 98k+ training samples and 100 testing samples for the atom number predictor.

Baselines PAFlow is compared with the following baseline methods: **LiGAN** [22] is a CNN-Based conditional VAE generating voxelized atomic density. **AR** [3] and **Pocket2Mol** [4] are autoregressive generative models where atoms are sampled one by one. **TargetDiff** [8], **DecompDiff** [9], **IPDiff** [10], **TAGMol** [24] and **ALiDiff** [11] are diffusion-based models that generate 3D molecules in a non-autoregressive manner. **MolCRAFT** [15] generates molecules in the continuous parameter space using Bayesian Flow Network (BFN) framework. **FlowSBDD** [14] is based on the rectified flow model, another form of FM that transports prior to data along linear paths, with a novel bond distance loss.

Evaluation metrics The generated molecules are evaluated from two aspects: **binding affinity** with the target protein and **molecular properties**. We employ the widely adopted AutoDock Vina[41] to estimate the mean and median values of affinity-based metrics (Vina Score, Vina Min, Vina Dock and High Affinity). Vina Score directly measures the binding affinity based on the generated poses;

Vina Min optimizes the pose through local minimization before estimation; Vina Dock performs re-docking to reflect the optimal binding affinity; High Affinity computes the percentage of generated molecules that bind better than the reference ligands per test protein. Following [3, 22], QED [42] (drug-likeness), SA [43] (synthesize accessibility), and diversity are adopted to evaluate critical molecular properties.

Table 1: Summary of binding affinity and molecular properties of reference molecules and molecules generated by PAFlow and other baselines. (↑)/(↓) denotes a larger / smaller number is better. Top 2 results are highlighted with **bold text** and underlined text, respectively.

Method		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.
Ref		-6.36	-6.46	-6.71	-6.49	-7.45	-7.26	-	-	0.48	0.47	0.73	0.74	-	-
Auto-regressive	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
Diffusion	TargetDiff	-5.47	-6.30	-6.64	-6.83	-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	-7.09	-8.39	-8.43	64.4%	71.0%	0.45	0.43	0.61	0.60	0.68	0.68
	IPDiff	-6.42	-7.01	-7.45	-7.48	-8.57	-8.51	69.5%	75.5%	0.52	0.53	0.61	0.59	0.74	0.73
	TAGMol	-7.02	-7.77	-7.95	-8.07	-8.59	-8.69	69.8%	76.4%	0.55	0.56	0.56	0.56	0.69	0.70
	ALiDiff	-7.07	-7.95	-8.09	-8.17	-8.90	-8.81	73.4%	81.4%	0.50	0.50	0.57	0.56	0.73	0.71
BFN	MolCRAFT	-6.59	-7.04	-7.27	-7.26	-7.92	-8.01	59.1%	62.6%	0.50	0.51	0.69	0.68	0.73	0.73
Flow	FlowSBDD	-3.62	-5.03	-6.72	-6.60	-8.50	-8.36	63.4%	70.9%	0.47	0.48	0.51	0.51	**0.75**	**0.75**
	Ours	**-8.31**	**-8.92**	**-8.79**	**-8.96**	**-9.46**	**-9.49**	**80.8%**	**93.7%**	0.49	0.50	0.57	0.57	0.71	0.70

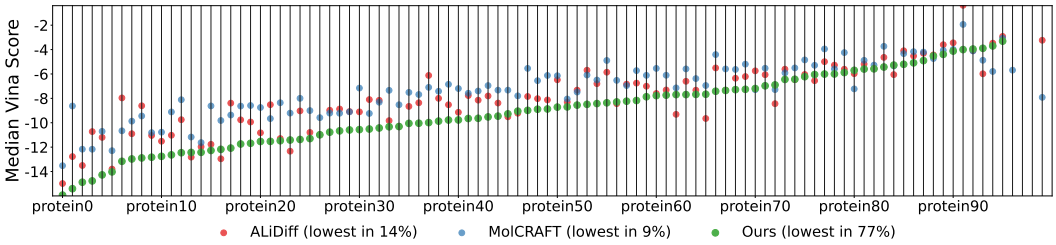


Figure 2: Median Vina energy for different generated molecules (ALiDiff, MolCRAFT, PAFlow) across 100 testing binding targets. The proteins are sorted by the median Vina energy of molecules generated from PAFlow.

5.2 Main Results

Target Binding Affinity and Molecular Properties We comprehensively evaluate the performance of PAFlow by comparing it against four types of SBDD methods: autoregressive, diffusion, BFN, and FM methods. 100 molecules are sampled for each test protein. As shown in Tab. 1, PAFlow significantly outperforms all baseline methods on all binding-related metrics, where the Vina score is computed without conformational optimization and serves as the most critical metric. Specifically, PAFlow surpasses the strong diffusion method ALiDiff by a large margin of 17.5%, 8.7%, and 6.3% on Avg. Vina Score, Vina Min, and Vina Dock, respectively. Furthermore, it also achieves superior performance over the novel BFN method MolCRAFT by 26.1%, 20.9%, and 19.4% on the same metrics. In terms of high-affinity binders, an average of 80.8% of the PAFlow molecules exhibit higher binding affinity than the reference ligands, significantly exceeding all other baselines. Notably, the performance of FlowSBDD—another FM-based model—is unremarkable in binding affinity due to the use of linear probability paths, inadequately capturing the complexity inherent to SBDD. In contrast, PAFlow designs appropriate probability paths for atomic coordinates and atom types, effectively generating molecules with high binding affinity. Fig. 2 presents the median Vina energy of all generated molecules for each binding pocket, comparing PAFlow with two SOTA methods ALiDiff and MolCRAFT. PAFlow achieves the highest binding affinity on 77% of the targets, significantly exceeding the other two methods.

PAFlow generates molecules with higher binding affinity in Tab. 1 while maintaining comparable QED, SA, and diversity to ALiDiff. These properties are not explicitly emphasized during generation, as in real-world drug discovery they are typically used for rough filtering and are considered acceptable

as long as they fall within a reasonable range. Based on the PAFlow framework, extending the molecule–protein interaction predictor to molecular properties holds promise for further improving these metrics. Fig. 3 presents examples of generated molecules along with their properties. The results show that the molecules generated by PAFlow exhibit favorable properties while maintain reasonable structures, suggesting strong potential as candidate ligands. We further conduct structural analysis experiments, with the results presented in Appendix E.7.

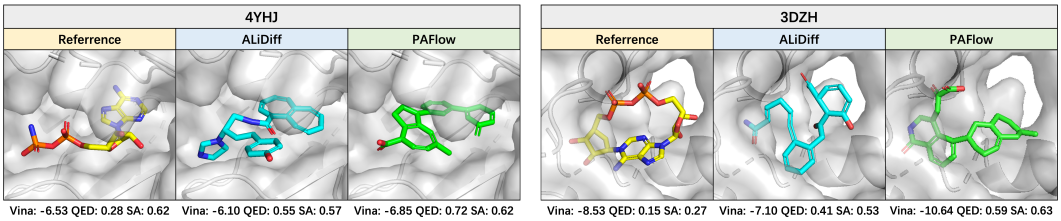


Figure 3: Visualizations of reference molecules and molecules generated by ALiDiff and PAFlow for protein pockets (4YHJ, 3DZH , 2Z3H and 2JJG). Vina Score, QED and SA are reported below.

Sampling Efficiency We select the most efficient model from each category of baselines to compare sampling speed with PAFlow. Additionally, we report the sampling efficiency of PAFlow using a reduced number of steps T = 20. Fig. 4 shows the inference time for generating 100 molecules on average. While TargetDiff and Pocket2Mol require 3968s and 4009s respectively, PAFlow only takes 717s, achieving a $5.5\times$ speedup. Although PAFlow is slightly slower than MolCRAFT, it produces molecules with significantly higher binding affinity, which compensates for the additional time cost. Moreover, when using fewer steps T = 20, PAFlow becomes even faster than MolCRAFT (402s vs 287s) while still generating molecules with better binding performance (see Appendix E.2). This allows the sampling step size to be adjusted according to the trade-off between generation speed and molecule quality in different scenarios.

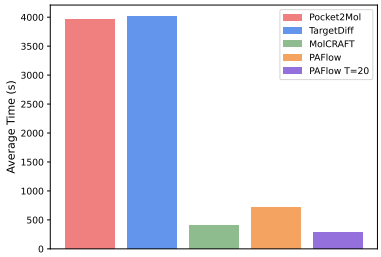
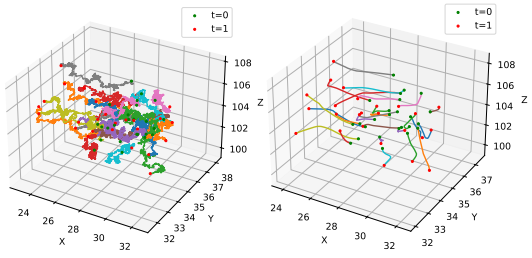


Figure 4: Average time required by different methods to generate 100 molecules for a target protein, with shorter times indicating higher sampling efficiency.

