C-MORAL: Controllable Multi-Objective Molecular Optimization with Reinforcement Alignment for LLMs

Anonymous ACL submission

Abstract

Large language models (LLMs) show promise for molecular optimization, but aligning them with selective and competing drug-design constraints remains challenging. We propose C-MORAL, a reinforcement learning post-training framework for controllable multi-objective molecular optimization. C-MORAL combines group-based relative optimization, property score alignment for heterogeneous objectives, and bottleneck-sensitive non-linear reward aggregation to improve stability across competing molecular properties. Experiments on C-MuMOInstruct and S^2-Bench MolOpt show that C-MORAL achieves the best performance among compared methods on both benchmarks. On C-MuMOInstruct, C-MORAL achieves the best Success Optimized Rate (SOR) of 48.9% on in-domain tasks and 39.5% on out-of-domain tasks while preserving scaffold similarity. On S^2-Bench MolOpt, it also achieves the strongest results across LogP, MR, and QED optimization tasks. These results suggest that C-MORAL is an effective way to align molecular LLMs with continuous and constrained molecular design objectives.

1 Introduction

Drug discovery is a lengthy and costly process, with the hit-to-lead transition and the lead optimization stage being particularly critical (Katsuno et al., 2015; Sadybekov and Katritch, 2023). This process is inherently a multi-objective optimization (MOO) problem (Guan et al., 2019).

Practical molecular optimization in drug discovery is rarely a matter of simply improving all properties (Fialková et al., 2021). In many cases, a lead compound already approaches desirable levels in a subset of attributes; therefore, optimization requires selective improvement of the remaining deficient properties while maintaining those that already meet the target criteria, and simultaneously preserving the core scaffold and synthetic feasibility (Zhou et al., 2024). For example, ControllableGPT (Liu et al., 2025) aims to increase binding affinity while preserving the core scaffold by making minimal, targeted edits that remove liabilities. Such tasks have been formalized as a benchmark setting for controllable multi-property, multi-objective optimization (C-MuMO) (Dey et al., 2025). C-MuMO highlights the need to specify property-wise objectives and thresholds for either improvement or preservation, thereby better reflecting the fine-grained requirements of real medicinal chemistry. In this context, "controllable" means that the model must follow explicit, property-wise directives—improving only the deficient attributes while preserving those already satisfying target criteria—instead of indiscriminately maximizing a single aggregate objective.

Despite its practical importance, controllable molecular optimization remains challenging. The difficulty lies not in improving molecular properties alone, but in doing so under explicit, property-wise requirements: a model must improve deficient attributes while preserving those already meeting target criteria based on source molecules. Existing methods struggle to satisfy such fine-grained control reliably. RL-based approaches typically optimize a scalarized reward over multiple objectives (Wang and Zhu, 2024; Park et al., 2025), which can lead to brittle trade-offs under strict and competing constraints. LLM-based approaches adapted by SFT or prompting can leverage strong chemical priors (Ye et al., 2023), but they often treat chemical constraints as soft instructions rather than hard requirements. As a result, they may achieve apparent score improvements through undesired edits, such as altering the core scaffold or violating preservation constraints (Guo et al., 2023).

To achieve genuine, controllable optimization and overcome the limitations of both traditional RL and SFT paradigms, we introduce C-MORAL,

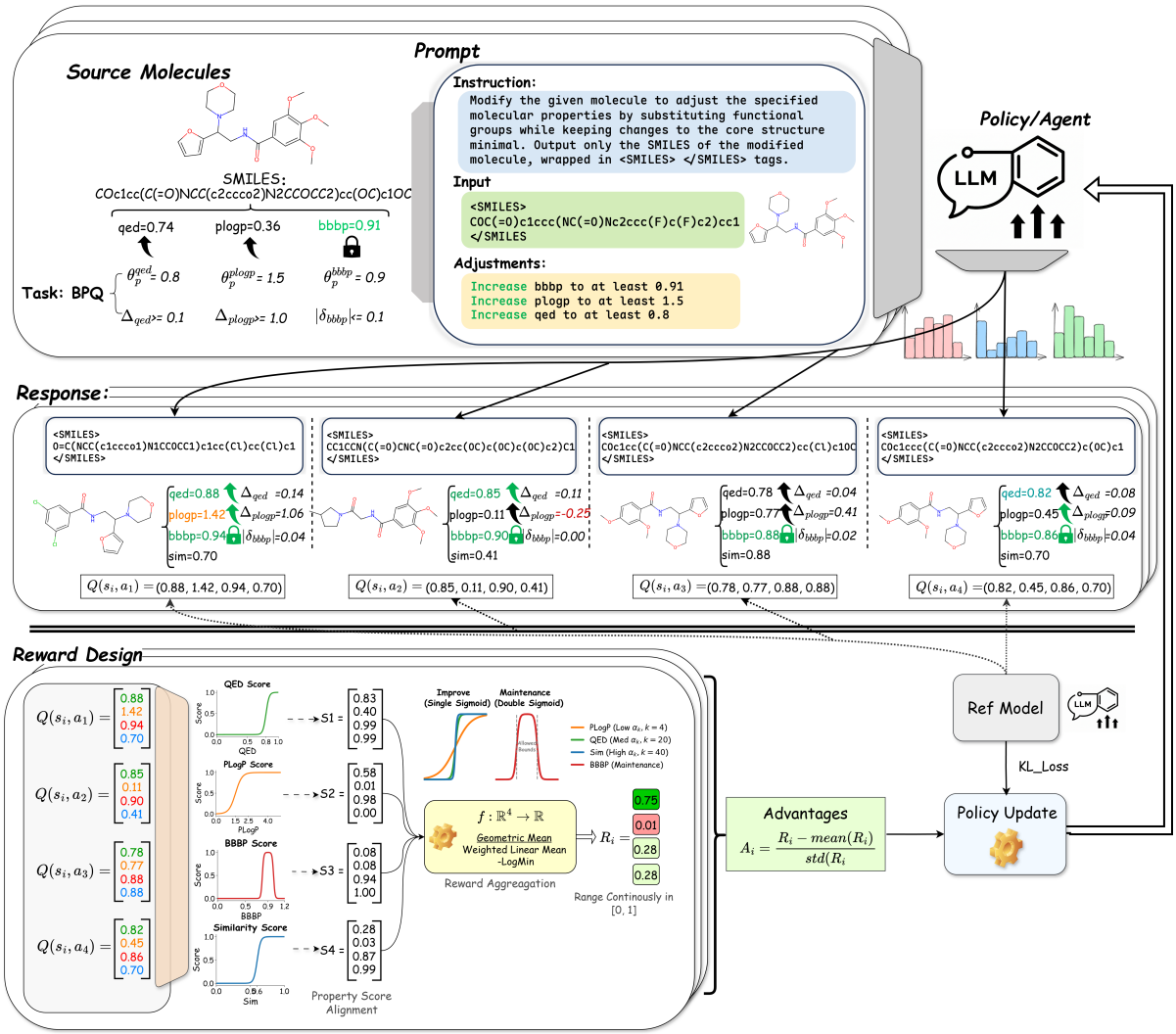


Figure 1: Overview of C-MORAL generation and training pipeline

a constraint-aware reinforcement learning post-training framework designed specifically for molecular LLMs. C-MORAL stands for **C**ontrollable Multi-**O**bjective **M**olecular **O**ptimization with **R**einforcement **A**lignment for **L**LMs. An overview of the framework is shown in Figure 1.

C-MORAL aims to bridge the gap between the robust chemical priors of LLMs and the strict multi-objective demands of lead optimization. Rather than introducing a new policy-gradient estimator, C-MORAL provides a task-specific RL alignment formulation for controllable molecular editing. Its main contribution lies in constraint-aware reward construction and group-relative training under heterogeneous molecular objectives. The key idea is to align molecular LLMs with property-wise scientific constraints through calibrated property scores, bottleneck-sensitive reward aggregation, and group-relative feedback.

Specifically, we adapt two group-relative RL algorithms, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) and Group Reward-Decoupled Normalization Policy Optimization (GDPO) (Liu et al., 2026), to instantiate two complementary variants of C-MORAL. C-MORAL-GRPO evaluates candidate molecules relative to peers generated from the same prompt and uses a sigmoid-aligned geometric mean reward to penalize property bottlenecks. C-MORAL-GDPO further decouples property-wise feedback and aggregates it with a smooth-min objective to reduce the implicit sacrifice of any attribute. Since both variants avoid a separate memory-intensive value network, they improve training efficiency and memory utilization.

Evaluated on the stringent C-MuMOInstruct benchmark (Dey et al., 2025) and S^2-Bench MolOpt, our framework significantly outperforms

strong baselines. In summary, our main contributions are:

- We propose C-MORAL, a constraint-aware RL post-training framework for controllable multi-objective molecular optimization with LLMs.
- We introduce a property-aware reward design that combines sigmoid score alignment with bottleneck-sensitive non-linear aggregation, and instantiate it with group-relative RL through GRPO and GDPO.
- C-MORAL achieves the best performance among compared methods on two molecular optimization benchmarks, C-MuMOInstruct and S^2-Bench MolOpt, with the highest SOR on both in-domain and out-of-domain C-MuMOInstruct tasks and the strongest MolOpt results across LogP, MR, and QED.

2 Related Work

2.1 Group-based Policy Optimization

Compared with PPO, which relies on an additional critic network, GRPO removes the value model by normalizing rewards within a sampled candidate group, thereby reducing computational overhead (Shao et al., 2024). GDPO further extends this idea by decoupling advantage normalization across reward dimensions, making it suitable for multi-reward settings with conflicting objectives (Liu et al., 2026). Our work builds on group-relative policy optimization, but does not introduce a new policy-gradient estimator. Instead, we adapt it to controllable molecular editing through property score alignment and bottleneck-sensitive reward aggregation under heterogeneous molecular constraints.

2.2 Molecular Language Models and Generative Models

Molecular generation has been studied with SMILES-based sequence models, graph generative models, and continuous structure refinement methods. Since SMILES represents molecules as sequences (Weininger, 1988), language modeling has become a natural paradigm for molecular generation. MolGPT treats molecule generation as autoregressive SMILES modeling (Bagal et al., 2021), while MolT5 formulates molecules and natural language as a sequence-to-sequence translation problem (Edwards et al., 2022). Recent LLM-based systems further support molecular understanding and interaction, including LLaMo (Park et al., 2024) and ChatDrug (Liu et al., 2023). These models demonstrate the promise of language models for chemistry, but are not primarily designed for strict controllable multi-objective optimization with property-wise improvement and preservation constraints.

2.3 Molecular Optimization

Molecular optimization aims to improve desired properties while maintaining chemical validity and structural fidelity. Existing approaches include heuristic search, reinforcement learning, and deep generative modeling (Gómez-Bombarelli et al., 2018; Zhou et al., 2019; Walters and Barzilay, 2020; Olivecrona et al., 2017; Wang et al., 2023), with action spaces ranging from token-level SMILES editing to graph construction and continuous structure refinement (Erikawa et al., 2023; Barrett and Westermayr, 2024). However, multi-objective molecular optimization remains challenging because scalarized rewards can lead to brittle trade-offs under competing objectives and hard constraints (Brown et al., 2019; Wang and Zhu, 2024). Recent LLM-based methods such as LICO and MOLLEO explore in-context surrogate modeling and evolutionary search for molecular design (Nguyen and Grover, 2025; Wang et al., 2025), while REPO studies GRPO-style post-training for instruction-based molecular generation (Li et al., 2026). In contrast, our work focuses on controllable multi-objective molecular editing, where deficient properties must be selectively improved while near-optimal properties and scaffold similarity are preserved.

3 Method

3.1 Problem Formulation

We formulate Controllable Multi-objective Molecule Optimization (C-MuMO) as a constrained generation task. Given an initial sub-optimal "hit" molecule M_x and a natural language instruction I, the objective is to generate an optimized "lead" molecule M_y that selectively improves specific molecular properties up to pharmaceutically relevant levels, while maintaining others that already meet the criteria (Dey et al., 2025).

Let $\mathcal{P}$ denote the set of target pharmacological properties. For each property $p \in \mathcal{P}$, we define a pharmaceutically relevant threshold Θ_p and an

improvement/stability margin Δ_p. Based on the initial molecule M_x, the properties are partitioned into two disjoint subsets:

- Sub-optimal properties ($\mathcal{P}_i$): Properties that require targeted enhancement, defined as $\mathcal{P}_i = \{p \in \mathcal{P} \mid p(M_x) \text{ is worse than } \Theta_p\}$.
- Near-optimal properties ($\mathcal{P}_s$): Properties that already meet the criteria and must be preserved, defined as $\mathcal{P}_s = \{p \in \mathcal{P} \mid p(M_x) \text{ is better than or equal to } \Theta_p\}$.

Detailed values of Δ_p and Θ_p for different properties are given in Table A4.

The generation policy $\pi_\Theta(M_y \mid M_x, I)$, parameterized by an LLM, is required to generate M_y that satisfies three non-differentiable constraints simultaneously. First, the structural similarity between M_x and M_y, measured by Tanimoto similarity (Bajusz et al., 2015), must remain above a predefined threshold. Second, for each sub-optimal property $p \in \mathcal{P}_i$, the generated molecule must improve in the desired direction. Specifically, we define an optimization direction $d_p \in \{+1, -1\}$, where $d_p = +1$ indicates that p should be increased and $d_p = -1$ indicates that p should be decreased; successful improvement requires $d_p\big(p(M_y) - p(M_x)\big) \geq \Delta_p$. Third, each near-optimal property $p \in \mathcal{P}_s$ must remain stable, with its absolute deviation bounded by Δ_p, i.e., $|p(M_y) - p(M_x)| \leq \Delta_p$.

