# LLM-GUIDED DYNAMIC ACTION SPACES FOR SYNTHESIZABLE MOLECULAR OPTIMIZATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Synthesizable molecular optimization seeks to improve target properties while ensuring that molecular modifications follow feasible synthetic pathways. Existing synthesis-aware methods typically rely on exploring a large space of candidate transformations defined by reaction templates and purchasable building blocks. This search becomes even more challenging when property improvement requires multiple reaction steps, as the space expands further along the pathway. To address this challenge, we introduce MOLREACT, which reformulates molecular optimization as search over compact reaction spaces proposed by a tool-augmented large language model (LLM). At each step, the LLM combines its prior chemical knowledge with cheminformatics tools to identify a molecule-specific set of compatible reactions, preserving synthesizability while making multi-step optimization feasible. Given this compact action space, we further leverage Group Relative Policy Optimization (GRPO) with the terminal oracle reward to improve long-term decision-making over multiple reaction steps. Across diverse molecular optimization tasks, MOLREACT achieves the highest Top-10 score on 11 of 14 tasks and the best sample efficiency on 12 of 14 tasks, outperforming existing baselines under limited oracle budgets. Beyond these gains, MOLREACT also provides each optimized molecule with a template-grounded synthetic pathway.

## 1 INTRODUCTION

Molecular optimization is a central task in drug discovery that seeks to identify molecules with improved target properties (Zhang et al., 2024). However, high property scores alone are not enough for practical use, since optimized molecules must also be chemically valid and practically accessible through feasible synthetic transformations. Therefore, improving target properties while ensuring that molecules can be reliably synthesized remains a key research challenge (Papidocha et al., 2026).

Prior work on synthesis-aware molecular optimization typically searches a combinatorial reaction space built from validated templates and predefined building blocks, using generative or search-based methods to construct feasible synthetic pathways (Swanson et al., 2024; Nakamura et al., 2025; Edwards et al., 2026; Pedawi et al., 2022; Korovina et al., 2020; Zhu et al., 2023; Koziarski et al., 2024; Cretu et al., 2025; Seo et al., 2025; Sun et al., 2022). However, searching over numerous reaction templates and large building-block libraries yields an enormous number of candidate reactions. Identifying promising reactions among these candidates requires extensive oracle calls and substantial computational resources. This problem becomes even worse over multiple reaction steps, as the search space expands further at each step, making multi-step optimization more difficult in practice.

To address this search-space bottleneck, we reformulate molecular optimization as search over compact reaction spaces proposed by a tool-augmented large language model (LLM). Pretrained on extensive scientific corpora, LLMs demonstrate strong capabilities in molecular understanding and reaction reasoning. (Guo et al., 2023; Jablonka et al., 2024; Park et al., 2024; Guo et al., 2024; Fang et al., 2024). Tool augmentation can further strengthen this capability by incorporating explicit chemical and structural information from external chemistry tools (M. Bran et al., 2024; Kim et al., 2025; Huang et al., 2025). This reformulation shifts the role of the LLM from complex optimized molecule generation to targeted search-space guidance, where at each step it dynamically constructs a compact reaction space tailored to the current molecule, making multi-step optimization feasible.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

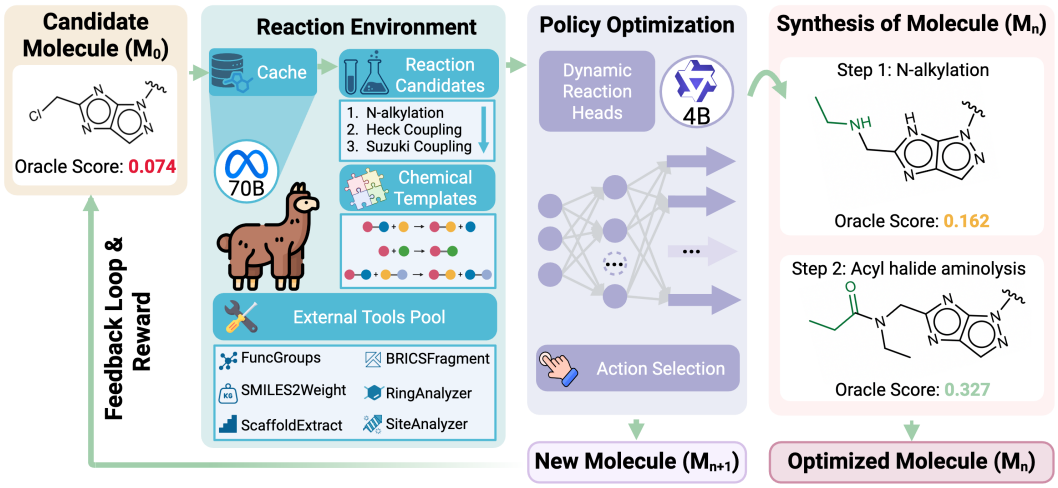



Figure 1: Overview of MolReAct. The reaction environment performs template matching and tool-augmented analysis to propose feasible candidates. The policy selects among candidates or stops the trajectory. The right panel shows a two-step optimization via N-alkylation and acyl halide aminolysis.

Building on this idea, we propose MOLREACT, a synthesis-aware framework for multi-step molecular optimization from a given starting molecule. At each step, substructure matching identifies reaction templates compatible with the current molecule. The LLM then combines its chemical knowledge with specialized cheminformatics tools to select promising reactions. Instantiating them with purchasable building blocks from the Enamine catalog (Enamine, 2026) yields a compact and synthesizable reaction space. However, reaching the target properties usually takes several reaction steps, so effective optimization requires planning across the entire pathway. We therefore deploy a policy model optimized with Group Relative Policy Optimization (GRPO), using the terminal oracle reward to evaluate the long-term impact of each reaction across the pathway for multi-step optimization. To keep this optimization efficient, we cache LLM proposals for previously seen molecules, avoiding repeated inference while maintaining a comparable computational cost. Across TDC benchmarks (Huang et al., 2021) and a proxy-based binding affinity task (Bengio et al., 2021), MOLREACT achieves the highest Top-10 score on 11 out of 14 tasks and the best sample efficiency on 12 out of 14 tasks, outperforming existing baselines under limited oracle budgets. Further analysis shows that tool-guided proposals improve candidate quality at each step, which in turn enables more effective policy optimization across multiple steps. By constructing a dynamic molecule-specific action space, MOLREACT provides each optimized molecule with a template-grounded pathway.

## 2 RELATED WORK

### 2.1 SYNTHESIZABLE MOLECULAR OPTIMIZATION

Molecular optimization often prioritizes target properties without enforcing synthetic feasibility. Recent approaches address this through two main strategies. The first restricts the search space using predefined building block libraries and validated reaction templates. Algorithms like variational autoencoders (Pedawi et al., 2022), Bayesian optimization (Korovina et al., 2020), Monte Carlo tree search (Swanson et al., 2024; Nakamura et al., 2025; Sun et al., 2022; Wang et al., 2024), and GFlowNets (Zhu et al., 2023; Koziarski et al., 2024; Cretu et al., 2025; Seo et al., 2025) navigate this space to construct valid reaction sequences. However, exploring these extensive reaction networks without directional guidance demands massive computational resources and heavy oracle evaluations. An alternative strategy projects given molecules into the synthesizable space to suggest valid analogs. These approaches generally learn this mapping by training separate projection models (Gao et al., 2025; Lee et al., 2026; Sun et al., 2025b; Luo et al., 2024), often integrated with genetic algorithms or by fine-tuning large language models directly (Edwards et al., 2026; Sun et al., 2025a). Although these methods produce synthetically feasible analogs, it incurs additional training overhead. Rather than enumerating predefined reactions or training separate projection models, MolReAct performs

molecule-conditioned optimization from a given starting molecule, using a tool-augmented LLM to dynamically construct a compact chemical action space explicitly tailored to its chemical context.

### 2.2 Tool-Augmented LLM for Chemical Reasoning

Large language models have shown strong capabilities in chemical reasoning and property interpretation (Guo et al., 2023; Jablonka et al., 2024; Park et al., 2024; Guo et al., 2024; Fang et al., 2024). However, when applied to generative molecular design, their internal knowledge alone often yields chemically invalid or implausible structures (Tao et al., 2025). To address this limitation, recent works augment LLMs with external chemistry tools to ground the generation process in explicit physical rules, such as reaction planners, safety checkers, and molecular databases (M. Bran et al., 2024; Kim et al., 2025; Huang et al., 2025). While these tool-augmented systems significantly improve the chemical validity of the outputs, they are primarily designed for executing predefined workflows, answering chemical queries, or performing single-step retrospective synthesis. In practice, these systems often struggle with long-term decision-making during multi-step optimization (Wang et al., 2025; Liu et al., 2025). Motivated by this, MolReAct combines the chemical reasoning of a tool-augmented LLM with trajectory-level decision-making through reinforcement learning. Guided by the LLM, MolReAct dynamically constructs a compact synthesis-constrained action space at each step, enabling the policy model to more effectively optimize over multi-step reaction pathways.

## 3 Methodology

### 3.1 Problem Formulation

We formulate synthesizable molecular optimization as a Markov Decision Process (MDP), $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$. At step $t$, the state $s_t \in \mathcal{S}$ is the chemically valid molecule. An action $a_t \in \mathcal{A}(s_t)$ either specifies a reaction through a template and purchasable building block or stops the optimization. The transition is deterministic, with $\mathcal{P}(s_{t+1} \mid s_t, a_t)$ determined by executing the selected reaction. An episode terminates when the policy selects the stop action $a_{\text{stop}}$, reaches the maximum depth $T_{\max}$, or no reaction template matches the current molecule. The oracle evaluates the final molecule $s_T$, giving the corresponding terminal reward $r_T = \text{Oracle}(s_T)$. The policy $\pi_\theta(a_t \mid s_t)$ is then trained to maximize the expected terminal reward, $J(\pi_\theta) = \mathbb{E}\pi\theta[r_T]$. Since every reaction action comes from a validated template, each optimization trajectory defines an explicit template-grounded pathway.

### 3.2 LLM Agent as a Dynamic Reaction Environment

We construct an action space tailored to each molecule through a two-stage process. The first stage grounds all actions in validated reaction templates to enforce template validity. The second leverages LLM chemical knowledge to select relevant transformations and instantiate concrete building blocks.

**Template Matching.** Following Gao et al. (2025) and Lee et al. (2026), we adopt a curated set of 115 validated reaction templates that include uni-, bi-, and tri-molecular reactions. Given the current molecule $s_t$ represented as a SMILES string, we perform substructure matching against the reactant patterns in each template. Only templates whose reactant patterns are present in $s_t$ are retained, yielding a matched subset $\mathcal{T}_{\text{match}}(s_t) \subseteq \mathcal{T}$. If no template matches, the trajectory terminates early.

**LLM-Guided Action Space.** The agent receives a task-specific natural-language description of the optimization objective (Appendix Table 10). Given $\mathcal{T}_{\text{match}}(s_t)$, the tool-augmented LLM analyzes the current molecule, selects the most chemically relevant templates, and proposes candidate reactions by specifying the building block required for each selected template. The agent operates within the ReAct framework (Yao et al., 2023), interleaving reasoning with invocations of specialized chemical analysis tools (Table 11). These tools provide information about functional groups, scaffolds, ring systems, BRICS fragments, reactive sites, and molecular weight, enabling the agent to understand the molecular structure, select appropriate templates, and propose chemically grounded building blocks.

