

1	<b>Appendix</b>	
2	<b>Contents</b>	
3	<b>A Methodology</b>	<b>2</b>
4	A.1 Conditional Diffusion Model . . . . .	2
5	A.2 Surrogate Models for Multi-objective Uncertainty Quantification . . . . .	4
6	A.3 RL-guided Optimization . . . . .	5
7	<b>B Experimental Settings</b>	<b>7</b>
8	B.1 Data Sources and Descriptions . . . . .	7
9	B.2 Data Preprocessing . . . . .	7
10	B.2.1 Target Protein Preprocessing . . . . .	7
11	B.2.2 Molecular Data Preprocessing . . . . .	7
12	B.3 Surrogate Model Training and Hyper-parameters . . . . .	9
13	B.4 Hyper-parameters for RL-guided Diffusion Model Training . . . . .	10
14	B.5 Baseline Implementation . . . . .	10
15	B.6 Ablation Study Design . . . . .	11
16	B.7 MD simulations and ADMET property prediction . . . . .	11
17	B.8 Evaluation . . . . .	12
18	B.8.1 Evaluation Metrics for Surrogate Models . . . . .	12
19	B.8.2 Evaluation Metrics for Diffusion Models With and Without Optimization .	13
20	B.9 Devices and Computational Setup . . . . .	15
21	<b>C Results and Discussion</b>	<b>16</b>
22	C.1 The Results of Surrogate Models . . . . .	16
23	C.2 Extended Discussion on Baseline Comparisons . . . . .	17
24	C.3 Extended Discussion on Ablation Studies . . . . .	18
25	C.4 Extended Discussion on MD and ADMET Prediction . . . . .	18
26	C.5 Visualization for Generated Molecules . . . . .	18
27	C.6 Analysis on Other Diffusion Models . . . . .	18
28	<b>D User Interface</b>	<b>20</b>

## 29 A Methodology

30 This section provides extended details to supplement Section 3 of the main manuscript.

### 31 A.1 Conditional Diffusion Model

32 **Notation.** We define  $\mathbf{x} \in \mathbb{R}^{M \times 3}$  as the coordinates of  $M$  atoms in 3D space, constrained to the  
 33 zero center of gravity subspace, meaning the sum of the coordinates  $\sum_i \mathbf{x}_i = 0$ . The atom features,  
 34  $\mathbf{h} \in \mathbb{R}^{M \times d}$ , are invariant to E(3) transformations and include attributes like one-hot encoded atom  
 35 types or charges. The conditioning variable  $\mathbf{c}$  represents a desired molecular property. At each  
 36 diffusion step  $t$ , the latent variable  $\mathbf{z}_t = [\mathbf{z}_t^{(x)}, \mathbf{z}_t^{(h)}]$  combines the noised coordinates  $\mathbf{z}_t^{(x)}$  and  
 37 features  $\mathbf{z}_t^{(h)}$ . The parameters  $\alpha_t$  and  $\sigma_t$  control the balance between signal retention and noise  
 38 addition, while  $\phi(\cdot, t, \mathbf{c})$  is an Equivariant Graph Neural Network (EGNN) that predicts noise during  
 39 denoising, conditioned on both the time step  $t$  and the property  $\mathbf{c}$ . Finally,  $\mathcal{N}_{xh}$  denotes a joint normal  
 40 distribution over coordinates and features, with coordinates constrained to the zero center of gravity  
 41 subspace.

42 **Forward Diffusion.** The forward diffusion process is the first key component of EDM, designed to  
 43 gradually corrupt the original data into noise, creating a sequence of latent variables  $\mathbf{z}_0, \mathbf{z}_1, \dots, \mathbf{z}_T$   
 44 that we can later denoise to generate new samples. In a standard diffusion model, the process adds  
 45 noise to a data point  $\mathbf{x}$  according to the formula:

$$q(\mathbf{z}_t | \mathbf{x}) = \mathcal{N}(\mathbf{z}_t; \alpha_t \mathbf{x}, \sigma_t^2 \mathbf{I}), \quad (1)$$

46 where  $\alpha_t$  determines how much of the original signal is retained, and  $\sigma_t^2$  controls the variance of  
 47 the added Gaussian noise. This formula captures the essence of diffusion: as  $t$  increases, the data is  
 48 progressively noised until it resembles pure noise at  $t = T$ .

49 In EDM, this idea is extended to jointly model both coordinates  $\mathbf{x}$  and features  $\mathbf{h}$ , since molecules  
 50 require both positional and categorical information. The forward process is Markovian, meaning  
 51 each step depends only on the previous one, and is expressed as:

$$q(\mathbf{z}_0, \mathbf{z}_1, \dots, \mathbf{z}_T | \mathbf{x}, \mathbf{h}) = q(\mathbf{z}_0 | \mathbf{x}, \mathbf{h}) \prod_{t=1}^T q(\mathbf{z}_t | \mathbf{z}_{t-1}). \quad (2)$$

52 The initial distribution at  $t = 0$  is defined as:

$$q(\mathbf{z}_0 | \mathbf{x}, \mathbf{h}) = \mathcal{N}_{xh}(\mathbf{z}_0 | \alpha_0[\mathbf{x}, \mathbf{h}], \sigma_0^2 \mathbf{I}), \quad (3)$$

53 and the transition from  $\mathbf{z}_{t-1}$  to  $\mathbf{z}_t$  is given by:

$$q(\mathbf{z}_t | \mathbf{z}_{t-1}) = \mathcal{N}_{xh}(\mathbf{z}_t | \alpha_{t|t-1} \mathbf{z}_{t-1}, \sigma_{t|t-1}^2 \mathbf{I}), \quad (4)$$

54 where  $\alpha_{t|t-1} = \alpha_t / \alpha_{t-1}$  adjusts the signal scaling between steps, and the noise variance is

$$\sigma_{t|t-1}^2 = \sigma_t^2 - \alpha_{t|t-1}^2 \sigma_{t-1}^2.$$

55 The joint distribution  $\mathcal{N}_{xh}$  factorizes into distributions for coordinates and features:

$$\mathcal{N}_{xh}(\mathbf{z}_t | \alpha_t[\mathbf{x}, \mathbf{h}], \sigma_t^2 \mathbf{I}) = \mathcal{N}_x(\mathbf{z}_t^{(x)} | \alpha_t \mathbf{x}, \sigma_t^2 \mathbf{I}) \cdot \mathcal{N}(\mathbf{z}_t^{(h)} | \alpha_t \mathbf{h}, \sigma_t^2 \mathbf{I}). \quad (5)$$

56 Here,  $\mathcal{N}_x$  is a normal distribution over the zero center of gravity subspace, ensuring the coordinates  
 57 remain translation-invariant, and is defined as:

$$\mathcal{N}_x(\mathbf{x} | \boldsymbol{\mu}, \sigma^2 \mathbf{I}) = (\sqrt{2\pi\sigma})^{-(M-1) \cdot 3} \exp(-\|\mathbf{x} - \boldsymbol{\mu}\|^2 / (2\sigma^2)), \quad (6)$$

58 with  $\boldsymbol{\mu}$  in the zero center of gravity subspace. The feature distribution  $\mathcal{N}$  is a standard normal  
 59 distribution, reflecting the invariance of  $\mathbf{h}$  to E(3) transformations.

The noise schedule is designed to be variance-preserving, meaning  $\alpha_t = \sqrt{1 - \sigma_t^2}$ , ensuring that the total variance of the data remains constant as noise is added. The specific form of  $\alpha_t$  is chosen to smoothly transition from retaining the signal to adding noise:

$$\alpha_t = (1 - 2s) \cdot (1 - (t/T)^2) + s, \quad (7)$$

where  $s = 10^{-5}$  is a small constant to prevent numerical issues, allowing  $\alpha_t$  to decrease from  $\alpha_0 \approx 1$  (almost pure signal) to  $\alpha_T \approx 0$  (almost pure noise). To quantify the balance between signal and noise at each step, we define the Signal-to-Noise Ratio (SNR):

$$\text{SNR}(t) = \alpha_t^2 / \sigma_t^2 = \alpha_t^2 / (1 - \alpha_t^2). \quad (8)$$

The SNR decreases as  $t$  increases, reflecting the increasing dominance of noise. For computational convenience during training and sampling, we also compute the negative log-SNR, defined as:

$$\gamma(t) = -\log \text{SNR}(t) = -\log \alpha_t^2 + \log \sigma_t^2. \quad (9)$$

The purpose of the negative log-SNR is to provide a numerically stable representation of the noise level at each step, which is particularly useful when optimizing the model or scheduling the noise. Using properties of the sigmoid function, we can express  $\alpha_t^2 = \text{sigmoid}(-\gamma(t))$  and  $\sigma_t^2 = \text{sigmoid}(\gamma(t))$ , which simplifies calculations and ensures that the noise schedule is well-behaved across all timesteps.

**Backward Diffusion.** The goal of the reverse process is to generate new molecules by denoising the latent variables, starting from pure noise at  $t = T$  and iteratively refining them back to a data sample at  $t = 0$ . In a conditional diffusion model, the reverse process approximates the denoising step, conditioned on  $\mathbf{c}$ , as:

$$p(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \mathbf{c}) = \mathcal{N}(\mathbf{z}_{t-1}; \mu_\theta(\mathbf{z}_t, t, \mathbf{c}), \sigma_t^2 \mathbf{I}), \quad (10)$$

where  $\mu_\theta(\mathbf{z}_t, t, \mathbf{c})$  is the mean predicted by EGNN, guiding the denoising process toward samples that satisfy the condition  $\mathbf{c}$ . This formula provides a high-level view of the reverse process, capturing how the model learns to reverse the noise addition step by step.

