

SkillMol: Skill-Conditioned Action Editing for Molecular Optimization

Jiajun Yu^{1,2}, Kun Li³, Zhihao Wu¹
Weifan Mao^{2,4}, Yizhen Zheng⁵, Haishuai Wang^{1,*}

¹Zhejiang Key Lab of Accessible Perception and Intelligent Systems, Zhejiang University

²Shanghai Innovation Institute ³Shanghai AI Lab

⁴Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University

⁵Department of Data Science and Artificial Intelligence, Monash University

Abstract

Existing LLM-based molecular optimization methods frame the task as molecule-to-molecule mapping, forcing models to implicitly learn structure-property relationships from paired SMILES alone. We argue this misses a key ingredient: task-level chemical knowledge that tells the model *which* structural motifs to prefer or avoid for a given property direction, before it ever looks at a specific molecule. We propose SkillMol, which constructs compact task *skills* – sets of recommended and discouraged structural motifs for each optimization objective – from domain knowledge, and pairs them with an atom-anchored action interface that decomposes editing into discrete, executable operations. Skills are used twice: as explicit training context so the model learns skill-conditioned edit actions, and as functional-group constraints during inference search. A deterministic RDKit executor applies the proposed graph edits. On the MoleculeSTM benchmark across 14 molecular editing tasks, SkillMol achieves 92.5% loose and 80.0% strict hit ratio on single-property tasks and 65.3% average across loose and strict multi-property columns, outperforming strong LLM baselines. Our code can be found at <https://github.com/jiajunyu1999/SkillMol>.

1 Introduction

Large language models (Guo et al., 2025; Achiam et al., 2023; Dubey et al., 2024; Yang et al., 2025; Yu et al., 2026b; Liang et al., 2025) have become a popular tool for molecular optimization (Liu et al., 2024; Ye et al., 2025; Wu et al., 2024; Yu et al., 2025), but the dominant approach still frames the

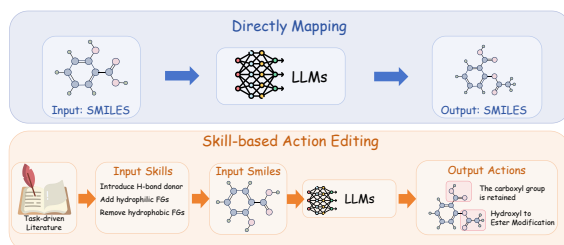


Figure 1: The SkillMol workflow: construct a task skill from domain knowledge, use the skill to predict an atom-anchored edit action, execute the edit, and rank candidates by property reward.

task as molecule-to-molecule mapping (Liu et al., 2023, 2024, 2025b): given a source molecule, generate a target that satisfies the desired property direction. This forces the model to implicitly recover structure-property relationships from paired SMILES alone, with no explicit signal about *why* a particular substitution improves logP or raises tPSA. The gap is most visible under strict evaluation, where a property change must exceed a meaningful threshold rather than merely point in the right direction. The missing ingredient is task-level chemical knowledge: a compact prior that tells the model which functional groups to prefer or avoid for a given property direction, independent of any specific molecule. A medicinal chemist brings this kind of knowledge before touching a molecule — which functional groups reliably increase solubility (Lipinski et al., 1997)? which substitutions raise tPSA without hurting permeability (Matsson and Kihlberg, 2017)? — and applies it to decide where to make a targeted local edit. This prior is not molecule-specific; it is task-specific, reusable, and grounded in accumulated chemical evidence.

*Corresponding author: haishuai.wang@zju.edu.cn

SkillMol operationalizes this design. For each optimization objective, we construct a task *skill* – a compact, reusable set of recommended and discouraged structural motifs – from domain knowledge, and pair it with an atom-anchored action interface that decomposes editing into discrete, executable operations. The skill is used in two places. During SFT, it is provided as explicit prompt context so the model learns to produce atom-anchored edit actions conditioned on chemical reasoning rather than on SMILES co-occurrence patterns. During inference, the same skill is converted into a functional-group constraint that narrows the search space before any candidate is sampled or executed. A deterministic RDKit (Landrum et al., 2013) executor applies each proposed action and sanitizes the resulting molecular graph, keeping graph construction out of the language model’s responsibility. Experiments on the MoleculeSTM (Liu et al., 2023) 14-task benchmark show that SkillMol, built on Qwen3-0.6B (Yang et al., 2025) with LoRA (Hu et al., 2021), achieves 92.5% loose and 80.0% strict hit ratio on single-property tasks and 65.3% on multi-property tasks, outperforming strong LLM-based baselines. Our contributions can be summarized as follows: (i) we show that incorporating compact, task-level chemical skills can substantially improve the molecular editing ability of LLMs by providing reusable structure-property priors beyond molecule-to-molecule memorization, (ii) we demonstrate that an action-based editing framework enables strong molecular optimization performance even with very small language models, suggesting that interface design matters more than parameter scale, (iii) across 14 molecular optimization tasks, SkillMol achieves the best average performance among existing LLM-based approaches under strict evaluation settings.

2 Related Work

Molecular optimization. Molecular optimization aims to improve target properties while retaining useful structural context. Prior work has studied reinforcement-learning graph generation (You et al., 2018; Zhou et al., 2019), graph-to-graph translation (Jin et al., 2018), genetic or search-based optimization (Jensen, 2019; Ahn et al., 2020), fragment-level modification (Chen et al., 2021; Yu et al., 2026a), and constrained edit sampling (Fu et al., 2021b,a). These methods provide useful mechanisms for local graph modification, but con-

trollability is often tied to task-specific objectives or search heuristics. Transformer-based molecular models (Vaswani et al., 2017), including SMILES generation systems such as Transformer-R (He et al., 2021) and LigGPT (Bagal et al.), further improved sequence-level molecular modeling, while multimodal molecule-text frameworks connected structure optimization with textual supervision (Liu et al., 2023).

LLMs for molecular optimization. Large language models have strong reasoning and instruction-following capabilities (Guo et al., 2025; Liu et al., 2026; Dubey et al., 2024; Yang et al., 2025), and have been adapted to scientific and chemical tasks (Taylor et al., 2022; Mirza et al., 2024; Zheng et al., 2024; Zhang et al., 2025; Li et al., 2026). Instruction tuning and lightweight adaptation (Wei et al., 2021; Ouyang et al., 2022; Hu et al., 2021), together with open-source LLMs (Taori et al., 2023; Chiang et al., 2023), enable molecular systems that follow natural-language objectives and scientific prompts (Meyer et al., 2023; Zheng et al., 2025, 2024; Liu et al., 2024, 2023). Recent LLM-based optimizers support text-conditioned retrieval and editing, prompt-based optimization, dialogue-based drug editing, scaffold-aware constraints, feedback-guided search, and agentic reasoning (Liu et al., 2023; Wu et al., 2024; Liu et al., 2024; Ye et al., 2025; Liu et al., 2025a,b; Li et al., 2026). However, many still generate complete target molecules or rely on iterative search, which can be costly and imprecise at the atom level. SkillMol instead predicts explicit atom-anchored edit actions, uses compact task skills as chemical priors.

3 Method

Overview. SkillMol follows a three-stage pipeline: (1) construct one task skill per objective offline from domain knowledge; (2) train the LLM to emit atom-anchored edit actions conditioned on the skill; (3) at inference, reuse the same skill to constrain functional-group choices during multi-step search, with a deterministic RDKit executor applying and sanitizing each graph edit.

3.1 Problem Formulation

Let $\mathcal{X}$ denote the molecular graph space and let $\mathcal{Q}$ denote the set of optimization tasks. Each task $q \in \mathcal{Q}$ is defined by one or more property-direction pairs $q = \{(p_j, s_j)\}_{j=1}^m$, where p_j is a molecular

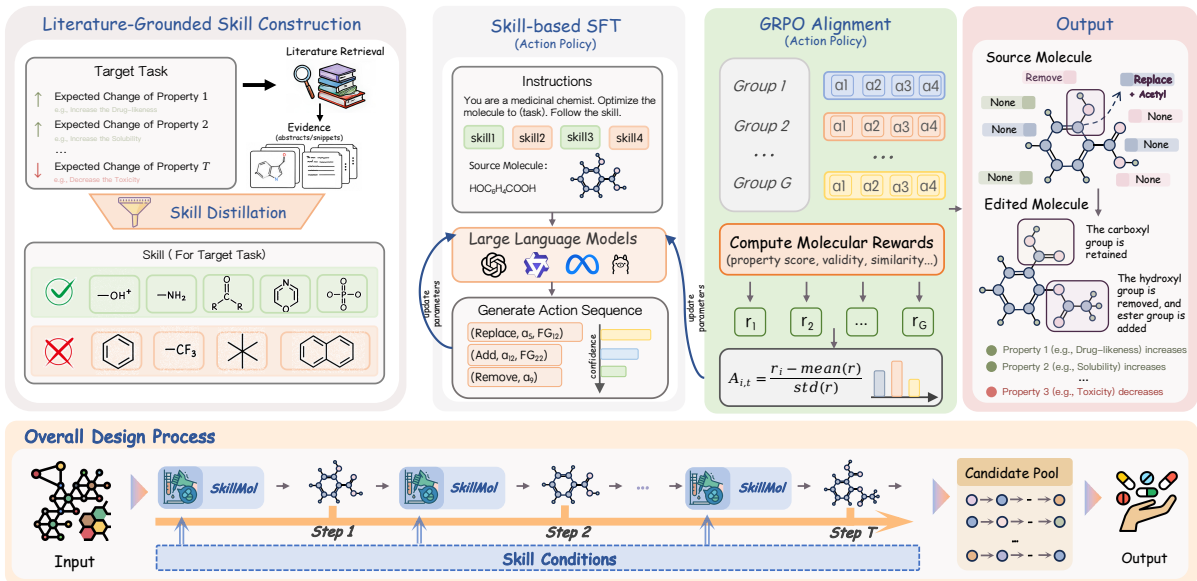


Figure 2: SkillMol pipeline. Task skills are built once from domain knowledge, used as prompt context for action SFT, optionally refined with execution rewards, and converted into functional-group constraints during search.

property and $s_j \in \{-1, +1\}$ indicates whether the property should decrease or increase. Given a source molecule $x_0 \in \mathcal{X}$, the objective is to generate an editing trajectory $(x_0, a_1, x_1, \dots, a_T, x_T)$ such that x_T satisfies the loose or strict hit criterion for q . Each action a_t belongs to a structured edit vocabulary $\mathcal{A}$ and is parameterized as

$$a_t = (o_t, u_t, v_t, g_t), \quad (1)$$

$$o_t \in \{\text{ADD}, \text{REMOVE}, \text{REPLACE}\},$$

where u_t is an atom-mapped anchor, v_t is an optional removed atom for destructive edits, and $g_t \in \mathcal{F}$ is an optional functional group for constructive edits. The transition $x_t = E(x_{t-1}, a_t)$ is executed by a deterministic RDKit-based edit executor E that applies the action to the current molecule and sanitizes the product. This formulation separates molecular reasoning into three parts: the skill gives the property-level chemical prior, the language model decides which local edit should be attempted, and the executor determines whether the proposed edit is chemically executable.

