Controllable Molecular Generative Foundation Models

Yihan Zhu
University of Notre Dame
yzhu25@nd.edu

Yuhan Liu
University of Notre Dame
yliu57@nd.edu

Weijiang Li
University of Notre Dame
wli27@nd.edu

Tengfei Luo
University of Notre Dame
tluo@nd.edu

Meng Jiang
University of Notre Dame
mjiang2@nd.edu

Abstract

Despite the success of foundation models in language and vision, molecular graph generation still lacks a unified framework for heterogeneous design tasks with reliable controllability. While reinforcement learning (RL) offers a natural post-training mechanism for task-specific optimization, applying it to graph generative models is hindered by the vast atom-wise action spaces and chemically invalid intermediate states. We propose **Co**ntrollable **Mole**cular Generative Foundation Models (CoMole), built with a unified motif-aware graph diffusion pipeline. By learning a motif-aware graph space, CoMole transfers pretrained structural priors into controllable generation, where RL optimizes conditional reverse policies over chemically meaningful decisions. We theoretically characterize the bottleneck of atom-level RL and justify motif-aware policy optimization. Across three heterogeneous benchmarks spanning materials and drug discovery, CoMole ranks first in controllability on all nine targets, reduces MAE by up to 48.2% relative to the strongest baselines, and maintains validity above 0.94 without rule-based correction or post-hoc filtering. We further show that CoMole transfers controllability to unseen properties by optimizing only task embeddings with the generator frozen, achieving performance competitive with strong task-specific baselines.

1 Introduction

Molecular inverse design, generating structures with desired functional properties, is a central challenge in scientific discovery, with applications spanning biomedicine and materials science. Recent graph diffusion methods have significantly advanced molecular generation [Vignac et al., 2023, Huang et al., 2023, Liu et al., 2024a]. However, these approaches remain largely task-specific and offer limited controllability over target properties [Qin et al., 2025, Liu et al., 2025]. In language and vision, foundation-model paradigms have enabled powerful generative systems through large-scale pretraining (PT), supervised fine-tuning (SFT), and reinforcement learning (RL)-based alignment [Bommasani et al., 2022, Zhang et al., 2023]. For molecular inverse design, the abundance of unlabeled chemical data alongside label-scarce downstream tasks motivates the development of unified molecular generative foundation models to bridge the controllability gap between current methods and practical inverse-design needs.

However, instantiating this paradigm with atom-level graph diffusion exposes a core bottleneck: beyond the trajectory collapse observed in atom-level diffusion RL [Dulac-Arnold et al., 2015, Liu et al., 2024b], the vast space of low-level atom-wise actions is poorly aligned with the chemically meaningful substructures through which chemists typically formulate structure-property rationales for molecular design. As illustrated in Figure 1, atom-level RL often fails to construct reliable substructures from local edits before it can associate structural changes with property rewards. This

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

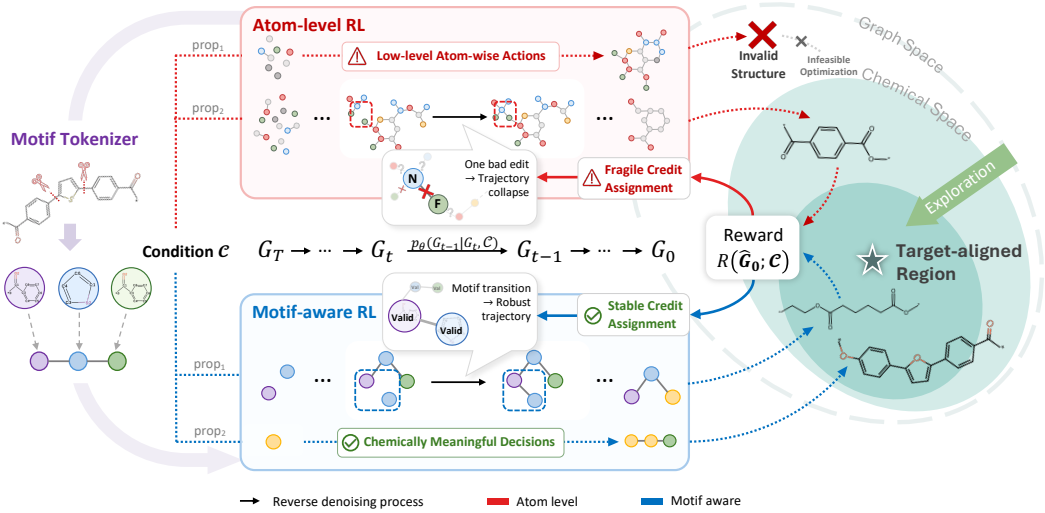

Figure 1: **Motif-aware RL as a key stage in training controllable molecular generative foundation models.** Atom-level RL over vast, low-level graph edits suffers trajectory collapse and fragile credit assignment, whereas motif-aware RL credits terminal rewards to chemically meaningful decisions, stabilizing policy updates.

fragility arises because each action must jointly coordinate atom types, bonds, and valence constraints. A single invalid choice can push the trajectory off the feasible chemical manifold and make later denoising difficult to recover. For example, the atom-level variant of our design collapses rule-free validity from 0.95 to 0.07 and worsens gas-permeability control MAE by 2.68× (Table 4).

To overcome this bottleneck, we reparameterize graph-diffusion RL with a motif-aware decision space. It preserves local structural flexibility while introducing rings, functional groups, and data-driven motifs as higher-level decisions. Since molecular properties depend on substructures and their complex interactions, this space lifts RL from fragile atom-wise construction paths to chemically meaningful decisions (e.g., attaching a benzene ring), allowing rewards on generated structures to be credited more directly to property-relevant actions. This abstraction also promotes chemical validity by preserving internally coherent substructures and reducing the risk of trajectory corruption from invalid atom-wise edits. Beyond using motifs mainly as generation priors [Jin et al., 2018, Kong et al., 2022], our design uses them to stabilize RL optimization toward controllable generation.

We introduce **CoMole**, to our knowledge the first family of **Co**ntrollable **Mole**cular Generative Foundation Models for heterogeneous inverse design. The central idea is to transfer pretrained structural knowledge into controllable inverse design by optimizing conditional reverse policies via RL over chemically meaningful decisions. Concretely, CoMole adopts the Node Pair Encoding (NPE) tokenization from DemoDiff [Liu et al., 2025] that preserves singleton atoms and attachment-level information while merging frequent adjacent units from the pretraining distribution into motif-aware graph states. Over this space, we propose a motif-aware policy-optimization formulation for graph diffusion. Building on this formulation, we further develop a foundation-modeling method for heterogeneous molecular inverse design: PT learns transferable structural priors, SFT introduces conditioning for multiple properties, and RL aligns the conditional reverse diffusion policy with terminal target-property rewards. In the main text, we optimize the proposed motif-aware structured policy with proximal policy optimization (PPO), while alternative policy-optimization objectives are discussed in Appendix D.5.3.

We theoretically characterize the structural bottleneck of atom-level RL and motivate motif-aware policy optimization. Empirically, we evaluate CoMole on three materials and drug-discovery benchmarks spanning numerical and categorical conditions. Across all nine targets, CoMole ranks first in controllability: on both polymer benchmarks, it reduces MAE by over 44% relative to the best baseline. On small-molecule tasks, it reduces FreeSolv MAE by 13.1% and achieves 1.0 accuracy on BACE classification. These gains are achieved while maintaining validity above 0.94 without rule-based correction or post-hoc filtering. We further show that CoMole transfers controllability to unseen property targets by learning only task embeddings while keeping the generator frozen, achieving performance competitive with baselines trained directly on those targets. Together, these results

suggest that our design learns transferable structure-property knowledge that supports controllable generation across heterogeneous inverse-design tasks.

2 Preliminaries

Notation. Let $x \in \mathcal{M}_{\text{val}}$ be a chemically valid molecular graph and ϕ a learned graph tokenizer. The tokenizer maps x to a motif graph $z_0 = \phi(x) \in \mathcal{Z}_0$, which serves as the motif state for diffusion generation and RL post-training. For conditional generation, $c = (k, y^\star) \in \mathcal{C}$ denotes the target condition, with task identity k and target value or label $y^\star$. We use D to denote the empirical training distribution.

2.1 Motif Graph Tokenization

We follow the Node Pair Encoding (NPE) algorithm introduced by DemoDiff [Liu et al., 2025] and learn a vocabulary from the pretraining dataset (Appendix C.2). The nodes of the resulting motif graph are atom-disjoint vocabulary units, including singleton atoms, preserved ring units, and data-driven merged substructures as higher-level motifs. The graph also stores inter-motif bond labels and directional attachment-position labels for lossless reconstruction. Since motif graphs have variable sizes, we pad them to a fixed number of motif slots and represent each state as $z = (X, E, P, m)$, where $m_i = 1$ indicates that slot i is active. The variable X_i encodes the motif type at slot i. For active motif pairs, $E_{ij} = E_{ji}$ denotes a symmetric categorical bond label, including a no-bond category. P_{ij} denotes the directional attachment-position label on motif i to motif j. Directed pairs without an attachment use a null attachment-position label. m is sampled once from a size prior and fixed throughout generation (see Appendix A.2). When no confusion arises, we omit m and write $z = (X, E, P)$.

2.2 Molecular Design with Graph Diffusion Transformers

Given a motif graph $z_0 = (X_0, E_0, P_0)$, we perform discrete diffusion over motif states $z_t = (X_t, E_t, P_t)$. The forward noising process is $q(z_{1:T} \mid z_0) = \prod_{t=1}^{T} q(z_t \mid z_{t-1})$, which progressively corrupts motif types, bond types, and attachment-position labels. Let $q_t(z_t \mid z_0)$ denote the induced marginal at step t. The reverse process starts from the prior $p(z_T)$ and is parameterized by a graph diffusion transformer: $p_\theta(z_{0:T} \mid c) = p(z_T) \prod_{t=1}^{T} \widetilde{p}_\theta(z_{t-1} \mid z_t, t, c)$. For unconditional pretraining, c is omitted. At each reverse step, the denoiser predicts the distributions $(\hat{X}_0, \hat{E}_0, \hat{P}_0) = f_\theta(z_t, t, c)$, which parameterize $\widetilde{p}_\theta(z_{t-1} \mid z_t, t, c)$ by first predicting the motif state $z_0 = (X_0, E_0, P_0)$. The model is trained with a masked denoising objective: $\mathcal{L}_{\text{diff}}(\theta) = \mathbb{E}_{(z_0,c)\sim \mathsf{D},\ t\sim\text{Unif}([T]),\ z_t\sim q_t(\cdot|z_0)} [\lambda_X \text{CE}_X + \lambda_E \text{CE}_E + \lambda_P \text{CE}_P]$. Here CE_X, CE_E, and CE_P are masked cross-entropy losses over active motif nodes, inter-motif edges, and directed motif pairs. Details are given in Appendix A.3.

3 Controllable Molecular Generative Foundation Modeling

Following Section 2, CoMole generates molecules by reversing a task-conditioned motif-graph diffusion process. Section 3.1 formulates PPO over the reverse trajectories, and Section 3.2 analyzes why our motif-aware design improves over atom-level RL.

3.1 Learning the Reverse Diffusion Process as a Policy

Given a condition c, reverse diffusion generates a graph through denoising z_T into z_0. Since intermediate noisy graphs are difficult to evaluate reliably with property oracles, we apply rewards only on the terminal graph. We therefore formulate reverse diffusion as a finite-horizon terminal-reward Markov decision process (MDP) and optimize the denoising policy over sampled trajectories.

MDP Formulation. We treat the reverse process as an MDP with horizon $H = T$. Given a target condition c, at MDP step $h = 0, \ldots, T-1$, the state $s_h = (z_{T-h}, T-h, c)$ contains the current noisy motif graph, timestep, and condition, and the action is the next reverse state $a_h = z_{T-h-1}$. During RL training, each rollout draws a condition $c \sim \mu_{\text{RL}}$ and samples $z_T \sim p(z_T)$, starting

from $s_0 = (z_T, T, c)$ and ending at $s_T = (z_0, 0, c)$. After the policy samples a_h, the next state is deterministically updated to $s_{h+1} = (a_h, T-h-1, c)$, so the only stochastic decision is the one-step reverse kernel:

$$\pi_\theta(a_h \mid s_h) = \widetilde{p}_\theta(z_{T-h-1} \mid z_{T-h}, T-h, c). \tag{1}$$

Although a_h is a structured motif-graph action, $\pi_\theta(a_h \mid s_h)$ factorizes over motif, bond, and attachment-position variables, allowing exact log-probability computation for RL. The explicit factorization is given in Appendix A.4.

Terminal Molecular Reward. Let $x_{\text{gen}} = \text{Dec}(z_0)$ be the molecule decoded from the final reverse state. For task k, let $\hat{o}_k(x)$ be the oracle output and define the target discrepancy $d_c(x) := \ell_k(\hat{o}_k(x), y^\star)$ for valid molecules. Here ℓ_k is task-specific, e.g., absolute error for regression or absolute probability-label difference for binary classification.

We use a terminal reward that combines validity and target satisfaction:

$$R(z_0; c) = w_{\text{val}}\, r_{\text{val}}(x_{\text{gen}}) + (1 - w_{\text{val}})\, r_{\text{prop}}(x_{\text{gen}}; c), \qquad w_{\text{val}} \in [0, 1]. \tag{2}$$

Here

$$r_{\text{val}}(x) = \begin{cases} 1, & x \in \mathcal{M}_{\text{val}}, \\ -1, & x \notin \mathcal{M}_{\text{val}}, \end{cases} \qquad r_{\text{prop}}(x; c) = \begin{cases} g_k(d_c(x)), & x \in \mathcal{M}_{\text{val}}, \\ 0, & x \notin \mathcal{M}_{\text{val}}. \end{cases} \tag{3}$$

For regression tasks, we use $g_k(d) = \exp[-(d/\sigma_k)^2]$, with $\sigma_k > 0$. For binary classification tasks, we set $g_k(d) = 1 - d$, so r_{prop} equals the oracle probability assigned to the target label. Thus $R(z_0; c) \in [-w_{\text{val}}, 1]$.

Policy Optimization. Given the rollout distribution μ_{RL}, the RL objective is

$$J(\pi_\theta) = \mathbb{E}_{c \sim \mu_{\text{RL}}} \mathbb{E}_{\tau = (z_T, \ldots, z_0) \sim p_{\pi_\theta}(\cdot \mid c)} \big[R(z_0; c) \big]. \tag{4}$$

We optimize (4) with PPO initialized from the SFT checkpoint. Because the reward is terminal-only and $\gamma = 1$, every reverse step in a rollout has return $R(z_0; c)$. At each MDP step h, we use an MLP value head to estimate the state value $V_\psi(s_h)$. The advantage is $A_h = R(z_0; c) - V_\psi(s_h)$.