We define this process more precisely to ensure E(3) equivariance. The true posterior for the reverse step is:

$$q(\mathbf{z}_s \mid \mathbf{x}, \mathbf{h}, \mathbf{z}_t) = \mathcal{N}_{xh}(\mathbf{z}_s \mid \mu_{t \rightarrow s}([\mathbf{x}, \mathbf{h}], \mathbf{z}_t), \sigma_{t \rightarrow s}^2 \mathbf{I}), \quad (11)$$

where  $s < t$ , and the mean and variance are:

$$\mu_{t \rightarrow s}([\mathbf{x}, \mathbf{h}], \mathbf{z}_t) = (\alpha_{t|s} \sigma_s^2 / \sigma_t^2) \mathbf{z}_t + (\alpha_s \sigma_{t|s}^2 / \sigma_t^2) [\mathbf{x}, \mathbf{h}], \quad (12)$$

$$\sigma_{t \rightarrow s} = (\sigma_{t|s} \sigma_s) / \sigma_t, \quad \alpha_{t|s} = \alpha_t / \alpha_s, \quad \sigma_{t|s}^2 = \sigma_t^2 - \alpha_{t|s}^2 \sigma_s^2. \quad (13)$$

Since  $\mathbf{x}$  and  $\mathbf{h}$  are unknown during generation, we approximate them using EGNN  $\phi(\mathbf{z}_t, t, \mathbf{c})$ , which predicts the noise  $\hat{\epsilon}_t = [\hat{\epsilon}_t^{(x)}, \hat{\epsilon}_t^{(h)}]$ . The estimated data is then computed as:

$$[\hat{\mathbf{x}}, \hat{\mathbf{h}}] = (\mathbf{z}_t / \alpha_t) - (\sigma_t / \alpha_t) \hat{\epsilon}_t, \quad (14)$$

where  $\hat{\epsilon}_t = \phi(\mathbf{z}_t, t, \mathbf{c})$ . Substituting this into the mean, the reverse transition distribution becomes:

$$p(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{c}) = \mathcal{N}_{xh}(\mathbf{z}_s \mid \mu_{t \rightarrow s}([\hat{\mathbf{x}}, \hat{\mathbf{h}}], \mathbf{z}_t), \sigma_{t \rightarrow s}^2 \mathbf{I}). \quad (15)$$

For sampling, we start with  $\mathbf{z}_T \sim \mathcal{N}_{xh}(0, \mathbf{I})$ , representing pure noise, and iteratively apply the reverse process for  $t = T$  down to  $t = 1$ , setting  $s = t - 1$ . At each step, we sample  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ , subtract the center of gravity from  $\mathbf{z}_t^{(x)}$ , compute the predicted noise  $\hat{\epsilon}_t = \phi(\mathbf{z}_t, t, \mathbf{c})$ , and update:

$$\mathbf{z}_s = (1 / \alpha_{t|s}) \mathbf{z}_t - (\sigma_{t|s}^2 / (\alpha_{t|s} \sigma_t)) \hat{\epsilon}_t + \sigma_{t \rightarrow s} \epsilon. \quad (16)$$

Finally, we sample the data  $(\mathbf{x}, \mathbf{h}) \sim p(\mathbf{x}, \mathbf{h} \mid \mathbf{z}_0, \mathbf{c})$ . EGNN ensures that the predicted noise for coordinates is equivariant:

$$\mathbf{R} \hat{\epsilon}_t^{(x)} = \phi^{(x)}(\mathbf{R} \mathbf{z}_t^{(x)}, \mathbf{z}_t^{(h)}, t, \mathbf{c}), \quad (17)$$

where  $\mathbf{R}$  is an orthogonal matrix.

**Training Objective.** To train the model, we aim to maximize the conditional log-likelihood  $\log p(\mathbf{x}, \mathbf{h} \mid \mathbf{c})$ , which is achieved by optimizing a variational lower bound:

$$\log p(\mathbf{x}, \mathbf{h} \mid \mathbf{c}) \geq \mathcal{L}_{c,0} + \mathcal{L}_{c,\text{base}} + \sum_{t=1}^T \mathcal{L}_{c,t}. \quad (18)$$

The term  $\mathcal{L}_{c,t} = -\text{KL}(q(\mathbf{z}_s \mid \mathbf{x}, \mathbf{h}, \mathbf{z}_t) \parallel p(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{c}))$  measures the divergence between the true and approximate reverse distributions for  $t = 1, \dots, T$ . The term  $\mathcal{L}_{c,0} = \log p(\mathbf{x}, \mathbf{h} \mid \mathbf{z}_0, \mathbf{c})$  evaluates the likelihood at the final step, and  $\mathcal{L}_{c,\text{base}} = -\text{KL}(q(\mathbf{z}_T \mid \mathbf{x}, \mathbf{h}) \parallel p(\mathbf{z}_T))$  compares the forward process at  $t = T$  to the prior. Since  $\alpha_T \approx 0$ ,  $\mathcal{L}_{c,\text{base}} \approx 0$ .

For  $t \geq 1$ , the KL divergence simplifies to a noise prediction objective:

$$\mathcal{L}_{c,t} = \mathbb{E}_{\epsilon \sim \mathcal{N}_{x_h}(0, \mathbf{I})} \left[ (1/2) w(t) \|\epsilon_t - \phi(\mathbf{z}_t, t, \mathbf{c})\|^2 \right], \quad (19)$$

where  $w(t) = 1 - \text{SNR}(t-1)/\text{SNR}(t)$ . Setting  $w(t) = 1$  simplifies training and improves sample quality, reducing the objective to minimizing the mean-squared error between the true noise  $\epsilon_t$  and the predicted noise  $\phi(\mathbf{z}_t, t, \mathbf{c})$ . During training, we sample  $t \sim \mathcal{U}(0, \dots, T)$ , noise  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ , compute  $\mathbf{z}_t = \alpha_t[\mathbf{x}, \mathbf{h}] + \sigma_t \epsilon$ , and minimize  $\|\epsilon - \phi(\mathbf{z}_t, t, \mathbf{c})\|^2$ .

The zero-term  $\mathcal{L}_{c,0}$  splits into contributions from coordinates and features:  $\mathcal{L}_{c,0} = \mathcal{L}_{c,0}^{(x)} + \mathcal{L}_{c,0}^{(h)}$ .

For coordinates, we approximate the posterior as:

$$q(\mathbf{x} \mid \mathbf{z}_0^{(x)}) \approx \mathcal{N}_x(\mathbf{x} \mid \mathbf{z}_0^{(x)}/\alpha_0, (\sigma_0^2/\alpha_0^2)\mathbf{I}), \quad (20)$$

and the generative distribution is:

$$p(\mathbf{x} \mid \mathbf{z}_0, \mathbf{c}) = \mathcal{N}(\mathbf{x} \mid (\mathbf{z}_0^{(x)}/\alpha_0) - (\sigma_0/\alpha_0)\epsilon_0^{(x)}, (\sigma_0^2/\alpha_0^2)\mathbf{I}), \quad (21)$$

yielding:

$$\mathcal{L}_{c,0}^{(x)} = \mathbb{E}_{\epsilon^{(x)} \sim \mathcal{N}_x(0, \mathbf{I})} \left[ \log Z - \frac{1}{2} \|\epsilon^{(x)} - \phi^{(x)}(\mathbf{z}_0, 0, \mathbf{c})\|^2 \right], \quad (22)$$

where  $Z = (\sqrt{2\pi} \cdot \sigma_0/\alpha_0)^{(M-1) \cdot 3}$ .

For categorical features, the noising is

$$q(\mathbf{z}_t^{(h)} \mid \mathbf{h}) = \mathcal{N}(\mathbf{z}_t^{(h)} \mid \alpha_t \mathbf{h}^{\text{onehot}}, \sigma_t^2 \mathbf{I}), \quad (23)$$

and the generative distribution is a categorical distribution, approximating  $\mathcal{L}_{c,0}^{(h)} \approx 0$  for small  $\sigma_0$ .

**Conditional Generation Mechanism** The purpose of conditional generation is to produce molecules  $(\mathbf{x}, \mathbf{h}) \sim p(\mathbf{x}, \mathbf{h} \mid \mathbf{c}, M)$  that satisfy the desired property  $\mathbf{c}$  and have  $M$  atoms. We first sample  $(\mathbf{c}, M) \sim p(\mathbf{c}, M)$  from a learned distribution estimated from the training data. Then, we sample pure noise  $\mathbf{z}_T \sim \mathcal{N}_{x_h}(0, \mathbf{I})$ , iteratively denoise using  $p(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \mathbf{c})$ , and finally sample the data  $(\mathbf{x}, \mathbf{h}) \sim p(\mathbf{x}, \mathbf{h} \mid \mathbf{z}_0, \mathbf{c})$ . The neural network  $\phi(\mathbf{z}_t, t, \mathbf{c})$  incorporates  $\mathbf{c}$  by concatenating it to the node features, guiding the denoising process toward the desired property.