3.2 Action Space and Edit Execution

Each edit action is a tuple (o, u, v, g) : an operation $o \in \{\text{ADD}, \text{REMOVE}, \text{REPLACE}\}$, an atom-map anchor u identifying the edit site, an optional removed atom v for destructive edits, and an optional functional-group identifier g for constructive edits. Every atom in the source molecule is assigned an explicit atom-map ID before prompting,

so the model selects a concrete anchor rather than describing a vague substructure. The model output is a structured text record rather than a product SMILES:

`<EDIT_SET> <OP> o <ANCHOR> u <REMOVE> v <FGID> g`

A deterministic RDKit executor parses the record and applies the graph transformation. ADD connects the selected anchor u to the attachment atom of the selected fragment g . REMOVE deletes the side branch reached from v after cutting the bond (u, v) ; a branch is removable only when v is outside the Murcko scaffold, the cut does not lie inside a ring, and deleting the connected component keeps the remaining molecule connected. REPLACE first performs the same branch removal and then attaches g to u through the fragment attachment point stored in the functional-group library. Ring atoms and aromatic bonds are not removed directly, and products are sanitized with RDKit after kekulization/aromaticity perception. For multi-step editing, atom-map IDs are reassigned after each executed edit before the next prompt is built, so the model always sees the current molecule with fresh, unambiguous atom identifiers.

3.3 Task Skill Construction

For each optimization objective q , we construct one task skill s_q offline. We use Codex-GPT5.5 high as a literature-search assistant: for each property direction, it issues web and PubMed-style queries combining the property name, direction, and medicinal-

chemistry edit terms (e.g., “logP decrease functional group”, “tPSA increase substituent”, and “lead optimization reduce HBD”). The retrieved evidence is limited to medicinal-chemistry reviews, highly cited papers, PubMed abstracts, and public compound resources, and is kept only when it directly supports the requested property direction. Codex-GPT5.5 high then summarizes the retained evidence into a structured skill: a set of recommended structural motifs $\mathcal{R}_q$ and a set of discouraged motifs $\mathcal{B}_q$ for the given property direction. We manually check the final skill table for direction consistency, duplicate motifs, and conflicts between recommended and discouraged sets. The skill is task-level and constructed entirely offline; no training molecule or generated result is used. The same skill $s_q = (\mathcal{R}_q, \mathcal{B}_q)$ is reused as prompt context during SFT and as the source of inference-time functional-group constraints. The complete skill library is listed in Appendix B.4.

3.4 Skill-Based Action SFT

Given a training set of molecular editing examples $\mathcal{D} = \{(x_i, q_i, y_i)\}_{i=1}^N$, where y_i is the target edit action sequence, we prepend the corresponding skill s_{q_i} to the instruction prompt and optimize the language model to produce edit actions. The input contains the source SMILES, atom-map indices for every atom, optimization objective, property direction, and the retrieved skill. The target action is also written in the same atom-ID space, so SFT directly teaches the model to choose an anchor ID and an edit type, using the same action record format defined in §3.2. Fields that do not apply to an operation are omitted. The record explicitly tells the executor the edit type, the atom ID of the anchor point, the atom ID of the removed branch when needed, and the inserted functional group when needed. Let θ denote the LoRA parameters and let $y_i = (y_{i,1}, \dots, y_{i,L_i})$ be the tokenized action sequence. The supervised objective is

$$\mathcal{L}_{\text{SFT}}(\theta) = - \sum_{i=1}^N \sum_{\ell=1}^{L_i} \log P_{\theta}(y_{i,\ell} \mid y_{i,<\ell}, x_i, q_i, s_{q_i}). \quad (2)$$

This objective teaches the model to associate a task skill with a local edit decision rather than a product SMILES. The fixed action schema also makes malformed outputs easy to detect before execution.

3.5 GRPO Alignment

After skill-based action SFT, we further align the policy with molecular execution rewards through GRPO. For each prompt $c = (x_0, q, s_q)$, the policy samples a group of K action trajectories, executes each with the RDKit executor, and computes a scalar reward r_k . The reward is hierarchical: a non-executed trajectory receives $r = -1$; an executed molecule receives a base score determined by hit status (strict hit $\geq$ loose hit $\geq$ executed-only), plus a continuous margin term $\delta_k = \min_j \frac{s_j \Delta p_j}{\theta_j}$ that measures how far the bottleneck property is from the strict threshold (where θ_j is the per-property strict threshold), and a small edit-count penalty ηn_k , where n_k is the number of executed edit steps. $\lambda > 0$ is a margin weight (set to $\lambda = 0.5$ in all experiments), $\eta = 0.05$, and ϵ is a small constant for numerical stability. Formally:

$$r_k = \begin{cases} -1 & \text{not executed} \\ 1 + \lambda \text{clip}(\delta_k, -0.75, 0.5) - \eta n_k & \text{strict hit} \\ 0.5 + \lambda \text{clip}(\delta_k, -0.75, 0.5) - \eta n_k & \text{loose hit} \\ \lambda \text{clip}(\delta_k, -0.75, 0.5) - \eta n_k & \text{executed only} \end{cases} \quad (3)$$

GRPO normalizes rewards within the sampled group to estimate relative advantages:

$$\mu_r = \frac{1}{K} \sum_{k=1}^K r_k, \sigma_r = \sqrt{\frac{1}{K} \sum_{k=1}^K (r_k - \mu_r)^2}, \hat{A}_k = \frac{r_k - \mu_r}{\sigma_r + \epsilon}. \quad (4)$$

Let τ_k denote the sampled action trajectory and $\rho_k(\theta) = \pi_{\theta}(\tau_k \mid c) / \pi_{\text{old}}(\tau_k \mid c)$ be the policy ratio. We update the policy with the clipped objective

$$\mathcal{J}_{\text{GRPO}}(\theta) = \frac{1}{K} \sum_{k=1}^K \min \left(\rho_k(\theta) \hat{A}_k, \bar{\rho}_k(\theta) \hat{A}_k \right) - \beta D_{\text{KL}}(\pi_{\theta} \parallel \pi_{\text{SFT}}), \quad (5)$$

where

$$\bar{\rho}_k(\theta) = \text{clip}(\rho_k(\theta), 1 - \epsilon_p, 1 + \epsilon_p). \quad (6)$$

Here ϵ_p is PPO clipping coefficient and β controls the KL penalty toward the SFT reference policy.

3.6 Skill-Guided Inference

At inference time, the same task skill s_q used during training is provided as prompt context, so the model already has the chemical prior when generating edit actions. The prompt format is identical to SFT: source SMILES with atom-map IDs, optimization objective, and skill text. The model outputs an action record in the same format as §3.2.

Table 1: Loose hit ratio on eight single-property tasks. A loose hit requires movement in the requested direction; AVG is the macro average over tasks. The gray row uses skill-FG search (§3.6). Bold marks the best result and underline marks the second best. Δ denotes the optimized property interval.

Method Δ	logP- 0.0	logP+ 0.0	QED+ 0.0	QED- 0.0	tPSA- 0.0	tPSA+ 0.0	HBA+ 0.0	HBD+ 0.0	AVG
Random	35.3 $\pm$ 1.0	43.4 $\pm$ 3.0	38.1 $\pm$ 2.0	37.0 $\pm$ 2.0	25.2 $\pm$ 2.0	16.8 $\pm$ 2.0	12.6 $\pm$ 1.0	3.0 $\pm$ 0.6	26.4
PCA	33.8 $\pm$ 3.0	39.4 $\pm$ 2.0	34.0 $\pm$ 3.0	35.2 $\pm$ 2.0	21.4 $\pm$ 0.0	15.5 $\pm$ 2.0	10.9 $\pm$ 2.0	4.0 $\pm$ 0.6	24.3
High Variance	33.5 $\pm$ 3.0	42.9 $\pm$ 2.0	36.2 $\pm$ 4.0	40.0 $\pm$ 0.0	22.0 $\pm$ 3.0	17.1 $\pm$ 1.0	11.8 $\pm$ 0.0	6.2 $\pm$ 0.7	26.2
GS-Mutate	52.0 $\pm$ 0.0	47.5 $\pm$ 0.0	28.0 $\pm$ 0.0	71.3 $\pm$ 0.0	22.0 $\pm$ 0.0	28.8 $\pm$ 1.0	21.2 $\pm$ 3.0	19.5 $\pm$ 2.9	36.3
MoleculeSTM-SMILES	61.9 $\pm$ 2.0	52.7 $\pm$ 1.0	36.5 $\pm$ 2.0	58.6 $\pm$ 1.0	57.7 $\pm$ 0.0	34.1 $\pm$ 0.0	54.0 $\pm$ 5.0	28.6 $\pm$ 0.8	48.0
MoleculeSTM-Graph	67.9 $\pm$ 3.5	64.8 $\pm$ 2.8	40.0 $\pm$ 4.3	<u>77.6 $\pm$ 2.8</u>	59.8 $\pm$ 0.8	31.8 $\pm$ 1.0	37.4 $\pm$ 0.8	61.0 $\pm$ 5.1	55.0
ChatDrug	<u>94.1 $\pm$ 1.0</u>	<u>96.9 $\pm$ 1.1</u>	48.7 $\pm$ 3.4	70.8 $\pm$ 2.9	56.6 $\pm$ 1.8	77.4 $\pm$ 2.0	<u>95.4 $\pm$ 0.6</u>	<u>96.5 $\pm$ 1.3</u>	<u>79.5</u>
MultiMol	91.3 $\pm$ 1.2	92.1 $\pm$ 1.1	49.3 $\pm$ 1.5	72.4 $\pm$ 1.3	51.2 $\pm$ 1.4	<u>86.9 $\pm$ 1.0</u>	89.3 $\pm$ 0.9	89.0 $\pm$ 1.1	77.7
LLaMA Chem Reasoner	51.5 $\pm$ 0.0	57.0 $\pm$ 0.0	32.0 $\pm$ 0.0	48.0 $\pm$ 0.0	44.0 $\pm$ 0.0	51.5 $\pm$ 0.0	51.5 $\pm$ 0.0	39.0 $\pm$ 0.0	46.8
LLaMA RL-Guider	60.0 $\pm$ 0.0	58.5 $\pm$ 0.0	43.0 $\pm$ 0.0	57.0 $\pm$ 0.0	62.0 $\pm$ 0.0	60.0 $\pm$ 0.0	61.0 $\pm$ 0.0	50.5 $\pm$ 0.0	56.5
DeepSeek Chem Reasoner	81.0 $\pm$ 0.0	87.5 $\pm$ 0.0	53.0 $\pm$ 0.0	73.0 $\pm$ 0.0	64.5 $\pm$ 0.0	74.0 $\pm$ 0.0	74.5 $\pm$ 0.0	50.0 $\pm$ 0.0	69.7
DeepSeek RL-Guider	82.0 $\pm$ 0.0	92.0 $\pm$ 0.0	78.0 $\pm$ 0.0	69.5 $\pm$ 0.0	<u>78.0 $\pm$ 0.0</u>	82.0 $\pm$ 0.0	75.0 $\pm$ 0.0	57.0 $\pm$ 0.0	76.7
SkillMol	98.0 $\pm$ 2.0	100.0 $\pm$ 0.0	<u>64.0 $\pm$ 6.8</u>	100.0 $\pm$ 0.0	80.0 $\pm$ 5.7	98.0 $\pm$ 2.0	100.0 $\pm$ 0.0	100.0 $\pm$ 0.0	92.5