5.3 Ablation Studies

Effectiveness of FM Framework As described in Sec. 4.1, PAFlow and TargetDiff share the same probability paths, but differ in their sampling strategies. To highlight the advantages of FM-based generation, we construct a variant denoted as PAFlow w/o PA, by removing prior guidance and replacing the atom count predictor with predefined sampling. This essentially corresponds to replacing the denoising strategy in TargetDiff with ODE-based sampling while keeping all other components unchanged. Evaluation results for the generated molecules are reported in Tab. 2. PAFlow w/o PA generally outperforms TargetDiff on the affinity-related metrics. Moreover, while maintaining the same SA score, it achieves even better QED and diversity than TargetDiff. These results demonstrate the superior generative performance of FM compared to diffusion-based denoising. Fig. 5 further visualizes the trajectories of atomic coordinates during generation, clearly showing that compared to diffusion the smoother probability path of FM could result in the better molecular quality.

Impact of Prior Guidance To validate the effectiveness of guiding the vector field with the protein-ligand interaction predictor, we generate 1,000 molecules in all test proteins with PAFlow w/o P (which excludes prior guidance during sampling) and PAFlow. As demonstrated in Tab. 3, when guidance is applied, the generated molecules exhibit substantial improvements across all affinity-related metrics, with increases of 60.4%, 22.5%, and 13.0% in Avg. Vina Score, Vina Min, and Vina Dock, respectively. Improvements are also observed in SA and Diversity. Although QED shows a slight decrease, it remains within an acceptable range. Given the superior performance of the guidance strategy, extending the predictor to molecular properties holds great promise for further enhancing these metrics. Additionally, the extent of improvement is influenced by the guidance strength γ, with related results presented in the Appendix E.3.



(a) (b)

Figure 5: Atomic coordinate trajectories from $t = 0$ to $t = 1$ under different sampling strategies with the same probability paths. (a) shows the generation trajectory of TargetDiff, while (b) presents that of PAFlow w/o PA.

Table 2: Evaluation of generation results on TargetDiff and PAFlow w/o PA. PAFlow w/o PA is derived from TargetDiff by retaining all components unchanged except for replacing the sampling strategy with ODE-based sampling.

Metrics	**TargetDiff**		**PAFlow w/o PA**	
	Avg.	Med.	Avg.	Med.
Vina Score	**-5.47**	-6.30	-5.13	**-6.35**
Vina Min	-6.64	-6.83	**-6.76**	**-7.12**
Vina Dock	-7.80	-7.91	**-8.08**	**-8.18**
QED	0.48	0.48	**0.53**	**0.53**
SA	**0.58**	**0.58**	**0.58**	**0.58**
Diversity	0.72	0.71	**0.73**	**0.72**

Table 3: Effect of using the molecular-protein interaction predictor for guidance. *PAFlow w/o P* refers to the variant that does not incorporate prior knowledge to guide the vector field during the generation process.

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.
PAFlow w/o P	-5.05	-6.78	-6.94	-7.48	-8.51	-8.53	70.3%	88.2%	**0.53**	**0.53**	0.56	0.56	**0.70**	0.69
PAFlow	**-8.10**	**-8.83**	**-8.50**	**-8.80**	**-9.62**	**-9.39**	**80.7%**	**100.0%**	0.49	0.50	**0.57**	**0.57**	**0.70**	**0.70**

Effect of Atom Number Predictor To assess the impact of atom number predictor, we design three variant models for comparison. All modifications are based on PAFlow w/o P to eliminate the influence of prior-guided sampling: (1) predefined: atom numbers are sampled from the predefined distributions; (2) $\delta = 0$: atom numbers are directly predicted by the predictor without added noise; and (3) $\delta = 0.01$: Gaussian noise $\mathcal{N}(0, 0.01^2)$ is added to the predicted result. We generate 1,000 molecules for each setting on the test proteins, and the evaluation results are summarized in Tab. 7. Using the predictor yields substantial improvements in all affinity-related metrics compared to sampling from the predefined distributions. For molecular properties, QED and SA are improved while maintaining diversity. This demonstrates that molecules generated with predicted atom numbers exhibit more appropriate molecular sizes and thus perform better overall. Furthermore, adding a small Gaussian to the predicted result leads to additional gains in performance. This randomness offers robustness that deterministic outputs lack and helps compensate for minor prediction errors. Moreover, introducing noise allows model to explore a broader design space and potentially discover molecules with superior performance.

Table 4: Comparison of different strategies for determining the number of atoms in molecules generated by PAFlow w/o P.

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.
predefined	-4.49	-6.50	-6.33	-7.28	-7.62	-8.46	61.2%	66.7%	0.52	0.51	0.57	0.57	**0.73**	**0.71**
$\delta = 0$	-5.72	-6.86	-7.30	-7.52	-8.30	-8.48	70.4%	**88.9%**	**0.55**	**0.56**	**0.58**	**0.58**	0.72	**0.71**
$\delta = 0.01$	**-6.17**	**-7.03**	**-7.50**	**-7.74**	**-8.70**	**-8.61**	**72.8%**	**88.9%**	**0.55**	**0.56**	0.57	0.57	0.71	**0.71**

6 Conclusion

In this work, we propose PAFlow, a flow matching-based SBDD method that separately models continuous atomic coordinates and discrete atom types using the VP path and a newly constructed CFM. By incorporating a protein-ligand interaction predictor, PAFlow effectively guides the vector field toward higher binding affinity. Additionally, an atom number predictor is introduced to determine the number of atoms in generated molecules, addressing the geometric incompatibility between generated molecules and target proteins and eliminating the dependence on prior knowledge from reference ligands. Extensive experiments on the CrossDocked2020 benchmark demonstrate that PAFlow achieves state-of-the-art performance in terms of binding affinity, with up to -8.31 Avg. Vina score, while maintaining desirable molecular properties. In future work, we plan to extend the

interaction predictor to molecular properties, enabling multi-objective optimization over both binding affinity and critical molecular properties for enhanced drug discovery potential.

7 Acknowledgements

This work was supported by the National Natural Science Foundation of China (grants No. 62172273), the Science and Technology Commission of Shanghai Municipality (grants No. 24510714300), and the Shanghai Municipal Science and Technology Major Project, China (Grant No. 2021SHZDZX0102).

References

[1] B. Yang, C. Xiang, T. Li, Y. Xu, and J. Li, "3d structure-based generative small molecule drug design: Are we there yet?" *bioRxiv*, pp. 2024–12, 2024.

[2] Q. Bai, T. Xu, J. Huang, and H. Pérez-Sánchez, "Geometric deep learning methods and applications in 3d structure-based drug design," *Drug Discovery Today*, p. 104024, 2024.

[3] S. Luo, J. Guan, J. Ma, and J. Peng, "A 3d generative model for structure-based drug design," *Advances in Neural Information Processing Systems*, vol. 34, pp. 6229–6239, 2021.

[4] X. Peng, S. Luo, J. Guan, Q. Xie, J. Peng, and J. Ma, "Pocket2mol: Efficient molecular sampling based on 3d protein pockets," in *International Conference on Machine Learning*. PMLR, 2022, pp. 17 644–17 655.

[5] M. Liu, Y. Luo, K. Uchino, K. Maruhashi, and S. Ji, "Generating 3d molecules for target protein binding," in *International Conference on Machine Learning (ICML)*, 2022.

[6] Z. Zhang, Y. Min, S. Zheng, and Q. Liu, "Molecule generation for target protein binding with structural motifs," in *The eleventh international conference on learning representations*, 2023.

[7] Z. Zhang and Q. Liu, "Learning subpocket prototypes for generalizable structure-based drug design," in *International Conference on Machine Learning*. PMLR, 2023, pp. 41 382–41 398.

[8] J. Guan, W. W. Qian, X. Peng, Y. Su, J. Peng, and J. Ma, "3d equivariant diffusion for target-aware molecule generation and affinity prediction," *arXiv preprint arXiv:2303.03543*, 2023.

[9] J. Guan, X. Zhou, Y. Yang, Y. Bao, J. Peng, J. Ma, Q. Liu, L. Wang, and Q. Gu, "Decompdiff: diffusion models with decomposed priors for structure-based drug design," *arXiv preprint arXiv:2403.07902*, 2024.

[10] Z. Huang, L. Yang, X. Zhou, Z. Zhang, W. Zhang, X. Zheng, J. Chen, Y. Wang, B. Cui, and W. Yang, "Protein-ligand interaction prior for binding-aware 3d molecule diffusion models," in *The Twelfth International Conference on Learning Representations*, 2024.

[11] S. Gu, M. Xu, A. Powers, W. Nie, T. Geffner, K. Kreis, J. Leskovec, A. Vahdat, and S. Ermon, "Aligning target-aware molecule diffusion models with exact energy optimization," *Advances in Neural Information Processing Systems*, vol. 37, pp. 44 040–44 063, 2024.

[12] X. Liu, C. Gong, and Q. Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," *arXiv preprint arXiv:2209.03003*, 2022.

[13] A. Graves, R. K. Srivastava, T. Atkinson, and F. Gomez, "Bayesian flow networks," *arXiv preprint arXiv:2308.07037*, 2023.

[14] D. Zhang, C. Gong, and Q. Liu, "Rectified flow for structure based drug design," *arXiv preprint arXiv:2412.01174*, 2024.