3.2 Property Score Sigmoid Alignment

Molecular properties inherently exist on vastly different scales. Directly aggregating raw values creates severe instability. To address this, we introduce a property-specific score shaping mechanism that normalizes all metrics into a unified $[0, 1]$ preference space.

Let $\sigma(x) = 1/(1 + \exp(-x))$ denote the standard sigmoid function. For brevity, let v_p denote the realized property value $v_p(M_y)$ of the generated molecule. For sub-optimal properties, we set the target threshold T_p according to the desired optimization direction. The direction-aware improvement score is defined as

$$s_p^{\text{imp}} = \sigma\left(\alpha_p \cdot d_p \cdot (v_p - T_p)\right),$$

where $d_p = +1$ for properties to be increased and $d_p = -1$ for properties to be decreased. Crucially, we parameterize α_p to be inversely proportional to the property-specific margin Δ_p (i.e., $\alpha_p \propto 5/\Delta_p$). This scaling ensures that achieving the required Δ_p for any metric—regardless of its original numerical scale—maps to an equivalent reward magnitude. It also prevents metrics with naturally broader ranges from dominating the gradient updates and reduces over-optimization of any single property due to saturation effects.

Conversely, for near-optimal properties ($\mathcal{P}_s$), the property must be constrained within a strict tolerance band $[L_p, U_p]$, where $L_p = p(M_x) - \Delta_p$ and $U_p = p(M_x) + \Delta_p$. To heavily penalize deviations, we design a Double Sigmoid score function:

$$s_p^{stab} = \sigma(\alpha_p \cdot (U_p - v_p)) \cdot \sigma(\alpha_p \cdot (v_p - L_p)) \quad (1)$$

This effective formulation creates a high-reward "plateau" within the acceptable range and imposes exponential decay upon any boundary violation.

3.3 Reward Aggregation

Each s_p denotes a property-level continuous preference score computed from the realized molecular property value v_p. After obtaining these property-level scores, we aggregate them using the geometric mean to enforce the simultaneous satisfaction of all constraints. Let $N = |\mathcal{P}_i| + |\mathcal{P}_s|$. The total reward $R_{total}(M_y)$ is:

$$R_{total}(M_y) = \left(\prod_{p \in \mathcal{P}_i} s_p^{imp} \prod_{q \in \mathcal{P}_s} s_q^{stab} \right)^{\frac{1}{N}} \quad (2)$$

Here, we deliberately employ the geometric mean rather than the standard arithmetic mean (linear scalarization) because it can be viewed as a continuous approximation of a min operator. The full explanation is in Appendix A.1.

3.4 GRPO Optimization

The final C-MORAL-GRPO policy follows the same loss $\mathcal{L}_{GRPO}(\Theta)$ in (Shao et al., 2024), employing an explicit KL divergence penalty to prevent catastrophic deviation from the reference model π_{ref}:

$$\mathcal{L}_{GRPO}(\Theta) = -\frac{1}{G} \sum_{i=1}^{G} \left[J_i(\Theta) - \beta D_{KL}(\pi_\Theta \| \pi_{ref}) \right] \quad (3)$$

By decoupling the absolute reward magnitude, C-MORAL-GRPO inherently preserves structural diversity and circumvents policy peaking. Detailed implementation is provided in Appendix F.1.

3.5 GDPO Optimization

To explicitly disentangle conflicting multi-objective feedback, we propose C-MORAL-GDPO. Rather than normalizing a single scalarized reward, GDPO evaluates relative superiority independently for each property across the generated group $\mathcal{G}$. These decoupled, property-specific advantages are then aggregated using a Log-Sum-Exp formulation. C-MORAL-GDPO mitigates the risk of "reward dominance," ensuring that easily optimized properties do not overshadow stricter constraints and preventing the implicit sacrifice of critical targets. Detailed mathematical formulations for the decoupled advantage and the aggregated loss are deferred to Appendix F.2. Further analysis of the reward aggregation choices is provided in Appendix A.2.

4 Experimental Setup

We evaluate the C-MORAL framework (**C**ontrollable Multi-**O**bjective **M**olecular **O**ptimization with **R**einforcement **A**lignment for **L**LMs) on two molecule optimization benchmarks: C-MuMOInstruct (Dey et al., 2025) and the MolOpt task from S^2-Bench (Li et al., 2025). We conduct experiments on three base models: two open-weight LLMs, LLAMA-3.1-8B (Touvron et al., 2023) and MISTRAL-7B (Jiang et al., 2023), and a smaller domain-specialized LLM, MOLT5 (Edwards et al., 2022).

To isolate the effect of RL post-training, the LLM policies are initialized from the same SFT checkpoints provided by Dey et al., namely GeLLM^{4}O-C (Dey et al., 2025). For MOLT5, we first perform supervised fine-tuning (SFT) on the molecular optimization dataset before applying RL post-training. The RL training is implemented using a custom training pipeline inspired by the design of the veRL framework (Sheng et al., 2025).

4.1 Datasets and Tasks

Both benchmarks formulate molecular optimization as conditional molecule generation.

For C-MuMOInstruct (Dey et al., 2025), we evaluate the model on 10 molecular optimization tasks and follow the predefined IND/OOD split summarized in Table A4. For OOD tasks, no paired optimized target molecules are available for the held-out property combinations. We therefore perform label-free RL adaptation using 10K unlabeled source molecules sampled only from the IND training split, with rewards computed from task-specific property oracles. The molecules in the OOD evaluation split are never used during SFT, RL adaptation, or model selection, ensuring that RL training molecules and final benchmark evaluation molecules are fully disjoint. Details are provided in Appendix B.

For the MolOpt task from S^2-Bench (Li et al., 2025), we further evaluate the model on three goal-oriented molecular optimization subtasks: molecular refractivity (MR), quantitative estimation of drug-likeness (QED), and LogP optimization. Details of the training datasets are in Appendix B.

4.2 Property Calculations

To provide accurate property feedback during both the reinforcement learning phase and the final evaluation, we employ two established computational oracles. Specifically, we use the open-source cheminformatics toolkit RDKit (Landrum et al., 2013) to validate generated SMILES strings, extract Morgan fingerprints, and compute fundamental physicochemical properties such as QED and Penalized LogP. For the remaining complex pharmacological properties, we use the ADMET-AI platform (Swanson et al., 2024) as our primary evaluator. These tools have been widely validated and adopted in recent molecular optimization studies (Averly et al., 2025; Zholus et al., 2025; Zheng et al., 2025).

4.3 Implementation and Hyperparameters

For large-scale models, we apply Low-Rank Adaptation (LoRA) (Hu et al., 2021) to the policy models instead of updating all parameters. The learning rate is set to 1×10^{-6}. Both GRPO and GDPO are trained with a batch size of 64 and a group size of 4. Policy updates are performed using 32 mini-batches over 2–3 optimization epochs per rollout. For MolT5, we use full fine-tuning for better adaptation. Detailed hyperparameters are provided in Tables A1 and A2. All RL post-training experiments are repeated with three random seeds.

4.4 Baselines and Evaluation Metrics

For C-MuMOInstruct, we compare C-MORAL with general-purpose LLMs, expert molecular generation models, and supervised fine-tuned molecular LLM baselines (Singh et al., 2026; OpenAI et al., 2024; Loeffler et al., 2024; Team et al., 2024). Following the standard benchmark protocol, all methods use the same 20-candidate generation budget and task-level selection rule, with details pro-

vided in Appendix D.1. For REINVENT-4, we perform task-specific staged training for each optimization setting and also generate 20 candidates at inference time. Therefore, all compared methods use the same candidate budget, property evaluators, and selection rule.

We evaluate C-MuMOInstruct using four metrics. **Success Optimized Rate (SOR)** measures whether the generated molecule improves the target properties while maintaining the required stability constraints. **Strict Success Optimized Rate (SSOR)** further requires all relevant properties to reach their prescribed near-optimal criteria. **Similarity (SIM)** is computed as the Tanimoto similarity between the generated and source molecules. **Relative Improvement (RI)** measures the relative improvement over the sub-optimal properties. More details are provided in Appendix D.1.

For S^2-Bench MolOpt, we follow the benchmark protocol and report property optimization performance with structural similarity in Appendix D.2.

5 Results

We evaluate C-MORAL on C-MuMOInstruct and S^2-Bench MolOpt. We first analyze the effect of reward design through ablations, then compare C-MORAL with strong baselines on C-MuMOInstruct under both IND and OOD settings. Finally, we report results on S^2-Bench MolOpt to test generalization to another molecular optimization benchmark.

5.1 Ablation Analysis

To validate our reward design, we conduct an ablation study on the IND tasks using MISTRAL. We examine two key components: Non-linear Reward Aggregation and Property Score Sigmoid Alignment. Table 1 reports the final performance, while Figure 2 visualizes the training dynamics on the challenging HLMPQ task.

Vulnerability of Linear AM and Reward Collapse. Linear AM performs poorly for both GRPO and GDPO, showing that simple averaging is not sufficient for constrained multi-objective optimization. In Table 1, GRPO with Linear AM drops below the SFT baseline in both SOR ($33.7\% \rightarrow 24.2\%$) and Sim ($0.58 \rightarrow 0.35$), while GDPO with Linear AM collapses further to only 9.5% SOR and 0.28 Sim. This trend is further supported by Figure 2, where AM-based training fails to keep all properties within their safe regions on the more challenging HLMPQ task.

Table 1: Ablation study on the IND task using MISTRAL. Non-linear aggregation prevents reward collapse and recovers molecular similarity, while Sigmoid alignment further improves overall and strict success rates. Best and second-best results are bold and underlined.

Method	IND Task			
	SOR(%)↑	**SSOR(%)↑**	**Sim↑**	**RI↑**
GeLLM^{4}O-C$_{\text{Mistral}}$	33.7	14.4	<u>0.58</u>	51.4
GRPO w/ Linear AM	24.2	11.2	0.35	21.5
+ Geometric Mean	39.7	17.1	<u>0.58</u>	<u>108.4</u>
+ Sigmoid Align	**48.9**	**25.1**	**0.59**	96.1
GDPO w/ Linear AM	9.5	4.5	0.28	12.3
+ LogSum-Exp	37.3	15.6	0.57	104.2
+ Sigmoid Align	<u>47.0</u>	<u>25.0</u>	**0.59**	**110.4**

Effectiveness of Non-Linear Reward Aggregation. Non-linear aggregation is the main factor that prevents reward collapse. In Table 1, replacing Linear AM with Geometric Mean improves GRPO from 24.2% to 39.7% SOR and restores Sim from 0.35 to 0.58. Similarly, replacing Linear AM with LogSum-Exp improves GDPO from 9.5% to 37.3% SOR and restores Sim from 0.28 to 0.57. Consistent with these final scores, Figure 2 shows that the non-linear variants maintain more stable property trajectories and avoid the boundary violations observed in AM-based training.

Effectiveness of Property Score Sigmoid Alignment. Sigmoid alignment is key to controllable optimization because it maps heterogeneous properties onto a shared scale. On top of non-linear aggregation, it increases GRPO from 39.7% to 48.9% SOR and from 17.1% to 25.1% SSOR. For GDPO, it improves SOR from 37.3% to <u>47.0%</u> and SSOR from 15.6% to <u>25.0%</u>. By reducing scale mismatch, it prevents metrics with broader ranges from dominating and shifts learning toward the remaining property bottlenecks.

Group-Relative Policy Optimization vs. PPO. We further compare the two group-relative methods with standard PPO. On IND tasks, the group-relative variants of C-MORAL achieve a $+8.4\%$ absolute improvement in SOR over standard PPO and converge approximately $3\times$ faster, as shown in Table A5 and Figure A4. These results suggest that the value-network-based PPO objective is less effective under the complex and conflicting constraints of molecular editing. Detailed performance comparisons and training curves are provided in Appendix E.

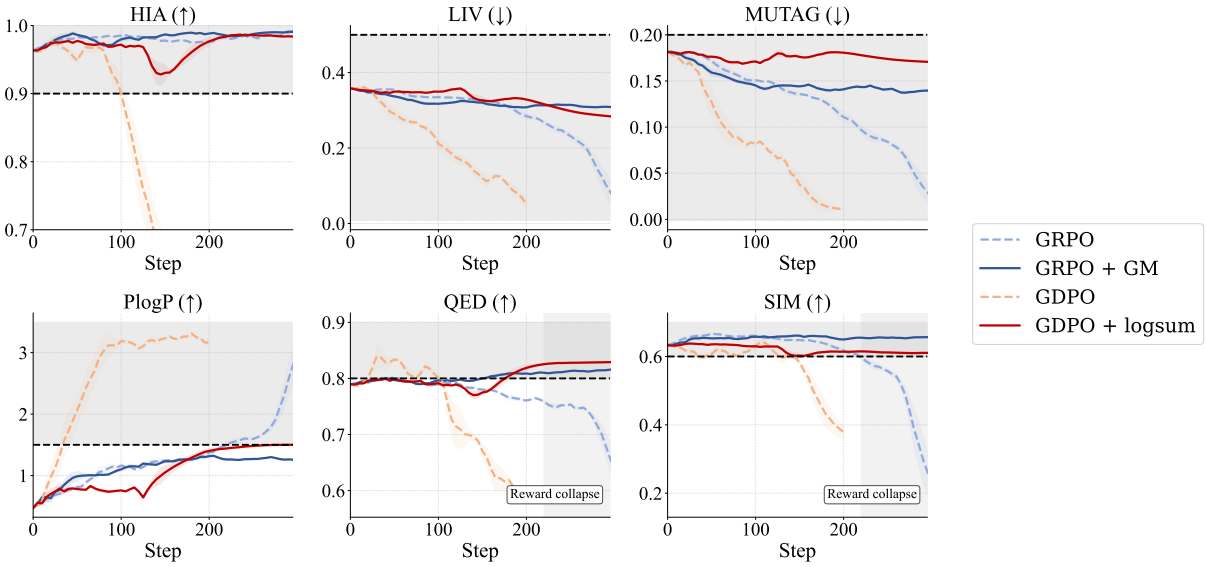


Figure 2: Ablation study of reward aggregation on the HLMPQ task using MISTRAL-7B over 300 training steps. Linear Arithmetic Mean (AM) suffers from reward collapse, while our non-linear aggregation methods maintain property values within safe boundaries.