Table 2: Strict hit ratio on eight single-property tasks. A strict hit must exceed the threshold in the Δ row, so this table measures both direction and edit magnitude. AVG is the macro average over tasks; the gray row uses skill-FG search (§3.6). Δ denotes the optimized property interval.

Method Δ	logP- 0.5	logP+ 0.5	QED+ 0.1	QED- 0.1	tPSA- 10.0	tPSA+ 10.0	HBA+ 1.0	HBD+ 1.0	AVG
Random	11.0 $\pm$ 2.4	19.8 $\pm$ 1.6	5.3 $\pm$ 0.2	6.2 $\pm$ 1.9	17.4 $\pm$ 1.4	11.0 $\pm$ 0.7	0.7 $\pm$ 0.0	0.0 $\pm$ 0.0	8.9
PCA	10.7 $\pm$ 3.2	15.1 $\pm$ 2.9	4.0 $\pm$ 0.1	5.3 $\pm$ 1.0	14.5 $\pm$ 0.8	10.6 $\pm$ 1.9	0.9 $\pm$ 0.8	0.0 $\pm$ 0.0	7.6
High Variance	10.9 $\pm$ 2.6	18.2 $\pm$ 3.2	4.4 $\pm$ 0.6	7.6 $\pm$ 0.3	14.7 $\pm$ 2.1	12.0 $\pm$ 1.0	0.7 $\pm$ 0.0	0.0 $\pm$ 0.0	8.6
GS-Mutate	14.7 $\pm$ 0.6	12.5 $\pm$ 0.8	6.3 $\pm$ 0.8	27.7 $\pm$ 3.8	6.2 $\pm$ 0.6	15.2 $\pm$ 1.0	1.8 $\pm$ 0.5	1.3 $\pm$ 0.2	10.7
MoleculeSTM-SMILES	49.0 $\pm$ 1.8	30.5 $\pm$ 3.3	8.8 $\pm$ 0.8	37.6 $\pm$ 1.8	47.5 $\pm$ 1.9	26.5 $\pm$ 1.0	27.3 $\pm$ 2.6	7.7 $\pm$ 0.6	29.4
MoleculeSTM-Graph	47.1 $\pm$ 3.4	54.4 $\pm$ 4.0	14.1 $\pm$ 3.2	54.2 $\pm$ 3.1	50.4 $\pm$ 2.7	19.8 $\pm$ 1.3	16.1 $\pm$ 2.9	32.4 $\pm$ 2.6	36.1
ChatDrug	70.1 $\pm$ 3.4	<u>88.7 $\pm$ 1.0</u>	<u>19.4 $\pm$ 5.5</u>	31.0 $\pm$ 2.7	43.1 $\pm$ 3.0	66.7 $\pm$ 2.7	72.6 $\pm$ 2.5	76.4 $\pm$ 3.3	58.5
MultiMol	81.6 $\pm$ 1.2	86.4 $\pm$ 1.5	30.4 $\pm$ 1.5	55.1 $\pm$ 1.2	59.9 $\pm$ 1.4	<u>86.6 $\pm$ 1.0</u>	95.1 $\pm$ 0.8	92.4 $\pm$ 1.1	73.4
LLaMA Chem Reasoner	31.5 $\pm$ 0.0	51.0 $\pm$ 0.0	5.0 $\pm$ 0.0	14.0 $\pm$ 0.0	34.0 $\pm$ 0.0	51.0 $\pm$ 0.0	13.5 $\pm$ 0.0	5.5 $\pm$ 0.0	25.7
LLaMA RL-Guider	52.5 $\pm$ 0.0	55.5 $\pm$ 0.0	9.5 $\pm$ 0.0	60.0 $\pm$ 0.0	44.5 $\pm$ 0.0	54.0 $\pm$ 0.0	46.0 $\pm$ 0.0	20.5 $\pm$ 0.0	42.8
DeepSeek Chem Reasoner	66.0 $\pm$ 0.0	81.0 $\pm$ 0.0	10.5 $\pm$ 0.0	13.0 $\pm$ 0.0	50.5 $\pm$ 0.0	74.5 $\pm$ 0.0	19.0 $\pm$ 0.0	10.5 $\pm$ 0.0	40.6
DeepSeek RL-Guider	75.5 $\pm$ 0.0	83.0 $\pm$ 0.0	10.0 $\pm$ 0.0	<u>70.5 $\pm$ 0.0</u>	<u>57.0 $\pm$ 0.0</u>	77.5 $\pm$ 0.0	65.0 $\pm$ 0.0	20.5 $\pm$ 0.0	57.4
SkillMol	98.0 $\pm$ 2.0	100.0 $\pm$ 0.0	12.0 $\pm$ 4.6	96.0 $\pm$ 2.8	54.0 $\pm$ 7.0	98.0 $\pm$ 2.0	<u>90.0 $\pm$ 4.2</u>	<u>92.0 $\pm$ 3.8</u>	80.0

In addition, the recommended motifs $\mathcal{R}_q$ from the skill are mapped to a permitted functional-group set $\mathcal{F}_q^+$, and logits for groups outside this set are masked to $-\infty$ before sampling. This constrains the proposal distribution toward task-consistent fragments before execution.

Inference supports multi-step editing: at each step, the policy samples multiple action records from the current molecule, RDKit executes the sampled records, and the highest-reward executed molecule advances to the next step. We use two editing steps ($T = 2$) in all main experiments. The full prompt templates for skill construction, SFT, and inference are given in Appendix B.3.

4 Experiments

4.1 Experimental Settings

Data and Tasks. Training pairs are constructed from the MoleculeSTM pre-training corpus (Liu

et al., 2023), converted into skill-conditioned action records (see Appendix A). Evaluation uses the MoleculeSTM test set of 200 molecules across 14 tasks: eight single-property (logP $\uparrow\downarrow$, QED $\uparrow\downarrow$, tPSA $\uparrow\downarrow$, HBA $\uparrow$, HBD $\uparrow$) and six multi-property tasks combining logP with HBA, HBD, or tPSA.

Metrics and Baselines. We report loose hit ratio (property moves in the correct direction) and strict hit ratio (property change exceeds a task-specific threshold). Baselines include Random, PCA, High Variance (Liu et al., 2022), GS-Mutate, MoleculeSTM-SMILES/Graph (Liu et al., 2023), ChatDrug (Liu et al., 2024), MultiMol (Yu et al., 2025), Chem Reasoner (Sprueill et al., 2024), and RL-Guider (Liu et al., 2025b). Chem Reasoner and RL-Guider are each evaluated with two backbone LLMs (LLaMA and DeepSeek), yielding four baseline rows in the tables.

Table 3: Loose and strict hit ratios on six multi-property tasks. Each task has a loose column and a strict column; a molecule is counted as a hit only if all requested property conditions are met. AVG is the macro average over all reported columns. Δ denotes the optimized property interval.