[15] Y. Qu, K. Qiu, Y. Song, J. Gong, J. Han, M. Zheng, H. Zhou, and W.-Y. Ma, "Molcraft: Structure-based drug design in continuous parameter space," *arXiv preprint arXiv:2404.12141*, 2024.

[16] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022.

[17] P. G. Francoeur, T. Masuda, J. Sunseri, A. Jia, R. B. Iovanisci, I. Snyder, and D. R. Koes, "Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design," *Journal of chemical information and modeling*, vol. 60, no. 9, pp. 4200–4215, 2020.

[18] A. C. Anderson, "The process of structure-based drug design," *Chemistry & biology*, vol. 10, no. 9, pp. 787–797, 2003.

[19] M. Skalic, D. Sabbadin, B. Sattarov, S. Sciabola, and G. De Fabritiis, "From target to drug: generative modeling for the multimodal structure-based ligand design," *Molecular pharmaceutics*, vol. 16, no. 10, pp. 4282–4291, 2019.

[20] H. Qian, C. Lin, D. Zhao, S. Tu, and L. Xu, "Alphadrug: protein target specific de novo molecular generation," *PNAS nexus*, vol. 1, no. 4, p. pgac227, 2022.

[21] C. Tan, Z. Gao, and S. Z. Li, "Target-aware molecular graph generation," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2023, pp. 410–427.

[22] M. Ragoza, T. Masuda, and D. R. Koes, "Generating 3d molecules conditional on receptor binding sites with deep generative models," *Chemical science*, vol. 13, no. 9, pp. 2701–2713, 2022.

[23] H. Qian, W. Huang, S. Tu, and L. Xu, "Kgdiff: towards explainable target-aware molecule generation with knowledge guidance," *Briefings in Bioinformatics*, vol. 25, no. 1, p. bbad435, 2024.

[24] V. Dorna, D. Subhalingam, K. Kolluru, S. Tuli, M. Singh, S. Singal, N. Krishnan, and S. Ranu, "Tagmol: Target-aware gradient-guided molecule generation," *arXiv preprint arXiv:2406.01650*, 2024.

[25] Q. Dao, H. Phung, B. Nguyen, and A. Tran, "Flow matching in latent space," *arXiv preprint arXiv:2307.08698*, 2023.

[26] A.-A. Pooladian, H. Ben-Hamu, C. Domingo-Enrich, B. Amos, Y. Lipman, and R. T. Chen, "Multisample flow matching: Straightening flows with minibatch couplings," *arXiv preprint arXiv:2304.14772*, 2023.

[27] J. Li, C. Cheng, Z. Wu, R. Guo, S. Luo, Z. Ren, J. Peng, and J. Ma, "Full-atom peptide design based on multi-modal flow matching," *arXiv preprint arXiv:2406.00735*, 2024.

[28] Z. Zhang, M. Zitnik, and Q. Liu, "Generalized protein pocket generation with prior-informed flow matching," *arXiv preprint arXiv:2409.19520*, 2024.

[29] A. Tong, K. Fatras, N. Malkin, G. Huguet, Y. Zhang, J. Rector-Brooks, G. Wolf, and Y. Bengio, "Improving and generalizing flow-based generative models with minibatch optimal transport," *arXiv preprint arXiv:2302.00482*, 2023.

[30] R. T. Chen and Y. Lipman, "Riemannian flow matching on general geometries," *arXiv e-prints*, pp. arXiv–2302, 2023.

[31] Y. Song, J. Gong, M. Xu, Z. Cao, Y. Lan, S. Ermon, H. Zhou, and W.-Y. Ma, "Equivariant flow matching with hybrid probability transport for 3d molecule generation," *Advances in Neural Information Processing Systems*, vol. 36, pp. 549–568, 2023.

[32] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical solution of initial-value problems in differential-algebraic equations*. SIAM, 1995.

[33] J. Köhler, L. Klein, and F. Noé, "Equivariant flows: exact likelihood generative learning for symmetric densities," in *International conference on machine learning*. PMLR, 2020, pp. 5361–5370.

[34] V. Garcia Satorras, E. Hoogeboom, F. Fuchs, I. Posner, and M. Welling, "E (n) equivariant normalizing flows," *Advances in Neural Information Processing Systems*, vol. 34, pp. 4181–4192, 2021.

[35] M. Xu, L. Yu, Y. Song, C. Shi, S. Ermon, and J. Tang, "Geodiff: A geometric diffusion model for molecular conformation generation," *arXiv preprint arXiv:2203.02923*, 2022.

[36] E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling, "Equivariant diffusion for molecule generation in 3d," in *International conference on machine learning*. PMLR, 2022, pp. 8867–8887.

[37] Q. Zheng, M. Le, N. Shaul, Y. Lipman, A. Grover, and R. T. Chen, "Guided flows for generative modeling and decision making," *arXiv preprint arXiv:2311.13443*, 2023.

[38] J. Liang, C. Woodward, and H. Edelsbrunner, "Anatomy of protein pockets and cavities: measurement of binding site geometry and implications for ligand design," *Protein science*, vol. 7, no. 9, pp. 1884–1897, 1998.

[39] C. Bilodeau, W. Jin, T. Jaakkola, R. Barzilay, and K. F. Jensen, "Generative models for molecular discovery: Recent advances and challenges," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 12, no. 5, p. e1608, 2022.

[40] J. V. d. S. Guerra, H. V. Ribeiro-Filho, G. E. Jara, L. O. Bortot, J. G. d. C. Pereira, and P. S. Lopes-de Oliveira, "pykvfinder: an efficient and integrable python package for biomolecular cavity detection and characterization in data science," *BMC bioinformatics*, vol. 22, pp. 1–13, 2021.

[41] O. Trott and A. J. Olson, "Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading," *Journal of computational chemistry*, vol. 31, no. 2, pp. 455–461, 2010.

[42] G. R. Bickerton, G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins, "Quantifying the chemical beauty of drugs," *Nature chemistry*, vol. 4, no. 2, pp. 90–98, 2012.

[43] P. Ertl and A. Schuffenhauer, "Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions," *Journal of cheminformatics*, vol. 1, pp. 1–11, 2009.

[44] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.

[45] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International conference on machine learning*. PMLR, 2021, pp. 8162–8171.

[46] L. Hu, M. L. Benson, R. D. Smith, M. G. Lerner, and H. A. Carlson, "Binding moad (mother of all databases)," *Proteins: Structure, Function, and Bioinformatics*, vol. 60, no. 3, pp. 333–340, 2005.

[47] A. Bairoch, "The enzyme database in 2000," *Nucleic acids research*, vol. 28, no. 1, pp. 304–305, 2000.

[48] A. Schneuing, Y. Du, C. Harris, A. Jamasb, I. Igashov, W. Du, T. Blundell, P. Lió, C. Gomes, M. Welling *et al.*, "Structure-based drug design with equivariant diffusion models," *arXiv preprint arXiv:2210.13695*, 2022.

[49] H. Lin, G. Zhao, O. Zhang, Y. Huang, L. Wu, Z. Liu, S. Li, C. Tan, Z. Gao, and S. Z. Li, "Cbgbench: fill in the blank of protein-molecule complex binding graph," *arXiv preprint arXiv:2406.10840*, 2024.

[50] C. Harris, K. Didi, A. R. Jamasb, C. K. Joshi, S. V. Mathis, P. Lio, and T. L. Blundell, "Posecheck: Generative models for 3d structure-based drug design produce unrealistic poses," 2023.

[51] Z. Zhang, M. Wang, and Q. Liu, "Flexsbdd: Structure-based drug design with flexible protein modeling," *Advances in Neural Information Processing Systems*, vol. 37, pp. 53 918–53 944, 2024.

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: The main claims in the abstract and introduction accurately reflect the paper's contributions and scope.

   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: Limitations are discussed in Appendix F.3.

   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: [Yes]

Justification: Yes, the paper provides the full set of assumptions and complete, correct proofs in Appendix A.

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: Yes, the paper fully discloses all the necessary information to reproduce the main experimental results.

   Guidelines:

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

5. **Open access to data and code**

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

Answer: [Yes]

Justification: Yes, we have provided all the codes needed to reproduce the results presented.

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: Implementation details are provided in Appendix D.

   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 follow previous works and do not report error bars. For each protein target, 100 molecules are sampled. We believe that reporting the mean and median well reflects the overall performance.

   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 computer resources details are provided in Appendix D.

   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 presented in the paper fully complies with the NeurIPS Code of Ethics in all respects.

   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: Yes, broader impacts are discussed in Appendix F.3.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All assets are properly cited 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]

    Justification: This paper does not release new assets.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [NA]

    Justification: No crowdsourcing and research with human subjects.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: This paper does not involve LLMs as any significant, original, or non-standard components.

Guidelines:

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

A Proofs

A.1 Proof of Equivariant Generation

An essential requirement for generated molecules is translational and rotational equivariance to the ligand–protein complex. Since atom types are always invariant to SE(3)-transformation, we only need to consider the transformation behavior of atomic coordinates. As the probability path used during training is identical to that of TargetDiff [8]—and the path of TargetDiff is invariant by design—we only have to focus on the generation process. Let T_g denote the group of SE-(3) transformation, e.g. $T_g(\mathbf{x}) = \mathbf{R}\mathbf{x} + \mathbf{b}$, where $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the rotation matrix and $\mathbf{b} \in \mathbb{R}^3$ is the translation vector.