Before execution, we canonicalize each additional reactant specified by the LLM and retain a proposal only if every required building block exactly matches an entry in the Enamine catalog of building blocks (Enamine, 2026). Reactions without an additional reactant are retained. The remaining proposals are executed via their templates to produce candidate products, and proposals that fail execution or yield invalid products are discarded. The resulting filtered action space at state $s_t$ is

$$\mathcal{A}(s_t) = \left\{ a \in \mathrm{LLM}_\phi\big(\mathrm{Tools}(s_t),\ \mathcal{T}_{\mathrm{match}}(s_t),\ \mathrm{obj}\big) : \mathrm{Catalog}(a) \wedge \mathrm{Valid}(a) \right\} \cup \{a_{\mathrm{stop}}\}, \quad (1)$$

where $\mathrm{Catalog}(a)$ denotes the Enamine match for all required additional reactants and $\mathrm{Valid}(a)$ denotes successful template execution with a valid product. The $\mathrm{obj}$ defines the specific task objective. The stop action $a_{\mathrm{stop}}$ immediately terminates the trajectory. Proposed reactions and the stop action jointly define a dynamic action space tailored to the current molecule at each optimization step.

**Reaction Caching.** To reduce redundant LLM invocations during RL exploration, we cache the action space $\mathcal{A}(s_t)$ and transition outcomes for each previously encountered molecule, keyed by SMILES within each task. During training, the environment queries this cache before invoking the LLM agent, reusing the same action space and transition outcomes after the first query. On a cache hit, reactions and pre-computed products are retrieved instantly. This significantly accelerates GRPO training by amortizing the LLM inference cost across repeated visits to the same molecular states.

### 3.3 Trajectory Optimization via Reinforcement Learning

Given the synthesis-constrained action space from Section 3.2, We further train a policy model to select candidate reactions over multi-step trajectories, maximizing the terminal oracle reward.

**Policy Architecture.** The policy model is a Qwen-3-4B language model with a linear action head. At each step $t$, it reads a single prompt containing the current molecule SMILES $s_t$ and all candidate reactions, each numbered and described by its template name, reactant SMILES, and product SMILES. Encoding the molecule and every candidate together in one forward pass lets self-attention compare them directly and link each number to the reaction it labels. The action head then reads the final hidden state at each numbered candidate position and outputs a single logit for it. Through this process, the policy learns to dynamically score the proposed reactions. Unused slots are masked with $-\infty$ before the softmax, giving a distribution $\pi_\theta(a_t \mid s_t)$ over the current valid actions.

**Group Relative Policy Optimization.** For each initial molecule, the policy samples a group of $G$ independent trajectories. Each trajectory receives a single terminal reward $r_T = \mathrm{Oracle}(s_T)$, which is standardized within the group by subtracting the group mean and dividing by the group standard deviation to obtain a group-relative advantage. This shared advantage is then assigned to every step in the corresponding trajectory, enabling trajectory-level credit assignment. The policy is then updated by maximizing the standard clipped surrogate policy objective with additional KL regularization

$$\mathcal{L}_{\mathrm{GRPO}}(\theta) = \mathbb{E}\left[\min(\rho_t(\theta)A, \mathrm{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)A)\right] - \beta\, \mathbb{D}_{\mathrm{KL}}(\pi_\theta(\cdot \mid s_t) \parallel \pi_{\mathrm{ref}}(\cdot \mid s_t)), \quad (2)$$

where $\rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t|s_t)}$ is the importance sampling ratio with $\pi_{\theta_{\mathrm{old}}}$ the behavior policy at rollout, $\pi_{\mathrm{ref}}$ the fixed initial policy used as a KL anchor, $\epsilon$ the clipping threshold, and $\beta$ the KL penalty strength preventing catastrophic deviation from $\pi_{\mathrm{ref}}$. An entropy bonus on $\pi_\theta$ is additionally applied during training to encourage exploration (Appendix A). The KL divergence is computed over valid actions at each state, naturally accommodating variable-size action spaces across different states.

## 4 Experiments

### 4.1 Experimental Setup

MolReAct is evaluated on 14 molecular optimization tasks against both reaction-driven and LLM-based baselines under a fixed oracle budget. All methods share the same held-out starting molecules and reaction template library, so performance differences primarily reflect their search strategies.

Table 1: Top-10 performance on 14 molecular optimization tasks. Best in **bold**, second-best underlined. ✓: synthesis-constrained methods. All methods report mean±sd over 3 runs.

| | *Initial* | Graph GA | ReaSyn | SynFormer | DrugAssist | LDMOL | mCLM | **MolReAct** |
|---|---|---|---|---|---|---|---|---|
| Synthesizable | – | × | ✓ | ✓ | × | × | ✓ | ✓ |
| amlodipine_mpo | 0.389 | **0.563**$_{\pm 0.076}$ | 0.438$_{\pm 0.005}$ | 0.452$_{\pm 0.006}$ | 0.428$_{\pm 0.008}$ | 0.402$_{\pm 0.008}$ | 0.387$_{\pm 0.002}$ | 0.498$_{\pm 0.013}$ |
| celecoxib_rediscovery | 0.182 | 0.351$_{\pm 0.247}$ | 0.224$_{\pm 0.005}$ | 0.370$_{\pm 0.005}$ | 0.200$_{\pm 0.003}$ | 0.198$_{\pm 0.007}$ | 0.213$_{\pm 0.004}$ | **0.393**$_{\pm 0.021}$ |
| DRD2 | 0.599 | 0.808$_{\pm 0.024}$ | **0.996**$_{\pm 0.003}$ | **0.996**$_{\pm 0.004}$ | 0.679$_{\pm 0.006}$ | 0.651$_{\pm 0.009}$ | 0.236$_{\pm 0.007}$ | 0.987$_{\pm 0.009}$ |
| fexofenadine_mpo | 0.427 | 0.713$_{\pm 0.035}$ | 0.684$_{\pm 0.005}$ | 0.760$_{\pm 0.075}$ | 0.589$_{\pm 0.002}$ | 0.535$_{\pm 0.005}$ | 0.545$_{\pm 0.006}$ | **0.787**$_{\pm 0.031}$ |
| GSK3_beta | 0.397 | 0.673$_{\pm 0.035}$ | 0.660$_{\pm 0.005}$ | 0.610$_{\pm 0.005}$ | 0.319$_{\pm 0.045}$ | 0.396$_{\pm 0.008}$ | 0.173$_{\pm 0.006}$ | **0.722**$_{\pm 0.028}$ |
| JNK3 | 0.269 | 0.422$_{\pm 0.178}$ | 0.470$_{\pm 0.003}$ | 0.470$_{\pm 0.006}$ | 0.243$_{\pm 0.003}$ | 0.283$_{\pm 0.010}$ | 0.078$_{\pm 0.002}$ | **0.541**$_{\pm 0.038}$ |
| median_1 | 0.100 | 0.214$_{\pm 0.028}$ | 0.175$_{\pm 0.008}$ | 0.234$_{\pm 0.003}$ | 0.102$_{\pm 0.005}$ | 0.108$_{\pm 0.008}$ | 0.125$_{\pm 0.008}$ | **0.253**$_{\pm 0.013}$ |
| median_2 | 0.126 | 0.141$_{\pm 0.003}$ | 0.166$_{\pm 0.002}$ | 0.151$_{\pm 0.007}$ | 0.135$_{\pm 0.002}$ | 0.139$_{\pm 0.009}$ | 0.141$_{\pm 0.009}$ | **0.186**$_{\pm 0.012}$ |
| osimertinib_mpo | 0.306 | 0.743$_{\pm 0.011}$ | 0.731$_{\pm 0.009}$ | 0.819$_{\pm 0.008}$ | 0.731$_{\pm 0.002}$ | 0.553$_{\pm 0.006}$ | 0.703$_{\pm 0.002}$ | **0.825**$_{\pm 0.020}$ |
| perindopril_mpo | 0.305 | 0.435$_{\pm 0.006}$ | 0.381$_{\pm 0.008}$ | 0.460$_{\pm 0.009}$ | 0.363$_{\pm 0.006}$ | 0.321$_{\pm 0.002}$ | 0.360$_{\pm 0.004}$ | **0.487**$_{\pm 0.014}$ |
| ranolazine_mpo | 0.081 | 0.645$_{\pm 0.034}$ | 0.657$_{\pm 0.012}$ | 0.795$_{\pm 0.011}$ | 0.458$_{\pm 0.012}$ | 0.170$_{\pm 0.012}$ | 0.195$_{\pm 0.011}$ | **0.812**$_{\pm 0.016}$ |
| sEH | 0.716 | 0.909$_{\pm 0.005}$ | **0.976**$_{\pm 0.005}$ | 0.918$_{\pm 0.058}$ | 0.652$_{\pm 0.030}$ | 0.724$_{\pm 0.007}$ | 0.654$_{\pm 0.006}$ | 0.934$_{\pm 0.013}$ |
| sitagliptin_mpo | 0.007 | 0.329$_{\pm 0.070}$ | 0.283$_{\pm 0.012}$ | 0.295$_{\pm 0.024}$ | 0.002$_{\pm 0.002}$ | 0.014$_{\pm 0.006}$ | 0.044$_{\pm 0.004}$ | **0.391**$_{\pm 0.017}$ |
| zaleplon_mpo | 0.133 | 0.393$_{\pm 0.023}$ | 0.326$_{\pm 0.006}$ | 0.429$_{\pm 0.010}$ | 0.236$_{\pm 0.011}$ | 0.217$_{\pm 0.005}$ | 0.347$_{\pm 0.008}$ | **0.476**$_{\pm 0.016}$ |
| Average score | 0.288 | 0.524$_{\pm 0.035}$ | 0.512$_{\pm 0.003}$ | 0.554$_{\pm 0.005}$ | 0.367$_{\pm 0.009}$ | 0.337$_{\pm 0.006}$ | 0.300$_{\pm 0.006}$ | **0.592**$_{\pm 0.014}$ |
| Rank (Gain %) | – | 3 (+82) | 4 (+78) | 2 (+92) | 5 (+27) | 6 (+17) | 7 (+4) | **1** (+106) |

**Tasks and Evaluation Metrics.** Following Gao et al. (2025) and Lee et al. (2026), we evaluate MolReAct on 13 property optimization tasks from the Therapeutic Data Commons (TDC) (Huang et al., 2021) and one proxy-based binding affinity task for soluble Epoxide Hydrolase (sEH) (Bengio et al., 2021). We adopt the molecule-conditioned setting (Sun et al., 2022), where each method improves the target property of a given starting molecule. All methods start from the same 100 held-out molecules and share the same 10,000-call oracle budget (Gao et al., 2025), which counts every query, including those made during policy training (Appendix A.5). All synthesizable methods also share the same 115-template reaction library (Lee et al., 2026). We report the *Top-10 score* over all evaluated molecules and compute its Area Under the Curve ($AUC_{10}$) as a function of oracle calls to evaluate sample efficiency. We also report the Top-10 score of the starting molecules and each method's relative gain over it for comparison.