## A.2 Surrogate Models for Multi-objective Uncertainty Quantification

**Uncertainty Modeling.** The uncertainty  $U_{\text{single}}(m; \delta)$  encompasses both aleatoric and epistemic uncertainties, combined into the total variance

$$\sigma_{\text{total}}^2(m) = \sigma_a^2(m) + \sigma_e^2(m). \quad (24)$$

Aleatoric uncertainty  $\sigma_a^2$  reflects inherent data noise, such as variability from computational simulations, while epistemic uncertainty  $\sigma_e^2$  arises from model limitations due to limited training data.

In the Chemprop model with evidential learning, these are estimated using the Normal Inverse-Gamma (NIG) parameters:

$$\sigma_a^2 = \frac{\beta}{\alpha - 1}, \quad \text{and} \quad \sigma_e^2 = \frac{\beta}{\nu(\alpha - 1)}, \quad (25)$$

where  $\beta$ ,  $\alpha$ , and  $\nu$  are predicted by the model.

123 The Gaussian distribution assumption, with mean  $\mu(m)$  and standard deviation

$$\sigma(m) = \sqrt{\sigma_{\text{total}}^2(m)}, \quad (26)$$

124 is reasonable because it allows the integral in  $U_{\text{single}}(m; \delta)$  to compute a probability, validated by the  
125 model’s calibration to real-world data variability.

126 **Uncertainty as a Threshold Probability.** The uncertainty  $\sigma(m)$  directly informs the probability  
127 that a molecule’s true property value exceeds the threshold  $\delta$ , computed as:

$$U_{\text{single}}(m; \delta) = \eta \int_{\delta}^{\infty} \frac{1}{\sigma(m)\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x - \mu(m)}{\sigma(m)}\right)^2\right) dx. \quad (27)$$

128 This integral represents the cumulative probability under a Gaussian distribution, where a larger  $\sigma(m)$   
129 reduces the likelihood of exceeding  $\delta$  due to increased spread, and a smaller  $\sigma(m)$  increases it if  
130  $\mu(m)$  is favorable.

131 The inclusion of total uncertainty ( $\sigma_{\text{total}}^2$ ) ensures that both aleatoric and epistemic contributions are  
132 considered, reflecting the overall reliability of the prediction.

133 **Multi-Objective Uncertainty.** The multi-objective uncertainty is defined as:

$$U_{\text{multi}}(m; \delta_1, \dots, \delta_k) = \prod_{i=1}^k U_{\text{single}}(m; \delta_i). \quad (28)$$

134 This product is justified by assuming conditional independence of property uncertainties given the  
135 molecule, meaning the likelihood of meeting all thresholds  $\delta_1, \dots, \delta_k$  is the joint probability of  
136 individual successes. Each  $U_{\text{single}}(m; \delta_i)$  measures the probability for one property, and the product  
137 ensures a molecule is rewarded only if it likely satisfies all objectives simultaneously.

138 This approach avoids trade-offs, penalizing molecules with low probability for any property, which is  
139 critical in RL where a balanced solution is needed, making it a coherent aggregation strategy.

### 140 A.3 RL-guided Optimization

141 **Transition Probability.** We first employ the trained diffusion model to sample a set of molecules  
142 and record their diffusion trajectories. Each trajectory can be treated as a Markov chain. The transition  
143 probability between two states on the chain are defined in Equation (4). Its PDF format is given by

$$p_{\theta}(\mathbf{z}_{t-1} | \mathbf{z}_t, \mathbf{c}) = \frac{1}{(2\pi\sigma_t^2)^{d/2}} \exp\left(-\frac{1}{2\sigma_t^2} \|\mathbf{z}_{t-1} - \mu_{\theta}(\mathbf{z}_t, t, \mathbf{c})\|^2\right), \quad (29)$$

144 relying on the variance  $\sigma_t^2$  and the latent variables  $\mathbf{z}_t$  and  $\mathbf{z}_{t-1}$ . The variance  $\sigma_t^2$  is computed based  
145 on the noise schedule in the diffusion process. Referring to Equation (13), we define

$$\sigma_{t \rightarrow s}^2 = \frac{\sigma_t^2}{\alpha_t} = \sigma_t^2 \frac{\sigma_s^2}{\alpha_s^2}, \quad (30)$$

146 where  $\sigma_t^2 = 1 - \alpha_t$ , ensuring that the variance increases as  $t$  grows, reflecting the progressive addition  
147 of noise in the forward process.

148 The latent variable  $\mathbf{z}_t$  is sampled iteratively during the backward process of the diffusion model.  
149 Starting from  $\mathbf{z}_T \sim \mathcal{N}(0, \mathbf{I})$ , each  $\mathbf{z}_t$  for  $t = T - 1, \dots, 0$  is computed using Equation (16):

$$\mathbf{z}_s = \frac{1}{\sqrt{\alpha_{t|s}}} \mathbf{z}_t - \left( \frac{\sigma_{t \rightarrow s}^2}{\sqrt{\alpha_{t|s}} \sigma_t} \right) \hat{\epsilon} + \sigma_{t \rightarrow s} \epsilon, \quad (31)$$

150 where  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$  and  $\hat{\epsilon} = \phi(\mathbf{z}_t, t, \mathbf{c})$ . Subsequently,  $\mathbf{z}_{t-1}$  is sampled from

$$\mathcal{N}(\mathbf{z}_{t-1} | \mu_{t \rightarrow s}([\mathbf{x}, \mathbf{h}], \mathbf{z}_t), \sigma_{t \rightarrow s}^2 \mathbf{I}), \quad (32)$$

151 ensuring the denoising trajectory aligns with the learned distribution.

152 **Reward Bonus Design.** The reward bonus  $R_{\text{bonus}}(m)$  is determined by three binary flags: validity  
 153 ( $v_m$ ), uniqueness ( $u_m$ ), and novelty ( $n_m$ ), each taking values in  $\{0, 1\}$ . These flags indicate whether  
 154 molecule  $m$  satisfies the respective criteria. The bonus is formulated as a weighted linear combination:

$$R_{\text{bonus}}(m) = b_v v_m + b_u u_m + b_n n_m, \quad (33)$$

155 where  $b_v$ ,  $b_u$ , and  $b_n$  are positive scalars representing the reward contributions of validity, uniqueness,  
 156 and novelty, respectively.

157 This reward-boosting design allows binary objectives related to generation quality to be integrated  
 158 without inducing reward sparsity, thereby enabling unified optimization of both binary and continuous  
 159 rewards in a multi-objective setting.

160 **Optimization.** We optimize the diffusion model using a PPO-style clipped surrogate loss. Gradients  
 161 are accumulated over multiple sampled timesteps based on fixed trajectories, enabling stable updates  
 162 under complex multi-objective rewards. The pseudocode is shown in Algorithm 1.

---

**Algorithm 1** RL-guided Optimization for Diffusion Models

---

```

1: Initialize diffusion model  $\pi_\theta$ , optimizer, and learning rate scheduler
2: for episode = 1 to  $N$  do
3:   Freeze current policy as  $\pi_{\theta_{\text{old}}}$ 
4:   Generate a batch of molecules using  $\pi_{\theta_{\text{old}}}$ 
5:   Record complete diffusion trajectories and transition probabilities
6:   Estimate uncertainty and compute reward probabilities
7:   Measure molecular diversity, validity, uniqueness, and novelty
8:   Compute total rewards
9:   for each policy update over fixed trajectories do
10:    Sample timesteps  $t_1, t_2, \dots, t_K \sim \mathcal{U}(0, T)$ 
11:    for each sampled timestep  $t$  do
12:      Evaluate log-probabilities under  $\pi_\theta$  at timestep  $t$ 
13:      Compute importance ratio  $r(m) = \exp(\log p_\theta - \log p_{\theta_{\text{old}}})$ 
14:      Compute clipped surrogate loss
15:      Accumulate gradients
16:    end for
17:    Update  $\pi_\theta$  using accumulated gradients
18:    Step learning rate scheduler
19:  end for
20: end for

```

---

## B Experimental Settings

As illustrated in Fig. 1, our pipeline consists of five key stages: data preprocessing, surrogate model training, diffusion model pretraining, RL-guided optimization of the diffusion model, and molecule generation with subsequent in silico evaluation. The following sections provide additional implementation details for each stage of the pipeline.

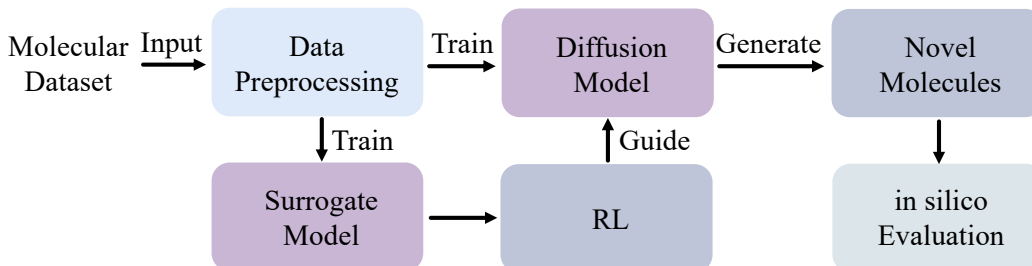


Figure 1: **Overall workflow of our framework.** A molecular dataset is first preprocessed to train both the diffusion model and the surrogate models. RL guides the optimization of the diffusion model using predictions from the surrogate models. The optimized diffusion model then generates candidate molecules for in silico evaluation.