5.2 Results on C-MuMOInstruct

We evaluate C-MORAL on C-MuMOInstruct across three backbones: MISTRAL, LLAMA, and MOLT5. As shown in Table 2(a), our framework consistently improves controllable multi-objective molecular optimization across different LLMs. Detailed per-task results are reported in Appendix H.

State-of-the-Art IND Performance. On IND tasks, C-MORAL outperforms all compared methods in controllable molecular optimization. As shown in Table 2(a), it obtains the best SOR of 48.9% with C-MORAL$_{\text{Mistral}}$ and the best SSOR of 25.4% with C-MORAL$_{\text{Llama}}$. These gains indicate that our RL framework better satisfies target property improvement and preservation constraints simultaneously. Meanwhile, C-MORAL maintains competitive similarity around 0.59–0.61 and strong molecular quality in validity, novelty, and synthetic accessibility, as detailed in Table A6(b), suggesting that the improvements are not primarily driven by structural drift or degenerate oracle-gaming artifacts. The radar visualization in Table 2(c) further supports this balanced improvement.

Adaptation to Unseen Objective Combinations. The OOD setting evaluates label-free adaptation to held-out objective combinations. As shown in Table 2(b), GRPO achieves the strongest OOD gains on both LLM backbones. For MISTRAL, it improves SOR from 24.4% to 39.5% (+61.9%) and SSOR from 9.8% to 20.8% (+112.2%). For LLAMA, it improves SOR from 25.2% to 37.2% (+47.6%) and SSOR from 10.2% to 19.6% (+92.2%). GDPO also remains highly competitive, reaching 38.3%/19.8% SOR/SSOR on MISTRAL and 35.6%/19.5% on LLAMA. These consistent OOD improvements, together with the molecular quality metrics in Table A7(b), suggest that C-MORAL adapts to unseen property combinations while maintaining valid, novel, and synthetically reasonable molecules.

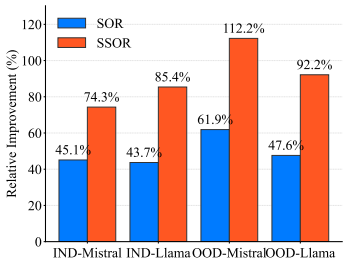
General Applicability across Model Scales. C-MORAL also improves the smaller molecule-text model MOLT5. Compared with MOLT5-Large (SFT), RL post-training increases IND SOR/SSOR from 13.9%/4.6% to 20.0%/8.8%, and OOD SOR/SSOR from 10.6%/3.6% to 19.2%/7.7%, while maintaining similarity around 0.60–0.61. Together with the stronger absolute performance of 7B/8B LLMs in Table 2(a), these results show that C-MORAL generalizes across model scales while model capacity remains important for complex controllable molecular editing.

GRPO vs. GDPO. GRPO and GDPO show complementary empirical strengths. GRPO achieves the best OOD SOR/SSOR on both MISTRAL (39.5%/20.8%) and LLAMA (37.2%/19.6%), suggesting that it is more favorable for strict success optimization under held-out objective combinations. In contrast, GDPO often achieves higher RI while maintaining comparable similarity, reach-

(a)

Model	In-Domain (IND)				Out-of-Domain (OOD)			
	SOR↑	SSOR↑	Sim↑	RI↑	SOR↑	SSOR↑	Sim↑	RI↑
General Purpose LLMs (Zero- & Few-Shot)								
Llama (0-shot)	3.0	0.8	**0.66**	8.0	2.0	0.5	**0.67**	6.3
Mistral (0-shot)	2.6	0.5	**0.66**	7.0	1.1	0.3	**0.67**	6.4
Llama (1-shot)	4.7	1.5	0.63	9.5	3.6	1.1	0.60	6.9
Mistral (1-shot)	5.8	2.3	0.61	9.3	4.3	1.4	0.62	9.2
GPT-5o (1-shot)	6.1	1.9	0.62	9.8	5.6	2.3	0.61	7.8
GPT-5o (2-shot)	7.8	2.9	0.62	13.5	7.7	2.8	0.60	10.3
Expert Models & SFT Baselines								
REINVENT-4	3.5	1.3	0.59	9.2	2.3	0.5	0.61	4.5
MolT5-Large	0.9	0.3	0.62	10.9	0.5	0.1	0.63	3.9
MolT5-Large (SFT)	13.9	4.6	0.62	20.3	10.6	3.6	0.62	5.8
GeLLM^{4}O-C$_{\text{Llama}}$	30.2	13.7	0.56	61.3	25.2	10.2	0.57	6.5
GeLLM^{4}O-C$_{\text{Mistral}}$	33.8	14.4	0.58	51.4	24.4	9.8	0.60	**27.3**
RL Post-Training (Ours: C-MORAL)								
GRPO Method								
C-MORAL$_{\text{MolT5}}$	20.0	7.9	0.61	25.4	19.2	7.7	0.61	4.6
C-MORAL$_{\text{Llama}}$	40.2	20.0	0.58	102.6	37.2	19.6	0.60	10.5
C-MORAL$_{\text{Mistral}}$	**48.9**	25.1	0.59	96.1	**39.5**	**20.8**	0.60	19.5
GDPO Method								
C-MORAL$_{\text{MolT5}}$	20.0	8.8	0.60	49.1	16.8	6.4	0.60	5.2
C-MORAL$_{\text{Llama}}$	43.4	**25.4**	0.57	110.3	35.6	19.5	0.59	19.1
C-MORAL$_{\text{Mistral}}$	47.0	24.9	0.59	**110.4**	38.3	19.8	0.59	27.2

(b)



(c)

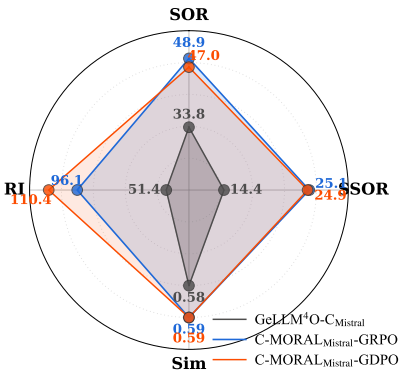


Table 2: Overall multi-objective optimization results on C-MuMOInstruct. (a) Comprehensive comparison of IND and OOD performance. For RL post-training methods, main-table results report the mean over three random seeds with beam search = 20; standard deviations are reported in Table A5. (b) Relative improvement margins in SOR and SSOR over the respective SFT baselines. (c) Multi-dimensional capability trade-offs on the Mistral backbone.

ing RI of 110.4 on MISTRAL IND and 110.3 on LLAMA IND. These results indicate that GRPO better supports constraint satisfaction, whereas GDPO is more beneficial for property improvement.

5.3 Results on S^2-Bench MolOpt

We further evaluate C-MORAL on S^2-Bench MolOpt, which considers both property improvement and structural similarity. As shown in Table A3, C-MORAL achieves the strongest overall performance across LogP, MR, and QED, consistently improving WSR over the corresponding GeLLM^{4}O-C backbones. For example, on LogP, WSR improves from 0.6554/0.6637 to 0.6756/0.6951 on the Mistral and LLaMA backbones, respectively. On MR, C-MORAL$_{\text{Mistral}}$ achieves the best WSR of 0.8126, with high SR (0.9546) and similarity (0.8513). These results show that C-MORAL generalizes beyond C-MuMOInstruct to standard goal-oriented molecular optimization.

6 Conclusion

We introduced C-MORAL, a constraint-aware RL post-training framework for controllable multi-objective molecular optimization with LLMs. The framework combines group-relative optimization with sigmoid-based property score alignment and bottleneck-sensitive non-linear reward aggregation. This design allows molecular LLMs to handle heterogeneous property scales, reduce reward dominance, and better satisfy competing improvement, stability, and scaffold-preservation constraints.

Empirical results on C-MuMOInstruct show that C-MORAL substantially improves both SOR and SSOR over strong SFT baselines across in-domain and out-of-domain settings, while maintaining competitive scaffold similarity. Ablation studies further demonstrate that non-linear aggregation prevents reward collapse and that sigmoid alignment improves controllability under heterogeneous molecular objectives. Results on S^2-Bench MolOpt also show consistent gains on standard goal-oriented optimization tasks, indicating that the proposed RL alignment strategy is not limited to the C-MuMO setting. Overall, C-MORAL highlights the potential of RL post-training as a general alignment paradigm for molecular LLMs, and suggests future opportunities to combine constraint-aware rewards with reasoning-enhanced molecular editing.

Limitations

Although C-MORAL shows strong performance on C-MuMOInstruct and S^2-Bench MolOpt, our study has several limitations. First, our experiments are conducted on existing molecular optimization benchmarks. While these benchmarks cover both controllable multi-objective optimization and standard MolOpt tasks, real-world drug discovery may involve additional constraints such as synthetic accessibility, target-specific binding affinity, and experimental validation. Second, due to computational resource constraints, we mainly evaluate two instruction-following LLM backbones, MISTRAL and LLAMA, together with the smaller molecule-text model MOLT5. Exploring substantially larger LLMs and more molecular foundation models remains future work. Third, our reward and evaluation signals rely on computational property oracles, which are widely used but may not perfectly reflect experimental molecular properties. Future work can further improve oracle reliability and explore reasoning-enhanced molecular optimization, where LLMs explicitly reason about property trade-offs and editing strategies before generating optimized molecules.

Ethics Statement

C-MORAL is an in silico framework for controllable molecular optimization. All experiments are conducted on standard molecular optimization benchmarks, and generated molecules are evaluated only through computational property oracles. As with other molecular generation methods, the model may carry potential dual-use risks if misused to design harmful compounds. We emphasize that generated molecules should be treated as computational hypotheses and require expert review, safety assessment, and laboratory validation before any real-world application. AI assistants were used to support language polishing, LaTeX formatting, and coding. All outputs were reviewed and verified by the authors.

References

Reza Averly, Frazier N. Baker, Ian A Watson, and Xia Ning. 2025. LIDDIA: Language-based intelligent drug discovery agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 12004–12028, Suzhou, China. Association for Computational Linguistics.

Viraj Bagal, Rishal Aggarwal, PK Vinod, and U Deva Priyakumar. 2021. Molgpt: molecular generation using a transformer-decoder model. *Journal of chemical information and modeling*, 62(9):2064–2076.

Dávid Bajusz, Anita Rácz, and Károly Héberger. 2015. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of cheminformatics*, 7(1):20.

Rhyan Barrett and Julia Westermayr. 2024. Reinforcement learning for traversing chemical structure space: Optimizing transition states and minimum energy paths of molecules. *The Journal of Physical Chemistry Letters*, 15(1):349–356.

Nathan Brown, Marco Fiscato, Marwin HS Segler, and Alain C Vaucher. 2019. Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108.

Vishal Dey, Xiao Hu, and Xia Ning. 2025. Large language models for controllable multi-property multi-objective molecule optimization. *molecules*, 1(331,586):433–166.

Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. 2022. Translation between molecules and natural language. *Preprint*, arXiv:2204.11817.

Daiki Erikawa, Nobuaki Yasuo, Takamasa Suzuki, Shogo Nakamura, and Masakazu Sekijima. 2023. Gargoyles: An open source graph-based molecular optimization method based on deep reinforcement learning. *ACS omega*, 8(40):37431–37441.

Vendy Fialková, Jiaxi Zhao, Kostas Papadopoulos, Ola Engkvist, Esben Jannik Bjerrum, Thierry Kogej, and Atanas Patronov. 2021. Libinvent: reaction-based generative scaffold decoration for in silico library design. *Journal of Chemical Information and Modeling*, 62(9):2046–2063.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. 2018. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276.

Longfei Guan, Hongbin Yang, Yingchun Cai, Lixia Sun, Peiwen Di, Weihua Li, Guixia Liu, and Yun Tang. 2019. Admet-score–a comprehensive scoring function for evaluation of chemical drug-likeness. *Medchemcomm*, 10(1):148–157.

Taicheng Guo, Kehan Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2023. What can large language models do in chemistry? a comprehensive benchmark on eight tasks. *Preprint*, arXiv:2305.18365.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Kei Katsuno, Jeremy N Burrows, Ken Duncan, Rob Hooft Van Huijsduijnen, Takushi Kaneko, Kiyoshi Kita, Charles E Mowbray, Dennis Schmatz, Peter Warner, and BT Slingsby. 2015. Hit and lead criteria in drug discovery for infectious diseases of the developing world. *Nature Reviews drug discovery*, 14(11):751–758.

Greg Landrum and 1 others. 2013. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8(31.10):5281.

Jiatong Li, Junxian Li, Weida Wang, Yunqing Liu, Changmeng Zheng, Dongzhan Zhou, Xiao yong Wei, and Qing Li. 2025. Speak-to-structure: Evaluating llms in open-domain natural language-driven molecule generation. *Preprint*, arXiv:2412.14642.

Xuan Li, Zhanke Zhou, Zongze Li, Jiangchao Yao, Yu Rong, Lu Zhang, and Bo Han. 2026. Reference-guided policy optimization for molecular optimization via llm reasoning. *arXiv preprint arXiv:2603.05900*.

Shengchao Liu, Jiongxiao Wang, Yijin Yang, Chengpeng Wang, Ling Liu, Hongyu Guo, and Chaowei Xiao. 2023. Chatgpt-powered conversational drug editing using retrieval and domain feedback. *Preprint*, arXiv:2305.18090.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, Jan Kautz, and Pavlo Molchanov. 2026. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *Preprint*, arXiv:2601.05242.