For on-policy rollouts collected by $\pi_{\theta_{\text{old}}}$, the PPO importance ratio is

$$\rho_h = \frac{\pi_\theta(a_h \mid s_h)}{\pi_{\theta_{\text{old}}}(a_h \mid s_h)}. \tag{5}$$

We maximize the clipped PPO surrogate

$$\mathcal{J}^{(K)}_{\text{clip}} = \mathbb{E}\left[\frac{1}{K} \sum_{h=T-K}^{T-1} \min\Big(\rho_h A_h,\ \text{clip}(\rho_h, 1-\epsilon, 1+\epsilon) A_h \Big) \right], \tag{6}$$

where $K = T$ gives the normalized full-horizon objective. In implementation, we set $K = 30$ and optimize the PPO loss with critic loss, entropy bonus, and KL regularization. Details are given in Sections A.5 and D.4.

3.2 Theoretical Analysis

We analyze the RL stage and the motif-aware decision space. Let π_{ref} denote the frozen SFT reference policy used for KL regularization, $\mathcal{A}(s)$ the finite reverse action space at state s, and $p_\pi(\tau \mid c)$ the trajectory distribution induced by policy π under condition c.

3.2.1 The Role of RL in Condition Control

As in Section 3.1, reverse diffusion defines a finite-horizon MDP with terminal reward $R(z_0; c)$ on the final clean motif graph z_0. Following KL-regularized control and control-as-inference formulations [Todorov, 2006, Levine, 2018], for a fixed condition c we consider the population objective

$$\mathcal{J}_\beta(\pi; c) = \mathbb{E}_{\tau = (z_T, \ldots, z_0) \sim p_\pi(\cdot \mid c)} \left[R(z_0; c) - \beta \sum_{h=0}^{T-1} \text{KL}(\pi(\cdot \mid s_h) \,\|\, \pi_{\text{ref}}(\cdot \mid s_h)) \right], \quad \beta > 0. \tag{7}$$

This objective is a population-level analytical reference, while PPO approximates the corresponding policy improvement in practice using sampled rollouts and a learned critic.

Proposition 3.1 (Regularized Bellman characterization)**.** *For a fixed condition c, assume policy π is supported on the support of π_{ref}, ensuring the KL term is finite. Let $V_h^\star(s)$ denote the optimal KL-regularized value from MDP step h. At the terminal step, for $s_T = (z_0, 0, c)$, $V_T^\star(s_T) = R(z_0; c)$. For $h = T-1, \ldots, 0$, define*

$$Q_h^\star(s,a) = V_{h+1}^\star(s'), \qquad s' = (a, T-h-1, c), \tag{8}$$

$$V_h^\star(s) = \beta \log \sum_{a \in \mathcal{A}(s)} \pi_{\mathrm{ref}}(a \mid s) \exp\left(\frac{Q_h^\star(s,a)}{\beta}\right). \tag{9}$$

Then the maximizer of Eq. (7) *is*

$$\pi_h^\star(a \mid s) = \pi_{\mathrm{ref}}(a \mid s) \exp\left(\frac{Q_h^\star(s,a) - V_h^\star(s)}{\beta}\right), \tag{10}$$

with support contained in that of the reference policy, i.e., $\pi_h^\star(a \mid s) = 0$ whenever $\pi_{\mathrm{ref}}(a \mid s) = 0$. The optimal value is

$$\mathbb{E}_{z_T \sim p(\cdot)}\left[V_0^\star((z_T, T, c))\right].$$

The optimal policy is an exponential reweighting of the reference reverse kernel toward actions with larger downstream value $Q_h^\star$. For any two actions a, a' with positive reference probability,

$$\frac{\pi_h^\star(a \mid s)/\pi_{\mathrm{ref}}(a \mid s)}{\pi_h^\star(a' \mid s)/\pi_{\mathrm{ref}}(a' \mid s)} = \exp\left(\frac{Q_h^\star(s,a) - Q_h^\star(s,a')}{\beta}\right). \tag{11}$$

Thus, actions with higher downstream value are amplified relative to π_{ref}. The proof follows by applying a standard Gibbs variational identity at each state, details are given in Appendix B.1.

Corollary 3.2 (Downstream-value amplification)**.** *Fix a state s, a reverse step h, and a subset of actions $\mathcal{G} \subseteq \mathcal{A}(s)$. Suppose there exist $b \in \mathbb{R}$ and $\Delta > 0$ such that*

$$Q_h^\star(s,a) \geq b + \Delta \quad \textit{for all } a \in \mathcal{G}, \qquad Q_h^\star(s,a) \leq b \quad \textit{for all } a \notin \mathcal{G}.$$

Let $p_G := \pi_{\mathrm{ref}}(\mathcal{G} \mid s)$. Then

$$\pi_h^\star(\mathcal{G} \mid s) \geq \frac{e^{\Delta/\beta} p_G}{e^{\Delta/\beta} p_G + 1 - p_G}. \tag{12}$$

In particular, high-value actions with nonzero reference support receive amplified probability mass, while unsupported actions remain unsupported.

This provides a mechanism by which RL can improve target controllability: KL-regularized optimization amplifies high-value denoising choices while staying close to the SFT reference policy. This amplification requires target-relevant actions to have nonzero support under π_{ref}. The readiness analysis in Appendix D.5.4 empirically probes this support through repeated sampling from the SFT checkpoint of CoMole.

3.2.2 Decision Complexity of Atom-level and Motif-aware RL

The reverse action $a_h = z_{T-h-1}$ is a structured graph-valued object. For a representation $r \in \{\mathrm{atom}, \mathrm{motif}\}$, write the one-step action as $a = (a_1, \ldots, a_{M_r(s)})$, where $M_r(s)$ is the number of stochastic categorical factors in the reverse kernel at state s. For motif graphs, these factors include motif types, inter-motif bonds, and directed attachment positions. Assume the policy and reference policy factorize as

$$\pi^{(r)}(a \mid s) = \prod_{j=1}^{M_r(s)} \pi_j^{(r)}(a_j \mid s, a_{<j}), \qquad \pi_{\mathrm{ref}}^{(r)}(a \mid s) = \prod_{j=1}^{M_r(s)} \pi_{\mathrm{ref},j}^{(r)}(a_j \mid s, a_{<j}). \tag{13}$$

Proposition 3.3 (Factorized one-step KL)**.** *Under Eq.* (13)*,*

$$\mathrm{KL}\Big(\pi^{(r)}(\cdot \mid s) \,\|\, \pi_{\mathrm{ref}}^{(r)}(\cdot \mid s)\Big) = \sum_{j=1}^{M_r(s)} \mathbb{E}_{a_{<j} \sim \pi^{(r)}}\Big[\mathrm{KL}\Big(\pi_j^{(r)}(\cdot \mid s, a_{<j}) \,\|\, \pi_{\mathrm{ref},j}^{(r)}(\cdot \mid s, a_{<j})\Big)\Big]. \tag{14}$$

Corollary 3.4 (KL budget per subdecision). *If* $\mathrm{KL}\Big(\pi^{(r)}(\cdot \mid s) \,\|\, \pi^{(r)}_{\mathrm{ref}}(\cdot \mid s)\Big) \le \eta$, *then*

$$\frac{1}{M_r(s)} \sum_{j=1}^{M_r(s)} \mathbb{E}_{a_{<j}\sim\pi^{(r)}} \Big[\mathrm{KL}\Big(\pi^{(r)}_j(\cdot \mid s, a_{<j}) \,\|\, \pi^{(r)}_{\mathrm{ref},j}(\cdot \mid s, a_{<j})\Big)\Big] \le \frac{\eta}{M_r(s)}. \tag{15}$$

Therefore, under the same reference-KL regularization, a larger $M_r(s)$ leaves less average policy change per categorical factor. Since atom-level representations induce larger $M_r(s)$ through atom and atom-pair decisions, terminal-reward policy improvement becomes difficult.

We next compare this decision size under atom-only and motif-aware representations. For a molecule x, let $n_{\mathrm{atom}}(x)$ and $n_{\mathrm{motif}}(x)$ denote its numbers of active atoms and active motifs. Since motifs partition the atom set, $1 \le n_{\mathrm{motif}}(x) \le n_{\mathrm{atom}}(x)$. To compare one-step reverse decision sizes, define

$$L_{\mathrm{atom}}(n) := n + \binom{n}{2} = \frac{n^2+n}{2}, \qquad L_{\mathrm{motif}}(n) \quad := n + \binom{n}{2} + n(n-1) = \frac{3n^2-n}{2}. \tag{16}$$

Here L_{atom} counts atom-type and atom-bond decisions in a standard graph diffusion model, while L_{motif} counts motif-type, inter-motif bond, and directed attachment-position decisions in the motif graph.

Corollary 3.5 (Motif-aware reduction of reverse decision size). *Let* $\chi(x) := n_{\mathrm{atom}}(x)/n_{\mathrm{motif}}(x)$ *be the atom-to-motif compression ratio. Then*

$$\frac{L_{\mathrm{motif}}(n_{\mathrm{motif}}(x))}{L_{\mathrm{atom}}(n_{\mathrm{atom}}(x))} = \frac{3n_{\mathrm{motif}}(x)^2 - n_{\mathrm{motif}}(x)}{n_{\mathrm{atom}}(x)^2 + n_{\mathrm{atom}}(x)} \le \frac{3}{\chi(x)^2}. \tag{17}$$

Thus, sufficiently large motif compression yields a substantially smaller one-step reverse decision size despite the additional attachment-position channel.

Combining Corollary 3.4 and Corollary 3.5, motif-aware actions improve the KL-regularized optimization geometry by reducing the number of categorical decisions that must be adjusted at each denoising step. In our implementation, the tokenizer yields an average atom-to-motif compression ratio above $5.5\times$ for polymers (see Appendix C.2), implying an upper-bound factor-count ratio of roughly $3/5.5^2 \approx 0.10$. The same tokenizer configuration yields weaker compression on small molecules, which is consistent with a relatively modest performance gain on FreeSolv in Table 3.

4 Experiments

RQ1: We validate the controllable generative power of CoMole against baselines from molecular optimization, generative modeling, and RL-based graph generation in Section 4.2. **RQ2**: We conduct further analysis to examine CoMole in Section 4.3.

4.1 Experimental Setup

We evaluate CoMole on three heterogeneous drug and material design task sets, covering both numerical and categorical properties. For each task set, CoMole is trained jointly across all tasks. We also include $\text{CoMole}_{\text{w/o RL}}$, an ablated variant without RL post-training, to isolate the effect of RL. We assess performance across seven metrics covering validity, distribution learning, and condition control. Full dataset statistics and implementation details are provided in Appendix C.1 and Appendix D.

Datasets For PT, we construct two unlabeled datasets: 13k real polymers for materials modeling and 10k molecules sampled from MoleculeNet [Wu et al., 2018] for drug-related tasks. For SFT, RL, and evaluation, we consider three task sets: (1) numerical polymer gas permeability conditions (O_2Perm, CO_2Perm, N_2Perm) [Thornton et al., 2012]; (2) numerical polymer density-functional-theory (DFT) properties (Eea, Egb, Egc) [Xu et al., 2023]; (3) a drug-related task set spanning physical chemistry (FreeSolv regression), biophysics (BACE classification), and physiology (BBBP classification) [Gao et al., 2022]. Two additional DFT properties, Ei and EPS, are held out from training for unseen-target generalization. We separate gas permeability and DFT properties for polymers since they represent distinct property families. A six-target polymer joint-training analysis is provided in Appendix D.5.1.

Table 1: Heterogeneous Conditional Molecular Generation of 10K Polymers: Results on three numerical properties (gas permeability for O_2, N_2, CO_2). MAE is calculated between the input conditions and the properties of the generated polymers using oracles. Best results are in **red**.

Model	Validity ↑	Distribution Learning			Condition Control			
	(w/o rule checking)	Diversity ↑	Similarity ↑	Distance ↓	O_2Perm ↓	N_2Perm ↓	CO_2Perm ↓	Avg. MAE ↓
Graph GA	1.0000 (N.A.)	0.8838	0.9233	9.1974	1.9921	2.3010	1.9577	2.0836
MARS	1.0000 (N.A.)	0.8302	0.9313	7.5231	1.8755	2.3397	1.6079	1.9410
LSTM-HC	0.9900 (N.A.)	0.8637	0.9492	6.3520	1.2168	1.5258	1.1370	1.2932
JTVAE-BO	1.0000 (N.A.)	0.7566	0.7524	24.6281	1.4501	1.2734	1.1529	1.2921
DiGress	0.9820 (0.2543)	0.8961	0.4350	26.0657	1.3147	1.5640	1.4411	1.4399
DiGress v2	0.9788 (0.3927)	0.9118	0.3027	18.9923	1.3355	1.5506	1.4427	1.4429
GDSS	0.9412 (0.8876)	0.7829	0.0030	31.2573	1.4821	1.5844	1.3503	1.4723
MOOD	0.9713 (0.9004)	0.8129	0.0924	35.8194	1.5727	1.6971	1.4646	1.5781
DeFoG	0.4609 (0.2988)	**0.9173**	0.6828	15.1224	1.4314	1.5850	1.1998	1.4054
Graph DiT	0.8176 (0.8536)	0.8725	0.9624	**5.8218**	0.8185	0.9472	0.7836	0.8498
FREED	0.9466 (N.A.)	0.8284	0.3046	13.9356	0.7376	0.8423	0.7652	0.7817
GDPO	0.2578 (0.2965)	0.8766	0.0973	28.4533	0.7934	0.8650	0.7527	0.8037
CoMole$_{\text{w/o RL}}$ (ours)	0.9883 (0.9844)	0.8564	**0.9630**	9.4849	0.6442	0.7888	0.6709	0.7013
CoMole (ours)	0.9539 (0.9500)	0.8644	0.9405	9.9854	**0.3975**	**0.4720**	**0.4250**	**0.4315**

Evaluation We use an 8:1:1 train/validation/test split and evaluate conditional generation on held-out test conditions. For each setting, we generate 10,000 samples and report: (1) molecular validity (Validity); (2) internal diversity among the generated examples (Diversity); (3) fragment-based similarity with the reference set (Similarity); (4) Fréchet ChemNet Distance with the reference set (Distance) [Preuer et al., 2018]; (5)-(7) MAE/Accuracy for the numerical/categorical task conditions (Property). Lower MAE or higher accuracy indicates stronger model controllability. We use Random Forests oracles for drug tasks [Gao et al., 2022] and GRIN for polymer tasks [Zhu et al., 2026]. Oracle analysis is provided in Appendix D.5.5. We additionally report novelty and uniqueness in Appendix D.5.6.