### B.1 Data Sources and Descriptions

We use three molecular datasets (QM9, ZINC15, and PubChem) as sources for training and evaluating surrogate and diffusion models. In addition, we use a curated crystallographic structure of mutant EGFR as the target receptor for molecular docking. Table 1 summarizes the sources, descriptions, and download options for each dataset used in this study.

### B.2 Data Preprocessing

#### B.2.1 Target Protein Preprocessing

We selected the human EGFR as the docking target in this study due to its central role in cell signaling and its clinical importance in various cancers. In particular, we used the 3D crystal structure with PDB ID 6VHN, which corresponds to the kinase domain of a mutant form of EGFR bound to a small-molecule inhibitor. This structure provides a biologically relevant conformation for evaluating ligand binding and offers sufficient resolution (2.5 Å) for structure-based modeling.

To prepare the protein for molecular docking, we applied a series of standard preprocessing steps to ensure structural completeness and compatibility with docking software. First, all water molecules, ions, and co-crystallized ligands not involved in the binding site were removed to eliminate potential artifacts and ensure a clean receptor surface. The structure was then processed to convert any non-standard residues into their canonical forms, and missing side chains were reconstructed using geometry-based modeling to ensure a complete atomic representation. Polar hydrogen atoms were added throughout the protein to restore potential hydrogen bonding interactions typically unresolved in crystallographic data. Finally, partial atomic charges were assigned to the protein using a standard method compatible with docking engines. The resulting structure was saved in the appropriate format and used as the receptor input in subsequent docking simulations.

#### B.2.2 Molecular Data Preprocessing

To process each dataset, we first remove invalid molecules (e.g., those with missing 3D coordinates or broken structures) and eliminate duplicates. For the remaining molecules, we annotate three key properties (QED, SAS, and binding affinity) for each molecule. Specifically, QED and SAS are computed using RDKit, while the binding affinity is predicted using AutoDock Vina through docking simulations with the EGFR receptor site. These property annotations will be used in both surrogate model training and diffusion model training.

Table 1: Sources and descriptions of datasets.

Dataset	Item	Details
QM9	Source	<a href="http://deepchem.io.s3-website-us-west-1.amazonaws.com/datasets/gdb9.tar.gz">http://deepchem.io.s3-website-us-west-1.amazonaws.com/datasets/gdb9.tar.gz</a>
	Description	Dataset containing quantum chemical properties of small organic molecules. QM9 is widely used in the development and evaluation of machine learning models for molecular property prediction.
	Download Option	Whole dataset (SDF format)
ZINC15	Source	<a href="https://zinc15.docking.org/tranches/home/#">https://zinc15.docking.org/tranches/home/#</a>
	Description	A free database of commercially available molecules for virtual screening, frequently used in drug discovery research.
	Download Option	3D lead-like molecules (SDF format); neutral or +0 charged; mid-range logP; in-stock availability; standard reactivity
PubChem	Source	<a href="https://pubchem.ncbi.nlm.nih.gov/#query=small%20molecule&amp;tab=compound">https://pubchem.ncbi.nlm.nih.gov/#query=small%20molecule&amp;tab=compound</a>
	Description	A public chemical database providing bioactivity data and annotations for small molecules. PubChem is extensively used in cheminformatics, drug discovery, and bioinformatics applications.
	Download Option	Small molecule subset (SDF format)
EGFR	Source	<a href="https://www.rcsb.org/structure/6VHN">https://www.rcsb.org/structure/6VHN</a>
	Description	Crystallographic structure of the kinase domain of mutant human EGFR (PDB ID: 6VHN), co-crystallized with a small-molecule inhibitor. This structure serves as the receptor target for binding affinity prediction via molecular docking.
	Download Option	Downloadable in .pdb format from RCSB PDB

After property computation, all hydrogen atoms are removed from the 3D molecular structures. This decision is motivated by the need to reduce computational overhead and simplify molecular graphs during preprocessing and training. Hydrogen atoms offer limited topological information and can be accurately reconstructed at a later stage based on standard valency rules. Because the core chemical and spatial information of each molecule remains intact, this step introduces no adverse impact on the training performance. Each molecule is ultimately represented by its SMILES string, 3D structure (including atom types and coordinates), and its drug-related properties. The SMILES string is used as molecular input when training surrogate models, while the 3D structure is used as input to the diffusion model.

To train the diffusion model, we further construct dataset-level configuration files based on the hydrogen-removed molecules. For each dataset, each distinct atom type in the molecule is assigned a numerical encoding via an atom encoder and its corresponding decoder. The unique atom types identified in each dataset are listed in Table 2. The number of atoms in each molecule is counted to determine the maximum graph size across the dataset. Fig 2 illustrates the number of molecules in each dataset grouped by heavy atom count. Molecules in QM9 typically contain fewer than 10 heavy atoms, while those in ZINC15 and PubChem span a broader range of sizes. Especially in PubChem, some structures exceed 40 atoms.

Table 2: Unique atom types observed in each dataset. (after hydrogen removal)

Dataset	Unique Atom Types
QM9	C, N, O, F
ZINC15	C, N, O, F, P, S, Cl, Br, I
PubChem	C, N, O, F, P, S, Cl, Br, I

To prepare the data for model training and evaluation, we apply different data splitting strategies tailored to the modeling task.



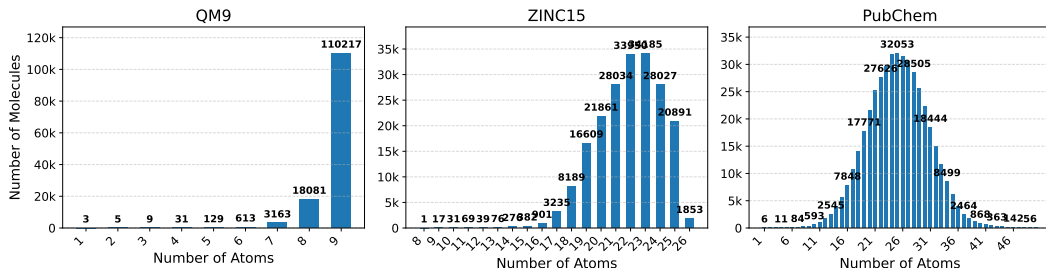


Figure 2: **Distribution of molecules by the number of heavy atoms in each dataset.** Subfigures (a), (b), and (c) correspond to QM9, ZINC15, and PubChem, respectively.

**Scaffold-based splitting.** For the surrogate model, we adopt scaffold-based splitting. Specifically, each molecule is first assigned a Bemis–Murcko scaffold by removing its side chains and retaining the core ring and linker structures. Molecules sharing the same scaffold are grouped together. These scaffold groups are then randomly shuffled and assigned to the training, validation, and test sets in such a way that no scaffold appears in more than one set. This ensures that the model is evaluated on structurally novel scaffolds not seen during training.

**Species-based splitting.** For training the diffusion model, we use species-based splitting. Each molecule is grouped based on the unique set of atom types it contains (i.e., its chemical species). These species groups are shuffled and assigned to disjoint training, validation, and test partitions, ensuring that each subset contains distinct combinations of atomic species. This strategy preserves the global diversity of atom types while promoting generalization to structurally diverse molecules in 3D space.

Table 3 summarizes the dataset statistics and splitting outcomes, including the total number of molecules, maximum number of heavy atoms, and the number of samples in the training, validation, and test sets for each dataset.

Table 3: **Dataset statistics after preprocessing.** Hydrogen atoms are excluded in max atom count.

	ZINC15	QM9	PubChem
# total molecules	198,626	132,251	444,287
# max atoms	26	9	50
# molecules in train set (80%)	158,900	105,800	355,429
# molecules in test set (10%)	19,862	13,225	44,428
# molecules in validate set (10%)	19,864	13,226	44,430

### B.3 Surrogate Model Training and Hyper-parameters

We train surrogate models using Chemprop, a graph neural network-based molecular property predictor. For each dataset, we use the available SMILES strings and their corresponding property labels (e.g., QED, SAS, or binding affinity) to train a separate model. The trained surrogate model estimates not only the predicted property value but also the uncertainty associated with whether the value exceeds a predefined threshold. Model hyperparameters are selected via grid search, as summarized in Table 4. The best hyperparameter configurations selected for each dataset are reported in Table 5.

Table 4: Hyperparameter search space for surrogate model training.

Hyperparameter	Search Range
activation	ReLU, SELU, GELU, Leaky ReLU
batch size	32, 64, 128, 512, 1024
dropout	0.0, 0.15, 0.1, 0.2
ensemble size	1, 5, 10, 20, 30
epochs	5, 10, 20, 30, 40, 50, 60
evidential regularization	1e-3, 5e-3, 1e-2
ffn_hidden_size	200, 300, 400, 600
final learning rate	1e-5, 1e-4
hidden size	200, 300, 400, 600
loss function	mse, evidential, mve
ffn_num_layers	2, 3
max learning rate	1e-3, 1e-2
initial learning rate	1e-4, 1e-3

Table 5: Hyperparameters for surrogate models across datasets and prediction tasks.