Table 2: $AUC_{10}$ sample efficiency comparison across all 14 optimization tasks. Best in **bold**, second-best underlined. All methods report mean±sd over 3 runs.

| Property | Graph GA | ReaSyn | SynFormer | **MolReAct** |
|---|---|---|---|---|
| amlodipine_mpo | 0.423$_{\pm 0.050}$ | 0.432$_{\pm 0.015}$ | 0.445$_{\pm 0.011}$ | **0.471**$_{\pm 0.017}$ |
| celecoxib_rediscovery | 0.312$_{\pm 0.165}$ | 0.222$_{\pm 0.006}$ | 0.344$_{\pm 0.021}$ | **0.387**$_{\pm 0.012}$ |
| DRD2 | 0.781$_{\pm 0.114}$ | 0.989$_{\pm 0.010}$ | **0.993**$_{\pm 0.005}$ | 0.980$_{\pm 0.007}$ |
| fexofenadine_mpo | 0.693$_{\pm 0.002}$ | 0.669$_{\pm 0.008}$ | 0.716$_{\pm 0.059}$ | **0.732**$_{\pm 0.019}$ |
| GSK3_beta | 0.598$_{\pm 0.012}$ | 0.649$_{\pm 0.107}$ | 0.605$_{\pm 0.093}$ | **0.684**$_{\pm 0.024}$ |
| JNK3 | 0.418$_{\pm 0.009}$ | 0.454$_{\pm 0.122}$ | 0.466$_{\pm 0.091}$ | **0.524**$_{\pm 0.031}$ |
| median_1 | 0.189$_{\pm 0.006}$ | 0.172$_{\pm 0.005}$ | 0.221$_{\pm 0.026}$ | **0.231**$_{\pm 0.023}$ |
| median_2 | 0.125$_{\pm 0.005}$ | 0.164$_{\pm 0.017}$ | 0.149$_{\pm 0.004}$ | **0.171**$_{\pm 0.011}$ |
| osimertinib_mpo | 0.719$_{\pm 0.004}$ | 0.726$_{\pm 0.021}$ | 0.791$_{\pm 0.011}$ | **0.810**$_{\pm 0.010}$ |
| perindopril_mpo | 0.398$_{\pm 0.011}$ | 0.378$_{\pm 0.016}$ | 0.439$_{\pm 0.010}$ | **0.475**$_{\pm 0.015}$ |
| ranolazine_mpo | 0.612$_{\pm 0.002}$ | 0.606$_{\pm 0.030}$ | 0.746$_{\pm 0.010}$ | **0.781**$_{\pm 0.016}$ |
| sEH | 0.877$_{\pm 0.031}$ | **0.915**$_{\pm 0.043}$ | 0.909$_{\pm 0.030}$ | 0.910$_{\pm 0.014}$ |
| sitagliptin_mpo | 0.310$_{\pm 0.014}$ | 0.281$_{\pm 0.086}$ | 0.281$_{\pm 0.054}$ | **0.355**$_{\pm 0.011}$ |
| zaleplon_mpo | 0.315$_{\pm 0.007}$ | 0.322$_{\pm 0.042}$ | 0.410$_{\pm 0.022}$ | **0.447**$_{\pm 0.022}$ |
| Average score | 0.484$_{\pm 0.010}$ | 0.499$_{\pm 0.025}$ | 0.537$_{\pm 0.009}$ | **0.568**$_{\pm 0.008}$ |

**Dataset and Starting Molecule Selection.** Starting molecules are drawn from ZINC-250K (Irwin et al., 2012) following the filtering protocol of Sun et al. (2022), which we instantiate per task as follows. We retain molecules with QED above 0.6 to ensure drug-likeness. For protein-target activity tasks (JNK3, DRD2, GSK3$\beta$, and sEH), starting molecules are restricted to oracle scores between 0.1 and 0.8, while for the remaining tasks, they are selected from the 60th–80th percentile of the corresponding property distribution. These criteria maintain initial molecular quality while preserving the potential for meaningful property improvement. From the filtered pool, we randomly sample 1,000 molecules for policy training and 100 held-out molecules as starting molecules for evaluation.

**Model and Training Configuration.** Our framework involves two language models serving distinct roles. A Llama-3.3-70B model (Grattafiori et al., 2024), deployed via the vLLM engine (Kwon et al., 2023), operates within the ReAct framework as the environment agent, leveraging chemical

tools to analyze molecular structures and propose template-grounded candidate reactions at each step. We retain those proposals with building blocks available in the Enamine catalog (Enamine, 2026) before reaction execution. The policy agent, a Qwen-3-4B model (Yang et al., 2025), is the sole trainable component, optimized via GRPO to learn reaction selection strategies that maximize long-term reward across multi-step trajectories. During training, we impose a maximum depth of 5 steps to prioritize synthetic efficiency and reduce side reactions from excessively long pathways.

**Baselines.** We compare against two categories of baselines. The first includes reaction-driven methods such as *GraphGA* (Jensen, 2019), *ReaSyn* (Lee et al., 2026), and *SynFormer* (Gao et al., 2025). *ReaSyn* and *SynFormer* project a molecule into synthesizable space to suggest analogs. The second category includes LLM-based frameworks such as *DrugAssist* (Ye et al., 2025) and *LDMol* (Chang & Ye, 2025), which generate modified molecules from natural language instructions, and *mCLM* (Edwards et al., 2026), which further prioritizes synthesizability. These LLM-based baselines use the same task-specific objective descriptions and initial molecules as MolReAct.

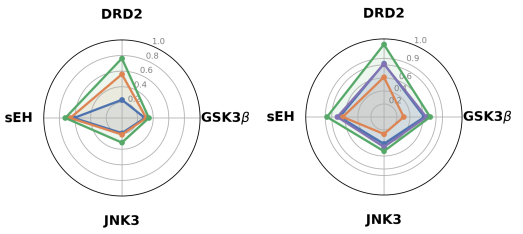
## 4.2 Results and Analysis

### 4.2.1 Comparative Performance

Table 1 reports the Top-10 scores across all 14 tasks. MolReAct achieves the highest average score of 0.592 and the largest relative gain over the starting Top-10 scores at 106%. Per-start analysis (Appendix A.7) shows that improvements occur for a substantial fraction of held-out molecules (43–63%) and that the optimized molecules remain structurally related to their starts (with Tanimoto similarity of 0.49–0.60). It performs best on 11 tasks and significantly outperforms all six baselines (Appendix D, $p < 0.05$). Table 2 reports $AUC_{10}$, which measures optimization quality over cumulative oracle calls. MolReAct achieves the best sample efficiency on 12 tasks, with an overall average $AUC_{10}$ of 0.568, indicating that MolReAct consistently finds high-scoring molecules.

Table 3: Molecular quality of the top-100 optimized molecules, averaged over the 14 tasks (mean±sd). Best in **bold**, second-best underlined.

| Method | IntDiv↑ | SA↓ | AiZynth↑ | QED↑ | MW↓ |
|---|---|---|---|---|---|
| *Initial* | $0.870_{\pm 0.007}$ | $2.98_{\pm 0.21}$ | $0.574_{\pm 0.107}$ | $0.782_{\pm 0.020}$ | $333_{\pm 8.1}$ |
| Graph GA | $0.518_{\pm 0.037}$ | $3.24_{\pm 0.21}$ | $0.255_{\pm 0.180}$ | $0.525_{\pm 0.029}$ | $376_{\pm 22.7}$ |
| ReaSyn | $0.576_{\pm 0.024}$ | $\mathbf{2.81}_{\pm 0.14}$ | $0.568_{\pm 0.331}$ | $0.668_{\pm 0.046}$ | $347_{\pm 15.4}$ |
| SynFormer | $0.631_{\pm 0.045}$ | $\underline{2.84}_{\pm 0.19}$ | $\mathbf{0.612}_{\pm 0.287}$ | $0.636_{\pm 0.022}$ | $393_{\pm 28.9}$ |
| DrugAssist | $0.873_{\pm 0.018}$ | $3.26_{\pm 0.25}$ | $0.433_{\pm 0.090}$ | $0.597_{\pm 0.038}$ | $389_{\pm 18.1}$ |
| LDMOL | $\underline{0.875}_{\pm 0.032}$ | $3.08_{\pm 0.15}$ | $0.506_{\pm 0.083}$ | $\mathbf{0.744}_{\pm 0.051}$ | $340_{\pm 25.6}$ |
| mCLM | $0.848_{\pm 0.049}$ | $3.29_{\pm 0.23}$ | $0.516_{\pm 0.080}$ | $0.635_{\pm 0.031}$ | $\underline{325}_{\pm 12.8}$ |
| **MolReAct** | $\mathbf{0.881}_{\pm 0.027}$ | $2.94_{\pm 0.17}$ | $\underline{0.581}_{\pm 0.073}$ | $\underline{0.673}_{\pm 0.043}$ | $\mathbf{321}_{\pm 30.3}$ |




(a) Reaction proposal: Random Template, Vanilla LLM, Tool-LLM

(b) Multi-step optimization: Greedy Search, Random Policy, MCTS, MolReAct

Figure 2: Ablation on tool-guided proposal and policy optimization across target activity tasks.

Table 3 evaluates the overall quality of the top-100 optimized molecules from each task in terms of internal diversity (IntDiv), synthetic accessibility (SA), AiZynthFinder retrosynthesis success rate (AiZynth), drug-likeness (QED), and molecular weight (MW). MolReAct achieves the highest IntDiv and the lowest MW among all methods, while preserving synthesizability, with SA and AiZynth remaining comparable to the starting molecules across tasks. Despite a slight decrease in QED, MolReAct remains competitive with the other methods. Overall, these results indicate that MolReAct consistently improves the target properties across diverse optimization tasks under a limited oracle budget, without relying on molecular-size inflation or substantially compromising diversity and synthetic accessibility.

### 4.2.2 Reaction Proposal and Policy Analysis

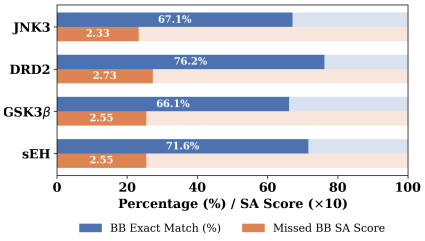
We conduct a progressive ablation to isolate the individual contributions of tool augmentation and RL-based policy optimization. We focus on the four protein-target activity tasks (JNK3, DRD2, GSK3$\beta$, sEH), as their well-defined biological targets and continuous activity scores provide the clearest and most interpretable signal for distinguishing the effect of each component. Results are visualized in Figure 2, with exact numerical values for all variants provided in Table 6 in the appendix.

**Tool-Guided Reaction Proposal.** We first evaluate the quality of single-step reaction proposals. For each starting molecule, the LLM proposes candidate reactions, and their products are then evaluated by the oracle. We report the average Top-10 score to assess proposal quality. Three different proposal variants are compared. **(i) Random Template**, which randomly samples structurally matched reaction templates and compatible building blocks; **(ii) Vanilla LLM**, which selects reactions based on the model's internal chemical knowledge for the target property; and **(iii) Tool LLM**, which equips the LLM with domain-specific analytical tools (*e.g.*, reactive site analysis, scaffold extraction) together with task-specific objective descriptions. Figure 2(a) shows that Vanilla LLM outperforms Random Template, while tool augmentation provides further gains and leads to the strongest overall performance for Tool LLM. We additionally run an **anonymized Tool LLM** that does not receive the objective description, which performs comparably to the Tool LLM across all evaluated tasks (Appendix Section A.4), indicating that proposal quality is driven primarily by the LLM's internal chemical knowledge and analytical tools rather than by any potential leakage from task information.