Baselines We compare against three families of baselines: (1) molecular optimization methods, including Graph-GA [Jensen, 2019], MARS [Xie et al., 2021], JTVAE with Bayesian optimization (JTVAE-BO) [Jin et al., 2018], and SMILES-LSTM with hill climbing (LSTM-HC) [Brown et al., 2019]; (2) graph generative models, including GDSS [Jo et al., 2022], DiGress [Vignac et al., 2023], MOOD [Lee et al., 2023], Graph DiT [Liu et al., 2024a], and DeFoG [Qin et al., 2025]; (3) RL-based graph generation methods, including FREED [Yang et al., 2021] and GDPO [Liu et al., 2024b]. Baselines are trained in their standard task-specialized setting. For task-set-level distribution metrics, we report the strongest specialized baseline.

4.2 RQ1: Heterogeneous Conditional Molecular Generation

As shown in Tables 1 to 3, CoMole **achieves the best controllability on all nine targets and maintains at least 0.94 validity even without rule checking**.

Chemical Validity Reported validity can overstate generative quality when it relies on hard-coded chemical rules. For example, graph-search methods such as GraphGA explicitly discard invalid molecules during mutation and crossover, thereby achieving perfect final validity. Removing rule checking in the decoding step, DiGress and MOOD suffer substantial validity drops (e.g., 0.98 to 0.25). Graph DiT is more robust, maintaining validity above 0.6, but still remains below rule-filtered search methods. GDPO achieves only 0.2965 validity on the gas-permeability task set, further highlighting the fragility of atom-level RL. In contrast, both CoMole and CoMole$_{\text{w/o RL}}$ maintain validity above 0.9 across all three datasets without rule-based checking, suggesting that the motif-aware graph space preserves chemical feasibility under both SFT and RL alignment.

Distribution Learning GraphGA and Graph DiT are strong in-distribution baselines, whereas several models such as GDSS and MOOD struggle to match the reference distribution. CoMole$_{\text{w/o RL}}$ is highly competitive, achieving the best fragment similarity on both polymer tasks (0.9630 and 0.9686) and the best Fréchet distance on the DFT task set. RL post-training shifts generation toward target-aligned regions, mildly on the polymer task sets but more substantially on the drug task set, where FreeSolv, BACE, and BBBP involve distinct physical-chemistry, target-bioactivity, and

Table 2: Heterogeneous Conditional Molecular Generation of 10K Polymers: Results on three numerical properties (Eea, Egb, Egc). MAE is calculated between the input conditions and the generated properties. Best results are in **red**.

Model	Validity ↑	Distribution Learning			Condition Control			
	(w/o rule checking)	Diversity ↑	Similarity ↑	Distance ↓	Eea ↓	Egb ↓	Egc ↓	Avg. MAE ↓
Graph GA	1.0000 (N.A.)	0.8552	0.8569	7.9635	1.2773	1.4989	1.3846	1.3869
MARS	1.0000 (N.A.)	0.8617	0.8899	7.9519	1.3798	1.6541	1.4233	1.4857
LSTM-HC	0.9976 (N.A.)	0.8641	0.9325	5.2350	1.1667	1.3761	1.2062	1.2497
JTVAE-BO	1.0000 (N.A.)	0.7859	0.7520	11.7901	1.2574	1.4103	1.2900	1.3192
DiGress	0.6970 (0.4834)	0.9112	0.4908	13.2764	0.8473	1.2600	0.7520	0.9531
DiGress v2	0.7521 (0.5020)	0.9055	0.5043	15.3710	0.8137	1.2511	0.7603	0.9417
GDSS	0.6240 (0.6220)	**0.9208**	0.0368	18.8047	1.2254	2.8129	1.8973	1.9785
MOOD	0.7417 (0.6835)	0.8246	0.2032	18.2511	1.6882	2.3201	1.7129	1.9071
DeFoG	0.6504 (0.5452)	0.8986	0.8033	6.0314	0.6714	0.9164	0.7453	0.7777
Graph DiT	0.6868 (0.6720)	0.8791	0.8965	5.4507	0.6075	0.5192	0.6739	0.6002
FREED	1.0000 (N.A.)	0.8308	0.2123	15.8095	0.5687	1.3398	0.9175	0.9420
GDPO	0.6504 (0.7598)	0.8902	0.8118	8.1592	0.5244	0.7081	0.5762	0.6029
CoMole$_{\text{w/o RL}}$ (ours)	0.9688 (0.9546)	0.8860	**0.9686**	**5.1352**	0.2874	0.4713	0.3772	0.3786
CoMole (ours)	0.9431 (0.9414)	0.8902	0.8527	9.3369	**0.2392**	**0.4129**	**0.2813**	**0.3111**

Table 3: Heterogeneous Conditional Molecular Generation of 10K Molecules: Results on one numerical property (FreeSolv) and two categorical properties (BACE, BBBP). MAE/Accuracy is calculated between the input conditions and the generated properties. Avg. Rank is obtained by ranking each method per metric and averaging across property metrics. Best results are in **red**.

Model	Validity ↑	Distribution Learning			Condition Control			
	(w/o rule checking)	Diversity ↑	Similarity ↑	Distance ↓	FreeSolv ↓	BACE ↑	BBBP ↑	Avg. Rank ↓
Graph GA	1.0000 (N.A.)	0.8806	**0.9658**	7.6709	2.0405	0.4691	0.3017	12.3333
MARS	1.0000 (N.A.)	0.8571	0.7914	8.9636	2.3988	0.5320	0.5389	9.0000
LSTM-HC	0.9802 (N.A.)	0.9122	0.8590	14.8023	1.8267	0.5076	0.5242	8.0000
JTVAE-BO	1.0000 (N.A.)	0.6996	0.6078	33.4123	1.9124	0.4632	0.4955	11.0000
DiGress	0.5061 (0.4203)	0.8961	0.6959	6.5017	1.9023	0.5263	0.6726	7.3333
DiGress v2	0.5172 (0.3921)	0.9016	0.7160	7.5507	1.8906	0.5215	0.6771	7.0000
GDSS	0.7928 (0.8053)	0.8968	0.5733	9.5036	3.5770	0.5036	0.5044	12.0000
MOOD	0.7619 (0.5787)	0.9126	0.1901	37.2665	2.0941	0.5066	0.4821	11.3333
DeFoG	0.5034 (0.3188)	**0.9265**	0.3836	8.1281	1.9216	0.5587	0.4636	9.3333
Graph DiT	0.8344 (0.8217)	0.8732	0.9147	9.1130	1.2390	0.9133	0.9342	2.6667
FREED	0.9980 (N.A.)	0.8326	0.1018	20.6090	2.4163	0.7756	0.9213	7.3333
GDPO	0.7307 (0.7266)	0.8992	0.7126	**5.3391**	1.3598	0.8457	0.8556	4.3333
CoMole$_{\text{w/o RL}}$ (ours)	0.9668 (0.9473)	0.8816	0.9616	7.6326	1.3169	0.9504	0.9439	2.3333
CoMole (ours)	0.9570 (0.9505)	0.8422	0.6401	17.6755	**1.0764**	**1.0000**	**0.9550**	**1.0000**

physiological-permeability objectives. This reflects the general controllability-distribution trade-off: when structure-property rationales are weakly correlated across tasks, improved controllability may require larger deviations from the learned generative prior.

Condition Controllability Graph DiT is a strong conditional generation baseline, and RL-based methods such as GDPO and FREED are competitive because they directly optimize target-aligned rewards. Nevertheless, CoMole achieves the best performance across all targets, with an average MAE reduction of 27% compared to its SFT-only variant CoMole$_{\text{w/o RL}}$, which itself ranks second overall in controllability. Compared with the best baseline in each setting, CoMole reduces MAE by 44.8% over FREED on gas-permeability properties and by 48.2% over Graph DiT on DFT properties. On the drug-related task set, CoMole reduces FreeSolv MAE by over 13% and achieves a BACE accuracy of 1.0.

4.3 RQ2: Ablation Studies and Model Analysis

Atom-level RL We replace the motif-aware graph representation in CoMole and CoMole$_{\text{w/o RL}}$ with an atom-level representation on the gas-permeability task set while keeping all other settings unchanged. As shown in Table 4, atom-level RL fails under $w_{\text{val}} = 0.1$: rule-free validity drops

Table 4: Atom-level RL ablation on the gas-permeability dataset. w_{val} is the validity-reward weight. Atom-level RL remains unstable and poorly controllable even with a larger validity weight.

Model	w_{val}	Validity ↑	Distribution Learning			Condition Control			
		(w/o rule checking)	Diversity ↑	Similarity ↑	Distance ↓	O_2Perm ↓	N_2Perm ↓	CO_2Perm ↓	Avg. MAE ↓
CoMole$_{\text{w/o RL}}$-atom	–	0.6876 (0.6374)	0.8902	0.8975	13.3321	1.0961	1.3941	1.0761	1.1888
CoMole$_{\text{w/o RL}}$	–	0.9883 (0.9844)	0.8564	0.9630	9.4849	0.6442	0.7888	0.6709	0.7013
CoMole-atom	0.1	0.2266 (0.0742)	0.8977	0.1849	32.9202	1.2423	1.8101	1.6688	1.5737
CoMole-atom	0.5	0.3164 (0.3086)	0.9074	0.0454	32.4114	1.3512	1.8947	1.5299	1.6076
CoMole	0.1	0.9539 (0.9500)	0.8644	0.9405	9.9854	0.3975	0.4720	0.4250	0.4315

Table 5: Generalization to unseen DFT targets (Ei, EPS). With frozen generator and task embeddings composed from existing tasks (Eea, Egb, Egc), CoMole achieves competitive performance against task-specific baselines.

Model	Validity ↑	Distribution Learning			Condition Control		
	(w/o rule checking)	Diversity ↑	Similarity ↑	Distance ↓	Ei ↓	EPS ↓	Avg. MAE ↓
LSTM-HC	0.9954 (N.A.)	0.8294	0.9388	4.0644	1.0602	1.2078	1.1340
DiGress	0.5460 (0.4630)	0.9100	0.3590	15.3168	0.7400	0.8143	0.7772
DeFoG	0.5624 (0.4948)	0.8811	0.7238	7.4383	0.6921	0.6674	0.6768
Graph DiT	0.6076 (0.7858)	0.8654	0.9124	5.6182	0.4586	0.4868	0.4727
CoMole (ours)	0.9182 (0.9037)	0.8537	0.6420	15.5045	0.7231	0.7610	0.7421

from 0.95 to 0.07, fragment similarity drops from 0.94 to 0.18, and Avg. MAE increases from 0.43 to 1.57. Increasing the validity weight to $w_{\text{val}} = 0.5$ partially recovers validity but further worsens controllability, indicating that reward reweighting cannot resolve the bottleneck of atom-level RL. The advantage of motif-aware representations is also visible before RL. These results support that atom-level RL is fragile under terminal rewards over long edit sequences, where stable structure-property relationships cannot be reliably learned.

Generalization We further evaluate whether CoMole can extend controllability to unseen property targets while keeping the generator frozen. For two held-out DFT properties, Ei and EPS, we split each into a 0.8/0.2 train/test split, using the training split to fit baselines and learn new task embeddings as convex combinations of the trained DFT task embeddings, e.g., $e_{\text{Ei}} = 0.68e_{\text{Eea}} + 0.12e_{\text{Egb}} + 0.20e_{\text{Egc}}$. As shown in Table 5, CoMole maintains high validity and achieves controllability competitive with baselines trained directly on these targets. These results suggest that CoMole captures structure-property knowledge that transfers across related targets, highlighting its potential for scalable and data-efficient extension to new properties.

5 Related Work

Molecular Optimization and Inverse Design Traditional molecular optimization searches chemical space with genetic algorithms, Monte Carlo tree search, or Bayesian optimization, typically requiring repeated black-box oracle queries [Jensen, 2019, Shahriari et al., 2016, Gao et al., 2022]. Recent graph generative models amortize this search by learning to sample molecules under target specifications, with diffusion and flow-based models such as GDSS, DiGress, Graph DiT, and DeFoG providing strong backbones for molecular graph generation [Jo et al., 2022, Vignac et al., 2023, Liu et al., 2024a, Qin et al., 2025]. In contrast, we target heterogeneous molecular inverse design and aim to bridge the controllability gap between existing methods and practical inverse-design needs.

Reinforcement Learning over Graph Generation Reinforcement learning provides a natural framework for aligning molecular generators with black-box objectives. Early methods optimize construction policies, such as fragment-based generation in FREED [Yang et al., 2021]. Recent work such as GDPO studies policy-gradient optimization for graph diffusion and highlights the instability of directly optimizing atom-level reverse trajectories [Liu et al., 2024b]. Our method performs RL over motif-aware graph diffusion instead, using chemically meaningful actions to stabilize policy optimization while improving controllability.

6 Conclusion

In this work, we introduce CoMole, controllable molecular generative foundation models for heterogeneous inverse design. CoMole turns pretrained graph-diffusion priors into controllable generation by optimizing conditional reverse policies via RL over motif-aware, chemically meaningful decisions. We theoretically characterize the decision complexity of atom-level RL and justify motif-aware policy optimization. Empirically, across three materials and drug-discovery benchmarks covering both numerical and categorical conditions, CoMole ranks first in controllability on all nine targets, maintains validity above 0.94 without rule-based correction, and generalizes to unseen properties by updating only task embeddings. These results suggest CoMole as a practical foundation for unified, scalable, and controllable molecular inverse design across heterogeneous property tasks.

Acknowledgement

This work was partially supported by NSF 2142827, 2146761, 2234058, 2332270, and 2341995 as well as AFOSR FA9550-25-1-0009. We also appreciate the support from the Foundation Models and Applications Lab at Notre Dame and ND-IBM Tech Ethics Lab. We thank Gang Liu for valuable early discussions on reasoning over molecules, including the use of motif vocabularies in that context.

References

T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019. URL https://arxiv.org/abs/1907.10902.

R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang. On the opportunities and risks of foundation models, 2022. URL https://arxiv.org/abs/2108.07258.

L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, Oct. 2001. ISSN 0885-6125. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

N. Brown, M. Fiscato, M. H. Segler, and A. C. Vaucher. Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108, 2019.

G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, T. Mann, T. Weber, T. Degris, and B. Coppin. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*, 2015.

W. Gao, T. Fu, J. Sun, and C. W. Coley. Sample efficiency matters: A benchmark for practical molecular optimization, 2022. URL https://arxiv.org/abs/2206.12411.

H. Huang, L. Sun, B. Du, and W. Lv. Conditional diffusion based on discrete graph structures for molecular graph generation, 2023. URL https://arxiv.org/abs/2301.00427.