First, we move the complex to achieve zero Center-of-Mass (CoM) on protein positions by the following linear transformation:

$$[\bar{\mathbf{x}}_M, \bar{\mathbf{x}}_P] = Q(\mathbf{x}_M, \mathbf{x}_P), \qquad where \quad Q = I_3 \otimes \begin{pmatrix} I_{N_M} & -\frac{1}{N_P}\mathbf{1}_{N_M}\mathbf{1}_{N_P}^T \\ \mathbf{0} & I_{N_P} - \frac{1}{N_P}\mathbf{1}_{N_P}\mathbf{1}_{N_P}^T \end{pmatrix}. \tag{17}$$

Then flow can be initialized from standard Gaussian, and during the generation process $\bar{\mathbf{x}}_0$ can be sampled from a standard Gaussian distribution. Meanwhile, for any T_g applied to the protein and molecule, we have $T_g(\bar{\mathbf{x}}_M, \bar{\mathbf{x}}_P) = \mathbf{R}(\bar{\mathbf{x}}_M, \bar{\mathbf{x}}_P)$. Thus $\bar{\mathbf{x}}_M$ and $\bar{\mathbf{x}}_P$ inherently satisfy translational invariance by definition. For brevity, we denote $\bar{\mathbf{x}}_M$, $\bar{\mathbf{x}}_P$ by $\mathbf{x}_M$, $\mathbf{x}_P$ in the following.

Given that ϕ_θ is an SE(3)-equivariant GNN, it follows from Eq. 9 that $\hat{\mathbf{x}}_1$ is SE-(3) equivariant to $\mathbf{x}_t$ and $\mathbf{x}_P$. At $t = 0$, based on Eq. 5 and 8 we obtain

$$\begin{aligned} \mathbf{x}_{\Delta t} &= \mathbf{x}_0 + v_\theta \Delta t \\ &= \mathbf{x}_0 + \frac{(\sqrt{\bar{\alpha}_{1-t}})'}{1-\bar{\alpha}_{1-t}}(\sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_0 - \hat{\mathbf{x}}_1). \end{aligned} \tag{18}$$

We can prove that $\mathbf{x}_{\Delta t}$ is SE(3)-equivariant to $\mathbf{x}_0$ and $\mathbf{x}_P$ as follows

$$\begin{aligned} T_g(\mathbf{x}_{\Delta t}(\mathbf{x}_0, \mathbf{x}_P)) &= T_g(\mathbf{x}_0) + \frac{(\sqrt{\bar{\alpha}_{1-t}})'}{1-\bar{\alpha}_{1-t}}(\sqrt{\bar{\alpha}_{1-t}}T_g(\mathbf{x}_0) - T_g(\hat{\mathbf{x}}_1)) \\ &= \mathbf{R}\mathbf{x}_0 + \mathbf{b} + \frac{(\sqrt{\bar{\alpha}_{1-t}})'}{1-\bar{\alpha}_{1-t}}(\sqrt{\bar{\alpha}_{1-t}}\mathbf{R}\mathbf{x}_0 + \sqrt{\bar{\alpha}_{1-t}}\mathbf{b} - \mathbf{R}\hat{\mathbf{x}}_1 - \mathbf{b}) \\ &= \mathbf{R}\mathbf{x}_0 + \frac{(\sqrt{\bar{\alpha}_{1-t}})'}{1-\bar{\alpha}_{1-t}}\mathbf{R}(\sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_0 - \hat{\mathbf{x}}_1)\Delta t + \tilde{\mathbf{b}} \\ &= \mathbf{x}_{\Delta t}(\mathbf{R}(\mathbf{x}_0, \mathbf{x}_P)) + \tilde{\mathbf{b}}, \end{aligned} \tag{19}$$

where $\tilde{\mathbf{b}} = [1 - (\sqrt{\bar{\alpha}_{1-t}})'(1 + \sqrt{\bar{\alpha}_{1-t}})\Delta t]\,\mathbf{b}$. Since translation invariance is achieved by moving CoM of the protein atoms to zero, $\tilde{\mathbf{b}}$ can be omitted thereby $T_g(\mathbf{x}_{\Delta t}(\mathbf{x}_0, \mathbf{x}_P)) = \mathbf{x}_{\Delta t}(\mathbf{R}(\mathbf{x}_0, \mathbf{x}_P)) = \mathbf{x}_{\Delta t}(T_g(\mathbf{x}_0, \mathbf{x}_P))$. Following the same reasoning, it can be derived that $\mathbf{x}_{t+\Delta t}$ is SE(3)-equivariant to $\mathbf{x}_t$ and $\mathbf{x}_P$. By mathematical induction, we conclude that $\mathbf{x}_1$ is SE(3)-equivariant to $\mathbf{x}_0$ and $\mathbf{x}_P$. Consequently, the entire generation process is SE(3)-equivariant with respect to $\mathbf{x}_0$ and $\mathbf{x}_P$.

A.2 Vector Field of Type Flow

Theorem 3 in [16] propose that for a flow of the form $\psi_t(x) = r_t(x_1)x + s_t(x_1)$, the corresponding vector field is given by:

$$u_t(x|x_1) = \frac{r_t'(x_1)}{r_t(x_1)}(x - s_t(x_1)) + s_t'(x_1), \tag{20}$$

where $r_t(x_1)$ and $s_t(x_1)$ can be arbitrary differentiable function satisfying the desired boundary conditions. Let $s_t = \bar{\alpha}_{1-t}\mathbf{a}_1$, $r_t = 1 - \bar{\alpha}_{1-t}$, then Eq. 20 is rewritten as

$$u_t(x|x_1) = -\frac{\bar{\alpha}_{1-t}'}{1-\bar{\alpha}_{1-t}}(x - \bar{\alpha}_{1-t}x_1) + \bar{\alpha}_{1-t}'x_1. \tag{21}$$

For atomic types we have $\mathbf{c}(\mathbf{a}, \mathbf{a}_1) = \bar{\alpha}_{1-t}\mathbf{a}_1 + (1 - \bar{\alpha}_{1-t})\mathbf{a}_0$, which can be substituted into Eq. 21 to obtain:

$$\begin{aligned} u_t(\mathbf{c}(\mathbf{a}, \mathbf{a}_1)|\mathbf{a}_1) &= -\frac{\bar{\alpha}'_{1-t}}{1-\bar{\alpha}_{1-t}}(\bar{\alpha}_{1-t}\mathbf{a}_1 + (1-\bar{\alpha}_{1-t})\mathbf{a}_0 - \bar{\alpha}_{1-t}\mathbf{a}_1) + \bar{\alpha}'_{1-t}\mathbf{a}_1 \\ &= -\frac{\bar{\alpha}'_{1-t}}{1-\bar{\alpha}_{1-t}}(1-\bar{\alpha}_{1-t})\mathbf{a}_0 + \bar{\alpha}'_{1-t}\mathbf{a}_1 \\ &= \bar{\alpha}'_{1-t}(\mathbf{a}_1 - \mathbf{a}_0). \end{aligned} \tag{22}$$

A.3 Prior Guidance of Vector Field

According to *Lemma 1* in [37], the guided vector field for a Gaussian Path $p_t(x|x_1) = \mathcal{N}(x|\mu_t x_1, \sigma_t^2 I)$ can be expressed as:

$$u_t(x|y) = a_t x_t + b_t \nabla \log p(x_t|y), \tag{23}$$

$$where \quad a_t = \frac{\mu'_t}{\mu_t}, \quad b_t = (\mu'_t\sigma_t - \mu_t\sigma'_t)\frac{\sigma_t}{\mu_t}. \tag{24}$$

The probability path for atomic coordinates follows the Gaussian distribution with $\mu_t = \sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_1$ and $\sigma_t = \sqrt{1-\bar{\alpha}_{1-t}}$. Substituting into Eq. 24 yields:

$$a_t = \frac{(\sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_1)'}{\sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_1} = \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}}, \tag{25}$$

$$\begin{aligned} b_t &= \left[\frac{\bar{\alpha}'_{1-t}\sqrt{1-\bar{\alpha}_{1-t}}}{2\sqrt{\bar{\alpha}_{1-t}}}x_1 + \frac{\bar{\alpha}'_{1-t}\sqrt{\bar{\alpha}_{1-t}}}{2\sqrt{1-\bar{\alpha}_{1-t}}}x_1\right]\frac{\sqrt{1-\bar{\alpha}_{1-t}}}{x_1\sqrt{\bar{\alpha}_{1-t}}} \\ &= \frac{\bar{\alpha}'_{1-t}}{2}\left(\frac{1-\bar{\alpha}_{1-t}}{\bar{\alpha}_{1-t}} + 1\right) \\ &= \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}}. \end{aligned} \tag{26}$$