Method	logP- & HBA+ 0-0	logP- & HBA+ 0.5-1	logP+ & HBA+ 0-0	logP+ & HBA+ 0.5-1	logP- & HBD+ 0-0	logP- & HBD+ 0.5-1	logP+ & HBD+ 0-0	logP+ & HBD+ 0.5-1	logP- & tPSA- 0-0	logP- & tPSA- 0.5-10	logP- & tPSA+ 0-0	logP- & tPSA+ 0.5-10	AVG
Δ													
Random	9.9 $\pm$ 1.0	0.2 $\pm$ 0.3	3.0 $\pm$ 0.4	0.5 $\pm$ 0.3	2.3 $\pm$ 1.2	0.0 $\pm$ 0.0	0.7 $\pm$ 0.6	0.0 $\pm$ 0.0	5.1 $\pm$ 1.2	1.2 $\pm$ 0.7	12.2 $\pm$ 1.1	6.2 $\pm$ 0.6	3.4
PCA	8.6 $\pm$ 2.1	0.5 $\pm$ 0.6	2.0 $\pm$ 0.6	0.0 $\pm$ 0.0	2.2 $\pm$ 1.2	0.0 $\pm$ 0.0	2.0 $\pm$ 0.9	0.0 $\pm$ 0.0	3.5 $\pm$ 0.4	0.7 $\pm$ 0.6	10.4 $\pm$ 2.9	6.2 $\pm$ 2.3	3.0
High Var	9.1 $\pm$ 1.3	0.2 $\pm$ 0.3	2.5 $\pm$ 0.7	0.2 $\pm$ 0.3	4.4 $\pm$ 0.6	0.0 $\pm$ 0.0	1.8 $\pm$ 0.7	0.0 $\pm$ 0.0	4.9 $\pm$ 2.2	0.7 $\pm$ 0.5	13.1 $\pm$ 2.3	6.7 $\pm$ 0.5	3.6
GS-Mutate	14.0 $\pm$ 2.5	0.7 $\pm$ 0.6	7.2 $\pm$ 0.9	0.2 $\pm$ 0.2	13.8 $\pm$ 3.0	0.0 $\pm$ 0.0	5.7 $\pm$ 0.6	0.0 $\pm$ 0.0	8.2 $\pm$ 1.0	0.0 $\pm$ 0.0	19.8 $\pm$ 2.5	4.8 $\pm$ 0.9	6.2
MolSTM-SMILES	27.9 $\pm$ 3.9	8.8 $\pm$ 0.0	8.6 $\pm$ 2.8	2.9 $\pm$ 0.3	33.5 $\pm$ 4.1	10.0 $\pm$ 1.0	17.0 $\pm$ 2.8	2.6 $\pm$ 1.1	35.7 $\pm$ 3.2	19.2 $\pm$ 0.7	44.4 $\pm$ 0.7	28.7 $\pm$ 2.2	19.9
MolSTM-Graph	27.4 $\pm$ 3.4	11.1 $\pm$ 1.8	8.2 $\pm$ 0.8	0.0 $\pm$ 0.0	49.2 $\pm$ 1.7	23.9 $\pm$ 1.1	14.4 $\pm$ 3.4	3.8 $\pm$ 0.7	39.7 $\pm$ 2.3	22.7 $\pm$ 1.9	30.9 $\pm$ 0.6	20.1 $\pm$ 1.3	21.0
ChatDrug	79.6 $\pm$ 0.6	49.6 $\pm$ 2.7	51.6 $\pm$ 3.8	24.9 $\pm$ 4.9	89.3 $\pm$ 1.0	53.6 $\pm$ 5.8	39.9 $\pm$ 3.9	24.2 $\pm$ 2.2	12.9 $\pm$ 2.7	10.4 $\pm$ 5.8	65.3 $\pm$ 2.2	52.9 $\pm$ 2.2	46.2
MultiMol	88.1 $\pm$ 1.5	85.0 $\pm$ 1.2	57.4 $\pm$ 1.5	33.7 $\pm$ 1.3	94.2 $\pm$ 1.0	85.4 $\pm$ 1.1	49.5 $\pm$ 1.4	25.3 $\pm$ 1.2	24.6 $\pm$ 1.5	11.2 $\pm$ 1.0	82.3 $\pm$ 1.3	61.2 $\pm$ 1.4	58.2
LLaMA-C	37.0 $\pm$ 0.0	8.0 $\pm$ 0.0	21.0 $\pm$ 0.0	7.0 $\pm$ 0.0	43.0 $\pm$ 0.0	11.5 $\pm$ 0.0	22.0 $\pm$ 0.0	5.0 $\pm$ 0.0	3.0 $\pm$ 0.0	2.0 $\pm$ 0.0	51.5 $\pm$ 0.0	33.5 $\pm$ 0.0	20.4
LLaMA-RL	50.0 $\pm$ 0.0	37.5 $\pm$ 0.0	37.5 $\pm$ 0.0	24.0 $\pm$ 0.0	45.0 $\pm$ 0.0	20.0 $\pm$ 0.0	23.0 $\pm$ 0.0	8.5 $\pm$ 0.0	8.0 $\pm$ 0.0	3.0 $\pm$ 0.0	44.0 $\pm$ 0.0	38.5 $\pm$ 0.0	28.3
DeepSeek-C	75.0 $\pm$ 0.0	28.5 $\pm$ 0.0	27.0 $\pm$ 0.0	3.0 $\pm$ 0.0	72.0 $\pm$ 0.0	23.5 $\pm$ 0.0	16.0 $\pm$ 0.0	1.5 $\pm$ 0.0	5.5 $\pm$ 0.0	4.5 $\pm$ 0.0	74.5 $\pm$ 0.0	52.5 $\pm$ 0.0	32.0
DeepSeek-RL	77.0 $\pm$ 0.0	63.0 $\pm$ 0.0	63.0 $\pm$ 0.0	6.0 $\pm$ 0.0	71.5 $\pm$ 0.0	68.5 $\pm$ 0.0	9.5 $\pm$ 0.0	10.5 $\pm$ 0.0	8.0 $\pm$ 0.0	5.5 $\pm$ 0.0	79.0 $\pm$ 0.0	73.0 $\pm$ 0.0	44.5
SkillMol	96.0 $\pm$ 2.8	90.0 $\pm$ 4.2	58.0 $\pm$ 7.0	34.0 $\pm$ 6.7	96.0 $\pm$ 2.8	74.0 $\pm$ 6.2	50.0 $\pm$ 7.1	16.0 $\pm$ 5.2	46.0 $\pm$ 7.0	24.0 $\pm$ 6.0	100.0 $\pm$ 0.0	100.0 $\pm$ 0.0	65.3

Table 4: Example skill path for logP decrease. The retrieved rule becomes a task skill, the skill supports a replace action, and the same skill masks functional groups toward polar fragments.

Stage	Example for logP $\downarrow$
Retrieved rule	Polar heteroatom insertion and hydrophobic-group removal are preferred for improving solubility.
Skill text	Replace hydrophobic alkyl substituents with hydroxyl, ether, amine, amide, or carbonyl-containing substituents; avoid alkyl or halogen growth.
Action target	<OP_REPLACE> at an atom-mapped anchor, followed by <FGID> for a polar fragment.
FG constraint	Permit polar groups such as amide, amine, morpholine, sulfonamide, pyridyl, and diazine IDs; suppress tert-butyl, CF ₃ , halogen, and hydrophobic aryl IDs.

Implementation. We use Qwen3-0.6B with LoRA (Hu et al., 2021; Yang et al., 2025), trained for 50 epochs with batch size 512 on two A100 (80GB) GPUs. Inference supports multi-step editing, and we use two editing steps ($T = 2$) in the main setting. Full training and inference details are in Appendix B.

4.2 Experimental Results

We report four groups of results. The main comparison evaluates SkillMol against existing molecular optimization systems. The ablation isolates the training skill, GRPO, and inference-time skill constraint. The backbone diagnostic checks whether scale alone explains the gains. The final analysis inspects latency, property movement, and selected functional groups.

Main Comparison. Tables 1–3 compare SkillMol with prior molecular optimization systems under the same tasks. Loose single-property results

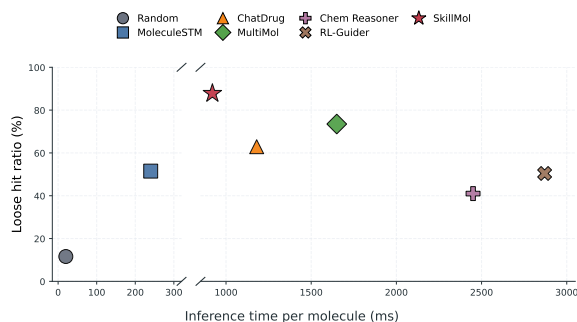


Figure 3: Efficiency-accuracy comparison. The x-axis is approximate inference latency per molecule and the y-axis is loose hit ratio. SkillMol trades more computation than classical edits for higher hit ratio, while avoiding the largest latencies of agentic LLM systems.

Table 5: Summary of the rule-random skill baseline. Scores are average hit ratios in percent; full task-level results are reported in Appendix B.6.

Method	Single Loose	Single Strict	Multi Loose	Multi Strict
Rule-random	37.6	21.6	34.7	6.8
SkillMol	92.5	80.0	74.3	56.3

measure whether edits move properties in the correct direction, while strict results require crossing the task threshold. Multi-property tasks are harder because all objectives must be satisfied simultaneously. SkillMol achieves 92.5% average loose hit ratio and 80.0% strict hit ratio on single-property tasks, and 65.3% average on multi-property tasks. The strict metric is most relevant for our action-based design because edits must be executable and sufficiently large. Weaknesses remain on QED increase (12.0% vs. 30.4% for MultiMol) and tPSA decrease (54.0% vs. 59.9%). Multi-property performance is stronger for compatible objectives and weaker under sharper trade-offs.

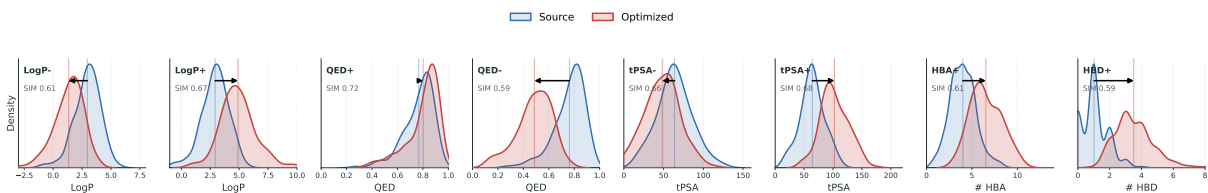


Figure 4: Single-property distributions before and after optimization. Blue curves are source molecules, red curves are optimized molecules, vertical lines mark means, and arrows show the average shift.

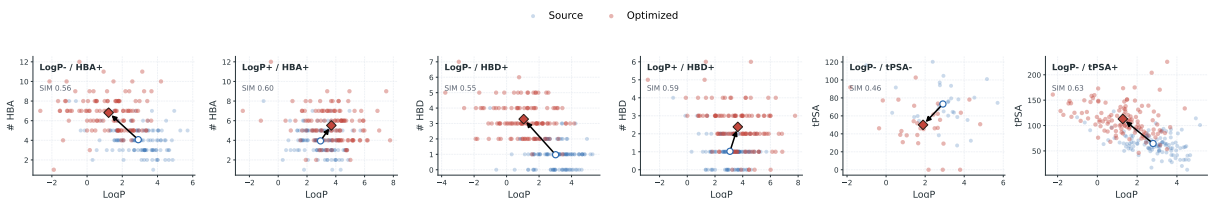


Figure 5: Two-property scatter plots for multi-property objectives. Each panel shows source and optimized molecules in the relevant property dimensions; the arrow connects the source mean to the optimized mean.

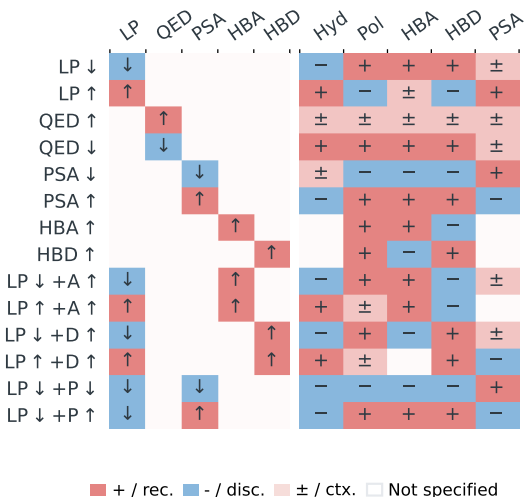


Figure 6: Task-skill alignment across optimization objectives. Left: requested property directions. Right: motif preferences induced by the task skill. LP, tPSA, A, and D denote logP, tPSA, HBA, and HBD.