J. H. Jensen. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572, 2019.

W. Jin, R. Barzilay, and T. Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pages 2323–2332. PMLR, 2018.

J. Jo, S. Lee, and S. J. Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International conference on machine learning*, pages 10362–10383. PMLR, 2022.

X. Kong, W. Huang, Z. Tan, and Y. Liu. Molecule generation by principal subgraph mining and assembling. *Advances in Neural Information Processing Systems*, 35:2550–2563, 2022.

S. Lee, J. Jo, and S. J. Hwang. Exploring chemical space with score-based out-of-distribution generation. In *International Conference on Machine Learning*, pages 18872–18892. PMLR, 2023.

S. Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review, 2018. URL https://arxiv.org/abs/1805.00909.

G. Liu, J. Xu, T. Luo, and M. Jiang. Graph diffusion transformers for multi-conditional molecular generation, 2024a. URL https://arxiv.org/abs/2401.13858.

G. Liu, J. Chen, Y. Zhu, M. Sun, T. Luo, N. V. Chawla, and M. Jiang. Graph diffusion transformers are in-context molecular designers, 2025. URL https://arxiv.org/abs/2510.08744.

Y. Liu, C. Du, T. Pang, C. Li, M. Lin, and W. Chen. Graph diffusion policy optimization, 2024b. URL https://arxiv.org/abs/2402.16302.

R. Ma and T. Luo. Pi1m: a benchmark database for polymer informatics. *Journal of Chemical Information and Modeling*, 60(10):4684–4690, 2020.

K. Preuer, P. Renz, T. Unterthiner, S. Hochreiter, and G. Klambauer. Fréchet chemnet distance: A metric for generative models for molecules in drug discovery. *Journal of Chemical Information and Modeling*, 58(9):1736–1741, 2018. doi: 10.1021/acs.jcim.8b00234. URL https://doi.org/10.1021/acs.jcim.8b00234. PMID: 30118593.

Y. Qin, M. Madeira, D. Thanou, and P. Frossard. Defog: Discrete flow matching for graph generation, 2025. URL https://arxiv.org/abs/2410.04263.

A. Z. Ren, J. Lidard, L. L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz. Diffusion policy policy optimization, 2024. URL https://arxiv.org/abs/2409.00588.

P. Renz, D. Van Rompaey, J. K. Wegner, S. Hochreiter, and G. Klambauer. On failure modes in molecule generation and optimization. *Drug Discovery Today: Technologies*, 32-33:55–63, 2019. ISSN 1740-6749. doi: https://doi.org/10.1016/j.ddtec.2020.09.003. URL https://www.sciencedirect.com/science/article/pii/S1740674920300159. Artificial Intelligence.

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016. doi: 10.1109/JPROC.2015.2494218.

A. W. Thornton, B. D. Freeman, and L. M. Robeson. Polymer gas separation membrane database. https://membrane-australasia.org/polymer-gas-separation-membrane-database/, 2012. Accessed: 2026-04-06.

E. Todorov. Linearly-solvable markov decision problems. *Advances in neural information processing systems*, 19, 2006.

A. Tripp and J. M. Hernández-Lobato. Genetic algorithms are strong baselines for molecule generation, 2023. URL https://arxiv.org/abs/2310.09267.

C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard. Digress: Discrete denoising diffusion for graph generation, 2023. URL https://arxiv.org/abs/2209.14734.

Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Y. Xie, C. Shi, H. Zhou, Y. Yang, W. Zhang, Y. Yu, and L. Li. Mars: Markov molecular sampling for multi-objective drug discovery. *arXiv preprint arXiv:2103.10432*, 2021.

C. Xu, Y. Wang, and A. Barati Farimani. Transpolymer: a transformer-based language model for polymer property predictions. *npj Computational Materials*, 9(1):64, 2023. doi: 10.1038/s41524-023-01016-5.

S. Yang, D. Hwang, S. Lee, S. Ryu, and S. J. Hwang. Hit and lead discovery with explorative rl and fragment-based molecule generation. *Advances in Neural Information Processing Systems*, 34: 7924–7936, 2021.

X. Zhang, Y. Zeng, J. Zhang, and H. Li. Toward building general foundation models for language, vision, and vision-language understanding tasks, 2023. URL https://arxiv.org/abs/2301.05065.

Y. Zhu, G. Liu, E. Inae, T. Luo, and M. Jiang. Learning repetition-invariant representations for polymer informatics, 2026. URL https://arxiv.org/abs/2505.10726.

A Details on Model

A.1 Model Architecture

Motif-aware graph-level tokens We instantiate the denoiser as a motif-aware DiT-style transformer over padded motif-graph states $z_t = (X_t, E_t, P_t, m)$. Following the graph-level token construction in DemoDiff [Liu et al., 2025], each transformer token corresponds to one motif slot and contains both its motif identity and its row-wise relation profile. For an active motif slot i, the initial token is

$$u_i = g_X(X_{t,i}) \parallel g_E(E_{t,i:}) \parallel g_P(P_{t,i:}),$$

where g_X is a motif-type embedding, and g_E, g_P are linear projections of the vectorized one-hot rows of inter-motif bond labels and directed attachment-position labels. The fixed padded slot order is used when vectorizing $E_{t,i:}$ and $P_{t,i:}$, so the j-th block always corresponds to the relation from motif slot i to motif slot j. Thus, pair-specific topology is injected before self-attention rather than pooled into an unordered summary. Null labels are used for non-bonded pairs and missing attachment positions, and entries involving padded slots are masked out.

Transformer backbone and conditioning The backbone is a stack of motif-slot self-attention blocks with DiT-style adaptive LayerNorm. The condition consists of a task identity and a target value or label: the task identity is represented by a learned embedding, the target is encoded by a small MLP, and the two are fused into a condition vector. This condition vector is added to the timestep embedding and used to modulate each transformer block. For unconditional pretraining, the condition vector is omitted. Since bond and attachment-position information is already included in the graph-level motif tokens, we do not add a separate edge bias to self-attention.

Output heads Let h_i be the final hidden representation of motif slot i, split as

$$h_i = h_i^X \| h_i^E \| h_i^P.$$

The three parts predict motif identities, bond rows, and attachment-position rows through residual row-wise heads:

$$\ell_i^X = W_X h_i^X + \mathrm{onehot}(X_{t,i}),$$

$$\widetilde{L}_i^E = \mathrm{reshape}(W_E h_i^E) + \mathrm{onehot}(E_{t,i:}), \qquad \ell_{ij}^E = \frac{1}{2}\left(\widetilde{L}_i^E[j] + \widetilde{L}_j^E[i]\right),$$

and

$$L_i^P = \mathrm{reshape}(W_P h_i^P) + \mathrm{onehot}(P_{t,i:}), \qquad \ell_{ij}^P = L_i^P[j].$$

The bond logits are symmetrized because $E_{ij} = E_{ji}$, while the attachment-position logits remain directional because P_{ij} and P_{ji} encode different attachment positions. All logits involving padded slots are masked out during training, sampling, and PPO log-probability computation. The resulting logits parameterize the endpoint-state predictions $(\hat{X}_0, \hat{E}_0, \hat{P}_0)$, which are converted into reverse-step probabilities through the x_0-prediction posterior.

A.2 Mask handling

All main-text distributions are conditional on a fixed node mask m. If m is not externally specified at sampling time, the full generative model can be written as

$$p_\theta(z_{0:T}, m \mid c) = \nu(m)\, p(z_T \mid m) \prod_{t=1}^{T} \widetilde{p}_\theta(z_{t-1} \mid z_t, t, c, m), \tag{18}$$

where ν is a prior over masks (equivalently, over graph sizes). The main text suppresses this conditioning for readability.

A.3 Masked denoising loss

Let $N_{\max}$ denote the fixed maximum number of motif nodes after padding. For a mask m, define the active node, undirected pair, and directed pair index sets

$$\Omega_X(m) = \{i \in [N_{\max}] : m_i = 1\},$$

$$\Omega_E(m) = \{(i,j) : 1 \le i < j \le N_{\max},\ m_i m_j = 1\},$$

and

$$\Omega_P(m) = \{(i,j) : i \ne j,\ m_i m_j = 1\}.$$

Here $\Omega_E(m)$ indexes undirected inter-motif bond variables, while $\Omega_P(m)$ indexes directional attachment-position variables. For directed pairs without an attachment, P_{ij} is assigned a null attachment-position label.

Then

$$\mathrm{CE}_X = -\frac{1}{|\Omega_X(m)|} \sum_{i \in \Omega_X(m)} \sum_{a=1}^{d_X} X_{0,ia} \log \hat{X}_{0,ia}, \tag{19}$$

$$\mathrm{CE}_E = -\frac{1}{|\Omega_E(m)|} \sum_{(i,j) \in \Omega_E(m)} \sum_{b=1}^{d_E} E_{0,ijb} \log \hat{E}_{0,ijb}, \tag{20}$$

$$\mathrm{CE}_P = -\frac{1}{|\Omega_P(m)|} \sum_{(i,j) \in \Omega_P(m)} \sum_{p=1}^{d_P} P_{0,ijp} \log \hat{P}_{0,ijp}. \tag{21}$$

If $\Omega_E(m)$ or $\Omega_P(m)$ is empty, the corresponding term is defined as 0.

A.4 Factorized reverse kernel and one-step log-probability

Conditioned on z_t, t, c, and the fixed mask m, we use a factorized categorical reverse kernel

$$\begin{aligned}\widetilde{p}_\theta(z_{t-1} \mid z_t, t, c, m) = &\prod_{i \in \Omega_X(m)} p^X_{\theta,t}(X_{t-1,i} \mid z_t, c, m) \prod_{(i,j) \in \Omega_E(m)} p^E_{\theta,t}(E_{t-1,ij} \mid z_t, c, m) \\ &\times \prod_{(i,j) \in \Omega_P(m)} p^P_{\theta,t}(P_{t-1,ij} \mid z_t, c, m)\,.\end{aligned} \tag{22}$$

The P-factor is defined over all active directed motif pairs. For pairs without an attachment, the corresponding categorical outcome is the null class 0.

Accordingly, for $a_h = z_{t-1}$ with $t = T - h$, the one-step log-probability used in PPO is

$$\begin{aligned}\log \pi_\theta(a_h \mid s_h) = &\sum_{i \in \Omega_X(m)} \log p^X_{\theta,t}(X_{t-1,i} \mid z_t, c, m) + \sum_{(i,j) \in \Omega_E(m)} \log p^E_{\theta,t}(E_{t-1,ij} \mid z_t, c, m) \\ &+ \sum_{(i,j) \in \Omega_P(m)} \log p^P_{\theta,t}(P_{t-1,ij} \mid z_t, c, m)\,.\end{aligned} \tag{23}$$

If the network predicts endpoint-state distributions $(\hat{X}_0, \hat{E}_0, \hat{P}_0)$, the factors above denote the corresponding z_{t-1}-probabilities induced by the reverse posterior parameterization.

A.5 PPO optimization details

Design rationale. The KL-regularized objective in Eq. (7) characterizes the desired policy improvement at the population level. In translating this objective into a practical PPO implementation, we use a standard surrogate decomposition: the critic is trained only against the terminal task reward, while KL regularization toward the frozen SFT reference policy is imposed as an explicit actor-side penalty. Specifically, the Monte Carlo return used for value learning is

$$G_h = R(z_0; c), \qquad h = 0, \ldots, T-1.$$

This keeps the value target tied to the fixed terminal reward function rather than to a policy-dependent KL cost that would change as the actor is updated. The KL coefficient c_{kl} in the actor loss therefore plays the empirical role of the regularization strength in the population objective, although it is not in one-to-one correspondence with β because PPO uses clipped importance ratios, finite on-policy batches, and normalized advantages. Thus, the implemented PPO update is best interpreted as a tractable regularized policy-improvement surrogate for Eq. (7), prioritizing stable value estimation and computational feasibility over exact dynamic-programming correspondence.

We initialize RL from the conditional diffusion model after SFT and denote the frozen reference policy by

$$\pi_{\mathrm{ref}} := \pi_{\theta_{\mathrm{ref}}}. \tag{24}$$

Let θ_{old} denote the behavior-policy parameters used to collect the current on-policy batch.

With a value estimator $V_\psi(s_h)$, the advantage is

$$A_h = G_h - V_\psi(s_h). \tag{25}$$

The PPO importance ratio is

$$\rho_h = \frac{\pi_\theta(a_h \mid s_h)}{\pi_{\theta_{\mathrm{old}}}(a_h \mid s_h)}. \tag{26}$$

The clipped actor loss, corresponding to the negative of the clipped surrogate in Eq. (6), is

$$\mathcal{L}_{\mathrm{clip}} = -\frac{1}{T}\sum_{h=0}^{T-1} \mathbb{E}\Big[\min\Big(\rho_h A_h,\ \mathrm{clip}(\rho_h, 1-\epsilon, 1+\epsilon)A_h\Big)\Big]. \tag{27}$$

The value loss, entropy bonus, and KL regularization toward the frozen reference policy are

$$\mathcal{L}_{\mathrm{value}} = \frac{1}{T}\sum_{h=0}^{T-1} \mathbb{E}\Big[\big(V_\psi(s_h) - G_h\big)^2\Big], \tag{28}$$

$$\mathcal{L}_{\mathrm{ent}} = \frac{1}{T}\sum_{h=0}^{T-1} \mathbb{E}\big[\mathcal{H}\big(\pi_\theta(\cdot \mid s_h)\big)\big], \tag{29}$$

$$\mathcal{L}_{\mathrm{KL}} = \frac{1}{T}\sum_{h=0}^{T-1} \mathbb{E}[\mathrm{KL}(\pi_\theta(\cdot \mid s_h) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s_h))]. \tag{30}$$

The total PPO loss minimized in practice is

$$\mathcal{L}_{\mathrm{PPO}} = \mathcal{L}_{\mathrm{clip}} + c_v\,\mathcal{L}_{\mathrm{value}} - c_e\,\mathcal{L}_{\mathrm{ent}} + c_{\mathrm{kl}}\,\mathcal{L}_{\mathrm{KL}}, \tag{31}$$

where $\epsilon > 0$ is the clipping radius and $c_v, c_e, c_{\mathrm{kl}} \geq 0$ are loss weights. All expectations are taken over $c \sim \mu_{\mathrm{RL}}$ and trajectories $\tau \sim p_{\pi_{\theta_{\mathrm{old}}}}(\cdot \mid c)$. In practice, advantages are normalized within each PPO batch.