Thus we have $a_t = b_t = \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}}$. Substituting this into Eq. 23 gives the guided vector field for atomic coordinates as follows:

$$\tilde{v}_\theta^x(\mathbf{m}_t, \mathbf{p}, y, t) = \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}}(\mathbf{x}_t + \nabla \log p(\mathbf{x}_t|y)). \tag{27}$$

According to the Bayes' rule, log-probability function can be decomposed as:

$$\begin{aligned} \nabla \log p(\mathbf{x}_t|y) &= \nabla(\log p(\mathbf{x}_t) + \log p(y|\mathbf{x}_t) - \log p(y)) \\ &= \nabla \log p(\mathbf{x}_t) + \nabla \log p(y|\mathbf{x}_t), \end{aligned} \tag{28}$$

where $\nabla \log p(y|\mathbf{x}_t)$ can be calculated using predictor with Eq. 15. Then we can reformulate Eq. 27 as:

$$\begin{aligned} \tilde{v}_\theta^x(\mathbf{m}_t, \mathbf{p}, y, t) &= \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}}(\mathbf{x}_t + \nabla \log p(\mathbf{x}_t)) + \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}}\nabla \log p(y|\mathbf{x}_t) \\ &= v_\theta^x(\mathbf{m}_t, \mathbf{p}, t) + \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}}\nabla \log p(y|\mathbf{x}_t). \end{aligned} \tag{29}$$

Moreover, scaling the predictor gradient by a constant factor $\gamma > 1$ is necessary to enhance the alignment of the generated samples with the desired condition. Similar to [44], we know that $\gamma \cdot \nabla \log p(y|\mathbf{x}_t) = \nabla \log \frac{1}{Z}p(y|\mathbf{x}_t)^\gamma$, where Z is an arbitrary constant. Consequently, the conditioning procedure remains theoretically grounded in a re-normalized predictor distribution proportional to $p(y|\mathbf{x}_t)^\gamma$. Higher probabilities are emphasized by the exponent when $\gamma > 1$, leading to a distribution more peaked than the original $p(y|\mathbf{x}_t)$. Thus increasing the gradient scale focuses more on the modes of the predictor, which promotes the generation of molecules with higher binding affinity.

A.4 Proof of Noise Injection Effectiveness

As introduced in Sec. 4.3, the predicted normalized number of atoms $\hat{n}_M$ in the generated molecule is obtained via the atom number predictor. For simplicity, we denote it as n in the following. Using n directly corresponds to deterministic sampling, i.e., $\tilde{n} = n$, whereas adding a small Gaussian noise corresponds to stochastic sampling with $\tilde{n} \sim \mathcal{N}(n, \delta^2)$. Let $f(n)$ denote the binding affinity as a function of the atom number n. The objective is to maximize the expected binding affinity, defined as $R = \mathbb{E}_{\tilde{n}}[f(\tilde{n})]$.

Under deterministic sampling, we have

$$R = \mathbb{E}_n[f(n)] = f(n). \tag{30}$$

When Gaussian noise is injected,

$$R = \mathbb{E}_{\tilde{n} \sim \mathcal{N}(n, \delta^2)}[f(\tilde{n})]. \tag{31}$$

The second-order Taylor expansion of $f(\tilde{n})$ at $n = \tilde{n}$ is given by:

$$f(\tilde{n}) \approx f(n) + f'(n)(\tilde{n} - n) + \frac{1}{2} f''(n)(\tilde{n} - n)^2. \tag{32}$$

Since $\tilde{n} \sim \mathcal{N}(n, \delta^2)$, the expectation can be rewritten as

$$\mathbb{E}_{\tilde{n}}[f(\tilde{n})] \approx f(n) + \frac{1}{2} f''(n)\delta^2. \tag{33}$$

As a result, the expected affinity increases over the deterministic value $f(n)$ if $f''(n) > 0$, i.e., if the function is locally convex around n. In practice, we observe that adding a small amount of noise (e.g., $\delta = 0.01$) consistently improves the binding affinity of the generated molecules (see Tab. 7). This implies that n lies near a local minimum of the affinity function, and noise injection effectively allows exploration of nearby data points that lead to better binding performance.

B Training Objective

The training objectives for atomic coordinates $\mathbf{x}$, atomic types $\mathbf{a}$, protein-ligand interactions y, and atom numbers n_M are defined as follows:

$$\mathcal{L}_x = || \mathbf{x}_1 - \hat{\mathbf{x}}_1 ||^2 \qquad \mathcal{L}_a = \sum_k c(\mathbf{a}_t, \mathbf{a}_1)_k \log \frac{c(\mathbf{a}_t, \mathbf{a}_1)_k}{c(\mathbf{a}_t, \hat{\mathbf{a}}_1)_k} \tag{34}$$

$$\mathcal{L}_y = || y - \hat{y} ||^2 \qquad \mathcal{L}_n = || n_M - \hat{n}_M ||^2, \tag{35}$$

where the loss for $\mathbf{a}$ is KL-divergence of categorical distributions, while the others are optimized using Mean Squared Error. KGDiff demonstrates that jointly training $\mathbf{x}$, $\mathbf{a}$, and y is more effective than training separately. Following this strategy, we adopt a joint training scheme with the combined loss defined as:

$$\mathcal{L} = \mathcal{L}_x + \lambda \mathcal{L}_a + \omega \mathcal{L}_y, \tag{36}$$

where λ and ω are scaling factors. The atom number predictor is trained independently.

C Algorithm

The training and sampling procedure of PAFlow are summarized below.

Algorithm 1 Training Procedure of PAFlow

Input: Protein-ligand complex $\{\mathbf{p}, \mathbf{m}, y\}_{i=1}^{N}$, neural network ϕ_θ, atom type loss weight λ and predictor loss weight ω

while ϕ_θ not converage **do**
  Sample $t \sim \mathcal{U}(0,1)$
  Shift the complex to make the CoM of protein atoms zero
  Obtain $\mathbf{x}_t$ and $\mathbf{a}_t$:
    $\mathbf{x}_t = \sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_1 + \sqrt{(1-\bar{\alpha}_{1-t})}\varepsilon$, where $\varepsilon \in \mathcal{N}(0, \mathbf{I})$
    $\log \mathbf{c} = \log(\bar{\alpha}_{1-t}\mathbf{a}_1 + (1-\bar{\alpha}_{1-t})/K)$
    $\mathbf{a}_t = one\_hot(\text{argmax}_i[g_i + \log c_i])$, where $g \sim \text{Gumbel}(0,1)$
  Predict $[\hat{\mathbf{x}}_1, \hat{\mathbf{a}}_1, \hat{y}]$ with ϕ_θ: $[\hat{\mathbf{x}}_1, \hat{\mathbf{a}}_1, \hat{y}] = \phi_\theta([\mathbf{x}_t, \mathbf{a}_t], t, \mathbf{p})$
  Compute the loss function using Eq. 36:
    $\mathcal{L} = || \mathbf{x}_1 - \hat{\mathbf{x}}_1 ||^2 + \lambda \text{KL}(\mathbf{c}(\mathbf{a}_t, \mathbf{a}_1) || \mathbf{c}(\mathbf{a}_t, \hat{\mathbf{a}}_1)) + \omega || y - \hat{y} ||^2$
  Update θ by minimizing $\mathcal{L}$
end while

Algorithm 2 Sampling Procedure of PAFlow

Input: Protein pocket $\mathbf{p}$, the learned model ϕ_θ, pretrained atom number predictor φ_ϵ, scale factor γ, standard deviation of Gaussian δ, total number of sampling steps T

Output: Generated ligand molecule $\mathbf{m}$

Obtain the number of atoms in $\mathbf{m}$ using Eq. 16
Move the CoM of protein atoms to zero
Initialize ligand atom coordinates $\mathbf{x}_0$ and atom types $\mathbf{a}_0$
$steps \leftarrow 0$, $t \leftarrow 0$, $\Delta t = 1/T$
while $steps \leq T - 1$ **do**
  Predict $[\hat{\mathbf{x}}_1, \hat{\mathbf{a}}_1, \hat{y}]$ with ϕ_θ: $[\hat{\mathbf{x}}_1, \hat{\mathbf{a}}_1, \hat{y}] = \phi_\theta([\mathbf{x}_t, \mathbf{a}_t], t, \mathbf{p})$
  Compute vector fields:
    $\tilde{v}_\theta^x = \frac{(\sqrt{\bar{\alpha}_{1-t}})'}{1-\bar{\alpha}_{1-t}}(\sqrt{\bar{\alpha}_{1-t}}\mathbf{x}_t - \hat{\mathbf{x}}_1) + \gamma \frac{\bar{\alpha}'_{1-t}}{2\bar{\alpha}_{1-t}} \nabla \log p(y=1|\mathbf{m}_t)$
    $v_\theta^a = \bar{\alpha}'_{1-t}(\hat{\mathbf{a}}_1 - \mathbf{a}_0)$
  Update $\mathbf{x}_{t+\Delta t}$ and $\mathbf{a}_{t+\Delta t}$:
    $\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + \tilde{v}_\theta^x \Delta t$
    $\mathbf{c}(\mathbf{a}_{t+\Delta t}, \hat{\mathbf{a}}_1) = \mathbf{c}(\mathbf{a}_t, \hat{\mathbf{a}}_1) + v_\theta^a \Delta t$
    Sample $\mathbf{a}_t$ from $\mathcal{C}(\mathbf{a}_{t+\Delta t} | \mathbf{c}(\mathbf{a}_{t+\Delta t}, \hat{\mathbf{a}}_1))$
end while

D Implementation Details

Data Following [8], the protein atom features include a one-hot indicator of the element type (H, C, N, O, S, Se), a one-hot vector indicating the amino acid type (20 dimension), and a one-dim flag indicating whether the atom belongs to the protein backbone. Ligand atom types are also represented as one-hot vectors that encode the element type (C, N, O, F, P, S, Cl) along with aromatic information. Two separate single-layer MLPs are utilized to embed the protein and ligand features into a 128-dimensional latent space.