Xuefeng Liu, Songhao Jiang, Bo Li, and Rick Stevens. 2025. Controllablegpt: A ground-up designed controllable gpt for molecule optimization. *arXiv preprint arXiv:2502.10631*.

Hannes H Loeffler, Jiazhen He, Alessandro Tibo, Jon Paul Janet, Alexey Voronov, Lewis H Mervin, and Ola Engkvist. 2024. Reinvent 4: Modern ai-driven generative molecule design. *Journal of Cheminformatics*, 16(1):20.

Tung Nguyen and Aditya Grover. 2025. Lico: Large language models for in-context molecular optimization. *Preprint*, arXiv:2406.18851.

Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):48.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Jinyeong Park, Jaegyoon Ahn, Jonghwan Choi, and Jibum Kim. 2025. Mol-air: Molecular reinforcement learning with adaptive intrinsic rewards for goal-directed molecular generation. *Journal of Chemical Information and Modeling*, 65(5):2283–2296.

Jinyoung Park, Minseong Bae, Dohwan Ko, and Hyunwoo J. Kim. 2024. Llamo: Large language model-based molecular graph assistant. *Preprint*, arXiv:2411.00871.

Anastasiia V Sadybekov and Vsevolod Katritch. 2023. Computational approaches streamlining drug discovery. *Nature*, 616(7958):673–685.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, page 1279–1297. ACM.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, and 467 others. 2026. Openai gpt-5 system card. *Preprint*, arXiv:2601.03267.

Kyle Swanson, Parker Walther, Jeremy Leitz, Souhrid Mukherjee, Joseph C Wu, Rabindra V Shivnaraine, and James Zou. 2024. Admet-ai: a machine learning admet platform for evaluation of large-scale chemical libraries. *Bioinformatics*, 40(7):btae416.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, Soroosh Mariooryad, Yifan Ding, Xinyang Geng, Fred Alcober, Roy Frostig, Mark Omernick, Lexi Walker, Cosmin Paduraru, Christina Sorokin, and 1118 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *Preprint*, arXiv:2403.05530.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

W Patrick Walters and Regina Barzilay. 2020. Applications of deep learning in molecule generation and molecular property prediction. *Accounts of chemical research*, 54(2):263–270.

Haorui Wang, Marta Skreta, Cher-Tian Ser, Wenhao Gao, Lingkai Kong, Felix Strieth-Kalthoff, Chenru Duan, Yuchen Zhuang, Yue Yu, Yanqiao Zhu, Yuanqi Du, Alán Aspuru-Guzik, Kirill Neklyudov, and Chao Zhang. 2025. Efficient evolutionary search over chemical space with large language models. *Preprint*, arXiv:2406.16976.

Jing Wang and Fei Zhu. 2024. Multi-objective molecular generation via clustered pareto-based reinforcement learning. *Neural Networks*, 179:106596.

Zichao Wang, Weili Nie, Zhuoran Qiao, Chaowei Xiao, Richard Baraniuk, and Anima Anandkumar. 2023. Retrieval-based controllable molecule generation. *Preprint*, arXiv:2208.11126.

David Weininger. 1988. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36.

Geyan Ye, Xibao Cai, Houtim Lai, Xing Wang, Junhong Huang, Longyue Wang, Wei Liu, and Xiangxiang Zeng. 2023. Drugassist: A large language model for molecule optimization. *Preprint*, arXiv:2401.10334.

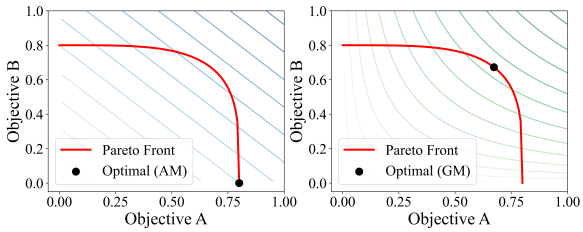
Yizhen Zheng, Huan Yee Koh, Jiaxin Ju, Anh TN Nguyen, Lauren T May, Geoffrey I Webb, and Shirui Pan. 2025. Large language models for scientific discovery in molecular property prediction. *Nature Machine Intelligence*, 7(3):437–447.

Artem Zholus, Maksim Kuznetsov, Roman Schutski, Rim Shayakhmetov, Daniil Polykovskiy, Sarath Chandar, and Alex Zhavoronkov. 2025. Bindgpt: A scalable framework for 3d molecular design via language modeling and reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):26083–26091.

Xiangxin Zhou, Xiwei Cheng, Yuwei Yang, Yu Bao, Liang Wang, and Quanquan Gu. 2024. Decompopt:

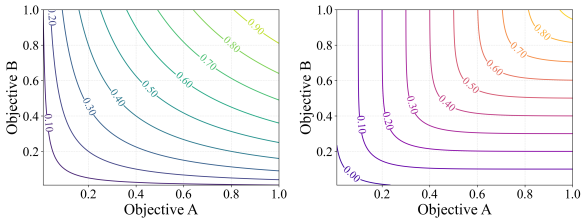
Controllable and decomposed diffusion models for structure-based molecular optimization. *Preprint*, arXiv:2403.13829.

Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. 2019. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):10752.



(a) Arithmetic Mean (b) Geometric Mean

Figure A1: Optimal solutions on a concave Pareto front. (a) Arithmetic Mean leads to extreme boundary solutions. (b) Geometric Mean forces balanced trade-offs.



(a) Geometric Mean (GM) (b) -Log-Sum-Exp (LogMin)

Figure A2: Contour plots of aggregation functions on two-dimensional objectives. (a) Geometric Mean (b) LogMin

A Details on Reward Aggregation

In multi-objective molecular optimization, the choice of reward aggregation directly determines the alignment behavior of the RL agent. We deliberately avoid standard linear scalarization in favor of non-linear aggregation methods, such as the Geometric Mean (GM) or the Smooth Minimum (-Log-Sum-Exp).

A.1 Reward Aggregation

The standard Arithmetic Mean (AM) defines the total reward as a linear combination of individual objectives: $R_{AM} = \frac{1}{N}\sum_{i=1}^{N} r_i$. While mathematically simple, it exhibits severe limitations when navigating complex chemical spaces.

As illustrated in Figure A1(a), the level curves of AM are straight lines. In molecular optimization, the true Pareto front between competing pharmacological properties (e.g., binding affinity versus drug-likeness) is typically strictly concave. Optimizing a linear objective over a concave front inevitably drives the solution toward the extremes (corner solutions). This mathematical artifact is the root cause of the "implicit sacrifice" phenomenon, where the LLM maximizes one easily optimizable property by completely violating another.

In contrast, Figure A1(b) demonstrates the behavior of the Geometric Mean:

$$R_{GM} = \left(\prod_{i=1}^{N} r_i\right)^{\frac{1}{N}}.$$

The GM constructs strictly convex, hyperbolic level sets. When these level sets intersect with the concave Pareto front, the optimal solution naturally settles near the center. This geometry enforces balanced trade-offs, effectively acting as a differentiable logical "AND" operator that requires all constraints to be reasonably satisfied.

A.2 Reward Aggregation: GRPO & GDPO

To enforce the simultaneous satisfaction of all constraints, the aggregation function must approximate the logical AND (min) operator. While both Geometric Mean (GM) and -Log-Sum-Exp (LogMin) provide continuous gradients, their mathematical properties dictate their specific applications in GRPO and GDPO.

As shown in Figure A2, GM and LogMin exhibit fundamentally different gradient behaviors.

For GM, defined as $R_{GM} = (\prod_{i=1}^{N} r_i)^{\frac{1}{N}}$, the gradient with respect to a single objective r_j is:

$$\frac{\partial R_{GM}}{\partial r_j} = \frac{1}{N}\frac{R_{GM}}{r_j}$$

This multiplicative nature ensures that the optimization of r_j is holistically scaled by the performance of all other objectives.

Conversely, LogMin acts as a strict bottleneck. Defined as $R_{LSE} = -\frac{1}{k}\log\sum_{i=1}^{N}\exp(-k\cdot x_i)$, its gradient is exactly the softmax distribution:

$$\frac{\partial R_{LSE}}{\partial x_j} = \frac{\exp(-k\cdot x_j)}{\sum_{i=1}^{N}\exp(-k\cdot x_i)}$$

When k is sufficiently large, if x_j is the worst-performing objective, its exponential term dominates the denominator. Consequently, $\frac{\partial R_{LSE}}{\partial x_j} \to 1$, while gradients for all other objectives approach 0. This mathematically explains the "L-shaped" contours in Figure A2(b), proving that LogMin forces the model to exclusively penalize the weakest property.

Domain Constraints Beyond gradients, the mathematical domain strictly determines algorithmic compatibility:

- **GRPO ($r_i \in [0,1]$):** GRPO maps raw properties to positive scores via non-linear reward shaping, perfectly satisfying the strict non-negativity requirement of the GM function.

- **GDPO ($A_i \in R$):** GDPO optimizes pairwise advantages, which naturally span negative values. Since GM is undefined for negative numbers, LogMin becomes mathematically mandatory. It seamlessly aggregates unbounded real numbers, isolating the most critical negative margin without breaking domain constraints.

B Details on Dataset

We provide additional details on the dataset usage and sampling protocol for reinforcement learning. Our experiments involve two molecule optimization benchmarks, C-MuMOInstruct and the MolOpt task from S^2-Bench. Both benchmarks are formulated as conditional molecular optimization, where each example contains a source molecule and an optimization instruction, and the model is expected to generate an optimized molecule satisfying the specified objective.

For C-MuMOInstruct, the In-Domain (IND) tasks provide paired training examples for each optimization task, where each pair consists of a source molecule and a corresponding target molecule satisfying the desired property constraints. Depending on the task, the number of available training pairs ranges from approximately 20K to 100K. In our reinforcement learning stage, instead of using the full training set, we randomly sample 10K source molecules from each IND task for RL training. To reduce the effect of sampling variance and provide a more robust evaluation, we repeat this sampling procedure with three different random seeds.

For the Out-of-Domain (OOD) tasks, no paired target molecules or supervised labels are available for the held-out property combinations. We therefore do not train on any ground-truth optimized OOD pairs. Instead, we perform label-free RL adaptation using 10K unlabeled source molecules sampled from the IND molecular pool. These molecules are reused only as starting points, while the optimization objectives are changed to the OOD property combinations. Rewards are computed directly from the generated molecules using task-specific property oracles and constraints, without using any target molecules.

For S^2-Bench, we use the training split of the MolOpt task, which evaluates goal-oriented molecular optimization over three property objectives: LogP, molecular refractivity (MR), and quantitative estimation of drug-likeness (QED). Consistent with our C-MuMOInstruct setting, we sample 10K molecules from the MolOpt training set for RL training. This ensures that all RL experiments are conducted under a comparable data budget across benchmarks.

C Details on Implementation

C.1 Hyperparameters

Hyperparameter	Value
Base Model	[Mistral-7B-v0.3, Llama3.1-8B]
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
Learning Rate	1×10^{-6}
Learning Rate Scheduler	Cosine with 10% warmup
LoRA Rank (r)	16
LoRA Alpha (α)	32
Target Modules	q_proj, v_proj, k_proj, o_proj, gate, up, down_proj, lm_head
Max Sequence Length	100 tokens
Temperature	1.0
RL Algorithm	GRPO/GDPO
Group Size (G)	4
KL Coefficient (Initial)	0.05
KL Target Value	1.0
Rollout Batch Size	32
Optimization Epochs	2
MiniBatch Size	32
Training Size	10000 mols per task
Hardware	$1 \times$ NVIDIA A100 (80GB)
Training Time	$\sim$6 Hours per task

Table A1: Hyperparameters for Mistral and Llama RL post-training stages.

Hyperparameter	Value
Shared Settings	
Base Model	MolT5
Training Method	Full-parameter fine-tuning
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
Learning Rate Scheduler	Cosine with 10% warmup
Max Sequence Length	100 tokens
Temperature	1.0
Training Size	10,000 molecules per task
Hardware	$1 \times$ NVIDIA A100 (80GB)
SFT Settings	
SFT Learning Rate	1×10^{-4}
SFT Batch Size	128
SFT Epochs	10
RL Settings	
RL Algorithm	GRPO/GDPO
GRPO/GDPO Learning Rate	5×10^{-5}
Group Size (G)	4
KL Coefficient (Initial)	0.05
KL Target Value	1.0
Rollout Batch Size	32
Optimization Epochs	2
MiniBatch Size	32
Hardware	$1 \times$ NVIDIA A100 (80GB)
Training Time	$\sim$3 Hours per task

Table A2: Hyperparameters for full-parameter SFT and RL post-training of MolT5.

```
%%% [INST]
Your task is to modify the given molecule to adjust specific molecular properties
so that the resulting molecule satisfies the given target thresholds. Keep
structural changes as minimal as possible. Your response should only contain a
valid SMILES representation of the modified molecule enclosed in <SMILES> </SMILES>
tags. The property values of the new molecule should meet or exceed the specified
targets enclosed in <THRESHOLD> </THRESHOLD> tags.

%%% Input:
<SMILES> COc1cc(C(=O)NCC(c2ccco2)N2CCOCC2)cc(OC)c1OC </SMILES>

%%% Adjustments:
change qed to be at least <THRESHOLD> 0.8 </THRESHOLD>,
change plogp to be at most <THRESHOLD> 1.5 </THRESHOLD>,
change bbbp to be at most <THERESHOLD> 0.91 </THRESHOLD>,

%%% Response:
<SMILES> O=C(NCC(c1ccco1)N1CCOCC1)c1cc(Cl)cc(Cl)c1 </SMILES>
```

Figure A3: An example of the highly structured prompt template used in C-MORAL. The prompt dynamically integrates the source molecule (Input) with explicit optimization directions and numerical thresholds (Adjustments) to guide the language model during the RL phase.