Hyperparameter	QM9			ZINC15			PubChem		
	QED	SAS	Affinity	QED	SAS	Affinity	QED	SAS	Affinity
activation	PReLU	PReLU	PReLU	PReLU	PReLU	PReLU	PReLU	PReLU	PReLU
batch size	64	64	64	64	64	64	64	64	64
dropout	0	0.15	0	0	0	0	0	0.1	0
ensemble size	1	1	1	1	1	1	1	1	1
epochs	40	40	40	40	40	40	40	40	40
evidential regularization	5e-3	5e-3	5e-3	5e-3	5e-3	5e-3	1e-2	5e-3	5e-3
ffn_hidden_size	300	300	300	300	300	300	300	300	300
final learning rate	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5
hidden size	300	300	300	300	300	300	300	300	300
loss function	evidential	evidential	evidential	evidential	evidential	evidential	evidential	evidential	evidential
ffn_num_layers	2	2	2	2	2	2	2	2	2
max learning rate	3e-3	3e-3	3e-3	3e-3	3e-3	3e-3	3e-3	3e-3	3e-3
initial learning rate	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4

#### B.4 Hyper-parameters for RL-guided Diffusion Model Training

We perform grid search over key hyperparameters for RL-guided diffusion model training, as summarized in Table 6. The final selected configurations for RL-guided diffusion models pre-trained on each dataset (QM9, ZINC15, PubChem) are reported in Table 7.

Table 6: Hyperparameter search space for RL-guided diffusion model training.

Hyperparameter	Search Range
valid_bonus	0.05, 0.2, 0.3, 0.35
unique_bonus	0.05, 0.2, 0.3, 0.35
novel_bonus	0.05, 0.2, 0.3, 0.35
clip_param	1e-4, 2e-4, 3e-4
learning rate	1e-5, 2e-5, 5e-5
reuse	1, 2, 3, 5, 10
n_samples	32, 64, 128, 256

#### B.5 Baseline Implementation

For each baseline, we follow the original framework design proposed in the corresponding papers. To adapt our multi-objective setting to these single-objective optimization frameworks, we convert the task into a binary reward formulation: a molecule receives a reward of 1 only if all target properties exceed their predefined cutoffs, and 0 otherwise. The cutoffs are dataset-specific and determined based on the top 10% of molecules ranked by each property, which provides a well-defined reward signal while mitigating the issue of extreme reward sparsity. For SFT-PG, we instead define the reward as the discrepancy between the property distribution of the generated molecules and an

Table 7: Final hyperparameter settings for RL-guided diffusion model training.

Hyperparameter	QM9	ZINC15	PubChem
valid_bonus	0.2	0.2	0.05
unique_bonus	0.35	0.35	0.3
novel_bonus	0.05	0.05	0.05
clip_param	3e-4	3e-4	3e-4
rl_lr	1e-5	1e-5	1e-5
reuse	3	3	3
n_samples	128	128	64

ideal target distribution derived from the unconditional diffusion model. We adopt hyperparameter configurations similar to the original implementations, using a clipping parameter of 0.2, a learning rate of  $1 \times 10^{-5}$ , and generating 128 samples per policy update.

## B.6 Ablation Study Design

We compare our method against three classes of widely used multi-objective optimization paradigms (scalarization-based, constraint-based, and gradient-based) to validate that the performance gains of our uncertainty-aware reward are not simply due to reformulating the objectives. These methods represent common and well-established strategies in multi-objective tasks.

We also compare to other uncertainty-based methods to demonstrate that our formulation has more actionable signals than generic uncertainty sampling or exploration strategies.

Lastly, we ablate key components in our reward module such as reward boosting, diversity penalty, and dynamic cutoff adjustment, which helps to understand their individual contributions and assess how each module improves molecule quality, stability, or balance across objectives.

## B.7 MD simulations and ADMET property prediction

To evaluate the dynamic stability and pharmacokinetic viability of candidate molecules beyond static docking, we conducted MD simulations and ADMET property predictions as two complementary computational assays. The MD simulations were implemented using AmberTools for system setup and OpenMM for GPU-accelerated production runs. Protein-ligand complexes were constructed by first assigning GAFF atom types to ligands using Antechamber, while proteins were parameterized with the ff14SB force field. All complexes were solvated in a rectangular TIP3P water box with a minimum 10 Å buffer in each direction and neutralized using counterions. The system topology and coordinate files were generated via tleap, and the entire simulation was conducted under periodic boundary conditions.

Each system underwent initial energy minimization and equilibration prior to production runs. The equilibration protocol consisted of a 100 ps NVT heating phase and a 200 ps NPT equilibration phase, gradually increasing the temperature to 300 K using a Langevin thermostat. The production simulations were run under the NPT ensemble at 1 atm and 300 K using Langevin dynamics with a 2 fs timestep, totaling 1,000,000 steps (approximately 4 ns). PME was used for long-range electrostatics with a 10 Å cutoff, and all hydrogen-containing bonds were constrained.

The primary readout of MD simulation was the Root-Mean-Square Deviation (RMSD) of protein backbone atoms, calculated over time using the initial apo structure as a reference. RMSD trajectories were used to assess the stability of the protein-ligand complex. Complexes demonstrating early convergence, low fluctuation (typically <0.5 nm), and the absence of sharp deviations were considered structurally stable. This dynamic screening step ensures that promising compounds maintain consistent binding behavior and do not induce substantial conformational drift or instability under physiologically relevant simulation conditions.

In parallel, ADMET properties were evaluated using ADMET-AI, a graph neural network-based platform trained on diverse experimental datasets. While the tool generates predictions for 41 pharmacokinetic and toxicological endpoints, we focused our analysis on a curated subset of 14 key properties that are particularly relevant for early-stage drug development. These include critical

parameters across all five ADMET domains, which are absorption (e.g., human intestinal absorption, Caco-2 permeability), distribution (e.g., volume of distribution, blood-brain barrier permeability), metabolism (e.g., CYP450 isoform inhibition for 3A4, 2D6, and 2C9), excretion (e.g., renal clearance), and toxicity (e.g., hERG inhibition, Ames mutagenicity, hepatotoxicity, and LD50).

The rationale for selecting these 14 properties stems from their strong empirical association with clinical trial outcomes and their routine use in pharmaceutical pipelines as early-stage go/no-go filters. Many of the remaining 27 endpoints, such as secondary CYP isoforms or advanced carcinogenicity markers, are either redundant with the selected subset or less predictive during the hit-to-lead phase. The 14-feature radar profile thus serves as a concise yet informative representation of a molecule’s drug-likeness and safety, enabling practical compound triage without overfitting to noisy or weakly generalizable predictions.

Each compound’s profile was evaluated both as an individual feature vector and as an integrated radar chart to identify potential liabilities. Compounds with strong absorption and distribution but poor metabolic or toxicity profiles were flagged for exclusion. Conversely, molecules with well-balanced and favorable values across all 14 categories were considered pharmacokinetically viable. This ADMET analysis complements the MD results by ensuring that structurally stable ligands also meet essential safety and efficacy criteria, thereby reinforcing their prioritization for further validation.

## B.8 Evaluation

### B.8.1 Evaluation Metrics for Surrogate Models

We adopt a widely used evaluation framework that includes both regression accuracy and uncertainty calibration metrics to assess the performance of surrogate models trained with uncertainty prediction.

**Regression Accuracy.** We use the coefficient of determination ( $R^2$ ) to assess the accuracy of the surrogate model’s predicted mean values. This metric quantifies how well the predicted outputs align with the ground-truth labels. It is defined as

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}, \quad (34)$$

where  $y_i$  and  $\hat{y}_i$  denote the ground-truth and predicted values, respectively, and  $\bar{y}$  is the mean of the ground-truth values. An  $R^2$  score of 1 indicates perfect prediction, whereas a score of 0 suggests that the model performs no better than simply predicting the mean. Higher  $R^2$  values therefore reflect stronger predictive performance and tighter alignment between the model’s outputs and observed data.

**Uncertainty Calibration.** To quantify how well predicted uncertainties reflect actual confidence, we compute the AUCE. This metric compares predicted confidence levels to the empirical frequency with which the true values fall within corresponding confidence intervals. It is defined as

$$\text{AUCE} = \int_0^1 |\hat{P}(c) - c| \, dc, \quad (35)$$

where  $c \in [0, 1]$  denotes the confidence level, and  $\hat{P}(c)$  is the proportion of predictions whose ground-truth values fall within the predicted  $c$ -level confidence interval. Lower AUCE values indicate better calibration.

**Visualization.** To complement the numerical metrics, we employ two types of visualizations for model diagnostics:

- **Parity plots:** compare predicted values against ground-truth targets, with predictive uncertainty shown as color gradients. We additionally include histograms and residual error plots to assess potential bias, variance, and distributional shifts.
- **Calibration curves:** plot empirical coverage  $\hat{P}(c)$  versus nominal confidence level  $c$ , allowing a visual assessment of how well the predicted uncertainties align with observed outcomes.

These visualizations provide intuitive insights into both regression accuracy and uncertainty calibration, and help identify systematic errors that may not be apparent from  $R^2$  or AUCE alone.

