REMEDI: INFERENCE-TIME REFINEMENT AND SELF-DISTILLATION FOR CONTROLLABLE MOLECULAR OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

In drug development, lead optimization must improve competing properties, such as increasing permeability and reducing toxicity, while limiting scaffold changes that could compromise the lead's potency. RL-based post-training, such as C-MORAL, addresses this problem by fine-tuning molecular language models to directly optimize molecular leads for objectives of interest, which relies on expensive task-specific datasets and faces challenges in achieving multi-objective optimization (MOO). We argue a pretrained model may already contain useful molecular edits that zero-shot generation cannot efficiently reach. Motivated by the limited search capabilities of zero-shot generation, we develop an efficient Metropolis–Hastings search strategy to better explore the proposal distribution. We propose ReMedI, which uses the molecular language model as a base proposer for inference-time alignment. Given a lead and a set of multi-objective tasks, the model proposes edits conditioned on the current molecule, while Metropolis–Hastings refinement guides sequential edits under an objective for property improvement, preservation, and lead similarity. On both C-MuMOInstruct and SMDD benchmarks, refinement improves frozen base models across three backbones, reaching significantly higher multi-objective optimization success rates compared to baselines. These search traces can then distilled into the base model which improves zero-shot generation and subsequent search using $\approx 7.8\times$ less training molecular leads than that of C-MORAL. ReMedI achieves $\approx 2.8\times$ the strict-success coverage of RL baseline C-MORAL, including on out-of-distribution tasks, with matched lead-relative similarity.

1 INTRODUCTION

Chemical language models (CLMs) have increasingly become popular for text-guided molecular generation (Edwards et al., 2022), multi-property optimization (Dey et al., 2025b;a), and the design of functional, makeable molecules (Edwards et al., 2026). A central application of CLMs is lead optimization, which starts with a promising molecule (lead) and seeks local edits that improve deficient properties, such as permeability or toxicity, while preserving satisfactory properties and maintaining lead-relative similarity. However, current CLMs do not reliably produce molecules that meet all property targets without drifting off-scaffold. RL post-training and reference-guided policy optimization methods try to align CLMs to property rewards to satisfy MOO (Gao et al., 2026; Li et al., 2026). For instance, C-MORAL combines group-based policy optimization with property-specific score transformations and nonlinear reward aggregation to improve selective molecular editing (Gao et al., 2026). This approach relies on task-specific RL adapters and editing data, which are costly to develop and can limit transfer to new combinations of objectives. Since chemists often optimize several properties together, the challenge therefore is to meet every property requirement and maintaining acceptable lead-relative similarity without adapting a separate model for each task.

Inference-time alignment offers an alternative by guiding generation while holding the model weights fixed. Existing approaches include similarity-guided decoding (Navratil et al., 2025) and iterative editing (Edwards et al., 2026). For instance, GP-MoLFormer-Sim guides decoding through reference similarity and uses an outer genetic algorithm for property optimization (Navratil et al., 2025), but its decoding guidance does not directly encode which properties should improve or re-

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.

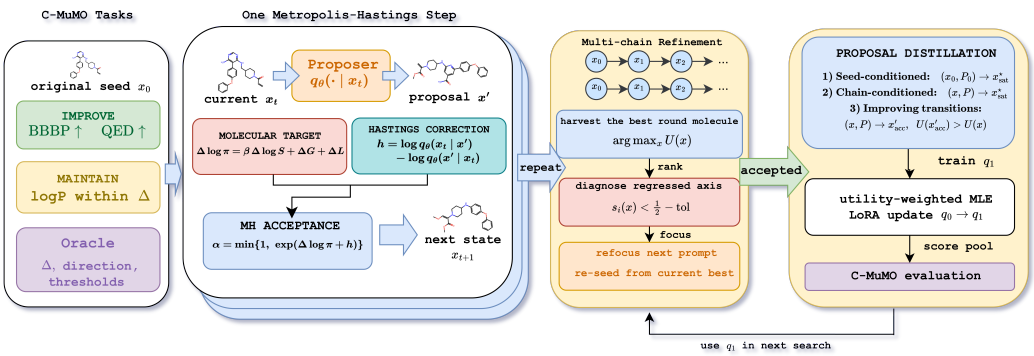


Figure 1: ReMedI separates the molecular edit proposal from the inference-time objective. Refinement discovers successful trajectories; lead- and state-conditioned examples from those trajectories train a shared proposal for later generation and search.