Model	LogP				MR				QED			
	SR↑	Sim↑	WSR↑	Validity↑	SR↑	Sim↑	WSR↑	Validity↑	SR↑	Sim↑	WSR↑	Validity↑
GPT-4o	0.7190	0.6586	0.4735	0.8796	0.6864	0.6420	0.4407	0.8352	0.3952	0.6180	0.2442	0.8570
Gemini-1.5-pro	0.7712	0.7022	0.5415	0.9274	0.7876	0.6744	0.5312	0.8926	0.4704	0.6077	0.2859	0.9484
Mistral-7B-Instruct-v0.2	0.2220	0.4501	0.0999	0.2802	0.1908	0.2578	0.0492	0.3795	0.1210	0.3244	0.0393	0.2532
Llama3-70B-Instruct (Int4)	0.5984	0.6028	0.3607	0.6482	0.5684	0.6032	0.3429	0.6272	0.2774	0.4828	0.1339	0.6340
MolT5-base	0.2074	0.1051	0.0218	0.4168	0.1856	0.1073	0.0199	0.3796	0.2358	0.1054	0.0249	0.4536
MolT5-large	0.4244	0.1015	0.0431	0.8156	0.4496	0.1072	0.0482	0.8678	0.4654	0.1190	0.0554	0.9214
Llama3.1-8B (OpenMolIns-large)	0.8054	0.6678	0.5378	0.8720	0.7122	0.6548	0.4663	0.8514	0.5224	0.6398	0.3342	0.8802
Llama3.1-8B (OpenMolIns-xlarge)	0.8822	0.6662	0.5877	0.9314	0.6982	0.6693	0.4673	0.9422	0.8648	0.6736	0.5825	0.9310
Galactica-125M (OpenMolIns-xlarge)	0.7362	0.5744	0.4229	0.8902	0.7124	0.5697	0.4059	0.8612	0.5786	0.5677	0.3285	0.8626
GeLLM^{4}O-C$_{\text{Mistral}}$	0.9010	0.7515	0.6554	**0.9998**	0.9420	0.7481	0.7047	0.9996	0.8760	0.7482	0.6770	0.9998
GeLLM^{4}O-C$_{\text{LLaMA}}$	0.8936	0.7427	0.6637	0.9996	0.9152	0.7539	0.6900	0.9994	0.8832	0.7189	0.6350	0.9994
C-MORAL$_{\text{Mistral}}$	**0.9650**	**0.8016**	0.6756	0.9996	**0.9546**	**0.8513**	**0.8126**	**0.9998**	0.9010	**0.7498**	**0.7735**	**1.0000**
C-MORAL$_{\text{LLaMA}}$	0.9215	0.7541	**0.6951**	**0.9998**	0.9372	0.7864	0.7370	0.9995	**0.9106**	0.7164	0.6523	0.9999

Table A3: Performance comparison on the MolOpt task from S^2-Bench. Baseline results above are directly reported from the original benchmark paper, while GeLLM^{4}O-C and C-MORAL are evaluated using the same MolOpt protocol. WSR is computed as SR $\times$ Sim. Best results are bolded and second-best results are underlined.

C.2 Prompt Design

We use structured prompt templates to help the language model interpret multi-objective molecular optimization tasks, as shown in Figures A3 and A6. For C-MORAL, the prompt is dynamically assembled for each episode: the `Input` field is filled with the sampled source SMILES, and the `Adjustments` field is generated by comparing the molecule's initial property values with the task thresholds Θ_p. Sub-optimal properties are converted into explicit improvement objectives, while already satisfactory properties are converted into preservation constraints. The quantitative targets are wrapped in `<THRESHOLD>` tags, and the model is instructed to return only the optimized molecule enclosed in `<SMILES>` tags.

For general-purpose LLM baselines, Figure A6 shows the corresponding few-shot prompting template. It provides a medicinal-chemistry system instruction, a molecular editing example, explicit property-wise thresholds, and the same constrained SMILES-only output format.

Algorithm 1 Candidate Selection

Require: Source molecule M_{src}, trained policy π_Θ, beam size $K = 20$
Require: Target properties $\mathcal{P}$, near-optimal thresholds Θ_p for $p \in \mathcal{P}$
Require: Optimization directions $d_p \in \{1, -1\}$ for $p \in \mathcal{P}$ ▷ 1: Maximize, -1: Minimize
Require: Target improvement margins Δ_p, and stability tolerance margins Δ_q
Ensure: The best aligned candidate M_{best}

Step 1: *Evaluate source molecule and dynamically formulate optimization tasks*
$\mathcal{P}_{sub} \leftarrow \{p \in \mathcal{P} \mid d_p \cdot p(M_{src}) < d_p \cdot \Theta_p\}$ ▷ Identify sub-optimal properties
$\mathcal{P}_{con} \leftarrow \{q \in \mathcal{P} \mid d_p \cdot q(M_{src}) \geq d_p \cdot \Theta_q\}$ ▷ Identify near-optimal properties

Step 2: *Generate candidates conditioned on the formulated tasks*
$\mathcal{M}_{gen} \leftarrow \text{BeamSearch}(\pi_\Theta(\cdot \mid M_{src}), K)$

Step 3: *Filter candidates to construct the SOR-compliant set $\mathcal{C}_{SOR}$*
$$\mathcal{C}_{SOR} \leftarrow \left\{ c \in \mathcal{M}_{gen} \;\middle|\; \begin{array}{l} \forall p \in \mathcal{P}_{sub} : d_p \cdot (p(c) - p(M_{src})) \geq \Delta_p \wedge \\ \forall q \in \mathcal{P}_{con} : |q(c) - q(M_{src})| \leq \Delta_q \end{array} \right\}$$

Step 4: *Select the optimal candidate based on Relative Improvement (RI)*
if $\mathcal{C}_{SOR} \neq \emptyset$ **then**
  $M_{best} \leftarrow \arg\max_{c \in \mathcal{C}_{SOR}} \text{RI}(c, M_{src})$
else
  $M_{best} \leftarrow \arg\max_{c \in \mathcal{M}_{gen}} \text{RI}(c, M_{src})$ ▷ Fallback: maximum RI relaxation
end if
return M_{best}

D Details on Evaluation Metrics

D.1 C-MuMOInstruct

Generation Protocols. Following the standard C-MuMOInstruct evaluation protocol (Dey et al., 2025), each method is evaluated under a fixed candidate budget of 20 generated molecules for each source molecule. For LLM-based methods, we use deterministic beam search with a beam width of 20 rather than stochastic sampling. Therefore, the 20 candidates correspond to the top beam hypotheses produced under the same decoding budget.

After candidate generation, the final output molecule is selected using the same task-level candidate selection rule as in the benchmark, as described in Algorithm 1. The selected molecule is then used to compute SOR, SSOR, SIM, and RI. This protocol is applied uniformly to all baselines and all C-MORAL variants, ensuring that all methods are compared under the same candidate budget, property evaluators, and selection rule.

Metrics To rigorously evaluate the performance of lead optimization, we formulate our four evaluation metrics mathematically. Let M_{src} denote the initial hit (source) molecule and M_{gen} denote the generated candidate. We partition the pharmacological properties into two sets: $\mathcal{P}_{sub}$ representing the targeted sub-optimal properties that require improvement, and $\mathcal{P}_{con}$ representing the stability constraints or near-optimal properties that must be maintained.

1. Similarity (SIM) To quantify structural preservation, we compute the Tanimoto similarity over Morgan fingerprints (radius 2, 2048 bits) between the generated candidate and the initial hit. All fingerprint generations and similarity calculations are implemented using the open-source cheminformatics RDKit. This ensures that the optimization strictly occurs within the valid chemical neighborhood of the lead compound.

$$\text{SIM}(M_{src}, M_{gen}) = \frac{|\text{Morgan}(M_{src}) \cap \text{Morgan}(M_{gen})|}{|\text{Morgan}(M_{src}) \cup \text{Morgan}(M_{gen})|}$$

The overall SIM score reported in our results is the average Tanimoto similarity across all valid generated molecules.

2. Success Optimized Rate (SOR) To ensure that the reported improvements are chemically

Type	Task	Target Properties ($\mathcal{P}$-Comb)
IND	BPQ	BBBP, PlogP, QED
	ELQ	hERG, LIV, QED
	ACEP	AMP, CARC, hERG, PlogP
	BDPQ	BBBP, DRD2, PlogP, QED
	DHMQ	DRD2, hERG, MUT, QED
OOD	CDE	CARC, DRD2, hERG
	ABMP	AMP, BBBP, MUT, PlogP
	BCMQ	BBBP, CARC, MUT, QED
	BDEQ	BBBP, DRD2, hERG, QED
	HLMPQ	hERG, LIV, MUT, PlogP, QED

Prop.	Δ_p	Θ_p	Prop.	Δ_p	Θ_p
AMP	0.1	0.8	HIA	0.1	0.9
BBBP	0.1	0.8	LIV	0.1	0.5
CARC	0.2	0.2	MUT	0.1	0.2
DRD2	0.1	0.4	PlogP	1.0	1.5
hERG	0.2	0.3	QED	0.1	0.9

Table A4: Overview of C-MuMOInstruct evaluation tasks and property thresholds. **(Top)** Target property combinations for both IND and OOD scenarios. **(Bottom)** The target improvement margins (Δ_p) and near-optimal constraints (Θ_p) for the 10 pharmacological properties.

meaningful and robust to model noise, we introduce a significance margin Δ_k. We divide the target properties into two subsets: $\mathcal{P}_{sub}$, containing properties that require directional optimization, and $\mathcal{P}_{con}$, containing properties that should remain stable relative to the source hit.

For each property $p \in \mathcal{P}_{sub}$, we define a direction indicator $s_p \in \{+1, -1\}$, where $s_p = +1$ denotes that the property is expected to increase and $s_p = -1$ denotes that it is expected to decrease. A generated molecule is considered successful on the optimization subset if all such properties improve beyond the significance margin:

$$C_{sub}^{(i)} = \prod_{p \in \mathcal{P}_{sub}} I\left(s_p\big(p(M_{gen}^{(i)}) - p(M_{src}^{(i)})\big) \geq \Delta_k\right).$$

For each property $q \in \mathcal{P}_{con}$, we require the generated molecule to remain within a tolerance band around the source hit:

$$C_{con}^{(i)} = \prod_{q \in \mathcal{P}_{con}} I\left(\left|q(M_{gen}^{(i)}) - q(M_{src}^{(i)})\right| \leq \Delta_k\right).$$

The overall SOR is then defined as the fraction of hit-candidate pairs that satisfy both conditions:

$$\text{SOR} = \frac{1}{N}\sum_{i=1}^{N}\left(C_{sub}^{(i)} \cdot C_{con}^{(i)}\right).$$

In our experiments, Δ_k is chosen according to the standard deviation or the known error bar of the corresponding property predictor.

3. Strict Success Optimized Rate (SSOR) While SOR evaluates whether a generated molecule achieves meaningful directional improvements on the target subset while preserving the constrained subset within tolerance, it does not require the final molecule to satisfy strict near-optimal criteria on all relevant properties. We therefore introduce the STRICT SUCCESS OPTIMIZED RATE (SSOR), which measures the fraction of generated molecules whose properties all fall within the desired near-optimal region.

Let Θ_r denote the near-optimal threshold for property $r \in \mathcal{P}$, where the satisfaction direction depends on the property type. We define the strict success indicator for the i-th pair as

$$C_{strict}^{(i)} = \prod_{r \in \mathcal{P}} I\big(\text{sat}_r(M_{gen}^{(i)}; \Theta_r) = 1\big),$$

where $\text{sat}_r(\cdot; \Theta_r)$ is a property-specific satisfaction function indicating whether property r meets its corresponding near-optimal threshold.

The overall SSOR is then defined as

$$\text{SSOR} = \frac{1}{N}\sum_{i=1}^{N} C_{strict}^{(i)}.$$

In other words, SSOR is a stricter metric than SOR, requiring all relevant properties to reach their prescribed near-optimal targets simultaneously.

4. Relative Improvement (RI) While SOR and SSOR measure whether a generated molecule satisfies the desired optimization criteria, they do not reflect the magnitude of improvement. We therefore define the RELATIVE IMPROVEMENT (RI) to quantify the average directional relative change on the subset of sub-optimal properties.

For each property $p \in \mathcal{P}_{sub}$, let $s_p \in \{+1, -1\}$ denote its desired optimization direction, where $s_p = +1$ indicates that the property is expected to increase and $s_p = -1$ indicates that it is expected to decrease. We first define the instance-level relative improvement as

$$\text{RI}^{(i)} = \frac{1}{|\mathcal{P}_{sub}|}\sum_{p \in \mathcal{P}_{sub}} \frac{s_p\big(p(M_{gen}^{(i)}) - p(M_{src}^{(i)})\big)}{\left|p(M_{src}^{(i)})\right|}.$$

The overall RI is then computed by averaging over all molecule pairs:

$$\text{RI} = \frac{1}{N}\sum_{i=1}^{N} \text{RI}^{(i)}.$$

In this way, RI measures the average relative change on sub-optimal properties in their desired optimization directions: positive values indicate improvement, while negative values indicate movement against the target direction.