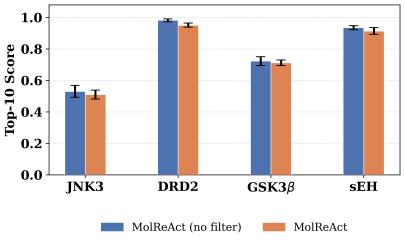
**Policy-Guided Multi-Step Optimization.** We then evaluate optimization strategies over multi-step trajectories. Fixing the reaction proposer as Tool LLM, we compare four policies: **(i) Random Policy**, which randomly selects a reaction from the proposed candidates at each step; **(ii) Greedy Search**, which selects the candidate with the largest local improvement; **(iii) Monte Carlo Tree Search**, which plans over the action space using UCT (Appendix C); and **(iv) MolReAct (GRPO)**, the full framework with a trained policy model. Figure 2(b) shows that Greedy Search outperforms Random Policy, while tending to get trapped in local optima. MCTS improves over Greedy Search on most tasks through multi-step planning. MolReAct performs best overall. These results show that the learned policy provides additional benefits beyond the action space by enabling long-term decision-making. Building on this policy-level analysis, we further evaluate an **anonymized MolReAct** at the end-to-end framework level, where objective information is withheld throughout the pipeline. Its overall performance remains comparable to the standard framework across all evaluated tasks (Appendix Section A.4), showing that MolReAct remains effective without explicit task descriptions.

### 4.2.3 Impact of Purchasability Constraints

MolReAct restricts its action space to purchasable building blocks throughout training and evaluation. A building block is retained only if its canonical SMILES exactly matches an entry in the Enamine catalog (Enamine, 2026). This constraint is not overly restrictive in practice, as the LLM already proposes mostly purchasable reagents. As shown in Figure 3a, 66.1–76.2% of the unique proposed building blocks are directly available in the catalog. The unmatched candidate building blocks also remain synthetically accessible, with SA scores ranging from 2.33 to 2.73 (all below 3). Because exact matching does not capture close analogs, salt forms, or availability from other suppliers, the actual availability is likely higher.




(a) Availability before catalog filtering.




(b) Top-10 with/without catalog filtering.

Figure 3: Effect of commercial filtering.

To quantify the performance cost introduced by this constraint, we compare MolReAct with an otherwise identical variant trained without the purchasability filter on the four protein-target tasks. Enforcing purchasability throughout training and evaluation changes the overall Top-10 score by only 1.4–3.8% (Figure 3b), which falls within observed run-to-run variation on three of the four targets. Thus, restricting the action space to purchasable building blocks preserves nearly all overall optimization performance while ensuring that every selected reaction uses a commercially available reagent.

### 4.2.4 Scaling of Environment and Policy Models

MolReAct contains two components, the environment LLM and policy model. We vary each while holding the other fixed, asking how much capacity each requires on four protein-target activity tasks.

**Environment LLM.** Since the environment LLM defines the action space, we first examine how model scale affects single-step proposal quality. Using the reaction-proposal protocol of Section 4.2.2 on the four protein-target tasks, we compare four Llama variants and report the Top-10 product score, coverage (the fraction of molecules with at least one valid reaction), and amortized GPU-seconds per molecule (Table 4a). Both metrics improve substantially from 3B to 70B, with coverage more than doubling from 34% to 79%. The 405B variant yields only a modest Top-10 gain and slightly lower coverage, suggesting that it is more selective and more likely to refrain when no confident transformation is available. Considering its roughly $6\times$ higher computational cost in practice, these results make 70B the best trade-off for our framework and motivate our choice of environment LLM.

**Policy Model.** We next study the contribution of the learned policy model by fixing the environment agent (Tool-LLM) and varying the model used to select among candidate reactions (Table 4b). To first assess the benefit of using a language model as the policy, we replace it with an MLP (Appendix A.6). The ECFP4+MLP policy performs substantially worse than Qwen3-4B in both Top-10 and $\text{AUC}_{10}$, indicating that pretrained chemical representations improve effective downstream reaction selection and sample efficiency under the same action space. We then vary the size of the language-model policy. Performance improves steadily from Qwen3-0.6B to Qwen3-4B and then largely saturates across tasks, with Qwen3-8B providing little additional gain while more than doubling the training time. These results suggest that the policy benefits from a pretrained language model capable of interpreting SMILES and reaction-template descriptions, with Qwen3-4B offering the best trade-off between performance and computational cost.

(a) Environment LLM.

| Environment LLM | Top-10 | Cov. | GPU-s |
|---|---|---|---|
| Llama-3.2-3B | 0.488 | 34% | 6 |
| Llama-3.1-8B | 0.515 | 35% | 21 |
| **Llama-3.3-70B** | **0.538** | **79%** | **201** |
| Llama-3.1-405B | 0.575 | 74% | 1238 |

(b) Policy model.

| Policy model | Top-10 | $\text{AUC}_{10}$ | Time (h) |
|---|---|---|---|
| ECFP4+MLP | 0.640 | 0.600 | 9 |
| Qwen3-0.6B | 0.618 | 0.605 | 13 |
| Qwen3-1.7B | 0.721 | 0.712 | 25 |
| **Qwen3-4B** | **0.796** | **0.775** | **40** |
| Qwen3-8B | 0.798 | 0.753 | 86 |

Table 4: Sensitivity analysis for (a) LLM scale and (b) policy model. Bold denotes the selected models.

### 4.2.5 Evaluation Scope for De Novo Generation

De novo synthesizable generation methods such as SynFlowNet (Cretu et al., 2025) and RxnFlow (Seo et al., 2025) take no starting molecule as input. We ask how often their unconditional outputs satisfy a molecule-conditioned query. We evaluate both on sEH under their default configurations and compare them with MolReAct and the two strongest baselines, Graph GA and SynFormer. For each start $s$ and threshold $\delta$, we count $s$ as solved if the method produces any molecule $g$ satisfying $\text{Tanimoto}(s, g) \geq \delta$ and $\text{Oracle}(g) > \text{Oracle}(s)$.

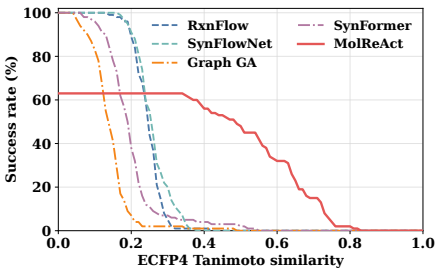



Figure 4: Success rate on sEH under increasing structural similarity constraints.

Figure 4 reports the fraction of solved starts as a function of the similarity threshold $\delta$. Without a similarity constraint the unconditional methods solve 100% of starts against 63% for MolReAct, but their success rates fall sharply as similarity is required. MolReAct retains 32% at $\delta = 0.6$, while all baselines reach zero. Thus, MolReAct is substantially more effective than these baselines when improvements need to remain structurally related to the starting molecule.

### 4.2.6 Computational Cost and Caching Efficiency

Environment-LLM inference is a major computational cost during RL exploration. The cache described in Section 3.2 stores proposed actions and reaction outcomes for each molecule, which are reused without another LLM call when GRPO revisits it. Across all 14 tasks, 15,702 of 27,828 environment calls are cache hits (56.4%). Including LLM inference and policy training, optimization takes 40 hours per task with caching, against an estimated 70 without, reducing overall cost by 43%.

Beyond the efficiency gains from caching, we also compare computational requirements across methods (Appendix B). MolReAct fine-tunes a policy model per task, whereas synthesizable baselines require multi-day pretraining of dedicated models on large molecular datasets. MolReAct therefore avoids pretraining corpora while maintaining a per-task cost broadly comparable to these baselines.

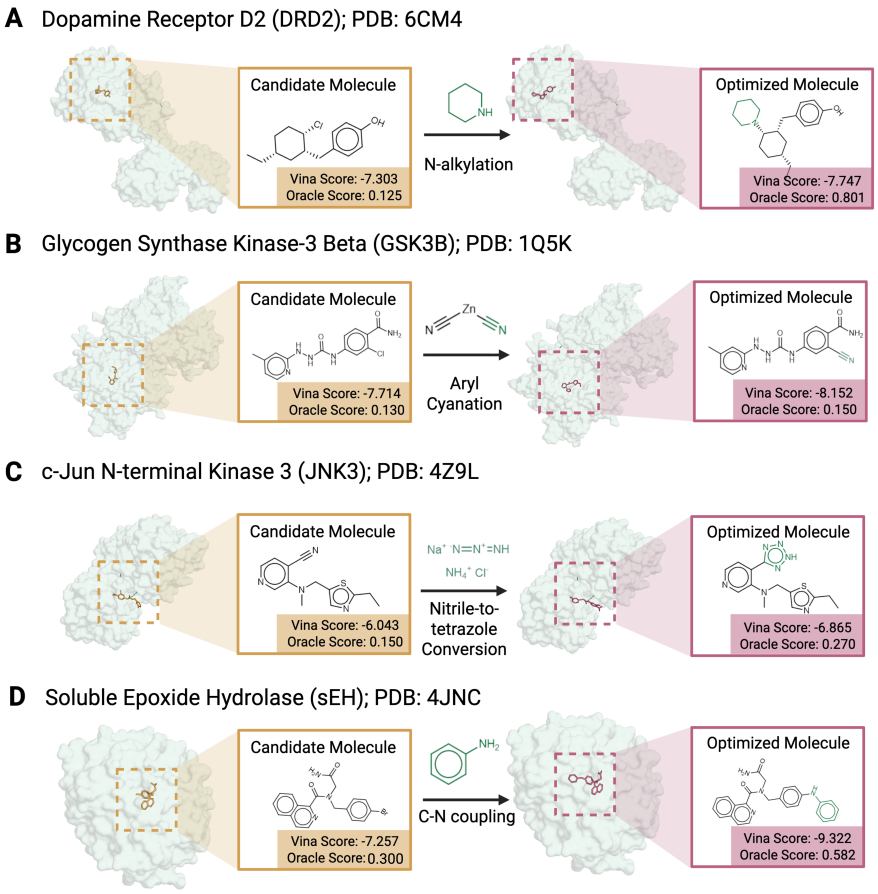



Figure 5: Representative single-step template-grounded reactions discovered by MolReAct on four protein-target tasks, with building blocks, oracle scores, and Vina scores for each molecule.

#### 4.2.7 Pathway and Search Analysis

Figure 5 shows representative single-step template-grounded transformations proposed by MolReAct, with AutoDock Vina scores (Trott & Olson, 2010) as a structure-based measure of binding. Across targets, MolReAct selects chemically meaningful edits that improve both oracle and Vina scores; additional trajectories appear in Appendix G. Beyond individual cases, the tool-augmented LLM proposes roughly four reactions per query (Figure 6a), while optimized molecules require about two steps (Figure 6b), indicating a compact action space that supports multi-step decision-making.

## 5 Discussion

This paper presents MolReAct, which reformulates synthesizable molecular optimization as search over compact action spaces proposed by a tool-augmented LLM, enabling multi-step optimization.

Nevertheless, several limitations remain. First, the quality of the action space at each step depends on the environment LLM, and weaker models may degrade proposal quality over multi-step trajectories. Beyond model capability, the purchasability filter ensures that selected building blocks are available from Enamine, but may still exclude useful reactions due to limited catalog coverage and exact-SMILES matching. On the transition side, template-grounded execution does not explicitly model reaction conditions or condition-dependent yields, and the resulting pathways therefore require experimental validation to confirm their real-world practical feasibility. More broadly, MolReAct targets early-stage drug discovery by restricting its search to validated reaction templates and purchasable building blocks, emphasizing established chemistry over less conventional synthetic reactions.