## B.8.2 Evaluation Metrics for Diffusion Models With and Without Optimization

To assess the quality generated molecules, we evaluate them using six key metrics: validity, uniqueness, novelty, atom stability, molecular stability, and the proportion of top molecules. Let  $\mathcal{M}_{\text{gen}}$  denote the set of molecules generated by the model, and  $\mathcal{M}_{\text{train}}$  the set of training molecules.

**Validity.** Validity measures the fraction of generated molecules that can be parsed into chemically valid molecular structures. Specifically, we consider a molecule valid if it can be successfully reconstructed from its atomic coordinates and types, converted into an RDKit molecule object, and translated into a canonical SMILES string without triggering errors during bond inference, sanitization, or stereochemistry assignment. This process includes constructing an XYZ block from the predicted positions and atom types, inferring bonds using RDKit’s internal heuristics, and applying standard sanitization and stereochemical processing. If these steps succeed without exceptions, the molecule is considered valid. Formally, the validity score is defined as:

$$\text{Validity} = \frac{|\{m \in \mathcal{M}_{\text{gen}} \mid m \text{ is valid}\}|}{|\mathcal{M}_{\text{gen}}|} \times 100\% \quad (36)$$

**Uniqueness.** Uniqueness denotes the proportion of valid molecules that are distinct from each other. It is computed by removing duplicates among all valid molecules based on their canonical SMILES representations. A higher uniqueness score indicates that the model is capable of generating a chemically diverse set of valid structures, rather than repeatedly producing similar or identical molecules. Conversely, a low uniqueness score may suggest issues such as overfitting to the training data or mode collapse, where the model fails to explore a wide range of chemical space. Formally, it is defined as:

$$\text{Uniqueness} = \frac{|\text{Unique}(\mathcal{M}_{\text{valid}})|}{|\mathcal{M}_{\text{valid}}|} \times 100\% \quad (37)$$

**Novelty.** Novelty quantifies the proportion of unique valid molecules generated by the model that do not appear in the training set. This comparison is performed using canonical SMILES strings to ensure consistency in molecular representation. A higher novelty score indicates that the model is capable of generating novel chemical structures beyond those it has seen during training, which is desirable in de novo molecular design. In contrast, a low novelty score may suggest that the model has overfit to the training data and is merely replicating known molecules. Formally, it is defined as:

$$\text{Novelty} = \frac{|\{m \in \text{Unique}(\mathcal{M}_{\text{valid}}) \mid m \notin \mathcal{M}_{\text{train}}\}|}{|\text{Unique}(\mathcal{M}_{\text{valid}})|} \times 100\% \quad (38)$$

**VUN.** VUN measures the proportion of generated molecules that are simultaneously valid, unique, and novel. It is computed as the product of Validity, Uniqueness, and Novelty. This metric reflects the overall ability of the model to generate truly de novo compounds that are chemically correct, structurally diverse, and absent from the training set. A high VUN indicates that a large fraction of the generated molecules can be considered meaningful de novo candidates. In contrast, a low VUN suggests that only a small portion of outputs are genuinely novel and chemically usable, limiting the model’s utility in de novo molecular design. Formally,

$$\text{VUN} = \text{Validity} \times \text{Uniqueness} \times \text{Novelty} = \frac{|\{m \in \text{Unique}(\mathcal{M}_{\text{valid}}) \setminus \mathcal{M}_{\text{train}}\}|}{|\mathcal{M}_{\text{gen}}|} \times 100\% \quad (39)$$

**Atom Stability.** Atom Stability measures the proportion of atoms in valid molecules that satisfy known valency rules. For each generated molecule, atom types and their 3D coordinates are used to reconstruct local bonding environments. Each atom is then evaluated based on whether its inferred number of bonds falls within its chemically allowed valence range. This metric captures whether the model respects fundamental atom-level chemical constraints. A high atom stability score indicates that most atoms are placed in chemically reasonable configurations with proper bonding patterns. Formally, the metric is defined as:

$$\text{Atom Stability} = \frac{|\{m \in \mathcal{M}_{\text{valid}} \mid \text{all atoms in } m \text{ obey valency}\}|}{|\mathcal{M}_{\text{valid}}|} \times 100\% \quad (40)$$

**Molecular Stability.** Molecular Stability measures the proportion of valid molecules that form chemically coherent structures as a whole. A molecule is considered stable if it forms a connected graph without unrealistic fragments, over-bonded atoms, or invalid substructures. Stability is assessed by reconstructing the full molecule from predicted positions and atom types, followed by heuristic bond inference and consistency checks. This metric reflects the model’s ability to generate molecules that are not only locally plausible (per atom), but also structurally sound on a global level. A high molecular stability score suggests that the majority of generated molecules form valid chemical graphs without structural discontinuities or violations of bonding rules. Formally, the metric is defined as:

$$\text{Molecular Stability} = \frac{|\{m \in \mathcal{M}_{\text{valid}} \mid m \text{ is chemically stable}\}|}{|\mathcal{M}_{\text{valid}}|} \times 100\% \quad (41)$$

**Top Molecules.** This metric calculates the percentage of generated molecules that simultaneously satisfy all three essential drug-relevant property constraints: QED  $\geq 0.4$ , SAS  $\leq 8.0$ , and binding affinity  $\leq -4.5$  kcal/mol (QED and SAS scores are computed using RDKit, while binding affinity is estimated using AutoDock Vina). Table 8 describes the meaning of different value ranges for QED, SAS, and binding affinity in the context of drug design. As a composite metric, it reflects the model’s capacity to generate candidate compounds that are not only chemically valid but also pharmacologically meaningful and synthetically accessible. We require a minimum QED score of 0.4 to ensure baseline drug-likeness. In practice, molecules with higher QED scores are more likely to exhibit favorable physicochemical properties, including appropriate lipophilicity, molecular weight, and hydrogen-bonding potential. These characteristics are typically associated with improved pharmacokinetics, reduced toxicity, and enhanced bioavailability in vivo. We also apply a SAS threshold of 8.0 to filter out molecules that are prohibitively difficult to synthesize. This constraint reflects real-world development considerations, where compounds that require low-yield or complex synthetic routes are often deemed infeasible regardless of their predicted activity. Finally, we impose a binding affinity threshold of  $-4.5$  kcal/mol to retain candidates that are thermodynamically capable of engaging the target protein with meaningful strength. For kinase targets like EGFR, effective binding at the Adenosine TriPhosphate (ATP) pocket is a prerequisite for competitive inhibition and downstream signaling blockade. Docking scores are inherently approximate. However, more negative binding energies often correlate with stronger interactions and, by extension, higher inhibitory potential. To avoid prematurely discarding promising compounds, particularly during the exploratory phase of molecular screening, we adopt this permissive but principled filtering strategy that balances chemical diversity with essential real-world constraints. By requiring all three properties to be satisfied simultaneously, the metric avoids overestimating molecules that perform well in only one aspect while failing in others. At the same time, it ensures that slightly suboptimal values in one property do not automatically disqualify a molecule, as long as the overall profile remains within a

Table 8: Reference ranges and interpretability of QED, SAS, and binding affinity scores.

Property	Range	Interpretation
QED	(0.0, 0.3)	Low drug-likeness; poor pharmaceutical potential
	(0.3, 0.5)	Suboptimal but potentially useful structures
	(0.5, 0.7)	Typical lead-like drug candidates
	(0.7, 0.9)	High drug-likeness, good design quality
	(0.9, 1.0)	Excellent drug-likeness (rare)
SAS	(1.0, 3.0)	Very easy to synthesize; trivial structures
	(3.0, 5.0)	Easy to synthesize using standard routes
	(5.0, 6.5)	Moderately challenging to synthesize
	(6.5, 8.0)	Synthesis may require complex procedures and conditions
	(8.0, 10.0)	Extremely difficult or infeasible to synthesize
Binding Affinity (kcal/mol)	(-15.0, -9.0)	Very high binding affinity; near irreversible
	(-9.0, -7.0)	Strong binding; ideal for inhibitors
	(-7.0, -5.5)	Moderate binding; potentially active
	(-5.5, -4.5)	Binding is weak; potential for initial target engagement
	(-4.5, 0.0)	Very weak binding; typically inactive

409 plausible range. This joint criterion enables a more holistic assessment of generated molecules and  
 410 better reflects the multidimensional requirements of early-stage drug-like candidates. Formally, the  
 411 Top Molecules score is defined as follows:

$$\text{Top Molecules} = \frac{|\{m \in \mathcal{M}_{\text{novel}} \mid \text{QED}(m) \geq 0.4 \wedge \text{SAS}(m) \leq 8 \wedge \text{Affinity}(m) \leq -4.5\}|}{|\mathcal{M}_{\text{gen}}|} \times 100\% \quad (42)$$