B Details on Theory

Throughout this appendix, we condition on a fixed target condition c and a fixed mask m, with m suppressed in the main notation. We denote by π_{ref} the frozen reverse policy induced by the SFT checkpoint, which serves as the KL reference policy during PPO. All state, action, and trajectory spaces are finite because $N_{\max}$, d_X, d_E, d_P, and T are finite. If the mask is sampled, the same derivations apply after treating m as part of the state. All policies are assumed to be absolutely continuous with respect to the reference policy at each state, i.e.,

$$\pi(\cdot \mid s) \ll \pi_{\mathrm{ref}}(\cdot \mid s),$$

so that the KL terms are well-defined.

B.1 A Gibbs variational identity used in the Bellman proof

Lemma B.1 (Gibbs variational identity)**.** *Let $\mathcal{A}$ be a finite action set, let π_{ref} be a probability distribution on $\mathcal{A}$, let $Q : \mathcal{A} \to \mathbb{R}$, and let $\beta > 0$. Then*

$$\sup_{\pi \ll \pi_{\mathrm{ref}}} \left\{ \sum_{a \in \mathcal{A}} \pi(a) Q(a) - \beta \, \mathrm{KL}(\pi \,\|\, \pi_{\mathrm{ref}}) \right\} = \beta \log \sum_{a \in \mathcal{A}} \pi_{\mathrm{ref}}(a) \exp(Q(a)/\beta). \tag{32}$$

The unique maximizer is

$$\pi^\star(a) = \frac{\pi_{\mathrm{ref}}(a) \exp(Q(a)/\beta)}{\sum_{a' \in \mathcal{A}} \pi_{\mathrm{ref}}(a') \exp(Q(a')/\beta)}. \tag{33}$$

Proof. Define

$$Z := \sum_{a \in \mathcal{A}} \pi_{\mathrm{ref}}(a) \exp(Q(a)/\beta)$$

and

$$\pi^\star(a) = \frac{\pi_{\mathrm{ref}}(a) \exp(Q(a)/\beta)}{Z}.$$

For any $\pi \ll \pi_{\mathrm{ref}}$,

$$\begin{aligned}
\mathrm{KL}(\pi \,\|\, \pi^\star) &= \sum_a \pi(a) \log \frac{\pi(a)}{\pi^\star(a)} \\
&= \sum_a \pi(a) \log \frac{\pi(a) Z}{\pi_{\mathrm{ref}}(a) \exp(Q(a)/\beta)} \\
&= \log Z - \frac{1}{\beta} \sum_a \pi(a) Q(a) + \mathrm{KL}(\pi \,\|\, \pi_{\mathrm{ref}}).
\end{aligned}$$

Rearranging gives

$$\sum_a \pi(a) Q(a) - \beta \, \mathrm{KL}(\pi \,\|\, \pi_{\mathrm{ref}}) = \beta \log Z - \beta \, \mathrm{KL}(\pi \,\|\, \pi^\star).$$

The right-hand side is maximized uniquely when $\pi = \pi^\star$, which proves the claim. □

B.2 Proof of Proposition 3.1

Proof. We prove the result by backward induction. At the terminal state $s_T = (z_0, 0, c)$, the value is

$$V_T^\star(s_T) = R(z_0; c).$$

Assume $V_{h+1}^\star$ gives the optimal regularized value from step $h+1$ onward. At state $s = s_h$, if the policy chooses a distribution $\pi_h(\cdot \mid s)$, then the one-step regularized objective is

$$\sum_{a \in \mathcal{A}(s)} \pi_h(a \mid s) Q_h^\star(s, a) - \beta \, \mathrm{KL}(\pi_h(\cdot \mid s) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s)),$$

where, because the MDP state update is deterministic,

$$Q_h^\star(s,a) = V_{h+1}^\star(s'), \qquad s' = (a, T-h-1, c).$$

Applying Lemma B.1 with $Q(a) = Q_h^\star(s,a)$ yields

$$V_h^\star(s) = \beta \log \sum_{a \in \mathcal{A}(s)} \pi_{\mathrm{ref}}(a \mid s) \exp(Q_h^\star(s,a)/\beta),$$

and the unique maximizing policy is

$$\pi_h^\star(a \mid s) = \pi_{\mathrm{ref}}(a \mid s) \exp\left(\frac{Q_h^\star(s,a) - V_h^\star(s)}{\beta}\right).$$

This proves the Bellman recursion and the policy form. Taking expectation over the initial noise $z_T \sim p(\cdot)$ gives the optimal value of $\mathcal{J}_\beta(\pi; c)$. □

B.3 Proof of Corollary 3.2

Proof. By Eq. (10),

$$\pi_h^\star(\mathcal{G} \mid s) = \frac{\sum_{a \in \mathcal{G}} \pi_{\mathrm{ref}}(a \mid s) \exp(Q_h^\star(s,a)/\beta)}{\sum_{a \in \mathcal{A}(s)} \pi_{\mathrm{ref}}(a \mid s) \exp(Q_h^\star(s,a)/\beta)}.$$

Let $p_G = \pi_{\mathrm{ref}}(\mathcal{G} \mid s)$. Using the assumptions

$$Q_h^\star(s,a) \geq b + \Delta \quad (a \in \mathcal{G}), \qquad Q_h^\star(s,a) \leq b \quad (a \notin \mathcal{G}),$$

we obtain

$$\sum_{a \in \mathcal{G}} \pi_{\mathrm{ref}}(a \mid s) \exp(Q_h^\star(s,a)/\beta) \geq e^{(b+\Delta)/\beta} p_G,$$

and

$$\sum_{a \notin \mathcal{G}} \pi_{\mathrm{ref}}(a \mid s) \exp(Q_h^\star(s,a)/\beta) \leq e^{b/\beta}(1 - p_G).$$

Therefore

$$\pi_h^\star(\mathcal{G} \mid s) \geq \frac{e^{(b+\Delta)/\beta} p_G}{e^{(b+\Delta)/\beta} p_G + e^{b/\beta}(1-p_G)} = \frac{e^{\Delta/\beta} p_G}{e^{\Delta/\beta} p_G + 1 - p_G}.$$

□

B.4 Proof of Proposition 3.3

Proof. For compactness, omit the representation superscript r and the state s. Let

$$\pi(a) = \prod_{j=1}^{M} \pi_j(a_j \mid a_{<j}), \qquad \pi_{\mathrm{ref}}(a) = \prod_{j=1}^{M} \pi_{\mathrm{ref},j}(a_j \mid a_{<j}).$$

Then

$$\log \frac{\pi(a)}{\pi_{\mathrm{ref}}(a)} = \sum_{j=1}^{M} \log \frac{\pi_j(a_j \mid a_{<j})}{\pi_{\mathrm{ref},j}(a_j \mid a_{<j})}.$$

Taking expectation under $a \sim \pi$,

$$\begin{aligned}
\mathrm{KL}(\pi \,\|\, \pi_{\mathrm{ref}}) &= \mathbb{E}_{a \sim \pi}\left[\sum_{j=1}^{M} \log \frac{\pi_j(a_j \mid a_{<j})}{\pi_{\mathrm{ref},j}(a_j \mid a_{<j})}\right] \\
&= \sum_{j=1}^{M} \mathbb{E}_{a_{<j} \sim \pi}\left[\sum_{a_j} \pi_j(a_j \mid a_{<j}) \log \frac{\pi_j(a_j \mid a_{<j})}{\pi_{\mathrm{ref},j}(a_j \mid a_{<j})}\right] \\
&= \sum_{j=1}^{M} \mathbb{E}_{a_{<j} \sim \pi}\left[\mathrm{KL}(\pi_j(\cdot \mid a_{<j}) \,\|\, \pi_{\mathrm{ref},j}(\cdot \mid a_{<j}))\right].
\end{aligned}$$

Restoring s and r gives Eq. (14). □

B.5 Proof of Corollary 3.4

Proof. By Proposition 3.3,

$$\mathrm{KL}\Big(\pi^{(r)}(\cdot \mid s) \,\|\, \pi^{(r)}_{\mathrm{ref}}(\cdot \mid s)\Big) = \sum_{j=1}^{M_r(s)} D_j(s),$$

where

$$D_j(s) := \mathbb{E}_{a_{<j}\sim\pi^{(r)}}\left[\mathrm{KL}\Big(\pi^{(r)}_j(\cdot \mid s, a_{<j}) \,\|\, \pi^{(r)}_{\mathrm{ref},j}(\cdot \mid s, a_{<j})\Big)\right].$$

If the left-hand side is at most η, then

$$\frac{1}{M_r(s)} \sum_{j=1}^{M_r(s)} D_j(s) \le \frac{\eta}{M_r(s)}.$$

□

B.6 Proof of Corollary 3.5

Proof. By definition,

$$L_{\mathrm{atom}}(n) = n + \binom{n}{2} = \frac{n^2+n}{2},$$

and

$$L_{\mathrm{motif}}(n) = n + \binom{n}{2} + n(n-1) = \frac{3n^2-n}{2}.$$

Therefore

$$\frac{L_{\mathrm{motif}}(n_{\mathrm{motif}}(x))}{L_{\mathrm{atom}}(n_{\mathrm{atom}}(x))} = \frac{3n_{\mathrm{motif}}(x)^2 - n_{\mathrm{motif}}(x)}{n_{\mathrm{atom}}(x)^2 + n_{\mathrm{atom}}(x)}.$$

Let

$$\chi(x) = \frac{n_{\mathrm{atom}}(x)}{n_{\mathrm{motif}}(x)}.$$

Since

$$3n_{\mathrm{motif}}(x)^2 - n_{\mathrm{motif}}(x) \le 3n_{\mathrm{motif}}(x)^2$$

and

$$n_{\mathrm{atom}}(x)^2 + n_{\mathrm{atom}}(x) \ge n_{\mathrm{atom}}(x)^2,$$

we obtain

$$\frac{L_{\mathrm{motif}}(n_{\mathrm{motif}}(x))}{L_{\mathrm{atom}}(n_{\mathrm{atom}}(x))} \le \frac{3n_{\mathrm{motif}}(x)^2}{n_{\mathrm{atom}}(x)^2} = \frac{3}{\chi(x)^2}.$$

□

Table 6: Structural statistics of the unconditional pretraining datasets. Values are reported as mean / maximum.

Dataset	#SMILES	Valid (%)	Real atoms	Hetero atoms	Rings	Mol. weight
Polymer	12,792	100.0	30.7 / 80	6.2 / 40	3.2 / 15	424.5 / 1282.5
Molecule	10,000	100.0	20.9 / 50	5.7 / 27	2.5 / 15	303.8 / 1295.7

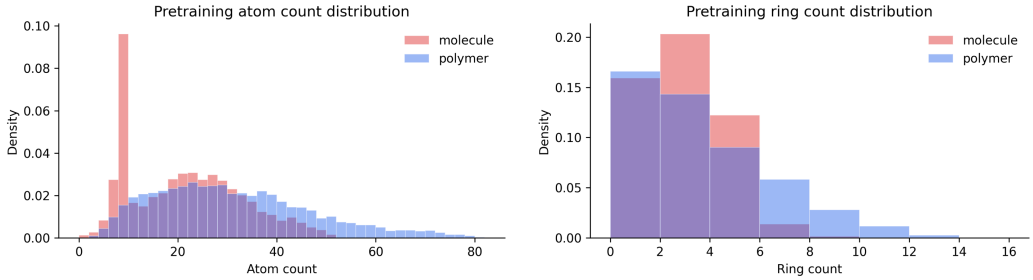

Figure 2: Atom and ring count for pretraining datasets.

C Details on Data

C.1 Datasets

For all random sampling, we set random seed to 42.

C.1.1 Pretraining Datasets

For unconditional pretraining (PT), we curate two unlabeled datasets.

- Polymer: The polymer corpus contains 12,792 unique real-world polymers.
- Small molecule: We curate a 10,000-molecule unlabeled corpus from MoleculeNet [Wu et al., 2018] by sampling valid and unique SMILES from datasets spanning physical chemistry, biophysics, and physiology.

No target labels or task identifiers are used during this stage, so PT remains fully unconditional. This design provides a broad structural prior before adapting the model to heterogeneous downstream molecular tasks.

We report RDKit-derived structural statistics in Table 6. We further visualize the real-atom and ring-count distributions in Figure 2.

C.1.2 Post-training Datasets

For SFT, RL, and evaluation, we construct three target-conditioned task sets, with statistics summarized in Table 7. The polymer DFT set contains electronic-structure properties relevant to functional polymer design, including electron affinity (Eea), bulk band gap (Egb), and chain band gap (Egc) for training, with ionization energy (Ei) and dielectric constant (EPS) held out as unseen targets [Xu et al., 2023]. The polymer gas set includes O_2, N_2, and CO_2 permeability targets for membrane materials [Thornton et al., 2012]. All gas-permeability values are transformed into log space following previous work [Ma and Luo, 2020]. The drug set spans physical chemistry, target bioactivity, and physiological permeability through the numerical FreeSolv dataset, and two class-balanced datasets BACE and BBBP from [Wu et al., 2018].

We visualize task-set-level target distributions in Figure 3. Regression tasks are shown as target-value histograms, while classification tasks are shown by class counts.

For RL post-training, we use the same data sources as SFT. Within each task set, we randomly subsample each task's training split to match the smallest training-set size among the tasks in that set. The validation and test splits are kept fixed between SFT and RL, ensuring that both stages are evaluated on the same held-out conditions.

Table 7: Task definitions and raw statistics for post-training datasets. Target ranges are reported in the original property units before any target transformation.