### AI USE STATEMENT

LLMs are components of the proposed method, with their prompts and configurations documented in Appendix A and Appendix F. Separately, we used generative AI tools to assist with code implementation and editing, data formatting, and improving readability. We did not use these tools to develop the conceptual framework, design the methodology or experiments, or interpret the results. All AI-assisted code, reported results, and edited text were carefully reviewed and independently verified for accuracy by the authors. We take full responsibility for the content of this work.

### ETHICS STATEMENT

This work involves no human subjects or private data, and all resources used are publicly available for research purposes. We acknowledge the dual-use potential of molecular optimization, as such methods could in principle be applied to harmful targets. This risk is limited by our use of benchmark-specific oracles and a constrained action space of common reaction templates and catalog-available building blocks. Since these oracles only approximate early-stage activity and do not establish safety or efficacy, any proposed molecule would require standard experimental and regulatory validation.

### REPRODUCIBILITY STATEMENT

The experimental protocol, including the oracle budget, starting-molecule filtering, evaluation metrics, and Enamine catalog filtering, is specified in Section 4.1, while the GRPO hyperparameters, environment-agent deployment settings, and reaction template library appear in Appendix A, the chemical tool interfaces in Appendix F, and the agent prompts verbatim in the appendix. All baselines run under their public default configurations with identical starting molecules and oracle budgets, all results are averaged over 3 independent runs with standard deviations, and per-task ablation numbers, significance testing, and computing requirements are reported in Appendix Table 6, Appendix D, and Appendix B, respectively. Source code, including the full SMARTS template list, will be released in anonymized form during the rebuttal period, as suggested by the conference guidelines.

## REFERENCES

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 27381–27394. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/e614f646836aaed9f89ce58e837e2310-Abstract.html.

Jinho Chang and Jong Chul Ye. LDMol: A text-to-molecule diffusion model with structurally informative latent space surpasses AR models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 7374–7392. PMLR, 2025. URL https://proceedings.mlr.press/v267/chang25d.html.

Harrison Chase. Langchain, October 2022. URL https://github.com/langchain-ai/langchain. Software; repository created 17 October 2022. Accessed 10 September 2026.

Miruna Cretu, Charles Harris, Ilia Igashov, Arne Schneuing, Marwin Segler, Bruno Correia, Julien Roy, Emmanuel Bengio, and Pietro Lio. Synflownet: Design of diverse and novel molecules with synthesis constraints. In *The Thirteenth International Conference on Learning Representations*, pp. 47101–47126, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/7495fa446f10e9edef6e47b2d327596e-Abstract-Conference.html.

Carl Edwards, Chi Han, Gawon Lee, Thao Nguyen, Sara Szymkuć, Chetan Kumar Prasad, Bowen Jin, Jiawei Han, Ying Diao, Ge Liu, Hao Peng, Bartosz Andrzej Grzybowski, Martin D. Burke, and Heng Ji. mCLM: A modular chemical language model that generates functional and makeable molecules. In *The Fourteenth International Conference on Learning Representations*, pp. 73066–73098, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/76f2614f3cabc22b8a1e0d9a6aa35b61-Abstract-Conference.html.

Enamine. Building blocks catalog, 2026. URL https://enamine.net/building-blocks/building-blocks-catalog. Undated online catalog. Accessed 10 September 2026.

Yin Fang, Xiaozhuan Liang, Ningyu Zhang, Kangwei Liu, Rui Huang, Zhuo Chen, Xiaohui Fan, and Huajun Chen. Mol-instructions: A large-scale biomolecular instruction dataset for large language models. In *The Twelfth International Conference on Learning Representations*, pp. 48221–48251, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/d346d91999074dd8d6073d4c3b13733b-Abstract-Conference.html.

Wenhao Gao, Tianfan Fu, Jimeng Sun, and Connor W. Coley. Sample efficiency matters: a benchmark for practical molecular optimization. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21342–21357, Red Hook, NY, USA, 2022. Curran Associates Inc. doi: 10.52202/068431-1551. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/8644353f7d307baaf29bc1e56fe8e0ec-Abstract-Datasets_and_Benchmarks.html.

Wenhao Gao, Shitong Luo, and Connor W. Coley. Generative AI for navigating synthesizable chemical space. *Proceedings of the National Academy of Sciences*, 122(41):e2415665122, 2025. doi: 10.1073/pnas.2415665122. URL https://doi.org/10.1073/pnas.2415665122.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Kehan Guo, Bozhao Nan, Yujun Zhou, Taicheng Guo, Zhichun Guo, Mihir Surve, Zhenwen Liang, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Can LLMs solve molecule puzzles? a multimodal benchmark for molecular structure elucidation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 134721–134746, 2024. doi: 10.52202/079017-4281. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f2b9e8e7a36d43ddfd3d55113d56b1e0-Abstract-Datasets_and_Benchmarks_Track.html.

Taicheng Guo, Kehan Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. What can large language models do in chemistry? a comprehensive benchmark on eight tasks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 59662–59688. Curran Associates Inc., 2023. doi: 10.52202/075280-2607. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/bbb330189ce02be00cf7346167028ab1-Abstract-Datasets_and_Benchmarks.html.

Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf H Roohani, Jure Leskovec, Connor W. Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/4c56ff4ce4aaf9573aa5dff913df997a-Abstract-round1.html.

Yue Huang, Zhengzhe Jiang, Xiaonan Luo, Kehan Guo, Haomin Zhuang, Yujun Zhou, Zhengqing Yuan, Xiaoqi Sun, Jules Schleinitz, Yanbo Wang, Shuhao Zhang, Mihir Surve, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Chemorch: Empowering LLMs with chemical intelligence via groundbreaking synthetic instructions. In *Advances in Neural Information Processing Systems*, volume 38, pp. 7552–7594. Curran Associates, Inc., 2025. doi: 10.52202/085713-0258. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/0b43289db08ed60edc6451cb2132e203-Abstract-Conference.html.

John J. Irwin, Teague Sterling, Michael M. Mysinger, Erin S. Bolstad, and Ryan G. Coleman. ZINC: A free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52(7):1757–1768, 2012. doi: 10.1021/ci3001277. URL https://doi.org/10.1021/ci3001277.

Kevin Maik Jablonka, Philippe Schwaller, Andres Ortega-Guerrero, and Berend Smit. Leveraging large language models for predictive chemistry. *Nature Machine Intelligence*, 6(2):161–169, 2024. doi: 10.1038/s42256-023-00788-1. URL https://doi.org/10.1038/s42256-023-00788-1.

Jan H. Jensen. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical Science*, 10(12):3567–3572, 2019. doi: 10.1039/C8SC05372C. URL https://doi.org/10.1039/C8SC05372C.

Hyomin Kim, Yunhui Jang, and Sungsoo Ahn. MT-mol: Multi agent system with tool-based reasoning for molecular optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 11544–11573, Suzhou, China, nov 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.619. URL https://aclanthology.org/2025.findings-emnlp.619/.

Ksenia Korovina, Sailun Xu, Kirthevasan Kandasamy, Willie Neiswanger, Barnabas Poczos, Jeff Schneider, and Eric Xing. ChemBO: Bayesian optimization of small organic molecules with synthesizable recommendations. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3393–3403. PMLR, 2020. URL https://proceedings.mlr.press/v108/korovina20a.html.

Michał Koziarski, Andrei Rekesh, Dmytro Shevchuk, Almer M. van der Sloot, Piotr Gaiński, Yoshua Bengio, Cheng-Hao Liu, Mike Tyers, and Robert A. Batey. RGFN: Synthesizable molecular generation using GFlownets. In *Advances in Neural Information Processing Systems*, volume 37, pp. 46908–46955. Curran Associates, Inc., 2024. doi: 10.52202/079017-1488. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/53704142f230054140418ecd8857f391-Abstract-Conference.html.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, New York, NY, USA, 2023. Association for Computing Machinery. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Greg Landrum, Paolo Tosco, Brian Kelley, Ricardo Rodriguez, David Cosgrove, Riccardo Vianello, sriniker, Peter Gedeck, Gareth Jones, Eisuke Kawashima, NadineSchneider, Dan Nealschneider, tadhurst cdd, Andrew Dalke, Matt Swain, Brian Cole, Samo Turk, Aleksandr Savelev, Niels Maeder, Alain Vaucher, Maciej Wójcikowski, Hussein Faara, Ichiru Take, Rachel Walker, Vincent F. Scalfani, Daniel Probst, Kazuya Ujihara, Axel Pahl, guillaume godin, and Juuso Lehtivarjo. rdkit/rdkit: 2025_09_5 (q3 2025) release. Zenodo, 2026. URL https://doi.org/10.5281/zenodo.18428170. Software release issued 30 January 2026.

Seul Lee, Karsten Kreis, Srimukh Prasad Veccham, Meng Liu, Danny Reidenbach, Saee Gopal Paliwal, Weili Nie, and Arash Vahdat. Exploring synthesizable chemical space with iterative pathway refinements. In *The Fourteenth International Conference on Learning Representations*, pp. 7497–7519, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0d0ca4fad5c2646349fc161047e06411-Abstract-Conference.html.

Wei Liu, Jiangtao Feng, Hongli Yu, Yuxuan Song, Yuqiang Li, Shufei Zhang, Lei Bai, Wei-Ying Ma, and Hao Zhou. Retro-r1: LLM-based agentic retrosynthesis. In *Advances in Neural Information Processing Systems*, volume 38, pp. 70709–70737. Curran Associates, Inc., 2025. doi: 10.52202/085713-2378. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/6697768644aeabfd163726c983f75795-Abstract-Conference.html.

Shitong Luo, Wenhao Gao, Zuofan Wu, Jian Peng, Connor W. Coley, and Jianzhu Ma. Projecting molecules into synthesizable chemical spaces. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 33289–33304. PMLR, 2024. URL https://proceedings.mlr.press/v235/luo24a.html.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6 (5):525–535, 2024. doi: 10.1038/s42256-024-00832-8. URL https://doi.org/10.1038/s42256-024-00832-8.

Shogo Nakamura, Nobuaki Yasuo, and Masakazu Sekijima. Molecular optimization using a conditional transformer for reaction-aware compound exploration with reinforcement learning. *Communications Chemistry*, 8(1):40, 2025. doi: 10.1038/s42004-025-01437-x. URL https://doi.org/10.1038/s42004-025-01437-x.

Sven M Papidocha, Andreas Burger, Varinia Bernales, and Alán Aspuru-Guzik. The elephant in the lab: synthesizability in generative small-molecule design. *Current Opinion in Chemical Engineering*, 51:101217, 2026. ISSN 2211-3398. doi: 10.1016/j.coche.2025.101217. URL https://www.sciencedirect.com/science/article/pii/S2211339825001297.

Jinyoung Park, Minseong Bae, Dohwan Ko, and Hyunwoo J. Kim. LLamo: Large language model-based molecular graph assistant. In *Advances in Neural Information Processing Systems*, volume 37, pp. 131972–132000, 2024. doi: 10.52202/079017-4195. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ee46288ab2aaf5c6e53aebebe719712c-Abstract-Conference.html.

Aryan Pedawi, Paweł Gniewek, Chaoyi Chang, Brandon M. Anderson, and Henry van den Bedem. An efficient graph generative model for navigating ultra-large combinatorial synthesis libraries. In *Advances in Neural Information Processing Systems*, volume 35, pp. 8731–8745. Curran Associates Inc., 2022. doi: 10.52202/068431-0635. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/39781da4b5d05bc2908ce08e43bc6404-Abstract-Conference.html.