The protein-ligand complex is adaptively represented as a k-nearest neighbors (knn) graph at the l-th layer, constructed using the known protein atom coordinates and the current ligand atom coordinates generated by the $l-1$-th layer. $k = 32$ is set in experiments. The edge features are obtained as the outer product between distance embeddings and bond type, while distances are expanded using radial basis functions centered at 20 points distributed between 0 Å and 10 Å. Bond types are encoded as a 4-dimensional one-hot vector indicating whether the connection is protein–protein, ligand–ligand, protein-to-ligand, or ligand-to-protein.

Model Information The SE(3)-equivariant network ϕ_θ consists of 9 equivariant layers, where each layer is implemented as a transformer with $\text{hidden\_dim} = 128$ and $\text{n\_head} = 16$. The key/value embeddings and attention scores are generated through 2-layer MLPs with ReLU activation and Layer Normalization. The protein-ligand interaction predictor employs a 2-layer MLP with ShiftedSoftplus activation. For the atom number predictor φ_ϵ, we adopt a 4-layer MLP with hidden dimensions of 128, 256, and 128. Each hidden layer is followed by Batch Normalization and ShiftedSoftplus activation, along with Dropout for regularization. The final output is produced by a linear projection to a scalar. We apply a sigmoid β schedule with $\beta_1 = 1\text{e}{-7}$ and $\beta_0 = 2\text{e}{-3}$ for atom coordinates, and a cosine β schedule proposed by [45] with $s = 0.01$ for atom types.

Training Details When training the SE(3)-Equivariant network, the Adam optimizer is employed to speed up convergence with $\text{init\_learning\_rate} = 5\text{e} - 4$, $\text{betas} = (0.95, 0.999)$, $\text{batch\_size} = 4$ and $\text{clip\_gradient\_norm} = 8$. The learning rate is scheduled to decay exponentially with a decay factor of 0.95, and we set the minimum learning rate to be $1\text{e} - 6$. We decay the learning rate if the validation loss is not improved for 15 consecutive evaluations. Following [23], we set λ, the weight corresponding to the atom feature term in loss function, is 100, and ω, the weight corresponding to the expert network, is 1. When generating, the scaling factor γ is 350 and the number of sampling steps is set to 50.

During the training of the atom number predictor, we use the Adam optimizer with $\text{init\_learning\_rate} = 5\text{e} - 4$, $\text{batch\_size} = 256$, and $\text{betas} = (0.95, 0.999)$. The learning rate follows an exponential decay schedule with a decay factor of 0.8, and a minimum learning rate of $1\text{e}{-5}$ is enforced. The learning rate is decayed when the validation loss does not improve for 5 consecutive evaluations. All models are trained on one NVIDIA A100 GPU (40 GB).

Table 5: Overview of the properties of the reference ligands and the molecules generated by different methods on the **Binding MOAD** dataset. (↑)/(↓) denotes a larger / smaller number is better. Top 2 results are highlighted with **bold text** and underlined text, respectively.

Method	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.
Ref	-6.32	-5.82	-7.22	-6.65	-7.95	-7.67	-	-	0.65	0.63	0.35	0.34	-	-
Pocket2Mol	-4.85	-4.55	-5.95	-5.54	-6.66	-6.31	46.2%	33.5%	**0.62**	**0.62**	**0.83**	**0.84**	**0.83**	**0.83**
TargetDiff	-5.60	-5.49	-6.48	-6.15	-7.47	-7.20	62.3%	63.0%	0.54	0.55	0.62	0.62	0.75	0.76
IPDiff	-6.64	-6.80	-7.50	-7.31	-8.57	-8.25	77.6%	86.4%	0.53	0.53	0.58	0.58	0.74	0.74
MolCRAFT	-7.33	-6.90	-7.54	-7.10	-7.90	-7.50	65.3%	77.4%	0.55	0.56	0.70	0.69	0.74	0.76
PAFlow	**-9.12**	**-8.87**	**-9.18**	**-8.78**	**-9.69**	**-9.19**	**83.4%**	**96.6%**	0.48	0.46	0.58	0.58	0.70	0.70

E More Experimental Results

E.1 Results on Binding MOAD

To investigate the performance of PAFlow across different datasets, we evaluate it on the Binding MOAD dataset [46] without additional retraining, which is a commonly used benchmark comprising experimentally determined protein–ligand complexes. We apply filtering based on the proteins' enzyme commission numbers [47] and exclude entries that could not be processed, resulting in 100 protein-ligand pairs for testing following previous work [48]. PAFlow and the baseline methods generate 100 molecules for each protein pocket, with the evaluation results summarized in Table 5. PAFlow achieves the best performance across all binding-related metrics, outperforming the second-best method by 24.4%, 21.8%, and 13.1% on Avg. Vina Score, Vina Min, and Vina Dock, respectively. Although there is a slight decline in molecular properties, they remain within a reasonable range. It is worth noting that PAFlow is not trained on the Binding MOAD dataset, indicating its strong generalization capability. Furthermore, it is reasonable to expect that performance could be further improved by training on this dataset.

E.2 Effect of Sampling Steps

We conduct an ablation study on the sampling steps of PAFlow by generating 10 molecules each for 100 test proteins using sampling strategies with different step (20, 50, 80, 200). The resulting Vina Score, Vina Dock, QED, and SA curves are shown in Fig. 6. It can be observed that PAFlow achieves very competitive results on affinity-related metrics even with only 20 steps, outperforming all baseline methods listed in Tab. 1. In contrast, diffusion-based models typically require 1000 sampling steps, indicating that PAFlow significantly improves sampling efficiency. Moreover, increasing the sampling step further enhances performance across all metrics, since more sampling steps allows model to more closely approximate the probability flow, leading to a smoother and more accurate trajectory toward the target distribution. Considering the trade-off between sampling efficiency and overall molecular quality, the number of sampling steps is set to 50 for PAFlow.

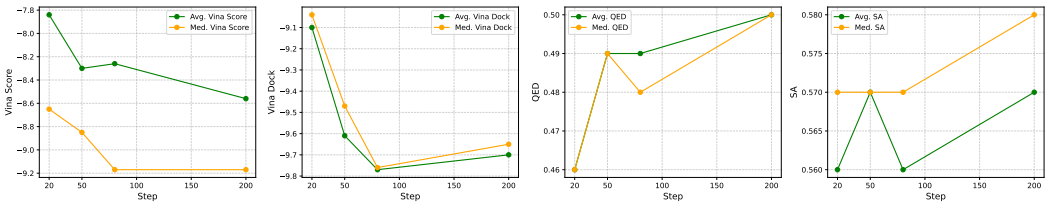


Figure 6: Ablation study on sampling steps. The performance of Vina Score, Vina Dock, QED, and SA under different numbers of sampling steps (20, 50, 80, and 200) is reported. Lower Vina Score and Vina Dock, as well as higher QED and SA, indicate better results.

E.3 Effect of Scaling Factor in Guidance

The scaling factor γ, introduced in Sec. 4.2, is utilized to control the strength of guidance applied to the vector field. We generate 10 molecules each for 100 test target proteins using PAFlow with different γ values (0, 10, 50, 150, 250, 350, 400). The results are presented in Fig. 7. A larger γ corresponds to stronger guidance, which leads to higher binding affinities for the generated molecules. However, a clear trade-off between binding affinity and QED can be observed, where as binding affinity improves, QED tends to decrease. This is consistent with the discussion in Sec. A.3 that larger gradient scales concentrate more on the modes of the predictor, potentially at the expense of other molecular properties. Considering both binding affinity and molecular properties, we select $\gamma = 350$ for PAFlow. Additionally, γ can be tuned during molecule generation to achieve different trade-offs between binding affinity and molecular properties, allowing adaptation to various drug design scenarios.

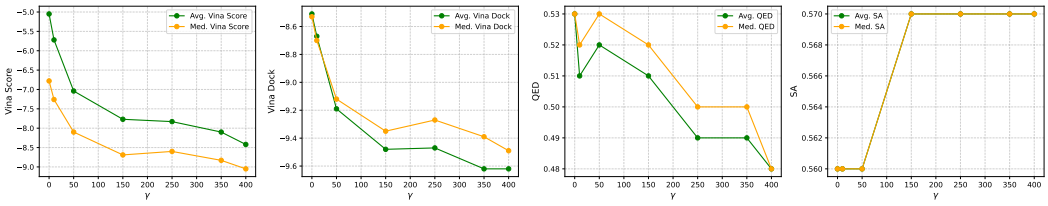


Figure 7: Ablation study of PAFlow under different guidance scaling factor γ. The performance on Vina Score, Vina Dock, QED, and SA is reported. In the SA plot, only one curve is shown because the average and median SA are identical.