D.2 S^2-Bench MolOpt

To evaluate the generalization and open-domain optimization capabilities of C-MORAL, we adopt the S^2-Bench (TOMG-Bench) (Li et al., 2025). The Molecule Optimization (MolOpt) task requires the model to modify a lead compound to achieve a specific property shift while preserving the core scaffold. We employ four primary metrics:

- **Validity:** The proportion of generated SMILES strings successfully parsed by RDKit.
- **Similarity:** Tanimoto similarity (Morgan fingerprints) between source and candidate molecules, ensuring minimal, targeted edits.
- **Success Rate (SR):** The proportion of valid, structurally similar candidates that satisfy the property target threshold.
- **Weighted Success Rate (WSR):** The similarity-weighted success rate, computed as $\text{WSR} = \text{SR} \times \text{Sim}$. It measures the joint quality of property success and structural preservation.

This weighted scoring mechanism prevents models from achieving high success rates by generating structurally distant, high-scoring molecules.

E Group Policy over PPO

Superiority of Group-Relative Alignment over PPO. A critical question in RL post-training is whether specialized group-relative frameworks are strictly necessary compared to standard Proximal Policy Optimization (PPO). As detailed in Table A5 and Figure A4, the answer is definitively yes. Empirically, PPO struggles to navigate the highly constrained, conflicting reward landscapes of molecular optimization. Its reliance on a separate, memory-intensive value network often leads to unstable advantage estimation when multiple heterogeneous objectives compete.

Consequently, C-MORAL's group-based variants achieve consistently higher success rates under both IND and OOD settings. On the representative Mistral IND tasks, C-MORAL delivers an absolute gain of $+8.4\%$ in SOR and $+5.5\%$ in SSOR over PPO, while also converging approximately $3.0\times$ faster (Figure A4). By normalizing advantages directly across peer candidates rather than struggling to fit an absolute value baseline, C-MORAL provides a much sharper and more stable gradient signal. This confirms that group-relative alignment is better suited for the delicate structural trade-offs required in lead optimization across both in-domain and out-of-domain objective settings.

F Group-Based Policy Optimization Implementation

F.1 Group Relative Policy Optimization

We provide the implementation details of GRPO used in our experiments. Consider a mini-batch of prompts $\{x_i\}_{i=1}^{B}$. For each prompt x_i, we sample a group of G responses

$$\mathcal{Y}_i = \{y_{i,1}, \ldots, y_{i,G}\}, \qquad y_{i,j} \sim \pi_{\Theta_{\text{old}}}(\cdot \mid x_i).$$

Assume there are M property-wise aligned scores. The m-th aligned score of response $y_{i,j}$ is denoted by

$$s_{i,j}^{(m)} = s_m(x_i, y_{i,j}), \qquad m = 1, \ldots, M.$$

In standard multi-reward GRPO, multiple reward dimensions are often aggregated into a single scalar reward before group-wise normalization. In C-MORAL-GRPO, we replace linear aggregation with the sigmoid-aligned geometric mean reward defined in Eq. 2. Specifically, the scalar reward for response $y_{i,j}$ is computed as

$$r_{i,j}^{\text{GRPO}} = \left(\prod_{m=1}^{M} s_{i,j}^{(m)}\right)^{1/M}$$

This scalar reward is then used for group-relative normalization.

For each prompt x_i, we then compute the group mean and group standard deviation over the G sampled responses:

$$\mu_i = \frac{1}{G}\sum_{j=1}^{G} r_{i,j}^{\text{GRPO}},$$

(a) Performance based on Mistral-7B architecture

Method	In-Domain (IND)				Out-of-Domain (OOD)			
	SOR$^{\uparrow}$	**SSOR**$^{\uparrow}$	**Sim**$^{\uparrow}$	**RI**$^{\uparrow}$	**SOR**$^{\uparrow}$	**SSOR**$^{\uparrow}$	**Sim**$^{\uparrow}$	**RI**$^{\uparrow}$
SFT Baseline	33.7	14.4	0.58	51.4	24.4	9.8	0.60	27.3
PPO	40.5±4.1	19.6±2.1	0.57±0.02	81.7±11.5	32.0±3.6	14.4±1.8	0.59±0.01	23.6±4.7
GRPO	48.9±5.0	25.1±3.0	0.59±0.01	96.1±10.7	39.5±3.9	20.8±2.8	0.60±0.03	19.5±3.3
GDPO	47.0±3.9	25.0±2.8	0.59±0.01	110.4±14.2	38.3±3.0	19.8±2.5	0.59±0.02	27.2±3.9

(b) Performance based on Llama-8B architecture

Method	In-Domain (IND)				Out-of-Domain (OOD)			
	SOR$^{\uparrow}$	**SSOR**$^{\uparrow}$	**Sim**$^{\uparrow}$	**RI**$^{\uparrow}$	**SOR**$^{\uparrow}$	**SSOR**$^{\uparrow}$	**Sim**$^{\uparrow}$	**RI**$^{\uparrow}$
SFT Baseline	30.2	13.7	0.56	61.3	25.2	10.2	0.57	6.5
PPO	35.9±3.8	16.1±1.1	0.57±0.03	68.6±6.2	29.1±2.1	12.8±1.2	0.58±0.02	19.4±4.1
GRPO	40.3±4.3	20.1±1.4	0.58±0.01	102.6±9.4	37.2±2.9	19.6±1.8	0.60±0.02	10.5±4.3
GDPO	43.4±5.7	25.4±1.6	0.57±0.02	110.3±10.1	35.6±3.0	19.5±1.6	0.59±0.01	19.1±5.1

Table A5: Comparison of RL algorithms under identical non-linear reward aggregation and sigmoid alignment. We report results on both Mistral-7B and Llama-8B backbones across In-Domain (IND) and Out-of-Domain (OOD) C-MuMOInstruct tasks. The SFT baseline is evaluated once from the fixed checkpoint, while all RL post-training results are averaged over three random seeds and reported as mean ± standard deviation.

$$\sigma_i = \sqrt{\frac{1}{G}\sum_{j=1}^{G}\big(r_{i,j}^{\text{GRPO}} - \mu_i\big)^2}.$$

The group-relative advantage is defined as

$$A_{i,j}^{\text{GRPO}} = \frac{r_{i,j}^{\text{GRPO}} - \mu_i}{\sigma_i + \epsilon_{\text{grp}}},$$

where ϵ_{grp} is a small constant for numerical stability.

For token t in response $y_{i,j}$, the importance ratio is

$$\rho_{i,j,t}(\Theta) = \frac{\pi_\Theta(y_{i,j,t} \mid x_i, y_{i,j,<t})}{\pi_{\Theta_{\text{old}}}(y_{i,j,t} \mid x_i, y_{i,j,<t})}.$$

The GRPO objective is

$$\mathcal{L}_{\text{GRPO}}(\Theta) = \frac{1}{B}\sum_{i=1}^{B}\frac{1}{G}\sum_{j=1}^{G}\frac{1}{|y_{i,j}|} \sum_{t=1}^{|y_{i,j}|}\min\Big(\rho_{i,j,t}(\Theta)A_{i,j}, \text{clip}\big(\rho_{i,j,t}(\Theta), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}}\big)A_{i,j}\Big).$$

In our implementation, we include a KL regularization term following the standard GRPO formulation. Compared with GDPO, GRPO performs normalization only after collapsing all reward dimensions into a single scalar reward, rather than normalizing each reward dimension separately before aggregation.

F.2 Group reward-Decoupled Normalization Policy Optimization

We provide the implementation details of GDPO used in our experiments. Consider a mini-batch of prompts $\{x_i\}_{i=1}^{B}$. For each prompt x_i, we sample a group of G responses

$$\mathcal{Y}_i = \{y_{i,1}, \ldots, y_{i,G}\}, \qquad y_{i,j} \sim \pi_{\Theta_{\text{old}}}(\cdot \mid x_i).$$

Assume there are M reward functions. The m-th reward of response $y_{i,j}$ is denoted by

$$r_{i,j}^{(m)} = r_m(x_i, y_{i,j}), \qquad m = 1, \ldots, M.$$

Step 1: Group-wise decoupled normalization. For each prompt x_i and each reward dimension m, we first compute the group mean and group standard deviation over the G sampled responses:

$$\mu_i^{(m)} = \frac{1}{G}\sum_{j=1}^{G} r_{i,j}^{(m)},$$

$$\sigma_i^{(m)} = \sqrt{\frac{1}{G}\sum_{j=1}^{G}\big(r_{i,j}^{(m)} - \mu_i^{(m)}\big)^2}.$$

Then the reward-specific normalized advantage is

$$A_{i,j}^{(m)} = \frac{r_{i,j}^{(m)} - \mu_i^{(m)}}{\sigma_i^{(m)} + \epsilon_{\text{grp}}}.$$

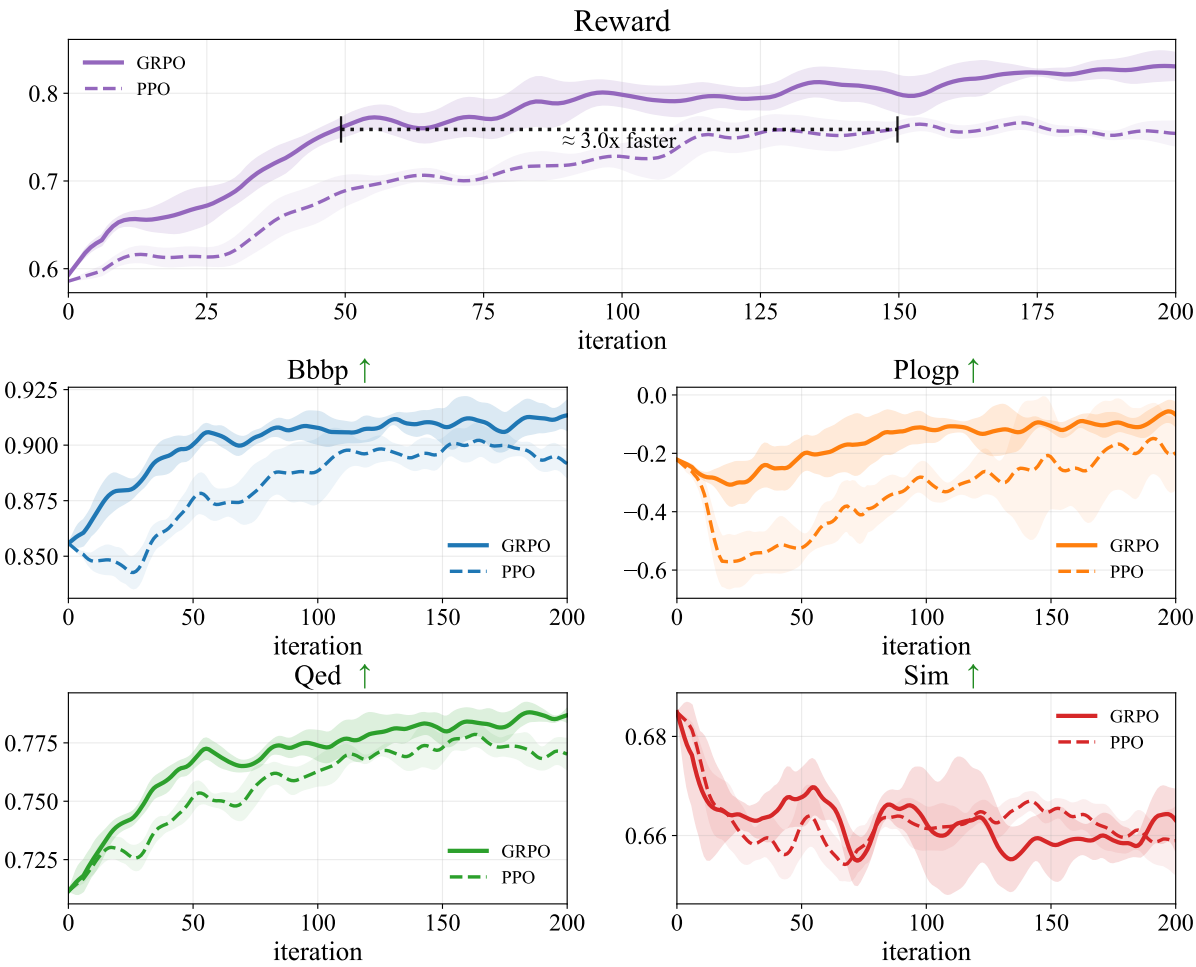


Figure A4: Training efficiency comparison of GRPO vs. PPO on the BPQ task (based on Mistral 7B). The learning curves for the overall reward and individual properties (Bbbp, Plogp, Qed, Sim) demonstrate that GRPO converges $\approx 3.0\times$ faster than PPO. An initial 30-iteration warmup phase applied to PPO is excluded from the plot.

Step 2: Smooth-min aggregation of decoupled advantages. After obtaining the reward-specific normalized advantages, we aggregate them using a smooth-min operator based on negative Log-Sum-Exp:

$$\tilde{A}_{i,j} = -\frac{1}{\tau} \log \left(\frac{1}{M} \sum_{m=1}^{M} \exp \left(-\tau w_m A_{i,j}^{(m)} \right) \right), \tag{4}$$

where $\tau > 0$ controls the sharpness of the smooth-min operator, and $w_m \geq 0$ is the weight of the m-th reward dimension. In our default setting, we use $w_m = 1$ for all reward dimensions unless otherwise specified. This aggregation gives larger influence to the weakest property-wise advantage, thereby reducing the implicit sacrifice of difficult molecular constraints.

Step 3: Batch-wise normalization (BN). To keep the numerical scale of advantages stable as the number of rewards increases, we further normalize $\tilde{A}_{i,j}$ over all responses in the current mini-batch. Let

$$\mu_{\mathcal{B}} = \frac{1}{BG} \sum_{i=1}^{B} \sum_{j=1}^{G} \tilde{A}_{i,j},$$

$$\sigma_{\mathcal{B}} = \sqrt{\frac{1}{BG} \sum_{i=1}^{B} \sum_{j=1}^{G} \left(\tilde{A}_{i,j} - \mu_{\mathcal{B}} \right)^2}.$$

The final GDPO advantage is

$$\hat{A}_{i,j} = \frac{\tilde{A}_{i,j} - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}} + \epsilon_{\text{bn}}}.$$

This step is crucial in the original paper. After testing, we find it also important for stable training.