main unchanged. MARS demonstrates that multi-property molecular optimization can benefit from iterative search over intermediate molecular states (Xie et al., 2021). We build on this principle in ReMedI by using an instruction-conditioned language model to propose successive edits with a search algorithm which determines which states to retain.

We introduce ReMedI, which uses a pretrained molecular language model to propose local edits and Metropolis–Hastings (MH) refinement to guide the search. Its lead-relative objective balances property improvement, preservation, and similarity to the lead. Because the proposal remains frozen during refinement, the improvement–similarity trade-off can be adjusted at inference time without task-specific RL adapters. Thus our method allows chemists to explore pharmacological spaces that are hard to reach via zero-shot prompting. On C-MuMOInstruct and SMDD, refinement improves success across GeLLMO-Llama, GeLLMO-Mistral, and mCLM-4B, and ReMedI achieves higher strict-success coverage than C-MORAL on matched GeLLMO-Mistral leads, including on combinations of properties held out during training.

We show that in ReMedI a single shared proposal adapter trained from only 500 lead–task instances achieves performance comparable to C-MORAL, which relies on five task-specific policies trained from 3,883 lead–task prompts. Our traces contain successful endpoints and intermediate states, so training does not require a separate specialized editing dataset. Search improves candidates at inference time and supplies examples that strengthen later generation and refinement.

Contributions.

- **Inference-time molecular alignment.** We use a chemical language model (CLM) to propose edits. Metropolis–Hastings (MH) accepts or rejects each proposal using a lead-relative objective for property improvement, preservation, and similarity, together with the corresponding LM probabilities. These priorities can be changed at inference time without retraining the model.
- **Learning from refinement.** We convert search-discovered endpoints, intermediate states, and improving transitions into supervised training pairs and distill them back into the base CLM which improves zero-shot generation and subsequent MH refinements.
- **Success under similarity requirements.** ReMedI achieves $\approx 2.8\times$ the strict-success coverage of C-MORAL on matched GeLLMO-Mistral leads. On C-MuMOInstruct, MH increases frozen-model SOR by 49–157% relative to zero-shot generation across three backbones and IND/OOD splits; on SMDD, it improves SOR by 12–50%.

2 Related Works

Post-training for molecular optimization. Molecular LLMs support text-guided generation, editing, and multi-property optimization (Wang et al., 2025). Early molecule–text models such as

MolT5 and MoleculeSTM connect molecular strings to natural language (Edwards et al., 2022; Liu et al., 2023a), while instruction- and interaction-based systems such as LlaSMol, PEIT-LLM, ChatDrug, and DrugAssist improve chemistry task following, property-conditioned generation, or conversational molecular editing (Yu et al., 2024; Lin et al., 2026; Liu et al., 2023b; Ye et al., 2023). For controllable multi-property optimization, Prompt-MolOpt steers property-specific transformations with prompt-conditioned embeddings (Wu et al., 2024), while GeLLMO-C and C-MuMOInstruct train models to follow multi-property instructions and prevent property regressions (Dey et al., 2025b;a). Earlier molecular optimization methods also applied reward-guided reinforcement learning to sequence generation and direct molecular edits (Olivecrona et al., 2017; Popova et al., 2018; Zhou et al., 2019). More broadly, post-training methods modify the generative policy through supervised fine-tuning, preference optimization, or reinforcement learning (Ouyang et al., 2022; Rafailov et al., 2024; Schulman et al., 2017). C-MORAL uses GRPO/GDPO-style optimization to align molecular LLM outputs with multi-objective constraints (Gao et al., 2026; Shao et al., 2024), while RePO uses reference-guided policy optimization (Li et al., 2026). These approaches adapt the model through specialized training. ReMedI instead keeps the CLM fixed during refinement and specifies the improvement–similarity trade-off through an explicit objective at inference time.