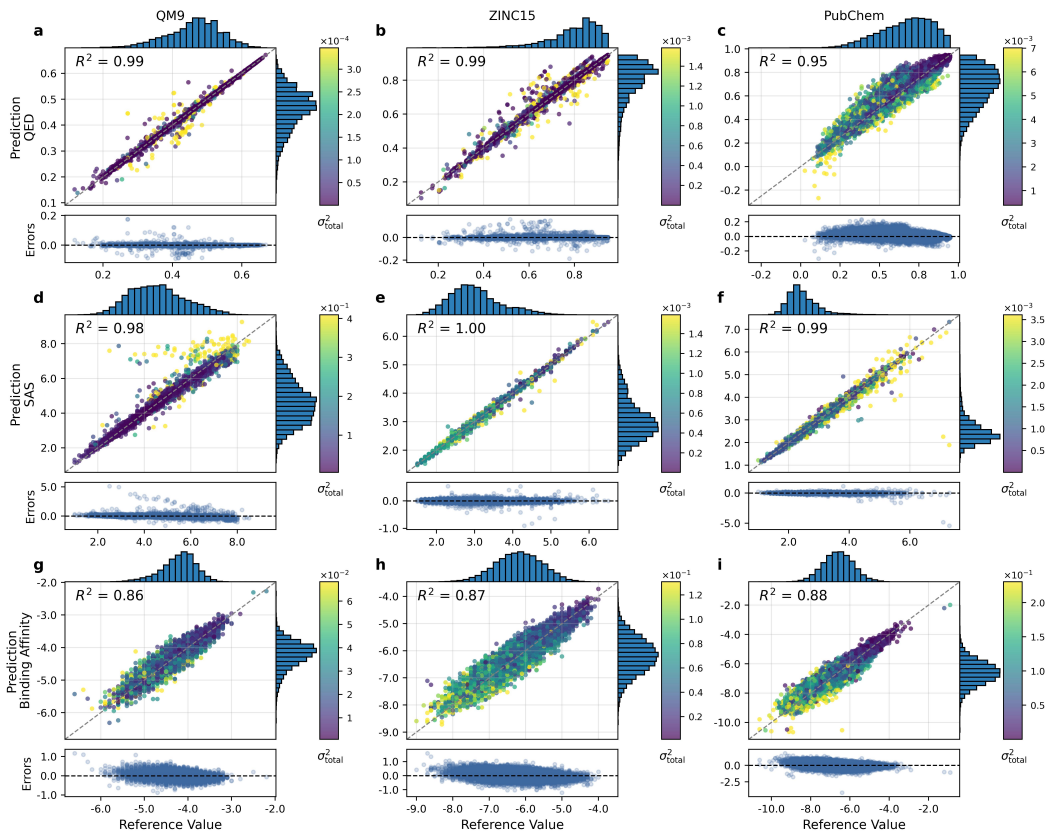
## 412 B.9 Devices and Computational Setup

413 All experiments are conducted on NVIDIA A100 GPUs with 80 GB of memory. On the QM9  
 414 dataset, both EDM and GeoLDM require approximately 90 seconds per training iteration during  
 415 pretraining. Due to its additional modules and guidance mechanisms, GFMDiff takes around 5  
 416 minutes per iteration. During RL-based EDM optimization, which includes molecule generation,  
 417 property evaluation, and sample reuse, each optimization step takes about 2 minutes. Generating  
 418 a single molecule using EDM model requires approximately 20 seconds. On larger datasets, the  
 419 training time increases accordingly. On ZINC15, one pretraining iteration of EDM takes around  
 420 7 minutes, while on PubChem it takes approximately 10 minutes. During optimization based on  
 421 pretrained EDM models, each optimization step takes about 3 minutes on ZINC15 and 5 minutes on  
 422 PubChem.

## 423 C Results and Discussion

### 424 C.1 The Results of Surrogate Models

425 We assess the performance and reliability of the trained surrogate models using parity plots (Fig.3)  
 426 and calibration curves (Fig.4). The parity plots demonstrate strong alignment between predicted  
 427 and ground-truth values across all datasets and properties, with  $R^2$  scores consistently above 0.8,  
 428 indicating high regression accuracy. The accompanying histograms show that the predictions cover  
 429 a wide and balanced range of property values, while the residual plots below each panel reveal no  
 430 major systematic bias, with residuals centered around zero. These observations confirm that the  
 431 models are not only accurate but also generalizable across molecular variations. The calibration  
 432 curves further show that the predicted uncertainties are well-calibrated, with AUCE values below 0.1  
 433 in most cases—suggesting that the uncertainty estimates meaningfully reflect the actual prediction  
 434 errors. Together, these results confirm that the surrogate models provide accurate and trustworthy  
 435 property predictions, supporting their use in uncertainty-aware reward construction.



**Figure 3: Parity plots comparing ground-truth property values with predictions from surrogate models.** The panels visualize results across three datasets (QM9, ZINC15, PubChem) and three properties (QED, SAS, binding affinity). Each point represents a molecule in the test set, colored by the predicted total uncertainty  $\sigma^2_{\text{total}}$ . The histograms indicate the distribution of predicted and true values. Scatter plots below each parity plot show residual errors.  $R^2$  is reported in each plot to indicate prediction accuracy.  $R^2$  values above 0.8 are generally considered to reflect strong predictive performance.



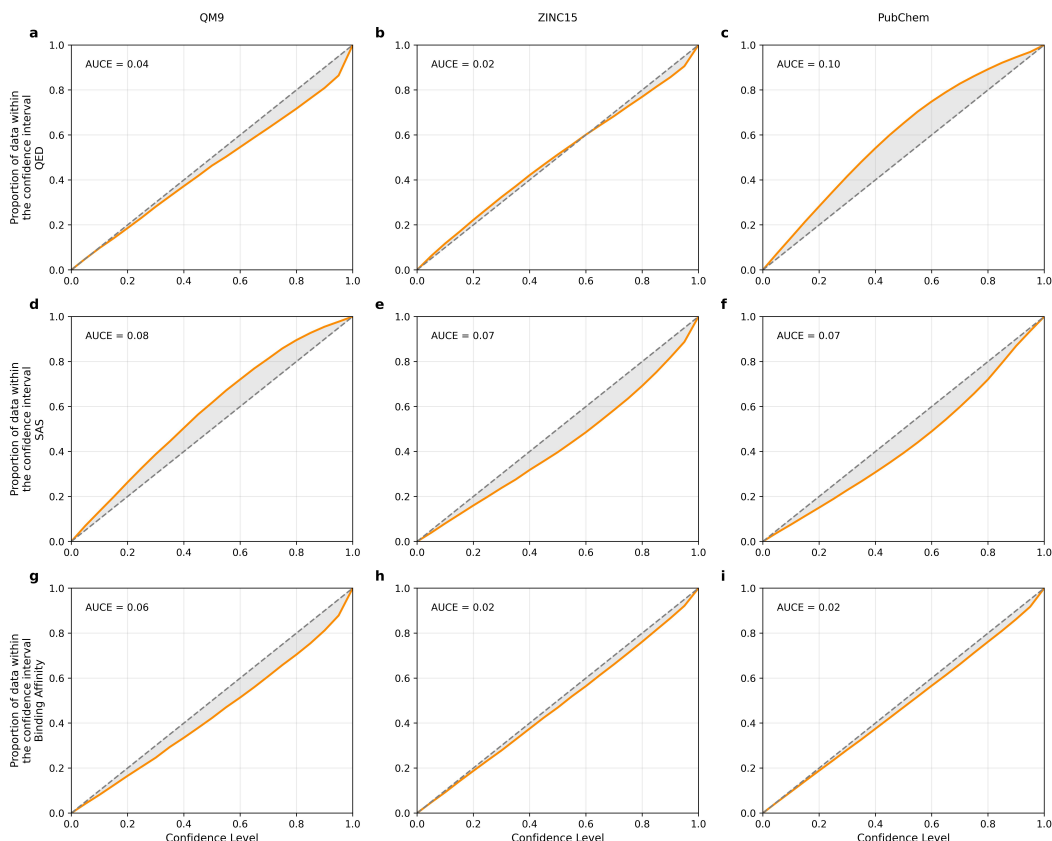


Figure 4: **Confidence-based calibration curves (orange) of our surrogate models.** The plots show calibration performance on the test set across three datasets (QM9, ZINC15, PubChem) and three properties (QED, SAS, binding affinity). AUC shown in each panel, quantifies miscalibration—lower AUC values indicate better uncertainty calibration. The gray region denotes the deviation from ideal calibration (diagonal dashed line). AUC values below 0.1 are generally considered indicative of well-calibrated uncertainty estimates.

## 436 C.2 Extended Discussion on Baseline Comparisons

437 Similar to many multi-objective optimization problems, our binary reward design for these baselines  
 438 creates a sparse and discontinuous optimization landscape. Molecules that are close to meeting the  
 439 cutoffs but fail in one property receive no learning signal, making it difficult for the policy to gradually  
 440 improve. This effect is exacerbated on challenging datasets where valid or near-optimal molecules  
 441 are rare in the initial stages of training. As a result, during the training of some baselines, nearly all  
 442 sampled molecules receive zero reward, leading to slow convergence and limited exploration.

443 SFT-PG approaches the problem from a different angle by minimizing the distributional gap between  
 444 the model’s generated property distributions and an ideal target distribution. While this avoids the  
 445 brittleness of hard thresholds, it still suffers from indirect optimization. Without explicit instance-level  
 446 reward feedback, the model may align distribution statistics while continuing to produce low-quality  
 447 individual samples. This is because the training objective only penalizes aggregate discrepancies  
 448 rather than enforcing quality at the individual molecule level. As a result, the model can satisfy  
 449 distributional targets by generating a large number of average-quality samples, even if many of them  
 450 fail to meet practical thresholds. This limitation is further exacerbated in settings with noisy or  
 451 imperfect surrogate labels, where lack of instance-level supervision prevents the model from learning  
 452 fine-grained improvements.