Rule-Only Skill Baseline. Table 5 tests whether the skill can be used as a direct editing rule. The rule-random baseline samples a recommended functional group or removal rule from the skill and applies it at a random feasible-looking site. This rule-only use of skills can sometimes move easy single properties in the right direction, but it fails under stricter thresholds and especially under multi-property objectives, where one edit must satisfy coupled and sometimes competing requirements. Thus, the skill is better understood as a chemical prior for a learned action policy, not as a complete editing algorithm by itself. More detailed per-task

Table 6: Ablation by task type. Single columns average over the eight single-property tasks; Multi columns average over the six multi-property tasks. Loose and Strict are hit ratios in percent.

Variant	Single Loose	Single Strict	Multi Loose	Multi Strict
SkillMol	92.5	80.0	74.3	56.3
w/o skill in SFT	59.9	40.3	36.3	17.0
w/o GRPO	86.7	75.3	71.0	52.3
w/o skill at inference	85.0	70.0	68.0	45.0

results are provided in Appendix B.6.

Ablation Study. Table 6 isolates three components: the task skill during SFT, GRPO alignment, and the inference-time skill constraint. The largest drop by far comes from removing skills during training: strict hit ratio falls from 80.0% to 40.3% on single-property tasks and from 56.3% to 17.0% on multi-property tasks, confirming that the task skill is a genuine training signal rather than a decoding hint. Removing GRPO gives a smaller but consistent reduction, especially under the strict criterion. Removing the inference-time skill constraint causes a further drop (80.0% $\rightarrow$ 70.0% strict single-property; 56.3% $\rightarrow$ 45.0% strict multi-property); crucially, this variant still substantially outperforms the no-skill-in-SFT variant (40.3% / 17.0%), confirming that skill-based SFT internalizes the chemical prior into the model weights and the inference-time constraint provides an additional complementary benefit.

Backbone Size. Table 7 evaluates whether a larger Qwen3 backbone improves the action policy under the same skill-guided editing interface.

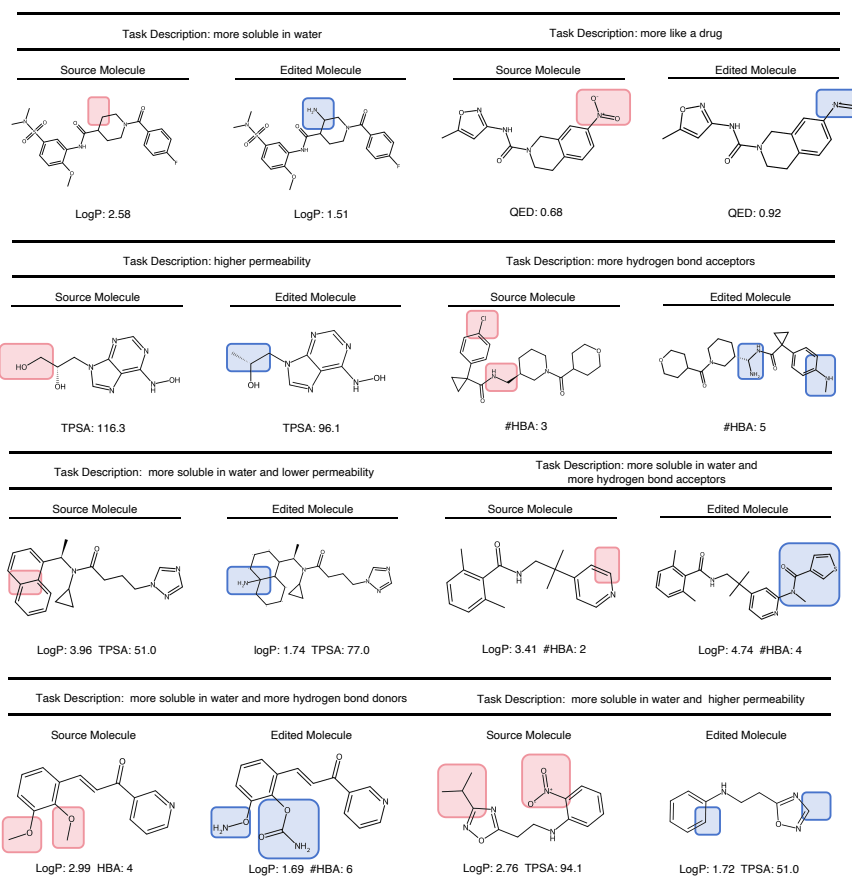


Figure 8: Case study of SkillMol editing trajectories. Each row shows a source molecule, the predicted edit action, and the optimized molecule with property values before and after the edit. Cases are selected to illustrate how the task skill guides functional-group choice across different optimization objectives.

Case Study Figure 8 illustrates representative editing trajectories produced by SkillMol on selected molecules from the test set. Each example shows the source molecule, the predicted edit action (operation, anchor atom, and functional group), and the resulting optimized molecule, together with the property values before and after the edit. The cases are drawn from tasks with different optimization directions to highlight how the task skill shapes the choice of functional group. For logP decrease, the policy consistently selects polar substitutions such as amide or hydroxyl at hydrophobic anchor sites. For tPSA increase, the policy inserts heteroatom-containing fragments such as morpholine or sulfonamide. For multi-property tasks, the selected edits reflect the combined skill constraint: the inserted fragment must satisfy both property directions simultaneously, which is visible in the logP↓+HBA↑ examples where ether or carbonyl groups are preferred over pure alkyl or pure donor

motifs. These qualitative patterns are consistent with the quantitative skill-alignment analysis in Figure 6 and confirm that the learned policy follows the chemical prior encoded in the task skill.

5 Conclusion

We presented SkillMol, a skill-conditioned framework for molecular optimization that combines compact task skills with an atom-anchored action interface for executable molecular edits. Results show that task skills are the main driver of performance, while inference-time functional-group constraints provide smaller additional gains. Scaling the backbone up to 8B brings only limited improvement, suggesting that the skill-action interface matters more than model size. Although effective on benchmark property-editing tasks, broader lead-optimization applications still require further medicinal-chemistry validation.

Limitations

SkillMol is evaluated on benchmark property-editing tasks rather than full medicinal-chemistry optimization campaigns. The reported properties are only proxy metrics and do not cover factors such as synthesis feasibility, assay uncertainty, or expert review. In addition, the learned skills act as coarse task-level priors and may be less effective for objectives without clear functional-group patterns or for scaffold-specific optimization scenarios.

Acknowledgments

This work was supported by Zhejiang Province Major Science and Technology Plan Project (2026C01021), the Provincial Key Laboratory Program (2025E10048), the Central Government-Guided Fund for Local Science and Technology Development (2025ZY01034), and Zhejiang Province Science and Technology Plan Project (2025C01128).

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Sungsoo Ahn, Junsu Kim, Hankook Lee, and Jinwoo Shin. 2020. Guiding deep molecular optimization with genetic exploration. *Advances in neural information processing systems*, 33:12008–12021.
- John A Arnott and Sonia Lobo Planey. 2012. The influence of lipophilicity in drug discovery and design. *Expert opinion on drug discovery*, 7(10):863–875.
- Viraj Bagal, Rishal Aggarwal, PK Vinod, and U Deva Priyakumar. Liggpt: Molecular generation using a transformer-decoder model. *ChemRxiv*.
- Mariana Pegrucci Barcelos, Suzane Quintana Gomes, Leonardo Bruno Federico, Isaque Antonio Galindo Francischini, Lorane Izabel da Silva Hage-Melim, Guilherme Martins Silva, and Carlos Henrique Tomich de Paula da Silva. 2022. Lead optimization in drug discovery. In *Research topics in bioactivity, environment and energy: experimental and theoretical tools*, pages 481–500. Springer.
- G Richard Bickerton, Gaia V Paolini, J  r  my Besnard, Sorel Muresan, and Andrew L Hopkins. 2012. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98.
- Ziqi Chen, Martin Renqiang Min, Srinivasan Parthasarathy, and Xia Ning. 2021. A deep generative model for molecule optimization via one fragment modification. *Nature machine intelligence*, 3(12):1040–1049.
- Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, and 1 others. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. See <https://vicuna.lmsys.org> (accessed 14 April 2023), 2(3):6.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Peter Ertl, Bernhard Rohde, and Paul Selzer. 2000. Fast calculation of molecular polar surface area as a sum of fragment-based contributions and its application to the prediction of drug transport properties. *Journal of medicinal chemistry*, 43(20):3714–3717.
- Tianfan Fu, Wenhao Gao, Cao Xiao, Jacob Yasonik, Connor W Coley, and Jimeng Sun. 2021a. Differentiable scaffolding tree for molecular optimization. *arXiv preprint arXiv:2109.10469*.
- Tianfan Fu, Cao Xiao, Xinhao Li, Lucas M Glass, and Jimeng Sun. 2021b. Mimoso: Multi-constraint molecule sampling for molecule optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 125–133.
- Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bis-san Al-Lazikani, and 1 others. 2012. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(D1):D1100–D1107.
- Ed Griffen, Andrew G Leach, Graeme R Robb, and Daniel J Warner. 2011. Matched molecular pairs as a medicinal chemistry tool: miniperspective. *Journal of medicinal chemistry*, 54(22):7739–7750.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shiron Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.
- Michael M Hann. 2011. Molecular obesity, potency and other addictions in drug discovery. *MedChemComm*, 2(5):349–355.
- J He, F Mattsson, M Forsberg, EJ Bjerrum, O Engkvist, C Tyrchan, W Czechtizky, and 1 others. 2021. Transformer neural network for structure constrained molecular optimization. *ChemRxiv*.