Inference-time alignment. A smaller line of work shifts optimization to inference time. MARS uses Markov-chain search for multi-objective molecular optimization with graph neural networks (Xie et al., 2021), while GP-MoLFormer-Sim provides training-free similarity-guided decoding for chemical language models (Navratil et al., 2025). ReMedI uses the CLM as a state-dependent proposal within a Metropolis–Hastings refinement loop (Metropolis et al., 1953; Hastings, 1970), with an explicit target that favors requested property improvements, preservation of guarded properties, and similarity to the lead. Search can also improve the proposal itself. Expert Iteration formalizes the search-and-distill pattern (Anthony et al., 2017), and we train a shared proposal adapter on successful refinement endpoints and intermediate-state examples, reconditioning multi-step solutions on the original lead as one-step targets. This combines inference-time refinement with proposal improvement and tests whether search-derived supervision transfers to unseen objective combinations.

3 PRELIMINARY

Metropolis–Hastings. Metropolis–Hastings (MH) draws samples from a distribution known only up to a normalizing constant. Given an unnormalized target π and a proposal $q(\cdot \mid x)$, from the current state x it draws a candidate $y \sim q(\cdot \mid x)$ and accepts it with probability

$$A(x \to y) = \min\left(1,\ \frac{\pi(y)}{\pi(x)}\,\frac{q(x \mid y)}{q(y \mid x)}\right), \tag{1}$$

where the second factor, the Hastings correction, offsets the asymmetry of a state-dependent proposal (a forward edit likely from x need not be equally likely in reverse). Since π appears only through a ratio, its normalizing constant cancels. The proposal controls exploration, while π determines the target distribution. The proposal can be changed or learned without changing that target, provided the Hastings correction is retained and the resulting chain remains irreducible and aperiodic over the target's support.

Nash bargaining and Pareto optimality. Let $\mathcal{X}$ be the feasible outcome set. Consider m objectives with utilities $u_i(x)$ and a reference (disagreement) point $d \in \mathbb{R}^m$ that specifies the baseline value for each objective. An outcome x is Pareto-optimal if there is no feasible x' such that $u_i(x') \geq u_i(x)$ for every i, with strict inequality for at least one objective. In other words, no objective can be improved without sacrificing another. If the feasible utility set is compact and convex and contains a point that strictly improves every objective over d, the Nash bargaining solution Nash & Jr. (1950) has a unique utility vector, attained by any maximizer x^{NBS} of

$$x^{\mathrm{NBS}} \in \underset{\substack{x \in \mathcal{X} \\ u_i(x) \geq d_i\ \forall i}}{\arg\max} \prod_{i=1}^{m} \big(u_i(x) - d_i\big), \tag{2}$$

which maximizes the *product* of gains over the reference point rather than their weighted sum. Equivalently, it maximizes $\sum_i \log\big(u_i(x) - d_i\big)$. This makes the solution invariant to positive affine

Algorithm 1 Focused refinement and trace generation. The inner chain is given in Alg. 2.

Require: lead x_0, task requirements t, base prompt $P_0(t)$, valid proposal q_θ^{valid}, stages R, chains C, steps N, similarity target τ_s, weights $\beta_{S,0}, \beta_G, \beta_L$, annealing rate ρ
Ensure: incumbent $x^\star$; transition log $\mathcal{T}$