E.4 Effect of the Gaussian standard deviation

To investigate the impact of different δ on molecular quality, we generated 10 molecules for each of 100 test proteins, where δ is the standard deviation of small Gaussian noise added to the output of the atom count predictor. Fig. 8 reports the averages of Vina Score, QED, SA, and the proportion of complete molecules under δ=(0, 0.02, 0.04, 0.06, 0.08, 0.1). When δ is small, both binding affinity and the proportion of complete molecules improve while maintaining molecular properties. This

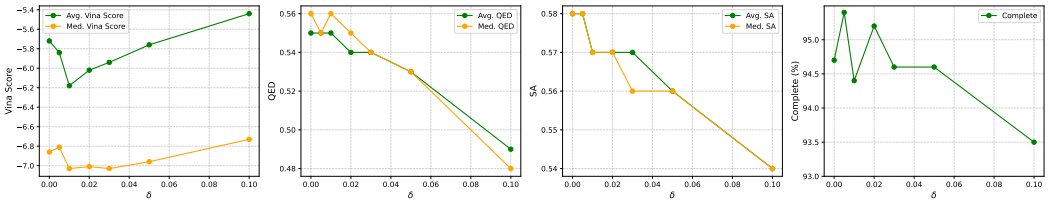


Figure 8: Ablation study using Gaussian noise with different standard deviations added to the predicted atom number. Vina Score, QED, SA, and the percentage of complete molecules are reported.

is because the added randomness introduces robustness that deterministic outputs lack, helping to compensate for minor prediction errors. However, when the standard deviation is larger, the noise injected overly disturbs the predicted values, leading to a decline in molecular quality. For PAFlow, we chose $\delta = 0.01$.

E.5 Factors Influencing the Number of Atoms in Ligands

To validate the effectiveness of the protein features used for predicting the atom numbers in generated molecules (including the number of pocket atoms N_P, binding site volume V, binding site surface area A, and space size S), we conduct a study on their relationships using the training dataset. V and A are computed using pyKVFinder, while S is the median value of the top 10 largest pairwise distances among protein atoms. Notably, existing non-autoregressive methods commonly use S alone to determine the atom number. The relationships between these features and N_M are shown in Fig. 9. All features exhibit significant correlations ($p < 0.05$) with N_M, supporting the reasonableness of using them for atom number prediction. Additionally, we compare the performance of using only V (denoted as Volume) versus using all four features (denoted as All), as reported in Tab. 6. The test loss refers to the loss of the trained predictor evaluated on the test set. Results show that incorporating all features leads to a lower test loss and better molecule quality, indicating that providing richer protein information allows the predictor to more accurately infer the appropriate N_M.

Table 6: Comparison of evaluation results using only binding site volume versus using all four features.

Metrics	Volume	All
Test Loss (e-3)	9.03	**3.80**
Vina Score	-4.88	**-5.72**
QED	**0.59**	0.55
SA	0.56	**0.58**
Diversity	0.69	**0.72**
Complete	94.0%	**94.7%**

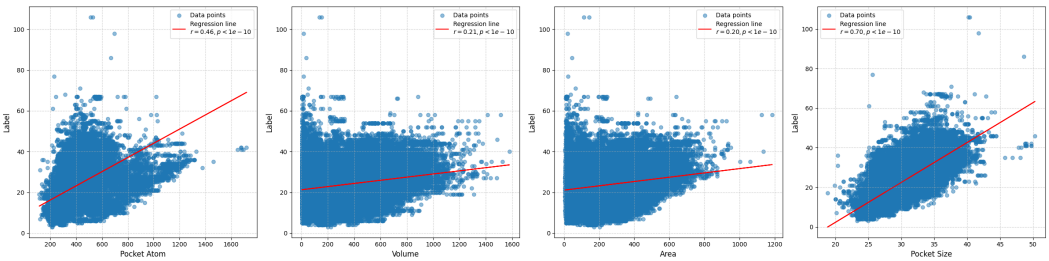


Figure 9: Relationships between ligand atom number and pocket atom number, binding site volume, surface area, and space size in the training set. Red lines show regression fits. Correlation coefficients and p-values (all $<$ 1e $-$ 10) are reported.

E.6 Validation for Atom Number Predictor

Prior research [49] has established a strong positive correlation between ligand size (i.e., number of atoms) and binding affinity. However, the performance gain in our method does not stem from simply generating molecules with more atoms. Instead, it results from generating molecules whose atom counts are closer to the reference ligand.

To more intuitively demonstrate the effectiveness of our predictor, we compare it with the predefined sampling by generating 1,000 atom numbers for each of three randomly selected test proteins. For each protein, we define 20% and 30% intervals around the reference atom number and compute the proportion of generated samples falling within these ranges. As shown in Table 7, most atom numbers predicted by our model fall within the 20% range of the reference ligand's atom count, without producing significantly larger molecules. In contrast, the traditional method—which samples atom numbers from predefined distributions—yields a substantial portion of results outside the 30% range. Such deviation increases the likelihood of generating molecules that are either too large or too small to fit well within the binding pocket, thereby negatively affecting the protein–ligand interaction.

Table 7: Comparison between predefined sampling and our predictor on atom number. For each of the three randomly selected test proteins, 1,000 atom numbers are generated.

PDB ID		<70%	[70%, 80%)	[80%, 100%)	[100%, 120%)	[120%, 130%)	>130%
4YHJ	Predefined	4.7%	10.9%	26.2%	34.2%	12.5%	11.5%
	Predict	0	0	64.0%	36.0%	0	0
2E24	Predefined	7.4%	7.8%	17.1%	17.4%	13.7%	36.6%
	Predict	0	0	2.9%	81.4%	15.5%	0.2%
3NFB	Predefined	25.2%	11.1%	29.8%	23.1%	5.8%	5.0%
	Predict	0	0.4%	97.2%	2.4%	0	0

E.7 Structure Analysis

To evaluate whether PAFlow can generate valid molecular conformations, a series of structure-related experiments are conducted. We report the average Jensen-Shannon divergence (JSD) between the bond length, bond angle, and torsion angle distributions of the generated and reference molecules, where a lower JSD indicates a distribution closer to that of real structures. In addition, we provide the ring size distribution of generated molecules. Together, these evaluations comprehensively assess the substructure stability. To evaluate global structural stability, the strain energy (SE) and steric clash (Clash) statistics are also reported, which are calculated by PoseCheck [50]. As shown in Table 8-10, PAFlow exhibits comparable performance to existing methods across both local and global structural metrics. Although classifier guidance is known to sometimes produce ill-formed conformations, PAFlow is still able to generate molecules with stable and chemically plausible geometries at both the local and global levels. In future work, we plan to incorporate physical rules and energy-based constraints to further improve molecular structural quality.

Table 8: Results of molecular conformation evaluation. $(\downarrow)$ denotes that lower values indicate better performance.

Methods	Length $(\downarrow)$	Angle $(\downarrow)$	Torsion $(\downarrow)$
AR	0.554	0.467	0.519
FLAG	0.511	0.284	-
IPDiff	0.402	0.415	0.386
ALiDiff	0.445	0.422	0.422
PAFlow	0.507	0.461	0.424

F Discussion

F.1 Difference Between PAFlow and Other FM-Based Methods

Flow Matching is a simulation-free approach for stably training continuous normalizing flows that demonstrates strong generative capabilities. Various types of probability paths can be used within the FM framework as summarized in Table 1 of [29], and the choice can be flexibly adapted based on the specific task requirements.

Table 9: Ratios of ring sizes for reference ligands and molecules generated by different methods.

Ring Size	Ref	IPDiff	TAGMoL	PAFlow
3	1.7%	0.0%	0.0%	0.0%
4	0.0%	3.2%	4.0%	3.4%
5	30.2%	25.8%	29.5%	19.6%
6	67.4%	42.4%	39.0%	38.4%
7	0.7%	18.6%	19.7%	26.0%
8	0.0%	7.0%	5.9%	9.2%
9	0.0%	2.9%	1.9%	3.4%

Table 10: Results on molecular conformation stability. For SE, the 25th percentile, median, and 75th percentile are reported.

Methods	SE (↓)			Clash (↓)	
	25%	50%	75%	Avg	Med
TargetDiff	363	1233	13773	10.77	7.00
ALiDiff	1607	56055	5779799	8.68	5.00
KGDiff	566	84186	124750452	7.82	5.00
PAFlow	3339	90239	5906610	7.74	5.00

FlowSBDD[14] employs Rectified Flow to learn the transport mapping of atom coordinates and types from the prior to the data distribution, aiming for straight-line trajectories between the two. FlexSBDD[51] adopts a similar modeling approach. While such straight-line paths are computationally efficient, they lack the capacity to model complex generation tasks effectively. In contrast, PAFlow utilizes different probability paths tailored to the distinct characteristics of atom coordinates and types. Specifically, continuous coordinates are modeled using Gaussian distributions with variance-preserving (VP) trajectories derived from diffusion models, while discrete atom types are modeled with categorical distributions, for which we specifically construct a FM formulation and conditional vector field (detailed derivations in Appendix A.2). In addition, we prove that the resulting generative process is consistent with SE(3)-transformation invariance, offering a theoretical justification for our modeling strategies (see Appendix A.1).