453 In contrast, our uncertainty-aware RL framework addresses these limitations through a probabilistic  
 454 reward function that estimates the likelihood of each molecule satisfying individual property con-

455 straints. This yields several advantages. First, it provides denser and more informative gradient  
456 signals, allowing the policy to improve even for partially successful molecules. Second, it enhances  
457 sample efficiency and convergence speed by avoiding reward sparsity. Third, by incorporating both  
458 aleatoric and epistemic uncertainty, the model learns to prioritize high-confidence regions of chemical  
459 space, which is especially important under noisy or limited data.

### 460 C.3 Extended Discussion on Ablation Studies

461 To assess the contribution of each component in our uncertainty-aware RL framework, we conduct  
462 ablation studies that isolate their individual effects. Results show that uncertainty-guided reward  
463 scaling is essential for stability and reliability. It prevents the model from overfitting to spurious high  
464 scores and ensures consistent improvement across multiple objectives. The diversity penalty further  
465 encourages structural exploration by discouraging repeated generation of similar molecules, leading  
466 to a broader search and a higher yield of top-performing candidates. Dynamic thresholding allows the  
467 model to adaptively respond to varying property distributions, avoiding premature convergence caused  
468 by rigid cutoffs. Lastly, the reward bonus effectively sharpens selection pressure toward high-quality  
469 molecules, enhancing validity while preserving uniqueness and novelty. Each component contributes  
470 to a different dimension of generation, and their combination yields robust and pharmaceutically  
471 meaningful outputs.

### 472 C.4 Extended Discussion on MD and ADMET Prediction

473 While the main text highlights the general trend of MD and ADMET results for generated molecules,  
474 several specific observations warrant further discussion. In the MD simulations, all three top-  
475 ranked candidates formed stable complexes with EGFR over the full 10 ns production trajectories.  
476 One candidate (Fig. 4c in main manuscript) exhibited excellent structural stability, maintaining  
477 a low RMSD relative to the equilibrated apo conformation, indicative of strong and persistent  
478 binding. Another (Fig. 4d in main manuscript) showed minimal conformational fluctuations across  
479 the simulation and preserved critical hydrogen bonds within the ATP-binding site, reinforcing its  
480 potential as a potent inhibitor.

481 Beyond structural dynamics, in silico pharmacokinetic profiling of these candidates using ADMET-AI  
482 revealed consistently favorable drug-like properties. All three molecules exhibited high predicted  
483 human intestinal absorption, low blood-brain barrier permeability, and no violations of Lipinski’s rule.  
484 Importantly, one of the molecule (Fig. 4d in main manuscript) showed low predicted hepatotoxicity  
485 and high metabolic stability, while another (Fig. 4b in main manuscript) scored favorably across  
486 P450 inhibition panels, suggesting low potential for metabolic drug-drug interactions.

487 These extended findings further support that top molecules generated by our framework are not only  
488 structurally valid and property-optimized, but also promising in terms of downstream pharmacological  
489 viability. Incorporating such analyses into post-generation filtering can substantially improve the  
490 relevance of AI-designed molecules for real-world drug discovery workflows.

### 491 C.5 Visualization for Generated Molecules

492 We visualize representative candidate molecules generated by RL-guided diffusion models pre-trained  
493 on different datasets in Fig. 5. For optimized model, we rank all generated molecules based on their  
494 multi-objective uncertainty scores. The top eight molecules with the highest scores from each model  
495 are shown. This selection strategy ensures that the displayed candidates are not only high-quality in  
496 terms of predicted properties, but also robust with respect to the model’s uncertainty.

### 497 C.6 Analysis on Other Diffusion Models

498 Table 9 summarizes the performance of our RL-guided optimization framework across other two  
499 diffusion models (GeoLDM and GFMDiff) on the QM9 dataset. Our method consistently improves  
500 the key evaluation metrics over the non-RL baselines, including validity, novelty and top molecules.  
501 This demonstrate that our method generalizes well across different diffusion models.

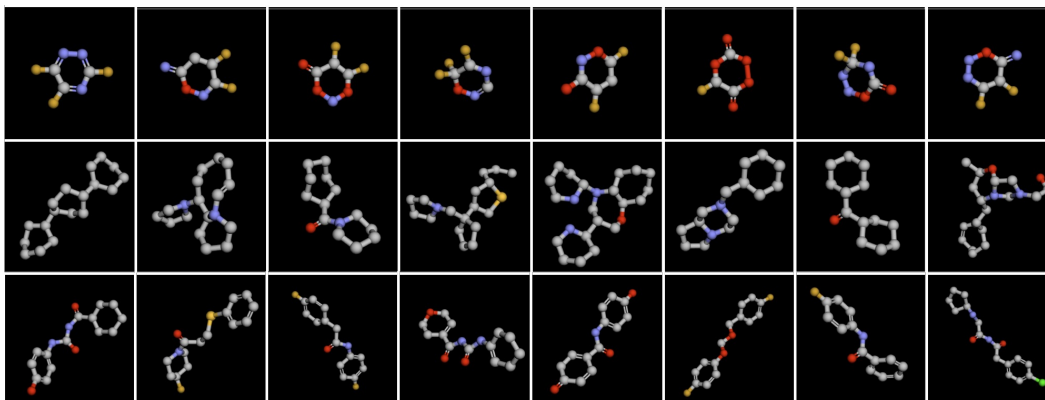


Figure 5: **Representative candidate molecules generated by RL-guided diffusion models.** Each model was pre-trained on one of the three datasets (QM9, ZINC15, PubChem) and generated 2,000 molecules. The top 8 molecules selected by composite properties are shown: first row—QM9, second row—ZINC15, third row—PubChem.

Table 9: **Performance of other diffusion models and our method on QM9 datasets.**

Model	Method	Val (%) (↑)	Uni (%) (↑)	Nov (%) (↑)	VUN (%) (↑)	ASta (%) (↑)	MSta (%) (↑)	Top (%) (↑)
GeoLDM	W/O RL	91.98 $\pm$ 0.17	95.13 $\pm$ 0.64	99.71 $\pm$ 0.07	87.25 $\pm$ 0.52	99.08 $\pm$ 0.16	95.67 $\pm$ 0.60	85.40 $\pm$ 0.00
	<b>Ours</b>	95.11 $\pm$ 1.24	94.12 $\pm$ 0.31	99.73 $\pm$ 0.13	89.28 $\pm$ 1.57	99.12 $\pm$ 0.17	95.54 $\pm$ 0.49	87.08 $\pm$ 0.02
GFMDiff	W/O RL	94.33 $\pm$ 0.17	96.34 $\pm$ 0.81	96.11 $\pm$ 1.03	87.35 $\pm$ 1.72	99.75 $\pm$ 0.03	98.23 $\pm$ 0.03	74.50 $\pm$ 0.00
	<b>Ours</b>	96.72 $\pm$ 1.27	95.93 $\pm$ 0.94	96.15 $\pm$ 0.21	89.21 $\pm$ 1.39	99.77 $\pm$ 0.10	98.35 $\pm$ 0.56	75.01 $\pm$ 0.01

Note: Each model generates 2,000 molecules per run. Results are averaged over three independent runs and reported as mean  $\pm$  95% confidence interval. "W/O RL" denotes vanilla diffusion models without RL. "Val", "Uni", and "Nov" represent the percentages of valid, unique, and novel molecules, respectively. "VUN" is their joint metric computed as Val  $\times$  Uni  $\times$  Nov, representing the percentage of molecules that are simultaneously valid, unique, and novel. "ASta" and "MSta" denote atom-level and molecule-level stability. "Top" indicates the proportion of generated molecules that simultaneously satisfy all three property constraints, using relaxed cutoffs (QED > 0.4, SAS < 8, and binding affinity < -4.5) to avoid missing potentially useful candidates. All metrics are reported as percentages, and higher values indicate better performance.

## 502 D User Interface

503 To improve reproducibility and practical applicability, we will release a Graphical User Interface  
 504 (GUI) alongside the source code on GitHub. As shown in Fig. 6, the interface is designed to  
 505 assist users without a programming background in generating drug-like candidate molecules for  
 506 specific target proteins. Users can upload a protein structure in PDB format and configure generation  
 507 parameters such as the desired number of molecules and the output format (PDB or PDBQT). Based  
 508 on these inputs, the program can design 3D candidate molecules for any target protein that exhibit  
 509 high binding affinity, strong drug-likeness, and good synthetic accessibility, thereby facilitating active  
 510 molecule design.

511 Although the GUI can be executed on machines with only a CPU, we strongly recommend running it  
 512 on systems equipped with a GPU with at least 16 GB of memory for better efficiency. The interface  
 513 also includes an optional “Train Model” feature. When this option is enabled, the system trains a  
 514 diffusion model from scratch and optimizes it specifically for the uploaded target protein, potentially  
 515 yielding molecules with better overall performance. If this option is disabled, the model instead  
 516 uses our pre-trained unconditional diffusion model and performs optimization based on the target.  
 517 While this still allows for the generation of promising drug-like candidates, the performance may be  
 518 slightly reduced. The downloaded molecules can be directly used for docking, MD simulations, or  
 519 visualization. Model training may take up to 40 hours, with an additional 10 hours for optimization,  
 520 while molecule generation only requires a few minutes.