- 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. *arXiv preprint arXiv:2106.09685*.
- Jan H Jensen. 2019. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572.
- Wengong Jin, Kevin Yang, Regina Barzilay, and Tommi Jaakkola. 2018. Learning multimodal graph-to-graph translation for molecular optimization. *arXiv preprint arXiv:1812.01070*.
- Peter W Kenny. 2022. Hydrogen-bond donors in drug design. *Journal of medicinal chemistry*, 65(21):14261–14275.
- Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, and 1 others. 2021. Pubchem in 2021: new data content and improved web interfaces. *Nucleic acids research*, 49(D1):D1388–D1395.
- Greg Landrum and 1 others. 2013. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8(31.10):5281.
- Andrew G Leach, Huw D Jones, David A Cosgrove, Peter W Kenny, Linette Ruston, Philip MacFaul, J Matthew Wood, Nicola Colclough, and Brian Law. 2006. Matched molecular pairs as a guide in the optimization of pharmaceutical properties; a study of aqueous solubility, plasma protein binding and oral exposure. *Journal of medicinal chemistry*, 49(23):6672–6682.
- Paul D Leeson and Brian Springthorpe. 2007. The influence of drug-like concepts on decision-making in medicinal chemistry. *Nature reviews Drug discovery*, 6(11):881–890.
- Shiyu Li, Ziqi Yan, Zhihao Wu, Jielong Lu, Weiran Liao, Jiajun Yu, Genjie Li, Zeyu Chu, Jiajun Bu, and Haishuai Wang. 2026. Deepbd: A grounded agentic workflow for variant prioritization and diagnosis of genetic birth defects. *arXiv preprint arXiv:2606.24779*.
- Yaozhen Liang, Xiao Liu, Jiajun Yu, Zhouhua Fang, Qunsheng Zou, Linghan Zheng, Yong Li, Zhiwei Liu, and Haishuai Wang. 2025. Explain: Enhancing retrieval-augmented generation with entity summary. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 1520–1529.
- Christopher A Lipinski, Franco Lombardo, Beryl W Dominy, and Paul J Feeney. 1997. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced drug delivery reviews*, 23(1-3):3–25.
- Shengchao Liu, Weili Nie, Chengpeng Wang, Jiarui Lu, Zhuoran Qiao, Ling Liu, Jian Tang, Chaowei Xiao, and Animashree Anandkumar. 2023. Multi-modal molecule structure–text model for text-based retrieval and editing. *Nature Machine Intelligence*, 5(12):1447–1457.
- Shengchao Liu, Chengpeng Wang, Weili Nie, Hanchen Wang, Jiarui Lu, Bolei Zhou, and Jian Tang. 2022. Graphcgc: Unsupervised discovery of steerable factors in graphs. *Openreview*.
- Shengchao Liu, Jiong Xiao Wang, Yijin Yang, Chengpeng Wang, Ling Liu, Hongyu Guo, and Chaowei Xiao. 2024. Conversational drug editing using retrieval and domain feedback. In *The Twelfth International Conference on Learning Representations*.
- Xuefeng Liu, Songhao Jiang, Ian Foster, Jinbo Xu, and Rick Stevens. 2025a. Scaffoldgpt: A scaffold-based gpt model for drug optimization. *arXiv preprint arXiv:2502.06891*.
- Xufeng Liu, Yixuan Ding, Jingxiang Qu, Yichi Zhang, Wenhan Gao, and Yi Liu. 2025b. RL-guider: Leveraging historical decisions and feedback for drug editing with large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 13121–13138.
- Zihang Liu, Fang Zhouhua, Hui Liu, Zhiwei Liu, Yong Li, and Haishuai Wang. 2026. Rfs-guard: Detecting reasoning hallucinations via cross-phase routing focus in large reasoning models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19371–19385.
- Pär Matsson and Jan Kihlberg. 2017. How big is too big for cell permeability? *Journal of Medicinal Chemistry*, 60(5):1662–1664.
- Nicholas A Meanwell. 2014. The influence of bioisosteres in drug design: tactical applications to address developability problems. In *Tactics in Contemporary Drug Design*, pages 283–381. Springer.
- Jesse G Meyer, Ryan J Urbanowicz, Patrick CN Martin, Karen O’Connor, Ruowang Li, Pei-Chen Peng, Tiffani J Bright, Nicholas Tatonetti, Kyoung Jae Won, Graciela Gonzalez-Hernandez, and 1 others. 2023. Chatgpt and large language models in academia: opportunities and challenges. *BioData Mining*, 16(1):20.
- Adrian Mirza, Nawaf Alampara, Sreekanth Kunchapu, Martiño Ríos-García, Benedict Emoekabu, Aswanth Krishnan, Tanya Gupta, Mara Schilling-Wilhelmi, Macjonathan Okereke, Anagha Aneesh, and 1 others. 2024. Are large language models superhuman chemists? *arXiv preprint arXiv:2404.01475*.
- Tamanna Narsinghani and Rajesh Sharma. 2014. Lead optimization on conventional non-steroidal anti-inflammatory drugs: an approach to reduce gastrointestinal toxicity. *Chemical biology & drug design*, 84(1):1–23.

- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.
- Henry W Sprueill, Carl Edwards, Khushbu Agarwal, Mariefel V Olarte, Udishnu Sanyal, Conrad Johnston, Hongbin Liu, Heng Ji, and Sutanay Choudhury. 2024. Chemreasoner: Heuristic search over a large language model’s knowledge space using quantum-chemical feedback. *arXiv preprint arXiv:2402.10980*.
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models.*, 3(6):7.
- Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- Daniel F Veber, Stephen R Johnson, Hung-Yuan Cheng, Brian R Smith, Keith W Ward, and Kenneth D Kopple. 2002. Molecular properties that influence the oral bioavailability of drug candidates. *Journal of medicinal chemistry*, 45(12):2615–2623.
- Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.
- Zhenxing Wu, Odin Zhang, Xiaorui Wang, Li Fu, Huifeng Zhao, Jike Wang, Hongyan Du, Dejun Jiang, Yafeng Deng, Dongsheng Cao, and 1 others. 2024. Leveraging language model for advanced multiproperty molecular optimization via prompt engineering. *Nature Machine Intelligence*, 6(11):1359–1369.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- Geyan Ye, Xibao Cai, Houtim Lai, Xing Wang, Junhong Huang, Longyue Wang, Wei Liu, and Xiangxiang Zeng. 2025. Drugassist: A large language model for molecule optimization. *Briefings in Bioinformatics*, 26(1):bbae693.
- Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. 2018. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31.
- Jiajun Yu, Zhihao Wu, Yizhen Zheng, Shirui Pan, Jiajun Bu, and Haishuai Wang. 2026a. Uniedit: A graph-based moe alternative to sequence generation for molecular editing. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 12739–12750.
- Jiajun Yu, Linghan Zheng, Hui Liu, Jiayuan Dong, Yue Shen, Zhiwei Liu, Yaozhen Liang, Yong Li, and Haishuai Wang. 2026b. Code-based english models reveal surprising performance on chinese qa pair extraction task. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 5060–5066.
- Jiajun Yu, Yizhen Zheng, Huan Yee Koh, Shirui Pan, Tianyue Wang, and Haishuai Wang. 2025. Collaborative expert llms guided multi-objective molecular optimization. *arXiv preprint arXiv:2503.03503*.
- Liying Zhang, Hongyao Zhu, Alan Mathiowetz, and Hua Gao. 2011. Deep understanding of structure–solubility relationship for a diverse set of organic compounds using matched molecular pairs. *Bioorganic & medicinal chemistry*, 19(19):5763–5770.
- Tianyun Zhang, Jiajun Yu, Shang Lou, Yaozhen Liang, Yue Liang, Zhekai Li, Haishuai Wang, Shanshan Pei, and Ning Shen. 2025. Systematic discovery of single-cell protein networks in cancer with shusi. *bioRxiv*, pages 2025–04.
- 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.
- Yizhen Zheng, Huan Yee Koh, Maddie Yang, Li Li, Lauren T May, Geoffrey I Webb, Shirui Pan, and George Church. 2024. Large language models in drug discovery and development: From disease mechanisms to clinical trials. *arXiv preprint arXiv:2409.04481*.
- 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 Dataset Construction

The SkillMol training set is derived from the same scaffold-preserving molecular editing construction used to create the supervised molecular editing pairs, but the final supervision is converted into text-form action records rather than graph-classification

labels. The starting pool is the training split of the MoleculeSTM pre-training corpus (Liu et al., 2023). Let $\mathcal{S}$ denote this base SMILES pool. To reduce redundancy and cover a broad chemical region, we compute Morgan fingerprints and use Tanimoto distance to select a diverse subset. For two molecules x_i and x_j with Morgan fingerprints f_i and f_j , where $\langle f_i, f_j \rangle = \sum_k f_{i,k} f_{j,k}$ is the number of common set bits, similarity and distance are defined as

$$\begin{aligned} \text{sim}(x_i, x_j) &= \frac{\langle f_i, f_j \rangle}{\|f_i\|_1 + \|f_j\|_1 - \langle f_i, f_j \rangle}, \\ d(x_i, x_j) &= 1 - \text{sim}(x_i, x_j). \end{aligned} \quad (7)$$

Starting from an initial selected set $\mathcal{S}_k$, the next molecule is chosen as the candidate whose minimum distance to the current selected set is largest:

$$\mathcal{S}_{k+1} = \mathcal{S}_k \cup \left\{ \arg \max_{x \in \mathcal{S} \setminus \mathcal{S}_k} \min_{u \in \mathcal{S}_k} d(x, u) \right\}. \quad (8)$$

This step yields a chemically diverse source-molecule set for generating editing supervision.

For each selected source molecule x , candidate target molecules are generated by scaffold-constrained structural editing. We extract the Murcko scaffold $M(x)$, identify scaffold-adjacent anchors, and apply a single-step edit transformation

$$y = \mathcal{T}(x; a, o, g), \quad (9)$$

where a is the atom anchor, $o \in \{\text{add}, \text{remove}, \text{replace}\}$ is the edit operation, and g is a functional group from a curated library $\mathcal{F}$. The library contains 27 medicinal-chemistry fragments, including amides, sulfonamides, heterocycles, alkyl groups, halogens, and polar substituents. Each retained pair (x, y) corresponds to an executable local modification that preserves the Murcko scaffold rather than an arbitrary target molecule.