Under the same experimental settings, a comparison between the results of these methods is presented. As shown in the Table 11 below, PAFlow significantly outperforms FlexSBDD and FlowSBDD on all affinity–related metrics while maintaining reasonable molecular properties. It is worth noting that FlexSBDD generates molecules with modeling protein flexibility, whereas PAFlow performs generation based on rigid proteins. Despite this difference, PAFlow still achieves superior performance, suggesting the greater effectiveness of our modeling strategy. Overall, although both PAFlow and FlexSBDD are built within the FM framework, they differ fundamentally in the design of probability paths and modeling strategies. We hope these extensions offer useful insights and contribute meaningfully to the advancement of FM-based molecular generation.

Table 11: Comparison of PAFlow with other FM-based methods. The table presents the mean values of each metric. The best results are highlighted in bold.

Methods	Vina Score (↓)	Vina Min (↓)	Vina Dock (↓)	High Affinity (↑)	QED (↑)	SA (↑)	Div (↑)
TargetDiff	-5.47	-6.64	-7.80	58.1%	0.48	0.58	0.72
FlowSBDD	-3.62	-6.72	-8.50	63.4%	0.47	0.51	0.75
FlexSBDD	-6.64	-8.27	-9.12	78.5%	**0.58**	**0.69**	**0.76**
PAFlow	**-8.31**	**-8.79**	**-9.46**	**80.8%**	0.49	0.57	0.71

F.2 Reasons for the Better Binding Affinity

There are two main reasons why the molecules generated by PAFlow achieve lower Vina scores compared to the ground truth in the test set. Firstly, we calculate the average and median Vina scores of the protein–ligand pairs in the training set to be -8.19 and -8.32 respectively, which are comparable to those of the molecules generated by PAFlow. This indicates that PAFlow effectively learned the binding patterns contained in the training data. Secondly, the interaction predictor is employed to guide the generation process. Since the predictor is trained using the Vina scores of the training set as labels, it captures prior domain knowledge about protein–ligand binding embedded in these scores. By incorporating this learned prior into the generation process, the model benefits not only from the information in the training data but also from the guidance, which together enable PAFlow to generate molecules with better binding affinity than those in the training set. Actually, the reference ground-truth ligands are excellent binders for their corresponding protein target according to certain experiments or predictions, but they may not be the optimal ones with the best Vina scores for the target.

F.3 Limitations, Future Work and Broader Impact

While PAFlow demonstrates outstanding performance, there remain several potential limitations that we aim to address in future work. First, the volume and surface area of binding pockets used to train the atom number predictor are computed using PyKVFinder, which provides only an approximate estimation and may introduce bias. In the future, we plan to incorporate experimentally measured pocket information to improve the performance of the predictor. Additionally, we have attempted to apply flow matching to rigid proteins and observed promising results. We intend to extend this approach to flexible proteins, which better reflect realistic scenarios in structural biology. Lastly, since PAFlow only focuses on optimizing binding affinity, we plan to extend the guidance strategy to include molecular properties for multi-objective optimization to generate ligands with higher pharmaceutical potential.

Our work holds promise for improving the efficiency of drug design and advancing the pharmaceutical industry. It can help streamline the drug development process, reducing both time and resource costs. In addition, regulatory oversight of structure-based drug design (SBDD) technique is necessary to ensure they are used for socially beneficial purposes and to prevent potential misuse that could lead to the generation of harmful molecules.

F.4 More Examples

The visualization of more ligand molecules generated by PAFlow, comparing to reference, ALiDiff and MolCRAFT, are provided in Fig. 11. We also present several unreasonable molecules generated by PAFlow as edge cases in Fig. 10 to illustrate its limitations, including improper double bonds, large rings, and fused rings, which occasionally occur. Incorporating physical rules and energy constraints is expected to alleviate this issue and improve the structural quality of the generated molecules.

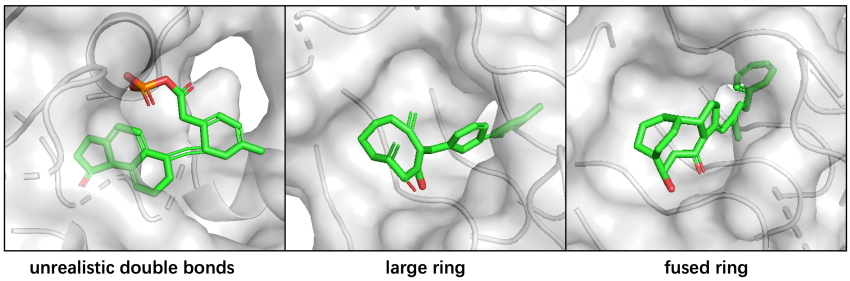


Figure 10: Some edge cases of molecules generated by PAFlow.

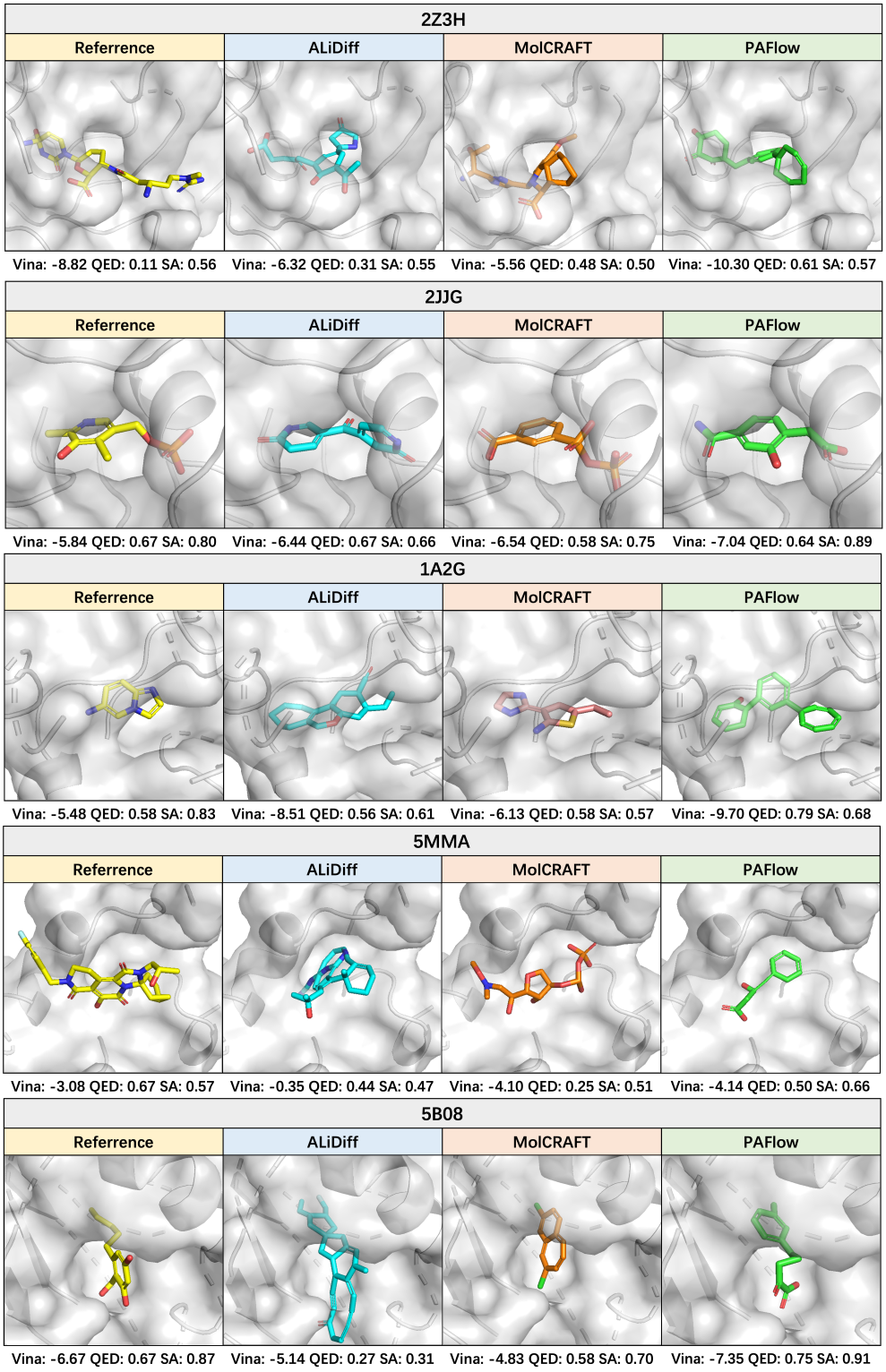


Figure 11: More visualizations of generated molecules and reference ligands for protein pockets. Carbon atoms of the reference ligands and molecules generated by AliDiff, MolCRAFT, and PAFlow are colored yellow, blue, orange, and green respectively. The corresponding Vina Score, QED, and SA for each molecule are also reported.