Step 4: Policy optimization objective. We then use the final normalized advantage $\hat{A}_{i,j}$ in a clipped

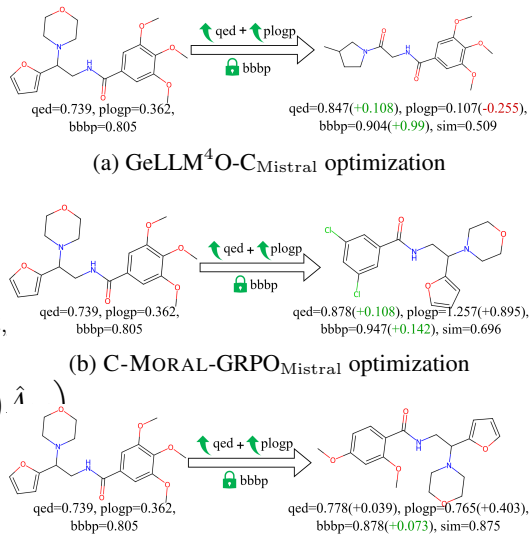
policy optimization objective. For token t in response $y_{i,j}$, define the importance ratio as

$$\rho_{i,j,t}(\Theta) = \frac{\pi_\Theta(y_{i,j,t} \mid x_i, y_{i,j,<t})}{\pi_{\Theta_{\text{old}}}(y_{i,j,t} \mid x_i, y_{i,j,<t})}.$$

The GDPO objective is

$$\mathcal{L}_{\text{GDPO}}(\Theta) = \frac{1}{B}\sum_{i=1}^{B}\frac{1}{G}\sum_{j=1}^{G}\frac{1}{|y_{i,j}|}\sum_{t=1}^{|y_{i,j}|}\min\Big(\rho_{i,j,t}(\Theta)\,\hat{A}_{i,} \\ \text{clip}\big(\rho_{i,j,t}(\Theta),\, 1-\epsilon_{\text{clip}},\, 1+\epsilon_{\text{clip}}\big)\,\hat{A}_{i,}\Big)$$

If a KL regularization term is used in practice, it can be added in the standard way. The key difference from multi-reward GRPO is that GDPO normalizes each reward *before* aggregation, and then applies an additional batch-wise normalization to the aggregated advantage.



(a) GeLLM^{4}O-C$_{\text{Mistral}}$ optimization

(b) C-MORAL-GRPO$_{\text{Mistral}}$ optimization

(c) C-MORAL-GDPO$_{\text{Mistral}}$ optimization

Figure A5: Optimization of different MISTRAL-based models on the BPQ task.

G Case Studies

G.1 BPQ Task

BPQ (BBBP, PlogP, QED) involves diverse combinations of property-specific objectives across BBBP, PlogP, and QED, three key properties for CNS drug design. Each optimization task may require improving one or more properties while maintaining or further enhancing the others. Optimizing these diverse multi-objective combinations simulates the early-stage filtering and refinement of CNS-active hit compounds.

Figure A5 presents one representative example from the BPQ task. Compared with the SFT baseline in Figure A5a, both RL post-trained models produce more favorable BPQ edits while better preserving meaningful structural motifs. GeLLM^{4}O-C$_{\text{Mistral}}$ achieves the target improvement mainly through a relatively aggressive rewrite of the left-half structure, replacing the original peripheral heterocyclic region with a more compact motif while keeping only part of the right aromatic scaffold. In contrast, C-MORAL-GRPO$_{\text{Mistral}}$ in Figure A5b performs a larger global restructuring, substantially changing both the aromatic core and the surrounding substituents; although this yields strong property improvement, it is less conservative in scaffold preservation. C-MORAL-GDPO$_{\text{Mistral}}$ in Figure A5c, however, makes more targeted local edits: it largely retains the original amide-linked aromatic core and the morpholine-containing motif, while modifying the peripheral substituents to improve QED and PlogP under the BBBP constraint. This qualitative example suggests that GDPO tends to achieve a better balance between property optimization and structural preservation, whereas GRPO explores more radical scaffold-level changes.

H Complete Experimental Results

H.1 S^2-Bench MolOpt

Table A3 reports the complete results on the MolOpt task from S^2-Bench. Following the benchmark protocol, we evaluate three goal-oriented molecular optimization tasks: LogP, molecular refractivity (MR), and QED. SR denotes the success rate of achieving the target property improvement, Sim denotes the Tanimoto similarity between the source and generated molecules, and WSR is computed as $\mathrm{SR} \times \mathrm{Sim}$ to measure the joint quality of property improvement and structural preservation. Validity denotes the percentage of generated SMILES that can be parsed as valid molecules by RDKit. Overall, C-MORAL achieves the strongest WSR across all three tasks while maintaining high validity, showing that the proposed RL alignment strategy generalizes beyond C-MuMOInstruct to standard goal-oriented molecular optimization.

H.2 C-MuMOInstruct

In Table A6 and A7, we provide complete per-task results on C-MuMOInstruct. Table A6 reports detailed results on IND tasks, while Table A7 reports detailed results on OOD tasks. For optimization performance, S/SS denotes SOR/SSOR: SOR measures whether the generated molecule satisfies the required property improvement and preservation constraints, while SSOR further requires all relevant properties to reach their prescribed near-optimal targets. Sim denotes the Tanimoto similarity between the source and generated molecules computed from Morgan fingerprints, and RI measures the relative improvement over sub-optimal properties. We also report molecular quality metrics: Val denotes the percentage of RDKit-valid generated SMILES, SAS denotes the synthetic accessibility score where lower is better, and Nov denotes the percentage of valid generated molecules whose RDKit-canonicalized SMILES do not appear in the training set.

I Licenses and Artifact Use

Table A8 summarizes the licenses, terms of use, and accessibility of the main artifacts used in this work. We use all artifacts only for research and evaluation purposes, following their intended use for molecular optimization, model evaluation, or computational property assessment. We do not redistribute the original datasets, model checkpoints, APIs, or external tools as part of this submission.

(a) Detailed IND performance

Model	BPQ			ELQ			ACEP			BDPQ			DHMQ		
	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑
General Purpose LLMs (Zero- & Few-Shot)															
Llama	4.4 / 1.9	**0.69**	0.79	4.3 / 1.2	**0.66**	0.32	3.1 / 0.7	0.67	1.21	1.1 / 0.2	0.64	19.1	0.9 / 0.1	0.65	18.7
Mistral	3.4 / 1.2	0.65	0.81	4.5 / 0.5	0.65	0.35	2.7 / 0.6	0.64	1.19	0.7 / 0.0	**0.70**	15.2	0.8 / 0.1	**0.68**	17.6
Llama (1-shot)	4.8 / 2.0	0.59	0.81	6.9 / 2.2	0.65	0.41	5.2 / 1.8	0.61	1.17	2.1 / 0.4	0.68	21.4	2.5 / 0.9	0.62	23.5
Mistral (1-shot)	10.8 / 4.5	0.58	1.21	7.5 / 3.1	0.63	0.39	5.1 / 1.9	0.61	1.52	3.0 / 1.3	0.59	16.7	2.6 / 0.7	0.62	26.6
GPT-5o (1-shot)	7.4 / 2.1	0.64	1.01	9.9 / 2.9	0.59	0.33	6.6 / 1.8	0.61	1.78	2.9 / 1.4	0.65	14.6	3.6 / 1.3	0.62	31.5
GPT-5o (2-shot)	11.6 / 4.3	0.58	1.43	12.7 / 4.6	0.60	0.35	6.8 / 2.1	**0.69**	2.32	3.7 / 1.8	0.59	11.1	4.0 / 1.9	0.63	52.4
Expert Models & SFT Baselines															
REINVENT-4	4.2 / 1.4	0.61	1.95	3.4 / 1.1	0.56	0.47	5.1 / 2.3	0.59	1.24	2.4 / 1.0	0.60	17.4	2.6 / 0.9	0.58	25.1
MolT5-Large (Base)	1.3 / 0.2	0.61	1.44	0.9 / 0.1	0.60	0.40	1.9 / 1.0	0.60	1.52	0.2 / 0.0	0.63	9.8	0.4 / 0.1	0.64	41.3
MolT5-Large (SFT)	24.8 / 7.8	0.62	2.39	22.2 / 9.2	0.60	0.45	12.8 / 4.0	0.63	3.95	5.6 / 1.4	0.60	13.5	4.0 / 0.8	0.64	81.4
GeLLM^{4}O-C$_{\text{Llama}}$	48.0 / 22.8	0.56	2.61	53.8 / 24.8	0.56	0.47	31.6 / 13.02	0.55	3.82	10.2 / 4.6	0.54	40.2	7.4 / 3.2	0.57	259.2
GeLLM^{4}O-C$_{\text{Mistral}}$	52.2 / 21.0	0.59	2.72	62.0 / 30.2	0.58	0.48	29.6 / 11.0	0.58	3.55	11.8 / 4.6	0.56	188.4	13.2 / 5.4	0.60	61.9
RL Post-Training (C-MORAL, Ours)															
GRPO Method															
C-MORAL$_{\text{MolT5}}$	31.0 / 11.6	0.61	2.97	37.0 / 15.6	0.59	**2.49**	18.2 / 8.4	0.61	2.45	8.6 / 2.1	0.60	93.7	5.2 / 1.8	0.64	25.23
C-MORAL$_{\text{Llama}}$	57.8 / 26.6	0.59	3.50	60.0 / 28.6	0.57	0.51	**45.2 / 25.0**	0.58	3.06	21.6 / 10.8	0.56	256.2	16.6 / 9.2	0.60	249.5
C-MORAL$_{\text{Mistral}}$	65.7 / 36.4	0.60	3.22	**73.4 / 38.8**	0.58	0.51	42.4 / 20.8	0.59	3.54	**24.8 / 11.6**	0.58	238.2	**38.2 / 18.0**	0.62	234.9
GDPO Method															
C-MORAL$_{\text{MolT5}}$	33.4 / 13.8	0.60	3.91	33.0 / 13.6	0.60	0.48	20.4 / 11.9	0.58	2.07	7.8 / 2.4	0.61	202.3	5.3 / 2.2	0.62	36.8
C-MORAL$_{\text{Llama}}$	66.0 / **44.8**	0.57	**4.43**	64.2 / 34.6	0.57	0.51	43.7 / 22.7	0.57	4.09	18.5 / 9.9	0.56	271.1	24.4 / 15.1	0.60	**271.6**
C-MORAL$_{\text{Mistral}}$	**69.8** / 40.2	0.58	4.05	72.8 / 37.0	0.57	0.53	44.4 / 23.0	0.57	**4.15**	22.6 / 10.0	0.58	**391.5**	25.6 / 14.2	0.63	151.5

(b) Additional IND molecular quality metrics

Model	BPQ			ELQ			ACEP			BDPQ			DHMQ		
	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑
General Purpose LLMs (Zero- & Few-Shot)															
Llama	96.20	2.76	100.00	97.80	2.89	100.00	95.20	2.72	99.79	98.00	3.02	99.59	95.80	2.99	100.00
Mistral	88.20	2.81	98.98	91.40	2.87	100.00	87.60	2.78	100.00	78.60	2.80	100.00	86.00	2.91	100.00
Llama (1-shot)	98.80	2.74	100.00	99.00	2.75	98.38	99.20	2.69	95.36	98.60	3.06	97.97	99.40	2.78	93.16
Mistral (1-shot)	97.80	2.62	95.91	99.40	2.79	96.38	98.80	2.71	93.12	99.20	2.58	96.57	99.00	2.76	93.94
GPT-5o (1-shot)	92.40	2.75	98.48	94.40	2.84	97.88	92.80	2.67	98.06	89.20	2.65	100.00	88.80	2.71	99.10
GPT-5o (2-shot)	94.80	2.69	79.96	95.60	2.80	83.68	92.60	2.71	84.23	91.00	2.69	89.01	90.40	2.75	90.71
Expert Models & SFT Baselines															
REINVENT-4	100.00	2.38	100.00	100.00	2.62	99.60	100.00	2.43	99.80	100.00	2.46	99.80	99.80	2.57	100.00
MolT5-Large (Base)	99.80	2.47	100.00	99.60	2.72	100.00	100.00	2.51	99.80	98.80	2.54	99.60	99.00	2.79	99.80
MolT5-Large (SFT)	98.20	2.35	97.34	98.20	2.68	98.16	99.40	2.31	98.19	98.80	2.51	99.19	99.00	2.67	98.38
GeLLM^{4}O-C$_{\text{Llama}}$	90.20	2.36	97.34	90.80	2.64	97.80	92.40	2.38	97.40	89.20	2.55	98.21	88.60	2.70	99.10
GeLLM^{4}O-C$_{\text{Mistral}}$	98.80	2.32	97.98	99.20	2.59	98.16	100.00	2.36	99.20	98.40	2.58	98.58	99.00	2.67	99.00
RL Post-Training (C-MORAL, Ours)															
GRPO Method															
C-MORAL$_{\text{MolT5}}$	98.40	2.26	98.98	99.20	2.49	98.79	99.20	2.12	95.77	98.80	2.29	97.17	98.40	2.69	97.36
C-MORAL$_{\text{Llama}}$	100.00	2.28	98.80	99.80	2.57	96.80	99.80	2.26	97.39	99.60	2.40	96.39	100.00	2.62	97.80
C-MORAL$_{\text{Mistral}}$	100.00	2.34	98.20	99.80	2.56	97.19	100.00	2.36	98.40	99.80	2.37	98.19	100.00	2.67	99.00
GDPO Method															
C-MORAL$_{\text{MolT5}}$	98.40	2.25	98.98	98.60	2.53	98.78	98.40	2.14	99.60	98.80	2.41	98.98	98.40	2.61	99.20
C-MORAL$_{\text{Llama}}$	100.00	2.13	98.80	100.00	2.68	99.00	100.00	2.09	98.80	100.00	2.50	99.20	100.00	2.59	99.40
C-MORAL$_{\text{Mistral}}$	100.00	2.24	99.20	99.80	2.58	99.20	100.00	2.23	98.80	99.80	2.33	97.19	100.00	2.61	98.60

Table A6: IND per-task results on C-MuMOInstruct. (a) Detailed performance comparison using S/SS, similarity, and RI. (b) Additional molecular quality metrics, including validity, synthetic accessibility score (SAS), and novelty. Val and Nov are reported in percentages; lower SAS indicates better synthetic accessibility.