Seonghwan Seo, Minsu Kim, Tony Shen, Martin Ester, Jinkyoo Park, Sungsoo Ahn, and Woo Youn Kim. Generative flows on synthetic pathway for drug design. In *The Thirteenth International Conference on Learning Representations*, pp. 14029–14060, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/24d36eee157559e0d2549455fba28f6a-Abstract-Conference.html.

Kunyang Sun, Dorian Bagni, Joseph M. Cavanagh, Yingze Wang, Jacob M. Sawyer, Bo Zhou, Andrew Gritsevskiy, Oufan Zhang, and Teresa Head-Gordon. Synllama: Generating synthesizable molecules and their analogs with large language models. *ACS Central Science*, 11(11): 2108–2120, 2025a. doi: 10.1021/acscentsci.5c01285. URL https://doi.org/10.1021/acscentsci.5c01285.

Mengying Sun, Jing Xing, Han Meng, Huijun Wang, Bin Chen, and Jiayu Zhou. Molsearch: Search-based multi-objective molecular generation and property optimization. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4724–4732. Association for Computing Machinery, 2022. doi: 10.1145/3534678.3542676. URL https://doi.org/10.1145/3534678.3542676.

Michael Sun, Alston Lo, Minghao Guo, Jie Chen, Connor W. Coley, and Wojciech Matusik. Procedural synthesis of synthesizable molecules. In *The Thirteenth International Conference on Learning Representations*, pp. 653–667, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/0280d6f1f6ec31076c4560e3a476cac3-Abstract-Conference.html.

Kyle Swanson, Gary Liu, Denise B. Catacutan, Autumn Arnold, James Zou, and Jonathan M. Stokes. Generative ai for designing and validating easily synthesizable and structurally novel antibiotics. *Nature Machine Intelligence*, 6(3):338–353, 2024. doi: 10.1038/s42256-024-00809-7. URL https://doi.org/10.1038/s42256-024-00809-7.

Wen Tao, Jing Tang, Alvin Chan, Bryan Hooi, Baolong Bi, Nanyun Peng, Yuansheng Liu, and Yiwei Wang. How to make large language models generate 100% valid molecules? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 26565–26580, Suzhou, China, nov 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1350. URL https://aclanthology.org/2025.emnlp-main.1350/.

Oleg Trott and Arthur J. Olson. AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of Computational Chemistry*, 31(2):455–461, 2010. doi: 10.1002/jcc.21334. URL https://doi.org/10.1002/jcc.21334.

Haorui Wang, Jeff Guo, Lingkai Kong, Rampi Ramprasad, Philippe Schwaller, Yuanqi Du, and Chao Zhang. LLM-augmented chemical synthesis and design decision programs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 62980–63001. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25ag.html.

Mingyang Wang, Shuai Li, Jike Wang, Odin Zhang, Hongyan Du, Dejun Jiang, Zhenxing Wu, Yafeng Deng, Yu Kang, Peichen Pan, Dan Li, Xiaorui Wang, Xiaojun Yao, Tingjun Hou, and Chang-Yu Hsieh. Clickgen: Directed exploration of synthesizable chemical space via modular reactions and reinforcement learning. *Nature Communications*, 15(1):10127, 2024. doi: 10.1038/s41467-024-54456-y. URL https://doi.org/10.1038/s41467-024-54456-y.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Geyan Ye, Xibao Cai, Houtim Lai, Xing Wang, Junhong Huang, Longyue Wang, Wei Liu, and Xiangxiang Zeng. Drugassist: a large language model for molecule optimization. *Briefings in Bioinformatics*, 26(1):bbae693, 01 2025. ISSN 1477-4054. doi: 10.1093/bib/bbae693. URL https://doi.org/10.1093/bib/bbae693.

Odin Zhang, Haitao Lin, Hui Zhang, Huifeng Zhao, Yufei Huang, Chang-Yu Hsieh, Peichen Pan, and Tingjun Hou. Deep lead optimization: Leveraging generative ai for structural modification. *Journal of the American Chemical Society*, 146(46):31357–31370, 11 2024. ISSN 0002-7863. doi: 10.1021/jacs.4c11686. URL https://doi.org/10.1021/jacs.4c11686.

Yiheng Zhu, Jialu Wu, Chaowen Hu, Jiahuan Yan, Chang-Yu Hsieh, Tingjun Hou, and Jian Wu. Sample-efficient multi-objective molecular optimization with GFlownets. In *Advances in Neural Information Processing Systems*, volume 36, pp. 79667–79684. Curran Associates, Inc., 2023. doi: 10.52202/075280-3487. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/fbc9981dd6316378aee7fd5975250f21-Abstract-Conference.html.

# A Implementation Details

## A.1 GRPO Training Hyperparameters

The policy model is Qwen3-4B-Instruct, trained with GRPO using the MemoryEfficientAdamW optimizer on a single NVIDIA RTX 6000 Ada GPU (48,GB), separate from the four GPUs serving the environment agent (Appendix A.2). Table 5 summarizes the hyperparameters shared across all 14 benchmark tasks in our experiments. A KL regularization term ($\beta_{\text{KL}}$=0.02) penalizes drift from a reference policy. Each training batch consists of 30 seed molecules, each expanded into 10 trajectories, accumulated over a micro-batch of 20. The policy is trained for 25 steps, sufficient for the small discrete classification at each step among at most 10 candidates. The maximum reaction chain per trajectory is 5 steps, and the LLM agent is allowed up to 10 ReAct iterations per step. Training on a single property takes approximately 40 hours on the same GPU, including LLM inference.

Table 5: GRPO hyperparameters for all benchmark tasks.

| **Hyperparameter** | **Value** |
|---|---|
| Policy model | Qwen3-4B-Instruct |
| Training dtype | bfloat16 |
| Optimizer | MemoryEfficientAdamW |
| Adam $(\beta_1, \beta_2)$ | $(0.9,\ 0.999)$ |
| Weight decay | 0.0 |
| Learning rate | $1.0 \times 10^{-5}$ |
| Training steps | 25 |
| Clip ratio $\varepsilon$ | 0.2 |
| Discount factor $\gamma$ | 1.0 |
| KL coefficient $\beta_{\text{KL}}$ | 0.02 |
| KL reference policy | Fixed (initial) |
| Entropy coefficient | 0.01 |
| Molecules per batch | 30 |
| Trajectories per molecule | 10 |
| Micro-batch size | 20 |
| Update epochs per rollout | 2 |
| Max reaction depth | 5 |
| Agent max iterations | 10 |

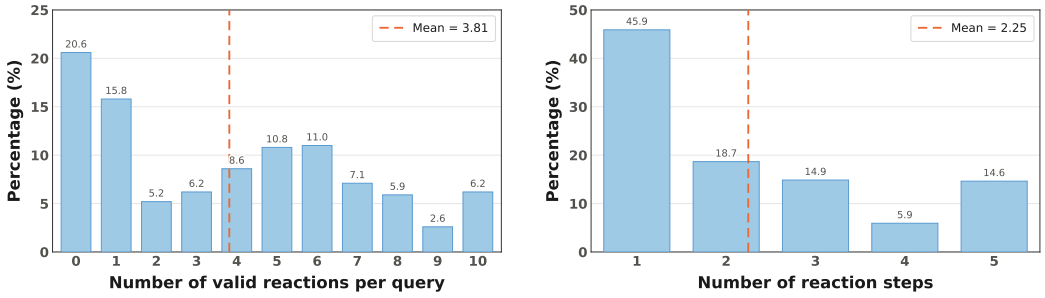
## A.2 Environment Agent Deployment

The environment agent runs Llama-3.3-70B-Instruct served via vLLM across four NVIDIA RTX 6000 Ada GPUs (48,GB each). Inference uses a low decoding temperature $\tau$=0.1 to favor near-deterministic and stable reaction selection. For each independent run, the reaction cache is initialized empty and populated anew, with no cache entries shared across runs. For each step the agent receives the current SMILES, task-specific objective description and tool-call outputs, then proposes up to 10 candidate reactions. Before template execution, we retain only proposals whose additional reactants exactly match canonical SMILES in the Enamine building-block catalog (Enamine, 2026). The 115-template library and this candidate cap bound the reactions the agent can propose.

## A.3 Reaction Template Library

The library contains 115 named organic reactions, each encoded as a SMARTS string specifying the reactant pattern and product transformation. The template set is identical to that used by SynFormer (Gao et al., 2025) and ReaSyn (Lee et al., 2026), ensuring direct comparability with the synthesis-aware baselines. At each step, applicable templates are identified by matching the molecule against every reactant SMARTS via RDKit `HasSubstructMatch` (Landrum et al., 2026), and only templates that pass this filter are exposed to the agent. The library spans major reaction classes:

C-C bond formation, heteroatom introduction, ring closure, and functional-group interconversion. When a template execution yields multiple products (*e.g.*, from repeated reactant matches), we canonicalize each output via RDKit `MolToSmiles` and retain the first valid product in the enumeration order returned by `RunReactants`. This deterministic tie-break ensures that the same (state, template, building block) tuple always produces the same successor molecule, enabling the SMILES-based caching in Section 3.2. The full SMARTS list and reaction-execution scripts are included in our source code, which we provide to reviewers during the discussion period.




(a) Distribution of valid reactions proposed per query.

(b) Distribution of reaction-path lengths.

Figure 6: Distributions characterizing MolReAct's search on the four protein-target activity tasks. (a) Valid reactions proposed per query during the search process. (b) Reaction-path lengths, the number of reactions applied before a trajectory ends at the stop action or the depth-5 limit.

### A.4 Task-Specific Information

We ask whether the performance of MolReAct depends on being told what each task optimizes. The anonymized variants leave the `property_description` field empty in the Question Prompt of Appendix H, removing both the task name and the natural-language objective description, so the environment agent sees only the current SMILES, the matched reaction templates, and the tool outputs. Everything else is identical to the corresponding full configuration: the same Llama-3.3-70B environment model and decoding temperature, the Enamine building-block filter, the maximum depth of 5, and the oracle budget of Appendix A.5. For the anonymized multi-step framework the description is withheld during both policy training and evaluation, so the policy is never exposed to it, while the terminal oracle reward is unchanged. Because these policies are evaluated at their final checkpoint, their values are not directly comparable to Table 1.

Table 6: Ablation results on four protein-target activity tasks. Panel (a) evaluates single-step reaction proposals, and panel (b) evaluates multi-step optimization policies. The learned-policy results are evaluated using final checkpoint, so their values are not directly comparable to those in Table 1 and Table 4b.

(a) Single-step reaction proposal variants.

| | DRD2 | GSK3$\beta$ | JNK3 | sEH |
|---|---|---|---|---|
| Random Template | 0.233 | 0.294 | 0.194 | 0.615 |
| Vanilla LLM | 0.561 | 0.311 | 0.212 | 0.661 |
| Tool-LLM | **0.764** | **0.345** | **0.316** | **0.725** |

(b) Multi-step optimization policies.

| | DRD2 | GSK3$\beta$ | JNK3 | sEH |
|---|---|---|---|---|
| Random Policy | 0.613 | 0.305 | 0.262 | 0.632 |
| Greedy Search | 0.819 | 0.622 | 0.413 | 0.682 |
| MCTS | 0.814 | 0.651 | 0.461 | 0.716 |
| MolReAct (GRPO) | **0.972** | **0.710** | **0.527** | **0.874** |

Table 7 reports both settings on all 14 benchmark tasks. Averaged over the tasks, the anonymized agent retains $94.8\%$ of the one-step proposal quality and $98.8\%$ of the multi-step optimization performance. The objective description therefore contributes a small but consistent gain to single-step proposals, and almost none to the full framework, indicating that the learned policy recovers most of the information the description provides rather than by explicit task information.