Benchmark	Task	Description	Type	#SMILES	Target range
Polymer DFT	Eea	DFT-computed electron affinity	Reg.	368	0.394–5.18
	Egb	DFT-computed bulk band gap	Reg.	561	0.391–10.1
	Egc	DFT-computed chain band gap	Reg.	3,380	0.0205–9.86
Polymer gas	O_2Perm	Oxygen permeability of polymer membranes	Reg.	443	$0.005–1.87\times10^4$
	N_2Perm	Nitrogen permeability of polymer membranes	Reg.	443	$0.001–1.66\times10^4$
	CO_2Perm	Carbon dioxide permeability of polymer membranes	Reg.	443	$0.029–4.70\times10^4$
Drug	FreeSolv	Hydration free energy of small molecules in water	Reg.	642	-25.5–3.43
	BACE	Inhibition label for human β-secretase 1	Cls.	1,332	0/1
	BBBP	Blood–brain barrier permeability label	Cls.	872	0/1
Generalization	Ei	DFT-computed ionization energy	Reg.	370	3.56–9.84
	EPS	DFT-computed dielectric constant	Reg.	382	2.61–9.09

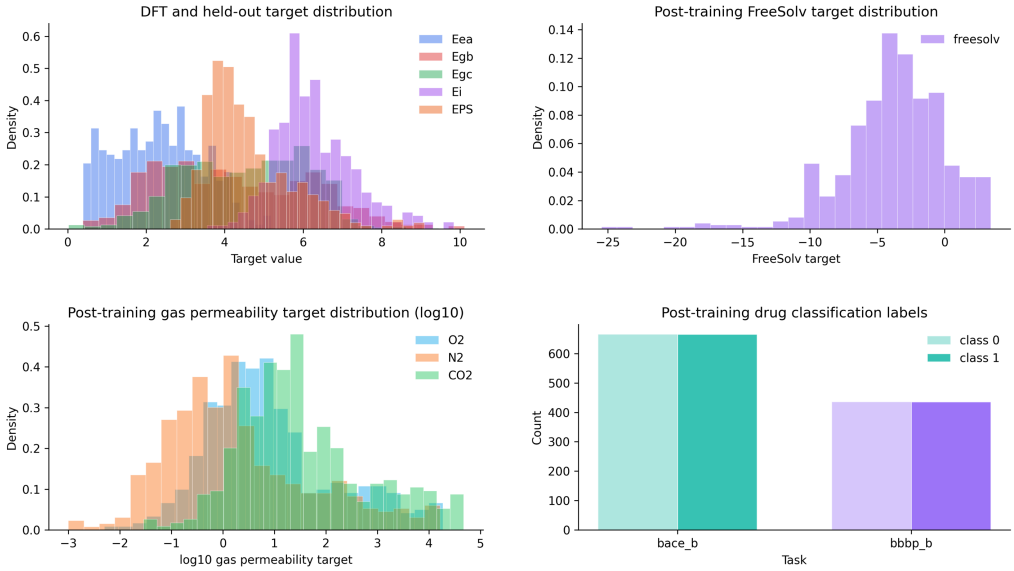

Figure 3: Target distributions for conditional training and evaluation datasets.

C.2 Tokenizer Preparation

We learn domain-specific tokenizers for the polymer and molecule domains using the same default configuration, V1000-R80, where V denotes the total motif vocabulary size and R denotes the ring vocabulary size.

C.2.1 Vocabulary Learning

We construct the motif tokenizer using the data-driven Node Pair Encoding (NPE) algorithm of DemoDiff [Liu et al., 2025]. Each molecule is represented as a graph of atom-disjoint connected units that cover the original atom-level graph. Inter-unit edges store the bond type and the attachment positions on both endpoint units, enabling lossless reconstruction to the original atom-level molecular graph.

Vocabulary learning starts from singleton atom tokens, including atom types observed in the pretraining corpus and the polymerization symbol "*". For constrained NPE, we further enumerate frequent maximal ring units and add the top-R rings to the initial vocabulary, treating them as indivisible units during later merging to avoid ambiguous ring attachments. NPE then repeatedly counts adjacent unit

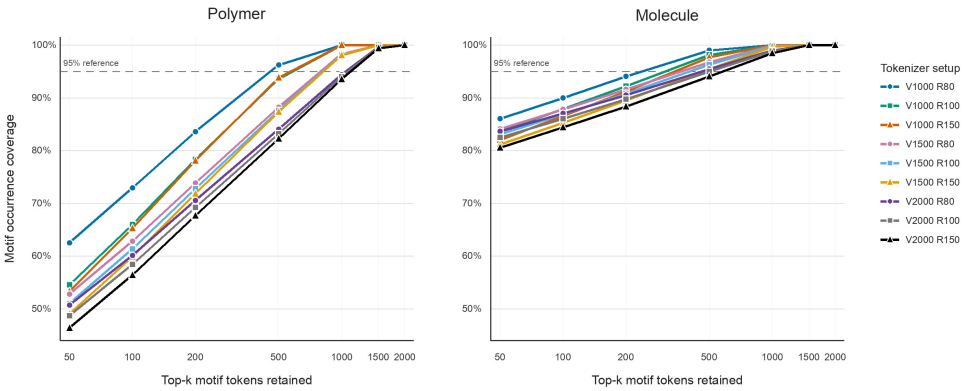

Figure 4: Motif-occurrence coverage under different tokenizer configurations. Coverage is the fraction of token occurrences captured by the top-k most frequent motifs after tokenization. V denotes the total motif vocabulary size, and R denotes the ring vocabulary size. The dashed line marks the 95% coverage reference.

Table 8: Average graph-size compression under representative tokenizer configurations. Compression is the ratio of average atom nodes to average motif nodes, and reduction is the corresponding percentage decrease in node count.

Dataset	Tokenizer	Avg. atom nodes	Avg. motif nodes	Compression	Reduction
Polymer	V1000-R80	32.71	5.85	5.59×	82.1%
	V1000-R100	32.71	5.14	6.37×	84.3%
	V1500-R80	32.71	4.90	6.68×	85.0%
Molecule	V1000-R80	20.87	9.22	2.26×	55.8%
	V1000-R100	20.87	8.39	2.49×	59.8%
	V1500-R80	20.87	8.50	2.45×	59.3%

pairs in the tokenized training graphs, merges the most frequent pair into a new vocabulary unit, and retokenizes the graphs until the target vocabulary size is reached.

C.2.2 Vocabulary Size Selection

We choose the tokenizer configuration based on two criteria: motif-occurrence coverage and graph-size compression.

Motif-occurrence Coverage. Motif-occurrence coverage measures the fraction of token occurrences explained by the top-k most frequent motifs after tokenization. As shown in Figure 4, V1000-R80 provides the strongest coverage efficiency across polymer and molecule pretraining datasets, especially under small and medium retained-token budgets. Larger vocabularies distribute occurrences over more rare motifs, reducing top-k coverage at the same retained vocabulary size.

Graph-size Compression. We further evaluate graph-size compression in Table 8. V1000-R80 reduces polymer graphs from 32.71 atom nodes, including dummy wildcard attachment atoms, to 5.85 motif nodes on average, and molecule graphs from 20.87 atom nodes to 9.22 motif nodes. This corresponds to an 82.1% node reduction for polymers and a 55.8% reduction for molecules. Although larger vocabularies or ring vocabularies can further reduce graph size, V1000-R80 offers a better coverage-compression trade-off. The weaker compression on small molecules is consistent with the more modest gains observed on the drug-related molecular task set in Table 3.

We therefore use V1000-R80 as the default tokenizer configuration. It provides the best occurrence-coverage efficiency while still achieving substantial graph compression, offering a practical balance between compactness, motif expressiveness, and stable training.

D Details on Experiments

Hardware Each training or evaluation run can be executed on a single NVIDIA A100 GPU, no distributed training is required.

D.1 Training Details

All CoMole variants use the same approximately 0.15B-parameter backbone, with transformer depth 12, hidden size 1024, 16 attention heads, and MLP ratio 4. We use $T = 500$ diffusion steps. We train one CoMole instance for each task set described in Section C.1.2. Each instance follows the same three-stage pipeline: PT, SFT, and PPO-based RL.

We use a consistent set of training hyperparameters across task sets:

- **PT** is run for 400 epochs with batch size 64, learning rate 2×10^{-5}, gradient clipping 0.1, and weight decay 10^{-12}.
- **SFT** is run for up to 3,000 epochs with batch size 64 and learning rate 2×10^{-5}. Checkpoints are selected by validation controllability averaged over tasks.
- **RL** is run for up to 400 epochs with batch size 32, learning rate 1×10^{-5}, PPO clip range 0.2, value loss coefficient 0.5, entropy coefficient 0.001, KL coefficient 0.01, one rollout collection pass, and two PPO update passes. For computational efficiency, we apply partial-suffix fine-tuning to the final 30 reverse denoising steps, following [Ren et al., 2024]. For regression tasks, the property reward uses a Gaussian target-alignment term with $\sigma = 0.5$ after target normalization. For classification tasks, the property reward is the evaluator-predicted probability assigned to the target class. The terminal reward combines the property term with the validity term using $w_{\text{val}} = 0.1$.

For baselines, we follow the official hyperparameter settings whenever available. When official settings are unavailable or incomplete, we use Optuna [Akiba et al., 2019] to tune hyperparameters on the validation set and report test performance using the selected configuration. All methods are evaluated on the same data splits, target conditions, and property evaluators.

D.2 Computational Cost and Execution Time

We report the wall-clock runtime and peak memory usage of the main training stages in Table 9. All runs are conducted on a single NVIDIA A100 GPU. Peak CPU memory refers to host-side process memory, while peak GPU memory refers to allocated device memory. The polymer pretraining run is shared by the two polymer benchmarks, namely Polymer gas and Polymer DFT, whereas the molecule pretraining run is used for the Drug benchmark.

The reported runtime and memory usage characterize the practical computational cost of training CoMole under our experimental setting. These measurements may vary with hardware configuration, software environment, I/O overhead, and cluster scheduling conditions.

Table 9: Runtime and peak memory usage of the main training stages on a single NVIDIA A100 GPU.

Stage	Data Type	Peak CPU Memory	Peak GPU Memory	Execution time
Pretraining	Polymer	∼5.8 GB	∼12.9 GB	3h 17m 20s
Pretraining	Molecule	∼5.8 GB	∼10.3 GB	1h 45m 33s
SFT	Polymer gas	∼8.0 GB	∼19.4 GB	6h 55m 45s
SFT	Polymer DFT	∼6.7 GB	∼10.0 GB	5h 35m 34s
SFT	Drug	∼6.1 GB	∼10.3 GB	4h 25m 1s
RL post-training	Polymer gas	∼6.0 GB	∼7.4 GB	2d 3h 49m
RL post-training	Polymer DFT	∼6.0 GB	∼7.5 GB	1d 18h 41m
RL post-training	Drug	∼6.3 GB	∼7.4 GB	2d 7h 6m

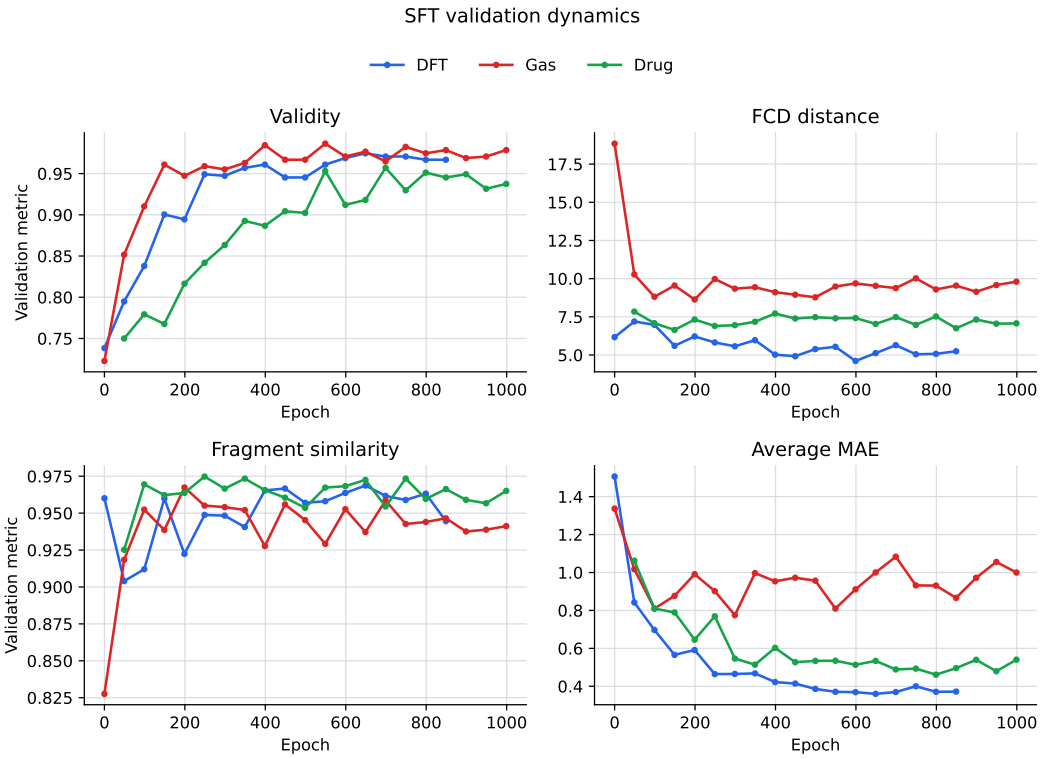

Figure 5: SFT validation dynamics across the polymer DFT, polymer gas-permeability, and drug benchmarks. Validity denotes raw validity without rule-based checking. SFT steadily improves controllability while maintaining high distribution fidelity.

D.3 Training Dynamics

We visualize validation dynamics during SFT and RL post-training in Figures 5 and 6. For the drug benchmark, Avg. MAE is computed as the macro-average over FreeSolv, BACE, and BBBP, giving each task equal weight. For categorical tasks such as BACE and BBBP, MAE is defined as the average absolute difference between the predicted positive-class probability and the binary target label.

During SFT, all three benchmarks exhibit stable training behavior. Validity increases rapidly and remains high throughout training, while fragment similarity stays close to the reference distribution. FCD decreases sharply in the early stage for the gas-permeability benchmark and remains relatively stable for the DFT and drug benchmarks. Avg. MAE decreases substantially for the DFT and drug benchmarks and improves overall for the gas-permeability benchmark, despite moderate fluctuations. These results indicate that SFT improves controllability while preserving the structural prior learned during pretraining.

RL post-training shows benchmark-dependent behavior. On the two polymer benchmarks, RL further improves Avg. MAE while largely maintaining distribution fidelity. The gas-permeability benchmark preserves high validity and stable fragment similarity, whereas the DFT benchmark improves controllability with moderate fluctuations in validity and distribution metrics.

In contrast, the drug benchmark exhibits a stronger distribution shift during RL. Although Avg. MAE decreases and validity remains high, FCD increases substantially and fragment similarity drops. This trend is consistent with the heterogeneous nature of the drug benchmark, where FreeSolv, BACE, and BBBP span physical chemistry, target-specific bioactivity, and physiological permeability, respectively [Wu et al., 2018]. Compared with a benchmark defined by a single drug-property family, this setting introduces more diverse target-alignment pressures, so RL improves controllability while shifting generation away from the aggregate reference distribution.