$x^\star \leftarrow x_0$; $\phi \leftarrow$ similarity; $P \leftarrow P_\phi(t)$; TRIED $\leftarrow \varnothing$; $\mathcal{T} \leftarrow \varnothing$
for $j = 1, \dots, R$ **do**
  $\mathcal{V} \leftarrow \{x^\star\}$
  for $c = 1, \dots, C$ **do**
    $\mathcal{V}_c, \mathcal{T}_c \leftarrow \text{MHCHAIN}(x^\star, x_0, P, N, q_\theta^{\text{valid}}, \beta_{S,0}, \rho, \beta_G, \beta_L, \tau_s)$
    $\mathcal{V} \leftarrow \mathcal{V} \cup \mathcal{V}_c$; $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{T}_c$
  end for
  $\hat{x} \leftarrow$ best-ranked candidate in $\mathcal{V}$ under the criteria above
  if $\hat{x}$ is preferred to $x^\star$ under the criteria above **then** $x^\star \leftarrow \hat{x}$; TRIED $\leftarrow \varnothing$
  else if ϕ is a property index **then** TRIED $\leftarrow$ TRIED $\cup \{\phi\}$
  end if
  if $\mathcal{R}_t(x^\star) = \varnothing$ **and** $\text{sim}(x^\star, x_0) \geq \tau_s$ **then break**
  end if
  $\mathcal{F} \leftarrow \mathcal{R}_t(x^\star) \setminus$ TRIED
  if $\mathcal{F} \neq \varnothing$ **then** $\phi \leftarrow \arg\min_{i \in \mathcal{F}} s_i(x^\star)$ ▷ worst $q_1$ied improve-property
  else $\phi \leftarrow$ similarity
  end if
  $P \leftarrow P_\phi(t)$
end for
return $x^\star, \mathcal{T}$

rescaling of an objective when its disagreement value is transformed in the same way, and favors balanced gains across objectives. Any outcome that leaves one objective at its reference value has a zero product, regardless of gains in the other objectives.

4 METHOD

Scoring and target. Let q_θ denote the LM's conditional proposal distribution, x_0 the lead, t the task requirements, and $P_0(t)$ the original prompt specifying those requirements. Let $p(x) \in \mathbb{R}^m$ be the oracle's property values and $d = p(x_0)$ the lead's values. The task specifies which properties to improve and which to guard. We denote these sets by $\mathcal{I}$ and $\mathcal{G}$, and use $\text{sgn}_i \in \{+1, -1\}$ for the requested direction of property i. We map each requested property to a lead-centred satisfaction

$$s_i(x) = \text{clip}_{[\epsilon, 1-\epsilon]}\big(\sigma\big(k_i \,\text{sgn}_i\,(p_i(x) - d_i)\big)\big), \qquad \sigma(z) = \tfrac{1}{1+e^{-z}}, \tag{3}$$

where $k_i > 0$ controls the response scale for property i, and $0 < \epsilon < 1/2$ keeps each satisfaction away from zero and one so its logarithm remains bounded. We aggregate the satisfactions of properties to improve as $S(x) = \prod_{i \in \mathcal{I}} s_i(x)$, so gains on one property cannot fully mask poor satisfaction on another. For guarded properties, the penalty $G(x)$ activates only when a property declines beyond its allowed margin. The similarity penalty $L(x)$ activates when similarity to the lead falls below τ_s.

$$G(x) = -\sum_{i \in \mathcal{G}} \max\big(0, -\tau_i^g - \text{sgn}_i(p_i(x) - d_i)\big), \qquad L(x) = -\max\big(0, \tau_s - \text{sim}(x, x_0)\big). \tag{4}$$

Here τ_i^g is the allowed decline for guarded property i, and τ_s is the minimum lead similarity. Combining the three terms gives the utility score and target distribution,

$$\begin{aligned} U(x) &= \beta_S \log S(x) + \beta_G\, G(x) + \beta_L\, L(x), \\ \pi(x) &\propto \exp\big(U(x)\big), \end{aligned} \tag{5}$$

Here $\log S(x) = \sum_{i \in \mathcal{I}} \log s_i(x)$ is the sum of per-property log-satisfactions and the log of the Nash product. The positive weights β_S, β_G, and β_L control improvement, guarding, and similarity, respectively.