Table 7: Top-10 quality with and without task-specific objective descriptions across all 14 tasks. Panel (a) evaluates single-step reaction proposals, and panel (b) evaluates the full multi-step framework, whose policies are evaluated using the final checkpoint. $\Delta$ denotes Task-Specific minus Anonymous.

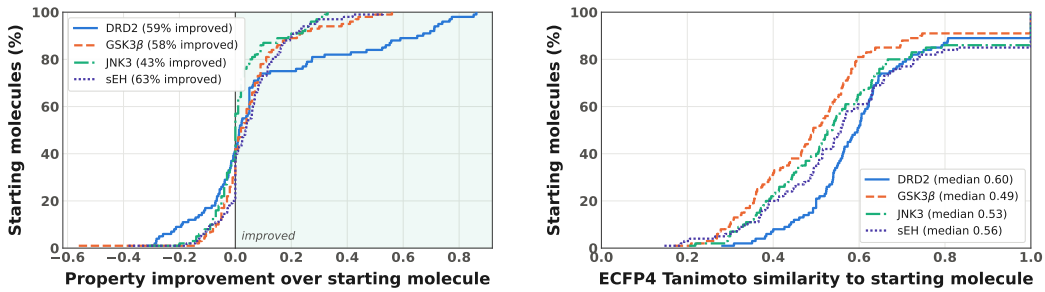
| | (a) One-step proposal | | | (b) Multi-step optimization | | |
|---|---|---|---|---|---|---|
| **Task** | **Anon.** | **Task-Spec.** | $\Delta$ | **Anon.** | **Task-Spec.** | $\Delta$ |
| *Multi-Parameter Optimization* | | | | | | |
| amlodipine_mpo | 0.430 | **0.462** | +0.032 | 0.442 | **0.474** | +0.032 |
| fexofenadine_mpo | 0.699 | **0.742** | +0.043 | **0.759** | 0.756 | -0.003 |
| osimertinib_mpo | 0.727 | **0.775** | +0.048 | 0.802 | **0.812** | +0.010 |
| perindopril_mpo | 0.408 | **0.438** | +0.030 | 0.447 | **0.456** | +0.009 |
| ranolazine_mpo | 0.697 | **0.749** | +0.052 | 0.781 | **0.794** | +0.013 |
| sitagliptin_mpo | 0.296 | **0.337** | +0.041 | 0.362 | **0.371** | +0.009 |
| zaleplon_mpo | **0.427** | 0.423 | -0.004 | 0.436 | **0.452** | +0.016 |
| *Rediscovery and Median* | | | | | | |
| celecoxib_rediscovery | **0.351** | 0.348 | -0.003 | 0.375 | **0.382** | +0.007 |
| median_1 | 0.181 | **0.218** | +0.037 | **0.264** | 0.243 | -0.021 |
| median_2 | 0.130 | **0.164** | +0.034 | **0.171** | 0.167 | -0.004 |
| *Protein Binding Activity* | | | | | | |
| DRD2 | 0.752 | **0.764** | +0.012 | 0.961 | **0.972** | +0.011 |
| GSK3$\beta$ | 0.331 | **0.345** | +0.014 | **0.712** | 0.710 | -0.002 |
| JNK3 | 0.313 | **0.316** | +0.003 | 0.513 | **0.527** | +0.014 |
| sEH | 0.707 | **0.725** | +0.018 | 0.873 | **0.874** | +0.001 |
| Average | 0.461 | **0.486** | +0.025 | 0.564 | **0.571** | +0.007 |

### A.5 Oracle Call Accounting

The 10,000-call budget of Gao et al. (2022) accounts for every oracle query made during policy training and evaluation. At each of the 25 training steps, we sample 30 molecules and generate 10 trajectories per molecule, resulting in 300 terminal oracle calls. We then evaluate the policy by generating one terminal product from each of the same 100 held-out starting molecules used by all methods, adding 100 evaluation calls. This gives 7,500 training calls and 2,500 evaluation calls within the budget of 10,000. Products generated during training are excluded from all reported metrics. For each run, the Top-10 score is computed over the 2,500 unique terminal products accumulated from the held-out evaluations. $\text{AUC}_{10}$ is computed from the cumulative Top-10 score over all held-out evaluation products generated up to each training step. Its horizontal axis counts both training and evaluation oracle calls, reaching 10,000 calls after 25 steps. The comparison therefore charges MolReAct for the cost of learning its policy rather than treating training as free.

### A.6 Fingerprint Policy Baseline

Since selecting among at most 10 candidates is a small discrete choice, we ask whether the language-model policy can be replaced by a lightweight fingerprint-based alternative. Each candidate reaction at state $s_t$ is described by the ECFP4 fingerprint of the product (2048 bits, Morgan radius 2), the difference between the product and state fingerprints, and a one-hot encoding of the reaction template over the 115-template library, giving a 4211-dimensional feature vector. A two-layer MLP with hidden widths 512 and 256 maps each feature vector to one logit, for 2.29M parameters against 4B for Qwen3-4B, with weights shared across candidates so that the variable-size action space is handled directly. Unused slots are masked with $-\infty$ and a softmax gives $\pi_\theta(a_t \mid s_t)$ as in Section 3.3, so this policy is trained against the same terminal reward. The environment agent remains the frozen Tool-LLM configuration, and the Enamine catalog filter, template library, cap of 10 candidates per step, maximum depth of 5, and oracle budget of Appendix A.5 are all unchanged.




(a) Cumulative distribution of property improvement. (b) Cumulative distribution of similarity to starting.

Figure 7: Per-starting-molecule results for the four protein-target activity tasks (DRD2, GSK3$\beta$, JNK3, and sEH); each panel contains one curve per task. (a) Property improvement $\Delta(s)$ over the 100 held-out starts. The legend reports the success rate, i.e., the percentage with $\Delta(s) > 0$. (b) ECFP4 Tanimoto similarity between each starting molecule and the product retained for it.

### A.7 Per-Starting-Molecule Property Improvement

To determine whether the aggregate gains are concentrated in a small subset of starting molecules, we analyze the 100 held-out starts separately for each protein-target activity task. For a start $s$, we retain the product with the highest oracle score across the held-out evaluation checkpoints and define its improvement as $\Delta(s) = \max_t \text{Oracle}(s_t) - \text{Oracle}(s)$. Figure 7(a) reports the empirical cumulative distribution of $\Delta(s)$; the legend gives the fraction of starts with $\Delta(s) > 0$. Figure 7(b) reports the ECFP4 (Morgan radius 2, 2048-bit) Tanimoto similarity between each start and its selected highest-scoring product. Improvements occur for a substantial fraction of the held-out starts on every task (43–63%), rather than being restricted to only a few starts, while the median similarities (0.49–0.60) show that the selected products remain structurally related to their starting molecules.

## B Computing Resource Comparison

The costs of MolReAct and of the baselines are not incurred in the same way in practice, so we separate them rather than report a single figure. The baselines pay a large one-time cost to pretrain a model that is then reused across all 14 tasks, whereas MolReAct pretrains nothing but trains a policy separately for each task, so its cost scales with the number of tasks. Table 8 therefore reports each method's cost as its own paper does, and the comparison below places them on a common basis by amortizing each one-time cost over the 14 benchmark tasks. Figures are taken from each baseline's original publication; entries marked "–" were not reported and we did not estimate them.

**Observations.** Amortized over the 14 tasks, ReaSyn's pretraining comes to roughly 690 GPU-h per task and SynFormer's to roughly 82, against 200 GPU-h per task for MolReAct. MolReAct is thus about an order of magnitude cheaper than ReaSyn and roughly twice as expensive as SynFormer, and because nothing is amortized, its total cost grows linearly with the number of tasks, reaching about 2,800 GPU-h across the full benchmark, with four of the five GPUs occupied by the frozen 70B environment model. We therefore do not claim a compute advantage over the synthesis-constrained baselines. What the table does show is a different cost profile. MolReAct trains no model from scratch and needs no reaction-pathway dataset, so a new objective requires only a 4B policy fine-tune on commodity 48 GB GPUs, with no 8×A100 pretraining run and no 120-GPU data-generation stage. Its recurring cost is also reducible in ways the pretraining cost is not: the environment LLM is frozen and used only for inference, so it can be served from a shared or hosted endpoint across concurrent tasks rather than held on dedicated hardware. Because these profiles are not commensurable, the oracle-call budget of Gao et al. (2022), which every method is held to identically, remains the fairness anchor for the comparisons in Section 4.1.

Table 8: Computing resources across baselines and MolReAct. Reported training cost is one-time pretraining for the baselines, which is reused across all 14 tasks, and per-task training for MolReAct, which is not. Search cost is excluded for all methods, as it is not reported consistently.

| **Method** | **Training framework (size)** | **Reported training cost** |
|---|---|---|
| Graph GA | Genetic algorithm (no neural model) | none |
| ReaSyn | AR transformer (166M) + Edit Bridge (174M) | 5d on 8×A100 + 3d data gen on 120×A100 |
| SynFormer | Encoder-decoder transformer (229M) | 6d on 8×A100 |
| DrugAssist | Llama2-7B-Chat with LoRA (7B) | – (10 epochs, batch 512) on 8×A100-40GB |
| LDMol | BERT-base AE + DiT-base (∼250M) | – on 8×A100-40GB |
| mCLM | Qwen2.5-3B + GNN (3B) | – (6 epochs pretrain, 5 epochs finetune) on 4×A100-80GB |
| MolReAct (Ours) | Qwen3-4B policy (4B) | 40 h on 5×RTX 6000 Ada (per task, no pretraining) |

## C Monte Carlo Tree Search Details

We implement Monte Carlo Tree Search (MCTS) as a planning baseline over the same Tool-LLM-generated reaction environment proposal provided by MolReAct. Each tree node represents a molecule, and its outgoing edges correspond to the proposed template-grounded reactions together with the stop action. During each simulation, tree selection follows the Upper Confidence Bound for Trees (UCT) criterion, $Q(s,a) + c\sqrt{\log N(s)/(N(s,a)+1)}$, where $Q(s,a)$ is the mean terminal reward backed up through action $a$, $N(s)$ and $N(s,a)$ are the corresponding visit counts, and the exploration constant is fixed to $c = \sqrt{2}$. After expanding an unvisited action, MCTS performs a uniform random rollout until the stop action is selected, the maximum reaction depth of five is reached, or no valid reaction is available. Only the terminal molecule is evaluated by the task oracle, and its reward is propagated through all nodes visited during the simulation.

We use the same MCTS configuration for DRD2, GSK3$\beta$, JNK3, and sEH, without task-specific hyperparameter tuning. For each task, MCTS runs 100 simulations independently from each of the 100 held-out starting molecules, resulting in 10,000 terminal oracle calls. Each state contains at most 10 reaction actions in addition to the stop action, matching the action-space configuration used by MolReAct. After search, we retain the highest-scoring terminal molecule discovered from each starting molecule and compute the Top-10 score over the resulting 100 outputs. Results are averaged over three independent MCTS seeds. MCTS uses neither a learned policy prior nor a learned value function; all search decisions are determined by UCT statistics and terminal oracle rewards.