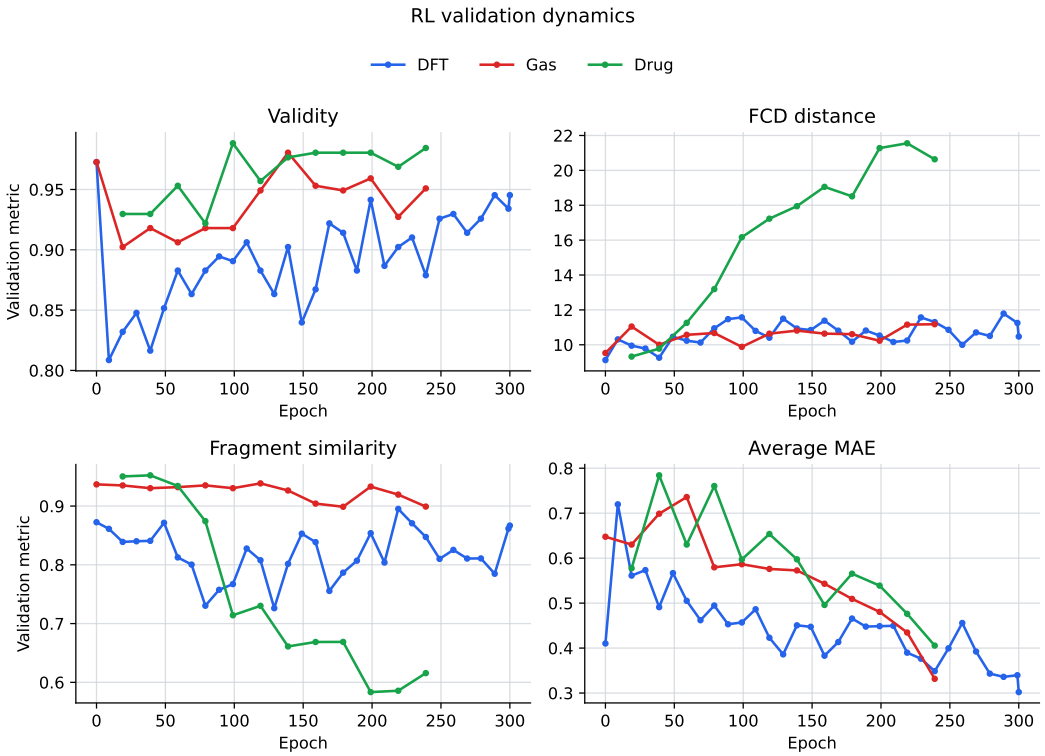

Figure 6: RL validation dynamics across the polymer DFT, polymer gas-permeability, and drug benchmarks. Validity denotes raw validity without rule-based checking. RL further improves target controllability. The polymer benchmarks retain relatively stable distribution metrics, whereas the drug benchmark exhibits a stronger shift toward target-aligned regions.

Table 10: Effect of the trainable reverse horizon on gas-property control. PPO results are reported at their respective evaluation checkpoints.

Model	Validity ↑	Avg. MAE ↓
$\text{CoMole}_{\text{w/o RL}}$	0.984	0.701
CoMole ($K = 30$)	0.950	**0.432**
CoMole ($K = 50$)	**0.989**	0.488

D.4 Effect of the Trainable Reverse Horizon

We investigate how the number of PPO-optimized reverse steps K affects controllability and computational cost on the gas-property tasks. With $T = 500$ reverse transitions, we consider $K \in \{30, 50, 100, 500\}$, where $K = T$ corresponds to full-horizon PPO. All variants use the same PPO configuration.

Control performance. Table 10 compares the SFT baseline with the $K = 30$ and $K = 50$ PPO variants. Both substantially improve property control over SFT. The $K = 30$ variant achieves the lowest average MAE (0.432), whereas $K = 50$ attains higher validity (0.989), illustrating a trade-off between property accuracy and structural validity.

Computational cost and optimization behavior. As shown in Table 11, increasing K substantially increases training cost. On a single NVIDIA A100, the per-epoch runtime grows by $5.51\times$ from $K = 30$ to $K = 500$. At the same number of training epochs, this corresponds to an increase from approximately 52 hours for our main configuration to nearly 12 days for full-horizon PPO.

To compare optimization behavior across horizons, we evaluate all four variants after 99 training epochs. Table 12 reports validity, average MAE, and the Pearson correlation between training epoch

Table 11: Measured PPO training cost on a single NVIDIA A100 GPU. Relative cost is normalized to $K = 30$.

K	Minutes/epoch	Relative cost
30	8.40	1.00×
50	9.81	1.17×
100	16.08	1.91×
500	46.30	5.51×

Table 12: Optimization diagnostics after 99 epochs. Reward–epoch correlation measures the Pearson correlation between epoch index and mean rollout reward over the first 99 epochs.

K	Reward–epoch corr.	Validity ↑	Avg. MAE ↓
30	$+0.378$	0.891	0.507
50	$+0.409$	0.902	0.541
100	$+0.181$	0.734	0.640
500	-0.413	0.715	0.707

and mean rollout reward. The correlation serves only as a diagnostic of the reward trend, rather than a convergence criterion.

At epoch 99, $K = 30$ and $K = 50$ achieve lower MAE and higher validity than the longer-horizon variants. The less favorable reward trends for $K = 100$ and $K = 500$ also suggest greater optimization difficulty under the evaluated PPO configuration. However, these comparisons fix the number of training epochs rather than total compute, and the longer-horizon variants have not necessarily converged. For example, $K = 100$ subsequently improves to 0.570 MAE and 0.820 validity by epoch 159. Thus, these results characterize optimization under the evaluated configuration rather than establishing an intrinsic limitation of longer-horizon PPO.

Reverse-trajectory diagnostics. To examine why suffix-constrained optimization can be effective, we analyze 500 complete reverse sampling trajectories generated by CoMole. Diffusion time decreases from $t = T = 500$ (initial noise) to $t = 0$ (final graph), such that $t = K$ marks the beginning of the final K reverse transitions.

At each intermediate state, we measure the fraction of connected motif graphs, the agreement of node, edge, and positional attributes ($X/E/P$) with the corresponding final state, and the Pearson correlation between post-hoc intermediate-state scores and final-state rewards across paired trajectories. These intermediate scores are used only for diagnostics and not for PPO training.

As shown in Table 13, substantial structural changes occur during the final reverse transitions. At $t = 100$, only 2% of intermediate motif graphs are connected, and intermediate-state scores exhibit weak correlation with final rewards (0.10). At $t = 30$, connectivity reaches 39%, while $X/E/P$ agreement with the final state remains limited, indicating that the terminal structure is still substantially undetermined.

Over the subsequent 20 transitions, connectivity increases to 64%, $X/E/P$ agreement rises to 0.82/0.86/0.74, and reward correlation reaches 0.70. These observations indicate that the final 30 transitions retain substantial capacity to modify the generated structure and terminal outcome, consistent with the effectiveness of suffix-constrained PPO.

Overall, the horizon ablations and trajectory diagnostics support $K = 30$ as an empirically effective operating point balancing property control and computational cost. They do not imply that earlier reverse transitions are unimportant or that longer-horizon optimization is inherently less effective; alternative optimization configurations may improve longer-horizon training.

D.5 More Results

In the following results, we report validity measured without rule-based checking, denoted as $\text{Validity}_{\text{w/o check}}$.

Table 13: Structural evolution across 500 paired reverse trajectories. Connected denotes the fraction of connected intermediate motif graphs. Reward correlation is computed between post-hoc intermediate-state scores and corresponding final-state rewards.

t	Connected	$X/E/P$ agreement	Reward corr.
100	0.02	0.23/0.13/0.35	0.10
30	0.39	0.32/0.48/0.28	0.22
10	0.64	0.82/0.86/0.74	0.70

Table 14: Joint training analysis on polymer tasks. *Separated* denotes the main setting where gas permeability and DFT properties are trained as two benchmark-specific models. *Joint* denotes a single model trained jointly on all six polymer targets. For *Separated*, validity and distribution metrics are arithmetic means of the two benchmark-level results, while Avg. MAE is the simple average over the six target-level MAEs. Best results are in **bold**.

Model	Validity$_{\text{w/o check}}$ ↑	Diversity ↑	Similarity ↑	Distance ↓	O_2Perm ↓	N_2Perm ↓	CO_2Perm ↓	Eea ↓	Egb ↓	Egc ↓	Avg. MAE ↓
CoMole$_{\text{w/o RL}}$-Joint	0.9609	0.8911	**0.9743**	**3.8751**	0.6906	0.8394	0.7580	0.3525	0.4787	0.3522	0.5786
CoMole$_{\text{w/o RL}}$-Separated	**0.9695**	0.8712	0.9658	7.3101	0.6442	0.7888	0.6709	0.2874	0.4713	0.3772	0.5400
CoMole-Joint	0.9277	**0.8923**	0.9524	4.6167	0.8245	1.0396	0.6889	0.2668	0.3522	0.3776	0.5916
CoMole-Separated	0.9457	0.8773	0.8966	9.6612	**0.3975**	**0.4720**	**0.4250**	**0.2392**	**0.4129**	**0.2813**	**0.3713**

D.5.1 Joint Training on Polymer Properties

We report gas permeability and DFT properties as separate polymer task sets in the main experiments because they correspond to distinct property families and empirical distributions. To examine whether they can be merged into a single heterogeneous polymer setting, we compare two training protocols:

- *Separated*: the main experimental setting, where gas-permeability and DFT targets are trained and evaluated using two benchmark-specific models.
- *Joint*: a single model trained jointly on all six polymer targets, including O_2Perm, N_2Perm, CO_2Perm, Eea, Egb, and Egc.

For the separated setting, target MAEs are taken from the corresponding gas and DFT benchmark runs. Validity$_{\text{w/o check}}$ and distribution metrics are reported as the arithmetic mean of the two benchmark-level results, while Avg. MAE is the simple average over all six target-level MAEs.

As shown in Table 14, joint training does not cause a general collapse in validity or distribution learning. However, merging the two polymer property families weakens target control. At the SFT stage, the six-target Avg. MAE increases from 0.5400 in the separated setting to 0.5786 under joint training. After RL, the degradation becomes more pronounced: Avg. MAE increases from 0.3713 to 0.5916. This drop is mainly driven by gas-permeability targets, whose average MAE increases from 0.4315 to 0.8510, whereas DFT targets are less affected, changing from 0.3111 to 0.3322. This suggests that gas-permeability targets introduce stronger distributional heterogeneity into the six-target setting.

These results indicate that merging heterogeneous polymer property families can introduce negative transfer in target controllability, even when validity and distribution-level metrics remain competitive. We therefore keep gas-permeability and DFT properties as separate polymer task sets in the main evaluation.

D.5.2 Robustness to the Data Split

We evaluate the sensitivity of PPO fine-tuning to the train/validation/test split on the DFT-property tasks. In addition to the original 8:1:1 split, we construct an independently resampled 6:2:2 split. For each split, SFT is initialized from the same pretrained checkpoint, and PPO is subsequently initialized from the corresponding SFT checkpoint.

PPO consistently reduces the MAE of all three properties under both splits, decreasing the average MAE from 0.379 to 0.311 under 8:1:1 and from 0.393 to 0.246 under 6:2:2. These results indicate that the benefit of PPO is not restricted to the original data partition.

Table 15: Sensitivity to the train/validation/test split on the DFT-property tasks. Each model is evaluated on the test set of its corresponding split.

Split	Model	Validity$_{\mathrm{w/o\ check}}\uparrow$	$E_{\mathrm{ea}}\downarrow$	$E_{\mathrm{gb}}\downarrow$	$E_{\mathrm{gc}}\downarrow$	Avg. MAE$\downarrow$
8:1:1	CoMole$_{\text{w/o RL}}$	0.969	0.287	0.471	0.377	0.379
	CoMole	0.943	0.239	0.413	0.281	0.311
6:2:2	CoMole$_{\text{w/o RL}}$	0.931	0.309	0.503	0.367	0.393
	CoMole	0.929	0.222	0.289	0.227	0.246

Table 16: Alternative RL algorithm on the gas-permeability benchmark. CoMole$_{\text{w/ GDPO}}$ replaces PPO in CoMole with a GDPO-style policy update, while keeping all other settings unchanged. Best results are in **bold**.

Model	Validity$_{\mathrm{w/o\ check}}\uparrow$	Diversity$\uparrow$	Similarity$\uparrow$	Distance$\downarrow$	O_2Perm$\downarrow$	N_2Perm$\downarrow$	CO_2Perm$\downarrow$	Avg. MAE$\downarrow$
GDPO	0.2965	**0.8766**	0.0973	28.4533	0.7934	0.8650	0.7527	0.8037
CoMole$_{\text{w/o RL}}$	**0.9844**	0.8564	**0.9630**	**9.4849**	0.6442	0.7888	0.6709	0.7013
CoMole$_{\text{w/ GDPO}}$	0.9286	0.7698	0.8423	13.9192	0.5842	0.6751	0.5044	0.5879
CoMole	0.9500	0.8644	0.9405	9.9854	**0.3975**	**0.4720**	**0.4250**	**0.4315**

D.5.3 Other RL Post-training Objective

Our main experiments instantiate RL post-training with PPO. To test whether our pipeline is specific to PPO, we additionally evaluate a GDPO [Liu et al., 2024b] policy optimization objective on the gas-permeability polymer benchmark. We denote this variant as CoMole$_{\text{w/ GDPO}}$, which keeps the motif tokenizer, SFT checkpoint, terminal reward, and evaluation protocol unchanged.

As shown in Table 16, the GDPO update is compatible with our RL pipeline and improves target control over the SFT checkpoint. Compared with CoMole$_{\text{w/o RL}}$, CoMole$_{\text{w/ GDPO}}$ reduces Avg. MAE from 0.7013 to 0.5879 while maintaining high raw validity. It also substantially outperforms the original atom-level GDPO baseline in validity and distribution similarity, highlighting the benefit of motif-aware actions.

However, PPO remains stronger in this setting. These results suggest that our pipeline is not tied to a specific RL algorithm, while PPO provides an effective instantiation for the main experiments.

D.5.4 Further Analysis of CoMole$_{\text{w/o RL}}$

In this section, we analyze whether the SFT reference policy π_{ref} mentioned in Section 3.2.1 already assigns trajectory-level probability to low-error generations before RL. Using the CoMole$_{\text{w/o RL}}$ trained on the polymer DFT benchmark, for each test condition c, we draw K independent samples and compute the absolute target error under that condition. We report two metrics:

- $\mathrm{mean\text{-}MAE@}K$ averages the errors over all sampled molecules and conditions, reflecting standard sampling quality.
- $\mathrm{best\text{-}MAE@}K$ first selects, for each condition, the lowest-error sample among the K draws, and then averages this best-sample error over conditions.

As shown in Table 17, $\mathrm{best\text{-}MAE@}K$ decreases from 0.3676 at $K = 1$ to 0.0261 at $K = 32$, while $\mathrm{mean\text{-}MAE@}K$ remains relatively stable. This suggests that the SFT reference policy already covers low-error candidates, but standard sampling does not select them reliably. RL can therefore improve controllability by increasing the probability of these target-aligned generations.