(a) Detailed OOD performance

Model	CDE			ABMP			BCMQ			BDEQ			HLMPQ		
	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑	S/SS↑	Sim↑	RI↑
General Purpose LLMs (Zero- & Few-Shot)															
Llama	1.3 / 0.5	**0.70**	19.8	1.9 / 0.6	**0.67**	1.83	4.1 / 1.2	0.66	0.38	0.7 / 0.0	0.68	8.9	2.2 / 0.1	**0.66**	0.75
Mistral	1.0 / 0.4	0.69	17.4	0.9 / 0.2	**0.67**	1.77	2.7 / 0.6	**0.68**	0.31	0.1 / 0.0	0.64	11.2	0.9 / 0.1	0.65	1.33
Llama (1-shot)	2.2 / 0.6	0.64	22.1	4.1 / 0.5	0.62	1.66	4.9 / 2.1	0.60	0.40	2.5 / 0.7	0.60	9.1	4.2 / 1.4	0.63	1.21
Mistral (1-shot)	3.1 / 1.2	0.64	29.7	4.5 / 0.8	0.61	2.01	6.8 / 2.7	0.60	0.43	3.0 / 0.9	0.61	12.5	4.1 / 1.3	0.63	1.57
GPT-5o (1-shot)	3.4 / 0.6	0.61	21.3	5.7 / 2.7	0.61	2.45	9.5 / 3.7	0.62	0.42	3.1 / 0.4	0.64	13.9	6.4 / 4.1	0.59	1.01
GPT-5o (2-shot)	4.6 / 1.3	0.60	33.4	11.2 / 4.8	0.61	2.71	10.5 / 2.9	0.63	0.41	**4.3 / 1.7**	0.58	13.7	8.0 / 3.2	0.57	1.15
Expert Models & SFT Baselines															
REINVENT-4	1.2 / 0.1	0.61	11.6	1.8 / 0.2	0.63	1.30	2.3 / 0.7	0.62	0.41	0.5 / 0.0	0.59	9.2	2.7 / 1.6	0.60	0.95
MolT5-Large (Base)	0.4 / 0.0	**0.70**	9.3	0.3 / 0.0	0.64	0.99	1.1 / 0.1	0.61	0.35	0.1 / 0.0	0.62	7.5	0.5 / 0.2	0.60	1.12
MolT5-Large (SFT)	1.0 / 0.2	0.59	14.2	23.0 / 8.4	0.60	1.85	17.2 / 7.8	0.62	0.46	0.4 / 0.2	**0.70**	10.17	11.4 / 1.6	0.58	**2.13**
GeLLM^{4}O-C$_{\text{Llama}}$	3.0 / 0.8	0.60	18.0	52.6 / 23.2	0.56	1.79	38.8 / 20.0	0.56	0.52	1.4 / 0.6	0.59	10.9	30.2 / 6.4	0.54	1.31
GeLLM^{4}O-C$_{\text{Mistral}}$	4.8 / 1.4	0.57	**120.1**	45.4 / 22.8	0.60	3.03	40.8 / 18.6	0.59	0.51	1.4 / 0.2	0.64	11.2	29.6 / 5.8	0.58	1.49
RL Post-Training (C-MORAL, Ours)															
GRPO Method															
C-MORAL$_{\text{MolT5}}$	2.3 / 0.4	0.63	6.99	41.0 / 19.6	0.59	2.62	26.0 / 10.4	0.62	0.47	0.4 / 0.1	0.59	10.93	26.4 / 8.0	0.60	1.92
C-MORAL$_{\text{Llama}}$	4.6 / 1.8	0.61	31.8	**76.2 / 49.8**	0.60	3.15	49.8 / 28.2	0.58	0.53	3.0 / 1.6	0.63	15.1	52.6 / 16.8	0.58	1.92
C-MORAL$_{\text{Mistral}}$	**7.6 / 2.6**	0.58	78.7	71.8 / 47.6	0.59	**3.67**	57.8 / 34.2	0.60	0.55	3.4 / 1.2	0.63	12.7	**57.0 / 18.2**	0.59	1.92
GDPO Method															
C-MORAL$_{\text{MolT5}}$	2.4 / 0.8	0.58	17.1	34.8 / 16.0	0.61	2.31	22.2 / 10.0	0.61	**2.51**	0.8 / 0.4	0.58	2.57	24.0 / 4.8	0.60	1.55
C-MORAL$_{\text{Llama}}$	4.5 / 2.0	0.59	72.4	73.1 / 47.7	0.58	3.24	53.3 / 33.9	0.58	0.54	2.9 / 1.5	0.62	**17.4**	44.3 / 12.3	0.58	1.91
C-MORAL$_{\text{Mistral}}$	6.4 / 2.4	0.57	114.1	72.6 / 45.8	0.58	3.40	**61.0 / 34.8**	0.58	0.55	2.8 / 1.2	0.62	15.9	48.8 / 14.8	0.59	1.90

(b) Additional OOD molecular quality metrics

Model	CDE			ABMP			BCMQ			BDEQ			HLMPQ		
	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑	Val↑	SAS↓	Nov↑
General Purpose LLMs (Zero- & Few-Shot)															
Llama	94.40	3.31	98.52	98.20	2.80	97.76	98.00	2.90	100.00	95.60	3.26	100.00	99.20	2.89	99.40
Mistral	88.40	3.03	100.00	86.20	2.85	100.00	88.80	2.84	100.00	83.60	2.96	100.00	86.40	2.90	99.54
Llama (1-shot)	99.40	3.41	91.55	99.20	2.77	96.37	99.80	2.62	88.98	99.60	3.19	83.33	99.60	2.73	87.98
Mistral (1-shot)	99.60	3.05	89.16	99.60	2.76	94.78	99.60	2.70	92.77	99.80	3.11	84.77	99.60	2.52	90.16
GPT-5o (1-shot)	94.60	3.10	98.94	93.80	2.79	98.08	90.00	2.68	100.00	92.40	2.81	98.48	93.60	2.75	98.72
GPT-5o (2-shot)	96.20	2.97	83.37	97.40	2.80	82.75	97.00	2.73	77.32	98.20	2.77	87.63	96.60	2.66	81.37
Expert Models & SFT Baselines															
REINVENT-4	100.00	2.93	100.00	99.80	2.72	99.40	100.00	3.04	99.80	99.60	3.23	100.00	100.00	2.96	100.00
MolT5-Large (Base)	99.60	2.89	99.20	100.00	2.76	100.00	100.00	2.92	99.40	99.80	2.95	98.80	100.00	2.84	99.40
MolT5-Large (SFT)	98.40	2.67	100.00	98.20	2.38	98.78	97.40	2.68	99.59	97.40	3.07	100.00	96.80	2.50	99.79
GeLLM^{4}O-C$_{\text{Llama}}$	99.20	2.75	99.40	99.80	2.32	99.20	98.80	2.55	100.00	97.00	2.96	98.76	99.60	2.32	99.40
GeLLM^{4}O-C$_{\text{Mistral}}$	100.00	2.85	100.00	99.80	2.36	99.40	100.00	2.60	98.80	99.80	3.04	98.40	99.80	2.37	99.00
RL Post-Training (C-MORAL, Ours)															
GRPO Method															
C-MORAL$_{\text{MolT5}}$	97.60	2.81	100.00	98.40	2.61	99.39	98.40	2.58	98.57	98.80	2.51	99.60	98.00	2.53	100.00
C-MORAL$_{\text{Llama}}$	100.00	2.94	99.40	100.00	2.75	99.80	100.00	2.62	99.20	99.60	2.90	100.00	99.80	2.45	99.60
C-MORAL$_{\text{Mistral}}$	100.00	2.90	99.80	100.00	3.13	100.00	100.00	2.66	98.80	100.00	2.99	99.00	100.00	2.50	99.40
GDPO Method															
C-MORAL$_{\text{MolT5}}$	98.40	2.80	99.19	97.00	2.67	99.80	98.80	2.50	100.00	99.00	2.50	99.00	96.80	2.39	99.80
C-MORAL$_{\text{Llama}}$	100.00	2.99	100.00	100.00	2.45	99.00	100.00	2.51	99.60	100.00	2.70	98.60	100.00	2.33	99.20
C-MORAL$_{\text{Mistral}}$	100.00	2.76	99.40	100.00	2.60	100.00	100.00	2.62	100.00	100.00	3.05	100.00	100.00	2.35	99.40

Table A7: OOD per-task results on C-MuMOInstruct. (a) Detailed performance comparison using S/SS, similarity, and RI. (b) Additional molecular quality metrics, including validity, synthetic accessibility score (SAS), and novelty. Val and Nov are reported in percentages; lower SAS indicates better synthetic accessibility.

```
<< SYS >>
You are an expert medicinal chemist specializing in lead optimization and
controllable molecular design. You understand how local structural modifications
affect common molecular properties, including QED, LogP/pLogP, molecular weight,
TPSA, hydrogen-bond donors/acceptors, rotatable bonds, synthetic accessibility, and
scaffold similarity. You are also familiar with key ADMET-related endpoints such as
BBB permeability, HIA, hERG inhibition, hepatotoxicity/liver injury, mutagenicity,
carcinogenicity, and receptor-related activity such as DRD2 inhibition.
<</ SYS >>

%%% [INST]
Your task is to modify the given molecule to adjust specific molecular properties
so that the resulting molecule satisfies the given target thresholds. Keep
structural changes as minimal as possible. Your response should only contain a
valid SMILES representation of the modified molecule enclosed in <SMILES> </SMILES>
tags. The property values of the new molecule should meet or exceed the specified
targets enclosed in <THRESHOLD> </THRESHOLD> tags.

Examples:
%%% Input : <SMILES> COc1ccc(OCC(O)CN2CCC(Nc3nc4ccccc4n3Cc3ccc(F)cc3)CC2)cc1
</SMILES>
%%% Adjustmments:
increase human intestinal adsorption ability with a goal of at least <THRESHOLD>
0.9 </THRESHOLD>,
decrease potential to cause liver disease with a goal of at most <THRESHOLD> 0.34
</THRESHOLD>,
decrease probability to induce genetic alterations (mutagenicity) with a goal of at
most <THRESHOLD> 0.25 </THRESHOLD>,
increase Penalized logP which is logP penalized by synthetic accessibility score
and number of large rings with a goal of at least <THRESHOLD> 3.34 </THRESHOLD>,
increase drug-likeness quantified by QED score with a goal of at least <THRESHOLD>
0.46 </THRESHOLD>
%%% Response: <SMILES>Fc1ccc(Cn2c(NC3CCN(Cc4ccccc4)CC3)nc3ccccc32)cc1</SMILES>

Task:
%%% Input:
<SMILES> COc1cc(C(=O)NCC(c2ccco2)N2CCOCC2)cc(OC)c1OC </SMILES>

%%% Adjustments:
increase human intestinal adsorption ability with a goal of at least
<THRESHOLD> 0.9 </THRESHOLD>,
decrease potential to cause liver disease with a goal of at most <THRESHOLD>
0.34 </THRESHOLD>,
decrease probability to induce genetic alterations (mutagenicity) with a goal
of at most <THRESHOLD> 0.25 </THRESHOLD>,
increase Penalized logP which is logP penalized by synthetic accessibility
score and number of large rings with a goal of at least <THRESHOLD> 3.34
</THRESHOLD>, increase drug-likeness quantified by QED score with a goal of
at least <THRESHOLD> 0.46 </THRESHOLD>

%%% Response:
```

Figure A6: An example of the structured prompt template used for general-purpose LLM prompting. The prompt includes a domain-specific system instruction, a few-shot molecular editing example, and a test query consisting of a source SMILES and property-wise adjustment targets. The model is instructed to generate only a valid optimized SMILES while preserving the molecular structure as much as possible.

Artifact	Source	License / Terms	Accessibility
C-MuMOInstruct	https://huggingface.co/datasets/NingLab/MuMOInstruct	CC BY 4.0	Dataset
GeLLM^{4}O-C	https://github.com/ninglab/GeLLMO-C	Original repository / model terms	Checkpoint / code
S^2-Bench MolOpt	https://github.com/phenixace/TOMG-Bench	Original benchmark terms	Dataset / benchmark
RDKit	https://www.rdkit.org/	BSD-style license	Open-source toolkit
ADMET-AI	https://admet.ai.greenstonebio.com/	Original platform / package terms	Web interface / package
MolT5	https://github.com/blender-nlp/MolT5	Original model terms	Checkpoint
LLaMA	https://huggingface.co/meta-llama	LLaMA Community License	Checkpoint
Mistral	https://huggingface.co/mistralai	Apache 2.0	Checkpoint
GPT-4o / GPT-5o	https://openai.com/	Proprietary API terms	API
Gemini-1.5-Pro	https://ai.google.dev/	Proprietary API terms	API

Table A8: Licenses, terms of use, and accessibility of the main artifacts used in this work. We use these artifacts only for research and evaluation purposes and do not redistribute them as part of this submission.