After generating candidate pairs, we compute five molecular properties with RDKit: QED, logP, tPSA, HBA, and HBD. The property vector and property change are

$$\begin{aligned} \mathbf{p}(x) &= [\text{QED}, \text{logP}, \text{tPSA}, \text{HBA}, \text{HBD}]^\top, \\ \Delta \mathbf{p} &= \mathbf{p}(y) - \mathbf{p}(x). \end{aligned} \quad (10)$$

For a task t with optimized property set $\mathcal{P}_t$ and direction vector $\mathbf{s}_t$, where $s_{t,k} = +1$ denotes increasing property k and $s_{t,k} = -1$ denotes decreasing it,

a pair is retained when it satisfies the task direction:

$$\bigwedge_{k \in \mathcal{P}_t} (s_{t,k} \Delta p_k > \epsilon), \text{score}_t(x, y) = \sum_{k \in \mathcal{P}_t} |\Delta p_k|. \quad (11)$$

Pairs are ranked by $\text{score}_t(x, y)$, the top fraction is retained, and operation and fragment frequencies are balanced to avoid overrepresenting a small number of easy edits.

The final difference from the graph-supervised construction is the representation of each retained example. Instead of using node-level operation labels directly, SkillMol converts every pair into a text instruction and a text target. The input text contains the source molecule, the optimization objective, the desired optimization direction, and the literature-retrieved skill s_t . The target text is an ordered action sequence: $y^{\text{text}} = [\langle \text{OP} \rangle, o, \langle \text{ANCHOR} \rangle, a, \langle \text{REMOVE} \rangle, r, \langle \text{FGID} \rangle, g]$ where r is included only for destructive or replacement edits and g is included only for constructive edits. Thus, all SkillMol SFT and GRPO data are built from the same source-target molecular editing pairs, but the model is trained to emit serialized edit decisions in natural-language-compatible text form.

B Experimental Setup Details

Training configuration. All models are trained with LoRA adaptation on top of the Qwen3-0.6B backbone. SFT uses a batch size of 512 and runs for 50 epochs with the AdamW optimizer. GRPO variants are initialized from the SFT checkpoint and fine-tuned with the same batch size for an additional 10 epochs, using a group size of $K = 8$ sampled trajectories per prompt. The LoRA rank is set to 16 with $\alpha = 32$, applied to the query and value projection matrices. All hyperparameters are selected on a held-out validation split and kept fixed across all backbone sizes.

Inference configuration. Unless otherwise specified, inference uses skill-FG search with two editing steps ($T = 2$). At each step, actions are sampled with temperature 0.8 and nucleus sampling $p = 0.95$, and the highest-reward molecule returned by the RDKit executor becomes the next input.

Hardware and runtime. All experiments are conducted on a computing cluster equipped with two NVIDIA A100 (80GB) GPUs and an Intel Xeon Platinum CPU. SFT for the 0.6B backbone

completes in approximately 4 hours; GRPO alignment requires an additional 6 hours. Inference over the full 200-molecule test set across all 14 tasks takes approximately 2 hours per model configuration.

Statistical reliability. We report the mean and standard deviation across 5 independent runs with different random seeds for SkillMol and all ablation variants. For Chem Reasoner and RL-Guider, we report results directly from their respective papers, which provided metrics from a single experimental run; these baselines therefore appear without standard deviation in the tables.

B.1 Molecular Quality Comparison

Table 8 compares SkillMol and MultiMol on four molecular quality metrics beyond hit ratio: novelty (fraction of optimized molecules not identical to the source), SA delta (change in synthetic accessibility score, lower is better), diversity (average pairwise Tanimoto distance among optimized molecules), and scaffold retention (fraction of optimized molecules sharing the Murcko scaffold with the source).

Table 8: Molecular quality metrics for SkillMol and MultiMol. SA delta is the mean change in synthetic accessibility score (positive means harder to synthesize). Higher is better for all metrics except SA delta.

Method	Novelty $\uparrow$	SA delta $\downarrow$	Diversity $\uparrow$	Scaffold $\uparrow$
MultiMol	93.13%	+0.2010	0.7833	93.44%
SkillMol	99.87%	+0.1851	0.8622	83.11%

B.2 Full Algorithm

Algorithm 1 summarizes the complete SkillMol pipeline, covering offline skill construction, skill-based SFT, optional GRPO alignment, and skill-FG guided search at inference time.

B.3 Prompt Templates

We use three prompt templates: one for skill construction (offline), one for SFT/GRPO training, and one for inference. All three share the same action output format.

Prompt 1: Skill Construction (offline, per task)

You are a medicinal chemistry expert. Below are excerpts from highly-cited papers on molecular property optimization.
Task: {property_direction} (e.g., "decrease logP")

Literature excerpts:
{paper_excerpts}

Based on these excerpts, extract a structured skill for this task:

1. Recommended structural motifs or edits (what to add/replace/remove)
2. Discouraged structural motifs or edits (what to avoid)

Output format:

RECOMMENDED: <comma-separated list of motifs>
DISCOURAGED: <comma-separated list of motifs>

Prompt 2: SFT / GRPO Training

You are a molecular editor. Given a molecule and an optimization objective, predict a single executable edit action.

Molecule (with atom IDs):
{smiles_with_atom_ids}

Objective: {property_direction}

Chemical skill for this objective:

- Recommended: {recommended_motifs}
- Discouraged: {discouraged_motifs}

Output a single edit action in the following format:

<EDIT_SET> <OP> {ADD|REMOVE|REPLACE} <ANCHOR>
{atom_id}
[<REMOVE> {atom_id}] [<FGID>
{functional_group_id}]

Prompt 3: Inference (multi-step search)

You are a molecular editor. Given a molecule and an optimization objective, predict a single executable edit action.

Molecule (with atom IDs):
{current_smiles_with_atom_ids}

Objective: {property_direction}

Chemical skill for this objective:

- Recommended: {recommended_motifs}
- Discouraged: {discouraged_motifs}

[Functional-group logits are masked to permitted set $\mathcal{F}_q^+$ before sampling.]

Output a single edit action:

<EDIT_SET> <OP> {ADD|REMOVE|REPLACE} <ANCHOR>
{atom_id}
[<REMOVE> {atom_id}] [<FGID>
{functional_group_id}]

(Repeated for T steps; at each step the highest-reward executed molecule becomes the next input.)

B.4 Skill Library

The task skills are constructed once and then reused for all molecules in the corresponding task. The evidence pool combines PubMed-indexed medicinal-chemistry abstracts, lead-optimization reviews, drug-likeness and permeability rules, and public compound-annotation resources such as PubChem and ChEMBL (Kim et al., 2021; Gaulton et al., 2012; Barcelos et al., 2022; Lipinski et al., 1997; Narsinghani and Sharma, 2014). For each property direction, Codex-GPT5.5 high retrieves pas-

sages with property and action keywords, keeps only direction-consistent evidence, and summarizes the retained passages into recommended transformations and avoid rules. Each skill is manually checked for consistency and then used as prompt context during SFT and converted into functional-group constraints during decoding. Appendix Table 9 lists the complete task-level skill set used in the experiments.

B.5 Skill Evidence Titles

To make the source of each task skill explicit, Table 10 back-maps the final skill text in Table 9 to representative retrieved medicinal-chemistry papers. The table reports real paper titles used as evidence for each property direction, but it is not intended to provide an exhaustive citation set for every individual motif.

B.6 Rule-Random Skill Baseline

We also test a deliberately simple non-LLM baseline to separate the value of the task skill from the value of learned atom selection. For each molecule and task, the baseline samples exactly one edit from the skill-derived rule set with a fixed random seed. For constructive tasks, it randomly chooses one recommended functional group for the task and one heavy-atom anchor, then asks RDKit to apply the ADD edit. For tasks whose skill recommends removal, it randomly chooses one removable non-scaffold or terminal branch and removes it. Invalid valence or unsanitizable products are counted as failures. There is no LLM, no retry, no candidate enumeration, and no property-based reranking.

This baseline is intentionally weak, but it is informative. The rules alone sometimes move easy single properties in the right direction, especially logP $\uparrow$ and tPSA $\uparrow$, but performance drops sharply under strict and multi-property criteria. The average valid-product rate is only 54.4%, because a randomly selected atom often cannot accept the sampled group. The contrast with SkillMol shows that the skill is useful as a prior, but the main gain comes from learning where and how to apply that prior rather than randomly attaching a recommended motif.

Algorithm 1 SkillMol: Skill-Conditioned Molecular Editing

Offline (once per task):

- 1: Use Codex-GPT5.5 high to search web and PubMed-style sources with property-direction keywords
- 2: Prompt Codex-GPT5.5 high to extract skill $s_q = (\mathcal{R}_q, \mathcal{B}_q)$ from retained evidence
- 3: Map $\mathcal{R}_q$ and $\mathcal{B}_q$ to functional-group constraint sets $\mathcal{F}_q^+$ and $\mathcal{F}_q^-$

Training:

- 4: // *Skill-based SFT*
- 5: **for** each training example (x_i, q_i, y_i) **do**
- 6: Assign atom-map IDs to all atoms in x_i
- 7: Construct prompt: source SMILES + atom IDs + objective q_i + skill s_{q_i}
- 8: Optimize $\mathcal{L}_{\text{SFT}}(\theta)$ to predict action record y_i
- 9: **end for**
- 10: // *Optional GRPO alignment*
- 11: **for** each prompt $c = (x_0, q, s_q)$ **do**
- 12: Sample K action trajectories $\{\tau_k\}$ from π_θ
- 13: Execute each τ_k with RDKit executor E ; compute reward r_k
- 14: Compute normalized advantage $\hat{A}_k$ within the group
- 15: Update θ by maximizing $\mathcal{J}_{\text{GRPO}}(\theta)$
- 16: **end for**

Inference (per source molecule x_0 , task q):

- 17: Set current molecule $x \leftarrow x_0$
 - 18: **for** step $t = 1, \dots, T$ **do**
 - 19: Assign atom-map IDs to x ; construct prompt with skill s_q
 - 20: Apply FG logit mask using $\mathcal{F}_q^+$; sample B action records
 - 21: Execute sampled action records with RDKit; score returned molecules by reward
 - 22: $x \leftarrow \arg \max$ reward among executed candidates
 - 23: **end for**
 - 24: **return** x
-