D.5.5 Oracle Robustness

To validate whether our conclusions depend on the property oracle used for reward and evaluation, we perform an oracle robustness analysis. During RL post-training, the property component of the terminal reward is computed by a learned oracle, so a model could in principle overfit to the biases of a specific evaluator.

We therefore freeze the generated candidates from the main experiments and re-score them using alternative predictors trained on the same training splits, including Random Forest [Breiman, 2001], Support Vector Machine, and GRIN [Zhu et al., 2026]. For each oracle, we compute the corresponding

Table 17: Repeated sampling from $\text{CoMole}_{\text{w/o RL}}$ on the polymer DFT benchmark, with exactly K draws per test condition.

K	$\text{Validity}_{\text{w/o check}} \uparrow$	mean-MAE@K $\downarrow$	best-MAE@K $\downarrow$
1	0.9815	0.3676	0.3676
4	0.9647	0.3743	0.0808
8	0.9641	0.3739	0.0476
16	0.9676	0.3725	0.0322
32	0.9671	0.3766	**0.0261**

Table 18: Oracle robustness for generation evaluation. Models are sorted by their average target-level rank across the nine evaluated properties, and only ranks 1–9 are shown here. The ordering of baselines varies across evaluators, but CoMole consistently ranks first.

Avg. Rank	Random Forest	Support Vector Machine	GRIN
1	**CoMole**	**CoMole**	**CoMole**
2	Graph DiT	FREED	GDPO
3	GDPO	GDPO	Graph DiT
4	DiGress v2	Graph DiT	DeFoG
5	FREED	LSTM-HC	DiGress v2
6	DiGress	JTVAE-BO	FREED
7	LSTM-HC	DiGress	DiGress
8	MARS	DiGress v2	LSTM-HC
9	DeFoG	DeFoG	JTVAE-BO

controllability metric for every target, rank models by target-level performance, and report the average rank across the nine evaluated properties in Table 18.

Although the ordering of some baselines varies across oracles, CoMole consistently ranks first under all three evaluators. This suggests that our main conclusion is robust to the choice of learned oracle and is not driven by a specific evaluator.

In the main experiments, we use the best-performing validation oracle for each domain: Random Forest for the drug-related molecular task set and GRIN for the polymer task sets. Since GRIN can be trained as a general graph predictor, its main advantage comes from repetition-aware augmentation for polymer repeat-unit representations, which provides limited benefit for small molecules.

D.5.6 Novelty and Uniqueness

We additionally report novelty and uniqueness as complementary generation-quality metrics. These metrics are computed over generated samples aggregated across the evaluated target conditions. Novelty measures the fraction of valid generated molecules absent from the training set, while uniqueness measures the fraction of non-duplicated valid generations after canonicalization.

For conditional generation, however, these metrics are mainly sanity checks against memorization and duplication. They do not measure whether generated molecules satisfy the desired target properties or lie in a practically useful design region. Prior work has shown that simple perturbations such as AddCarbon can achieve near-perfect scores without producing useful candidates [Renz et al., 2019, Tripp and Hernández-Lobato, 2023]. Therefore, we treat novelty and uniqueness as complementary rather than primary indicators of inverse-design quality in our main experiments.

As shown in Table 19, $\text{CoMole}_{\text{w/o RL}}$ and CoMole retain novelty and uniqueness, suggesting that CoMole improves target controllability without collapsing to memorized or highly duplicated generations.

D.6 Visualization

Given the quantitative results in Tables 1 and 2, we further visualize whether low-sample target-conditioned generation produces chemically plausible structures. For selected O_2Perm, CO_2Perm, Eea, and Egb test conditions, we sample 10 candidates from CoMole and show the rank-1 generated structure. The rank is computed by averaging the candidate's rank in target-value error and its rank

Table 19: Complementary novelty and uniqueness metrics across evaluated conditions. Novelty and uniqueness are computed over valid generated molecules after canonicalization.

Method	**Novelty**	**Uniqueness**
Graph GA	0.9950	1.0000
MARS	1.0000	0.7500
LSTM-HC	0.9421	0.9535
JTVAE-BO	1.0000	0.6847
DiGress	0.9908	0.7811
DiGress v2	0.9801	0.7720
GDSS	0.9136	0.2932
MOOD	0.9867	0.9761
DeFoG	0.9548	0.8819
Graph DiT	0.9139	0.8739
FREED	1.0000	1.0000
GDPO	1.0000	0.9577
$\text{CoMole}_{\text{w/o RL}}$	0.9026	0.8817
CoMole	0.9390	0.8476

in structural similarity to the corresponding held-out reference structure. The reference structure is used only for qualitative comparison and is not treated as the unique solution to the inverse-design problem.

As shown in Figures 7 and 8, the selected generated structures often retain chemically meaningful motifs or scaffold patterns related to the held-out references, such as thiophene-containing units for Eea and Egb, aliphatic ester repeat units for high-Egb, aromatic imide motifs for CO_2Perm, and aromatic sulfone or bulky aromatic ester scaffolds for O_2Perm. These examples qualitatively complement the quantitative results by showing that target-conditioned samples can remain close to plausible structure chemistry under limited sampling.

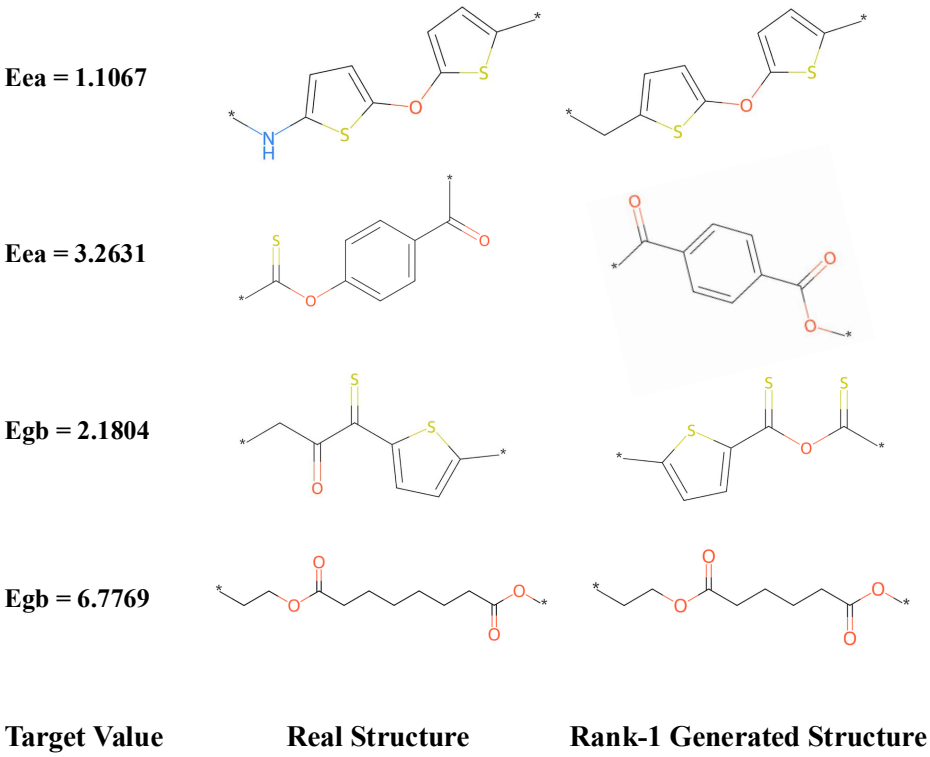

Figure 7: Rank-1 generated structures selected from 10 generated samples separately for Eea and Egb conditions.

E Limitations and Future Directions

Limitations First, our evaluation covers representative but not exhaustive chemical design settings. For polymers, we focus on gas-permeability and DFT-derived electronic properties, while other practically important targets, such as thermal properties (T_g and T_c) and mechanical properties, remain unexplored. For small molecules, our benchmark includes regression and binary classification tasks, but does not yet cover broader condition types such as multi-class labels. Expanding the task coverage would provide a more comprehensive assessment of heterogeneous controllable generation.

Second, we adopt a largely shared training configuration across benchmarks. For example, CoMole uses the same configuration for the polymer and molecule tokenizers, as described in Section C.2. Although the V1000-R80 tokenizer provides strong compression for polymers, yielding an 82.1% node reduction, the compression is less pronounced for small molecules. In particular, the current tokenizer reduces the RL decision space and credit-assignment burden more substantially for polymers than for small molecules.

Future Directions Future work could develop stronger and more adaptive tokenizers to better balance compression, expressiveness, and controllability. Promising directions include larger molecular vocabularies and unified tokenizers for polymers and small molecules that are less tied to a specific training corpus.

Another important direction is simultaneous multi-property control, where generated molecules or polymers must satisfy multiple numerical or categorical conditions at once. This setting better reflects practical scientific design, where candidates are typically selected under coupled requirements involving property targets, chemical validity, synthesizability, and distributional plausibility.

More broadly, the long-term goal of controllable molecular generation is to advance scientific discovery by bridging the gap between hypothetical structure generation and practical inverse design. Such models can help researchers prioritize more promising molecular candidates under heterogeneous property constraints before committing to expensive simulations, experiments, or screening cycles.

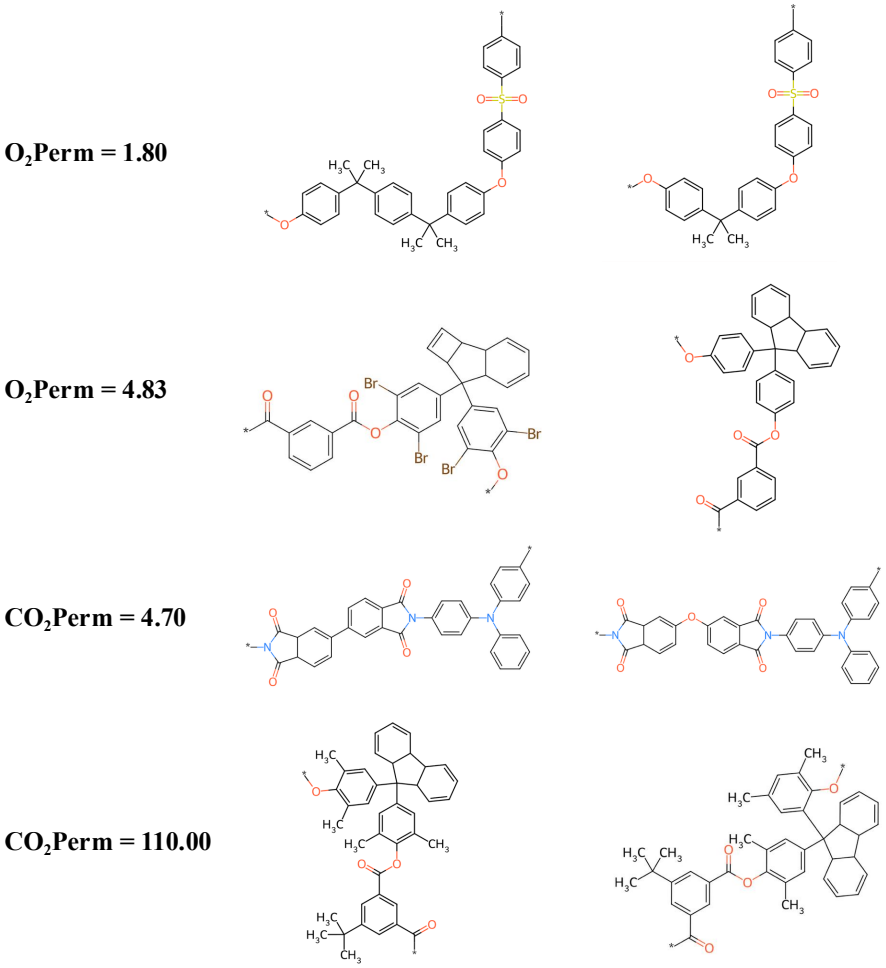

Figure 8: Rank-1 generated structures selected from 10 generated samples separately for O_2Perm and CO_2Perm conditions.

By combining transferable structural priors with task-conditioned adaptation and reward-guided post-training, CoMole provides a step toward foundation-model-based design systems for molecules and polymers.

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 abstract and introduction accurately reflect the paper's scope and contributions. The scope is clearly framed around the controllability challenge in molecular inverse design. The central contribution is CoMole, which is, to our knowledge, the first family of controllable molecular generative foundation models for heterogeneous inverse design.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The paper discusses the main limitations of the proposed approach in Appendix E.

   Guidelines:

   - The answer [N/A] 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: The paper states the assumptions for each theoretical result in Section 3.2. More statements and proofs are provided in the Appendix B.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The paper discloses the information needed to reproduce the main experimental results in Sections A, C.1 and D.

Guidelines:

- The answer [N/A] 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: Code, sample data and detailed README are provided in the supplementary materials. Due to supplementary file-size limits, full checkpoints will be released through the project repository.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: All the training and test details are included in Appendix D.1 and Appendix D.5.5.

   Guidelines:

   - The answer [N/A] 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: The main experiments report aggregate generation metrics computed from 10,000 generated structures per setting. This large-sample protocol reduces Monte Carlo noise. Sample-level uncertainty estimates would only quantify finite-sampling variability under a fixed checkpoint, split, and oracle, and would not capture variability from retraining, data resplitting, hyperparameter selection, or oracle training. The main conclusions are instead supported by consistent gaps across three benchmarks, SFT-vs-RL comparisons, motif-vs-atom ablations, and oracle-robustness analysis.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We report the compute resources required to reproduce the experiments, including the GPU environment, runtimes and memory usage for all stages and benchmarks in Appendix D.2.

   Guidelines:

   - The answer [N/A] 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 conforms to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: The paper discusses societal impacts of controllable molecular generation for accelerating materials and drug discovery in Appendix E.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This work does not have a high risk for misuse. Experiments are conducted on standard chemical benchmarks with computational evaluators. We do not target hazardous compounds or provide synthesis protocols, wet-lab instructions, or autonomous deployment pipelines.

Guidelines:

- The answer [N/A] 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: We use public datasets only for research benchmarking. MoleculeNet is used under the MIT license, and polymer DFT datasets are credited to Polymer Genome and used under CC BY-NC-ND 4.0.

Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: We release the CoMole implementation and experiment configurations, accompanied by documentation covering preprocessing, training, evaluation, and model settings.

    Guidelines:

    - The answer [N/A] 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: [N/A]

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

    Guidelines:

    - The answer [N/A] 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: [N/A]

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

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method, experiments, datasets, and evaluations do not involve LLMs as an important, original, or non-standard component. Any LLM assistance, if used, was limited to writing, editing, or formatting and did not affect the methodology, scientific claims, or experimental results.

Guidelines:

- The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.