Algorithm 2 MHCHAIN performs one refinement chain of length N.

function MHCHAIN($x_0, P, N, q_\theta^{\text{valid}}, \beta_{S,0}, \rho, \beta_G, \beta_L, \tau_s$)
  $x \leftarrow x_0$; $\mathcal{V} \leftarrow \{x\}$; $\mathcal{T} \leftarrow \varnothing$
  for $n = 1, \dots, N$ **do**
    $\beta_{S,n} \leftarrow \beta_{S,0}\rho^{n-1}$
    $x' \sim q_\theta^{\text{valid}}(\cdot \mid x, P)$
    $\Delta \log \pi_n \leftarrow \beta_{S,n}\,\Delta \log S + \beta_G\,\Delta G + \beta_L\,\Delta L$; $\log h \leftarrow \log q_\theta^{\text{valid}}(x \mid x', P) - \log q_\theta^{\text{valid}}(x' \mid x, P)$
    $\log r \leftarrow \Delta \log \pi_n + \log h$; $\alpha \leftarrow \min\{1, \exp(\log r)\}$
    $x_{\text{prev}} \leftarrow x$; $b \sim \text{Bernoulli}(\alpha)$
    if $b = 1$ **then** $x \leftarrow x'$; $\mathcal{V} \leftarrow \mathcal{V} \cup \{x\}$
    end if
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{(x_0, x_{\text{prev}}, P, x', U(x'), \log q_\theta^{\text{valid}}(x' \mid x_{\text{prev}}, P), \log h, \alpha, b)\}$
  end for
  return $\mathcal{V}, \mathcal{T}$
end function

Sampling and focused refinement. At each refinement stage, we set the focused prompt to $P = P_\phi(t) = P_0(t) \oplus \text{clause}(\phi)$, where ϕ identifies the property or similarity constraint to focus on. The model proposes edits from $q_\theta^{\text{valid}}(\cdot \mid x, P)$. For a proposed move from x to x', write $\Delta F = F(x') - F(x)$ for any score F. At MH iteration n, we anneal the satisfaction weight as $\beta_{S,n} = \beta_{S,0}\rho^{n-1}$, giving the step-specific target $\pi_n(x) \propto \exp\big(\beta_{S,n} \log S(x) + \beta_G G(x) + \beta_L L(x)\big)$. The Metropolis–Hastings acceptance ratio is

$$\log r = \underbrace{\beta_{S,n}\,\Delta \log S + \beta_G\,\Delta G + \beta_L\,\Delta L}_{\Delta \log \pi_n} + \underbrace{\log q_\theta^{\text{valid}}(x \mid x', P) - \log q_\theta^{\text{valid}}(x' \mid x, P)}_{\log h}, \tag{6}$$

We accept the move with probability $\alpha = \min\{1, e^{\log r}\}$. Each iteration applies the MH rule for its step-specific target π_n. We trim and parse proposals to valid molecules and retry invalid draws. The focus prompt guides which edits the model proposes, while the target score is determined by S, G, and L. The acceptance ratio accounts for the prompt by comparing the probabilities of proposing x' from x and x from x'. Computing the target ratio requires oracle scores for both states.

Proposal distillation. The search produces traces of proposed edits and their outcomes. We use these traces as supervision to train the proposal model for future searches, curating a corpus $\mathcal{D}$ of context–molecule pairs with three types of examples. *Seed-conditioned* examples pair the lead and its task prompt $(x_0, P_0(t))$ with a satisfying endpoint, teaching the model to generate directly from the lead. *Chain-conditioned* examples pair an intermediate state and its focused task prompt $(x, P_\phi(t))$ with a satisfying endpoint reached from that state. We also include some *accepted transitions* that do not meet every requirement, so the model learns incremental edits that keep refinement moving. We fit q_θ by weighted maximum likelihood,

$$\mathcal{L}(\theta) = -\sum_{(c,x') \in \mathcal{D}} w(x')\, \log q_\theta(x' \mid c), \qquad w(x') \propto \exp\big(U(x')/\beta_{\text{dist}}\big), \tag{7}$$

where the positive temperature β_{dist} controls how strongly the weights favor higher-scoring targets. Seed- and chain-conditioned examples match the two proposal contexts used at inference, so one distilled model can improve both direct generation and refinement. The process can be repeated by generating new traces with the current proposal and refitting Eq. 7.