Objective	Preferred motifs / edits	Discouraged motifs / edits
<i>Single-property skills</i>		
logP↓	Add polar heteroatoms; replace hydrophobic alkyl with hydroxyl, ether, amine, amide, or carbonyl; shorten alkyl chains.	Alkyl growth, uncompensated halogens, removing polar groups, hydrophobic expansion.
logP↑	Add alkyl, cycloalkyl, phenyl, benzyl, halogen, or CF ₃ ; mask polar donors by alkylation.	Hydroxyl, carboxyl, primary amine, sulfonamide, ionic groups, strong TPSA increase.
QED↑	Trim excessive size; moderate LogP/TPSA; use stable bioisosteres; balance HBA/HBD counts.	Large MW increase, extreme polarity/lipophilicity, excessive HBA/HBD, unstable motifs, scaffold damage.
QED↓	Add bulky hydrophobic groups, excessive polar groups, flexible chains, or complex substituents.	Conservative edits that move properties back toward drug-like ranges.
TPSA↓	Remove or mask hydroxyl, amine, carboxyl, sulfonamide, or amide; replace hydroxyl with methoxy/alkyl.	Adding high-TPSA groups, increasing HBA/HBD, introducing redundant heteroatoms.
TPSA↑	Add hydroxyl, amine, amide, carboxyl, sulfonamide, urea, carbamate, carbonyl, heteroaryl, morpholine, or piperazine.	Masking polar groups, removing heteroatoms, adding only alkyl/halogen, raising LogP without TPSA gain.
HBA↑	Add ether oxygen, carbonyl, tertiary amine, heteroaryl, nitrile, sulfone, sulfoxide, or heteroatom linker.	Donor-only groups, protonated motifs that do not count as acceptors, removing acceptors.
HBD↑	Add hydroxyl, primary/secondary amine, amide NH, urea, sulfonamide, carbamate, hydrazide, or NH heterocycle.	N-methylation, masking OH/NH, acceptor-only additions, removing donors.
<i>Multi-property skills</i>		
logP↓+HBA↑	Replace hydrophobic groups with ether, carbonyl, tertiary amine, amide, ester, heteroaryl, morpholine, nitrile, or sulfone.	Donor-only groups, alkyl/halogen growth, hydrophobic HBA additions, LogP increase.
logP↑+HBA↑	Add lipophilic acceptors such as methoxy, ethoxy, aryl ether, ester, ketone, tertiary amide, nitrile, or halogenated heteroaryl.	Highly polar acceptors, hydroxyl, primary amine, carboxyl, sulfonamide, unnecessary HBD increase.
logP↓+HBD↑	Add compact donor-containing polar groups: hydroxyl, amine, amide NH, sulfonamide NH, carbamate NH, or urea NH.	N-methylating donors, alkyl/halogen additions, HBA-only additions, masking OH/NH.
logP↑+HBD↑	Add one donor in a hydrophobic context, e.g., anilide, secondary amide, sulfonamide, NH heterocycle, or aryl/alkyl-supported amine.	Multiple OH/NH without hydrophobic compensation, carboxylic acids, excessive TPSA, donor-only polar edits.
logP↓+TPSA↓	Remove bulky hydrophobic substituents while masking/removing polar donors; replace high-TPSA motifs with lower-TPSA bioisosteres.	Lowering LogP by adding polar atoms, decreasing TPSA by hydrophobic growth, new heteroatoms, single-objective edits.
logP↓+TPSA↑	Add polar groups: hydroxyl, amine, amide, carboxyl, sulfonamide, urea, carbamate, carbonyl, polar heteroaryl, morpholine, piperazine.	Masking polar groups, alkyl/aryl/halogen/CF ₃ without polar compensation, removing heteroatoms, decreasing TPSA.

Table 9: Literature-retrieved task skills used by SkillMol. Skills are grouped by single-property and multi-property objectives. Each row lists the preferred edit motifs and the motifs discouraged when constructing SFT prompts and inference-time functional-group constraints.

Skill	Representative retrieved paper titles
logP↓	<i>Experimental and Computational Approaches to Estimate Solubility and Permeability in Drug Discovery and Development Settings</i> (Lipinski et al., 1997); <i>The Influence of Lipophilicity in Drug Discovery and Design</i> (Arnott and Planey, 2012); <i>Matched Molecular Pairs as a Guide in the Optimization of Pharmaceutical Properties; a Study of Aqueous Solubility, Plasma Protein Binding and Oral Exposure</i> (Leach et al., 2006).
logP↑	<i>The Influence of Lipophilicity in Drug Discovery and Design</i> (Arnott and Planey, 2012); <i>Molecular Obesity, Potency and Other Additions in Drug Discovery</i> (Hann, 2011); <i>The Influence of Drug-like Concepts on Decision-Making in Medicinal Chemistry</i> (Leeson and Springthorpe, 2007).
QED↑	<i>Quantifying the Chemical Beauty of Drugs</i> (Bickerton et al., 2012); <i>The Influence of Drug-like Concepts on Decision-Making in Medicinal Chemistry</i> (Leeson and Springthorpe, 2007); <i>Molecular Obesity, Potency and Other Additions in Drug Discovery</i> (Hann, 2011).
QED↓	<i>Quantifying the Chemical Beauty of Drugs</i> (Bickerton et al., 2012); <i>Molecular Obesity, Potency and Other Additions in Drug Discovery</i> (Hann, 2011); <i>The Influence of Drug-like Concepts on Decision-Making in Medicinal Chemistry</i> (Leeson and Springthorpe, 2007).
TPSA↓	<i>Fast Calculation of Molecular Polar Surface Area as a Sum of Fragment-Based Contributions and Its Application to the Prediction of Drug Transport Properties</i> (Ertl et al., 2000); <i>Molecular Properties That Influence the Oral Bioavailability of Drug Candidates</i> (Veber et al., 2002); <i>Hydrogen-Bond Donors in Drug Design</i> (Kenny, 2022).
TPSA↑	<i>Fast Calculation of Molecular Polar Surface Area as a Sum of Fragment-Based Contributions and Its Application to the Prediction of Drug Transport Properties</i> (Ertl et al., 2000); <i>Molecular Properties That Influence the Oral Bioavailability of Drug Candidates</i> (Veber et al., 2002); <i>Experimental and Computational Approaches to Estimate Solubility and Permeability in Drug Discovery and Development Settings</i> (Lipinski et al., 1997).
HBA↑	<i>Molecular Properties That Influence the Oral Bioavailability of Drug Candidates</i> (Veber et al., 2002); <i>Experimental and Computational Approaches to Estimate Solubility and Permeability in Drug Discovery and Development Settings</i> (Lipinski et al., 1997); <i>Hydrogen-Bond Donors in Drug Design</i> (Kenny, 2022).
HBD↑	<i>Molecular Properties That Influence the Oral Bioavailability of Drug Candidates</i> (Veber et al., 2002); <i>Hydrogen-Bond Donors in Drug Design</i> (Kenny, 2022); <i>The Influence of Drug-like Concepts on Decision-Making in Medicinal Chemistry</i> (Leeson and Springthorpe, 2007).
logP↓+HBA↑	<i>The Influence of Lipophilicity in Drug Discovery and Design</i> (Arnott and Planey, 2012); <i>Molecular Properties That Influence the Oral Bioavailability of Drug Candidates</i> (Veber et al., 2002); <i>Matched Molecular Pairs as a Medicinal Chemistry Tool</i> (Griffen et al., 2011).
logP↑+HBA↑	<i>The Influence of Lipophilicity in Drug Discovery and Design</i> (Arnott and Planey, 2012); <i>Molecular Properties That Influence the Oral Bioavailability of Drug Candidates</i> (Veber et al., 2002); <i>The Influence of Bioisosteres in Drug Design: Tactical Applications to Address Developability Problems</i> (Meanwell, 2014).
logP↓+HBD↑	<i>The Influence of Lipophilicity in Drug Discovery and Design</i> (Arnott and Planey, 2012); <i>Hydrogen-Bond Donors in Drug Design</i> (Kenny, 2022); <i>Experimental and Computational Approaches to Estimate Solubility and Permeability in Drug Discovery and Development Settings</i> (Lipinski et al., 1997).
logP↑+HBD↑	<i>The Influence of Lipophilicity in Drug Discovery and Design</i> (Arnott and Planey, 2012); <i>Hydrogen-Bond Donors in Drug Design</i> (Kenny, 2022); <i>The Influence of Drug-like Concepts on Decision-Making in Medicinal Chemistry</i> (Leeson and Springthorpe, 2007).
logP↓+TPSA↓	<i>The Influence of Lipophilicity in Drug Discovery and Design</i> (Arnott and Planey, 2012); <i>Fast Calculation of Molecular Polar Surface Area as a Sum of Fragment-Based Contributions and Its Application to the Prediction of Drug Transport Properties</i> (Ertl et al., 2000); <i>Matched Molecular Pairs as a Guide in the Optimization of Pharmaceutical Properties; a Study of Aqueous Solubility, Plasma Protein Binding and Oral Exposure</i> (Leach et al., 2006).
logP↓+TPSA↑	<i>Experimental and Computational Approaches to Estimate Solubility and Permeability in Drug Discovery and Development Settings</i> (Lipinski et al., 1997); <i>Fast Calculation of Molecular Polar Surface Area as a Sum of Fragment-Based Contributions and Its Application to the Prediction of Drug Transport Properties</i> (Ertl et al., 2000); <i>Deep Understanding of Structure-Solubility Relationship for a Diverse Set of Organic Compounds Using Matched Molecular Pairs</i> (Zhang et al., 2011).

Table 10: Representative evidence titles back-mapped from each final task skill. These titles record the papers retrieved to support the direction of the skill rules.

Table 11: Rule-random baseline on eight single-property tasks. Loose rows require movement in the requested direction; strict rows use the same thresholds as Table 2. AVG is the macro average over tasks.

Method	logP↓	logP↑	QED↑	QED↓	tPSA↓	tPSA↑	HBA↑	HBD↑	AVG
Rule-random (loose)	37.5	49.0	7.0	32.5	30.0	47.0	50.5	47.5	37.6
SkillMol (loose)	98.0	100.0	64.0	100.0	80.0	98.0	100.0	100.0	92.5
Rule-random (strict)	12.5	45.0	1.5	21.5	20.0	46.5	14.0	12.0	21.6
SkillMol (strict)	98.0	100.0	12.0	96.0	54.0	98.0	90.0	92.0	80.0

Table 12: Rule-random baseline on six multi-property tasks. A hit requires all requested property directions to be satisfied. AVG averages the six task columns within each row.

Method	logP↓ & HBA↑	logP↑ & HBA↑	logP↓ & HBD↑	logP↑ & HBD↑	logP↓ & tPSA↓	logP↓ & tPSA↑	AVG
Rule-random (loose)	30.0	21.5	49.5	47.0	15.0	45.0	34.7
SkillMol (loose)	96.0	58.0	96.0	50.0	46.0	100.0	74.3
Rule-random (strict)	2.5	0.5	9.0	7.0	3.0	18.5	6.8
SkillMol (strict)	90.0	34.0	74.0	16.0	24.0	100.0	56.3