Table 9: Paired Wilcoxon signed-rank test of MolReAct against each baseline across 14 tasks. Wins/losses count tasks where the MolReAct score exceeds/trails the baseline.

| | **Top-10** | | | **AUC$_{10}$** | | |
|---|---|---|---|---|---|---|
| **vs Baseline** | W/L | two-sided $p$ | one-sided $p$ | W/L | two-sided $p$ | one-sided $p$ |
| Graph GA | 13/1 | 0.003 | 0.002 | 14/0 | $< 0.001$ | $< 0.001$ |
| ReaSyn | 12/2 | $< 0.001$ | $< 0.001$ | 12/2 | $< 0.001$ | $< 0.001$ |
| SynFormer | 13/1 | $< 0.001$ | $< 0.001$ | 13/1 | 0.001 | $< 0.001$ |
| DrugAssist | 14/0 | $< 0.001$ | $< 0.001$ | – | – | – |
| LDMol | 14/0 | $< 0.001$ | $< 0.001$ | – | – | – |
| mCLM | 14/0 | $< 0.001$ | $< 0.001$ | – | – | – |

## D Statistical Significance

To assess whether the per-task improvements reflect systematic gains rather than run-to-run variance, we conduct paired Wilcoxon signed-rank tests across all 14 benchmark tasks. Top-10 comparisons include all six baselines in Table 1, whereas AUC$_{10}$ comparisons include the three baselines reported in Table 2. Table 9 reports both two-sided and one-sided $p$-values (testing the directional hypothesis that MolReAct exceeds the baseline), together with per-task win/loss counts.

**Observations.** MolReAct significantly outperforms all six baselines on Top-10 and all three reported baselines on $AUC_{10}$ (two-sided $p < 0.05$). The gain over the strongest synthesis-constrained baseline, SynFormer, is significant on both metrics (13/1 wins, two-sided $p < 0.001$).

Table 10: Optimization objectives for all 14 benchmark tasks.

| Property | Objective |
|---|---|
| *Multi-Parameter Optimization (MPO)* | |
| Amlodipine MPO | Maximize similarity to Amlodipine; keep total ring count ≈3. |
| Fexofenadine MPO | Maximize similarity to Fexofenadine; raise TPSA ($\geq 90$); keep logP $\leq 4$. |
| Osimertinib MPO | Maximize similarity to Osimertinib; raise TPSA ($\geq 100$); push logP very low ($\leq 1$). |
| Perindopril MPO | Maximize similarity to Perindopril; match aromatic ring count $\approx 2$. |
| Ranolazine MPO | Maximize similarity to Ranolazine; raise TPSA ($\geq 95$) and logP ($\geq 7$); keep fluorine count $\approx 1$. |
| Sitagliptin MPO | Maximize dissimilarity to Sitagliptin; match logP $\approx 2$ and TPSA $\approx 77$. |
| Zaleplon MPO | Maximize fingerprint similarity to Zaleplon. |
| *Rediscovery & Median* | |
| Celecoxib Rediscovery | Recover the exact scaffold and substituent pattern of Celecoxib via fingerprint similarity. |
| Median 1 | Maximize simultaneous similarity to camphor and menthol (terpene-like intermediate). |
| Median 2 | Maximize simultaneous similarity to tadalafil and sildenafil (multi-ring heteroaromatic intermediate). |
| *Protein Binding Activity* | |
| JNK3 | Maximize predicted inhibitory activity against JNK3 kinase. |
| DRD2 | Maximize predicted activity against DRD2 (aminergic GPCR). |
| GSK3$\beta$ | Maximize predicted inhibitory activity against GSK3$\beta$ kinase. |
| sEH | Maximize predicted binding affinity for soluble epoxide hydrolase (sEH/EPHX2). |

## E Benchmark Property Descriptions

The benchmark includes 13 property optimization tasks from TDC and one proxy-based binding affinity task for soluble epoxide hydrolase (sEH). Table 10 summarizes their optimization objectives across MPO, rediscovery and median objectives, and protein-target activity prediction.

## F Chemical Tool Details

All six tools are implemented with RDKit (v2024.03) and registered as LangChain `BaseTool` instances (Chase, 2022). Each tool takes the current molecule SMILES as input and runs locally, returning a structured plain-text string for the agent to reason over in the ReAct scratchpad. Table 11 lists each tool alongside its chemical role and a representative output. `SiteAnalyzer` detects five reactive motifs: electrophilic carbonyl, nucleophilic amine, alcohol/phenol, halogen, and Michael acceptor. Together, these tools provide the agent with a compact structural profile sufficient to select structurally compatible reaction templates without querying any external service.

Table 11: Chemical analysis tools for LLM agent. All accept the SMILES and return structured text.

| Tool | Role | Example output |
|---|---|---|
| `SMILES2Weight` | Molecular weight | `335.14` |
| `FuncGroups` | Functional group detection | `carboxylic acid (1), primary amine (1)` |
| `ScaffoldExtract` | Murcko scaffold extraction | `[Scaffold] c1ccc2ncccc2c1; rings=2; hetero=1` |
| `BRICSFragment` | BRICS decomposition | `frag_count=3; frags=[CCN, c1ccncc1, C(=O)O]` |
| `RingAnalyzer` | Ring analysis | `total=3; arom=2; hetero=1; [6A,6A,5H]` |
| `SiteAnalyzer` | Reactive site identification | `count=2; sites=[carbonyl, nucl_amine]` |

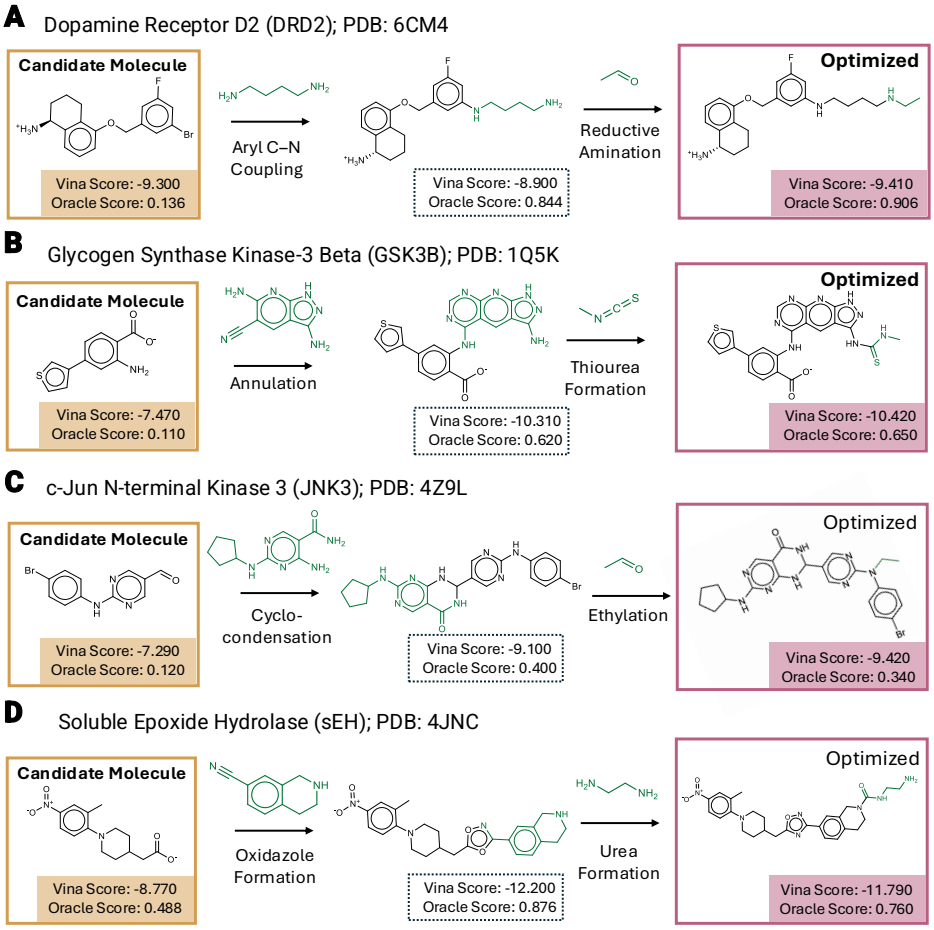



Figure 8: Representative two-step template-grounded pathways discovered by MolReAct for four protein-target activity tasks. Each panel shows the candidate molecule, two selected reactions with their building blocks, the intermediate product, and the final optimized product. The displayed task-oracle and AutoDock Vina scores are reported for each of the three molecules.

## G Additional Pathway Case Studies

Figure 8 shows two-step template-grounded trajectories for DRD2, GSK3$\beta$, JNK3, and sEH. Each panel records the candidate molecule, two selected reactions and their building blocks, the intermediate product, and the final optimized product. We report both the task oracle score and the computed AutoDock Vina score for the starting, intermediate, and final molecules. Across the examples, final products improve on their starts in both oracle score and Vina score, illustrating how individually valid template-grounded edits can be composed into multi-step optimization trajectories.

# H LLM AGENT PROMPT

**System Role**

You are an expert medicinal chemist.

**Format Instructions**

You can respond in two ways:

**Thought:** (reflect on your progress and decide what to do next)
**Action:** (the action name, should be one of [{tool_names}])
**Action Input:** (the input string to the action)

OR

**Final Answer:**

**Reaction 1:**
- Reaction name: [reaction name[id] from the feasible template list above]
- Reactants: [ONLY the missing reactant(s) SMILES -- see rules below]

**Reaction 2:**
- Reaction name: [reaction name[id]]
- Reactants: [ONLY the missing reactant(s) SMILES]

[List up to 10 reactions from the templates provided. Choose based on the provided optimization objective. Use a DIVERSE set of templates. If reusing the same template, you MUST use a different reactant each time.]

**Critical Rules**

(1) Reaction name must be EXACTLY one of the templates listed above.

(2) Reactants field -- STRICT requirements:
- ONLY the missing/additional reactant SMILES. NEVER include the input molecule.
- The number of reactants MUST match the template's missing count.
- Each reactant must be COMPLETE, VALID with $\geq$2 heavy atoms. Single atoms (O, N, C, Cl, Br) are **FORBIDDEN**.
- If the template says "reactants: none", write "Reactants: none".

(3) How to read SMARTS and pick the right molecule:
(a) identify the atom chain; (b) build a real molecule; (c) close any aromatic rings.

*Common SMARTS → reactant examples:*
[O&H1$(Oc)] (phenol) → Oc1ccccc1 [N&X3;H1,H2$(N[#6])] (amine) → NCC, NC(C)C
[#6][C&H1]=O (aldehyde) → CC=O [#6]C(=O)[O&H1] (acid) → CC(=O)O [Cl,Br,I][#6] → ClCC

Never output a bare atom. Always output a full molecule.

**Important:** You may use tools to understand the molecule. After every Observation, write a Thought, then either Action + Action Input, or Final Answer. Use exactly "**Final Answer:**" before **Reaction 1**.

**Question Prompt**

Answer the question below using the following Tools:

{tool_strings}

Question:
Input molecule (SMILES): {smiles}
Optimization objective: {property_description}
Design up to 10 diverse, chemically feasible single-step reactions.
Reaction templates: {matched_templates}