Let $\mathcal{R}_t(x)$ be the unmet improvement requirements for task t. We prefer candidates with fewer unmet requirements, then those meeting τ_s, then higher utility $U(x)$. The focus ϕ is the least-satisfied unmet requirement, or similarity if none remain, and its clause tells the model what to focus on.

Table 1: C-MuMOInstruct results on 50 held-out leads per task. q_0 denotes the original proposal and q_1 the proposal after one distillation round; *base* denotes direct generation and *MH* refinement. Best-of-20 selection prioritizes SOR, then SSOR, then similarity. Success–similarity tradeoffs are shown in Fig. 2. Bold marks the best value in each row.

Backbone	Metric	Split	C-MORAL	q_0-base	q_0-MH	q_1-base	q_1-MH
GeLLMO-Llama	SSOR ↑	IND	0.030	0.092	0.200	0.140	**0.264**
		OOD	0.140	0.180	0.276	0.240	**0.332**
	SOR ↑	IND	0.320	0.340	0.632	0.532	**0.756**
		OOD	0.290	0.300	0.448	0.424	**0.524**
GeLLMO-Mistral	SSOR ↑	IND	0.056	0.068	0.124	0.120	**0.196**
		OOD	0.122	0.164	0.236	0.280	**0.356**
	SOR ↑	IND	0.400	0.324	0.528	0.524	**0.692**
		OOD	0.322	0.268	0.400	0.372	**0.532**
mCLM-4B	SSOR ↑	IND	0.044	0.012	0.088	0.068	**0.156**
		OOD	0.116	0.080	0.204	0.096	**0.216**
	SOR ↑	IND	0.420	0.168	0.432	0.220	**0.448**
		OOD	0.273	0.152	**0.356**	0.212	**0.356**

5 EXPERIMENTS

5.1 SETUP

Models and benchmarks. We evaluate ReMedI primarily on C-MuMOInstruct, which contains ten multi-property editing tasks: five training-seen property combinations (IND) and five held-out combinations (OOD) (Dey et al., 2025a). We use GeLLMO-C Llama-3.1-8B, GeLLMO-C Mistral-7B, and mCLM-4B as proposal models (Dey et al., 2025b; Edwards et al., 2026), with property evaluation provided by ADMET-AI and PyTDC (Swanson et al., 2024; Huang et al., 2022). IND and OOD follow the benchmark's original task split; our shared proposal adapter is distilled only from trajectories spanning the five IND tasks. We additionally evaluate ReMedI on 314 instances from SMDD-Bench Task 4 (Lead Optimization) Han et al. (2026). We use a non-agentic evaluation in which both the lead and generated candidate are scored with ADMET-AI and RDKit. We compare direct generation (*base*) and MH refinement (*MH*) using the original proposal (q_0) and the proposal after one distillation round (q_1), against the RL-trained baseline C-MORAL. Since C-MORAL's task objectives do not align with SMDD Task 4, we pool the five IND seed files and train one shared adapter per backbone. Each seed retains its own improvement and hold-constant keys, while the oracle scores the union of IND properties.

Metrics. Following C-MORAL (Gao et al., 2026), we report Success Optimized Rate (SOR), the proportion of molecules that meet the requested property-improvement criteria, and Strict Success Optimized Rate (SSOR), the proportion whose relevant properties all satisfy prescribed near-optimal thresholds. For SMDD-Bench (Han et al., 2026), an SOR success requires every requested objective to improve by its instance-specific threshold. Similarity (Sim) is the Morgan fingerprint (Morgan, 1965; Rogers & Hahn, 2010; Landrum, 2026) Tanimoto similarity (Bajusz et al., 2015; Maggiora et al., 2014) between each generated molecule and its lead. Clearance is the rate at which the body, mainly the liver, removes a drug from plasma (Yim et al., 2021). For the 144 SMDD-Bench instances requesting reduced clearance, CLR is the fraction whose selected molecule's predicted clearance decreases by at least the instance-specific threshold (Swanson et al., 2024). We select the best of 20 candidates by SOR and emphasize SSOR because it captures simultaneous satisfaction of all property targets at near-optimality.

5.2 MAIN RESULTS

Refinement across backbones. Table 1 shows effect of inference-time refinement across three model families on both IND/OOD splits. MH improves q_0 SOR by $\approx 80\%$ and SSOR by $\approx 90\%$

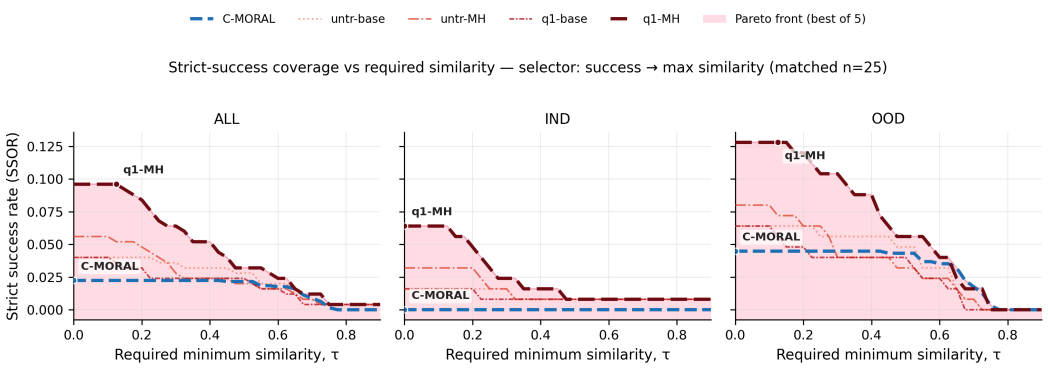


Figure 2: Strict-success coverage versus minimum lead similarity on 25 matched leads per task. Each method selects candidates by SOR success, then maximum similarity; coverage counts selected outputs that also satisfy SSOR and the indicated similarity threshold. C-MORAL OOD results are averaged over five task-specific GRPO policies.

Table 2: SMDD-Bench Task 4 results on 314 instances. ReMedI-MH uses the frozen proposal, and pooled C-MORAL uses one shared adapter per backbone at the same 20-candidate inference budget.

Backbone	SOR ↑		Sim ↑		CLR ↑	
	Ours	C-MORAL	Ours	C-MORAL	Ours	C-MORAL
GeLLMO-Llama	**0.764**	0.580	0.633	**0.707**	**0.764**	0.583
GeLLMO-Mistral	**0.771**	0.640	0.663	**0.737**	**0.792**	0.681
mCLM-4B	**0.796**	0.506	0.316	**0.361**	**0.806**	0.576

on average. The same pattern persists after distillation, with q_1-MH outperforming q_1-base on every backbone and split. These gains show that substantial optimization capability is already present in the proposal models but is not recovered by one-shot decoding; sequential MH refinement exposes this capability by composing and selecting edits under the target objective.

Success under similarity requirements. Figure 2 evaluates the tradeoff between strict multi-objective success and similarity to the starting lead. Across increasingly stringent similarity requirements, ReMedI forms empirical Pareto frontier in SSOR–similarity space, maintaining higher strict-success coverage than C-MORAL over the evaluated range. This advantage holds on the pooled evaluation and extends to the held-out OOD property combinations. Because SSOR requires all relevant properties to reach their prescribed near-optimal thresholds (Gao et al., 2026), the frontier measures how often a method can satisfy all molecular objectives while remaining close to the original lead. These results show that ReMedI improves the strict-success–locality tradeoff rather than obtaining higher success simply through larger structural changes.

Search and proposal distillation. Table 1 shows that proposal distillation substantially strengthens the base model. On GeLLMO-Mistral, one distillation round improves zero-shot base SOR by 62% on IND (0.324 → 0.524) and 39% on OOD (0.268 → 0.372). The trained base model also improves subsequent MH refinement by 31% on IND (0.528 → 0.692) and 33% on OOD (0.400 → 0.532). The same pattern holds for GeLLMO-Llama, where distillation improves zero-shot base by 57%/41% and MH refinement by 20%/17% on IND/OOD. These results suggest not only could inference-time MH improve multi-objective optimization: distilling search trajectories back into the base CLM produces an improved base model for both zero-shot generation and subsequent search.

Generalization to SMDD-Bench. On 314 SMDD-Bench instances, ReMedI-MH improves SOR by 0.13–0.29 and clearance-target success on every backbone compared with pooled C-MORAL,

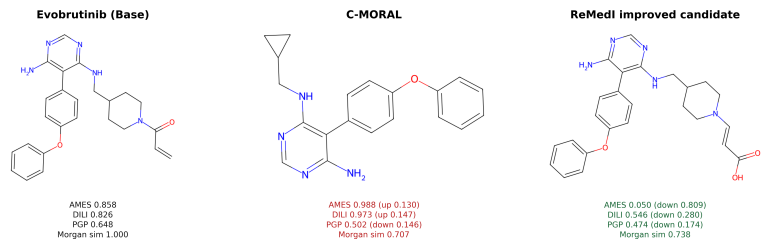


Figure 3: **Evobrutinib case study.** Starting from the same "fallen angel" drug lead Edwards et al. (2026), ReMedI identifies a candidate that simultaneously reduces all three predicted liabilities while preserving substantial structural similarity. This example illustrates how inference-time refinement can incorporate multiple beneficial edits into a single molecule while retaining the lead scaffold.

while C-MORAL retains higher similarity (Table 2). ReMedI's shared adapter uses trajectories from 500 lead–task instances, about $7.8\times$ fewer than C-MORAL's 3,883 RL training prompts (Gao et al., 2026).

Evobrutinib case study. Figure 3 compares optimization of the failed drug Evobrutinib. ReMedI simultaneously reduces AMES ($0.858 \rightarrow 0.050$), DILI ($0.826 \rightarrow 0.546$), and PGP ($0.648 \rightarrow 0.474$) while retaining similarity 0.738. By contrast, C-MORAL improves PGP but worsens DILI and AMES, with similarity 0.707. Whereas the mCLM example applies sequential property-specific edits, ReMedI finds a single search-derived candidate that improves all three liabilities simultaneously.

6 Ablations

Effect of search strategy. Figure 4 compares four search strategies under the same evaluated candidate budget: (i) *Independent best-of-K*, which samples every candidate directly from zero-shot generations; (ii) *unguided sequential editing*, which repeatedly conditions on the most recent molecule and always advances to the next proposal; (iii) *greedy feedback*, which edits sequentially but always accepts proposals that improve the objective; and (iv) *MH*, which is our method. Sequential methods all outperform independent sampling, among those, MH achieves the highest SOR at every tested budget, reaching 0.41 at $K = 20$ compared with 0.39 for unguided editing and 0.38 for greedy feedback. MH also preserves greater lead similarity than unguided editing (0.44 vs. 0.39), indicating that its gain does not come simply from allowing larger structural changes. Overall, the figure demonstrates the advantage of sequential edits over one-shot sampling, and that MH gives the best success–similarity tradeoff among the sequential strategies.

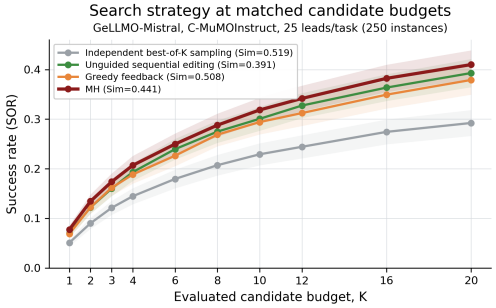
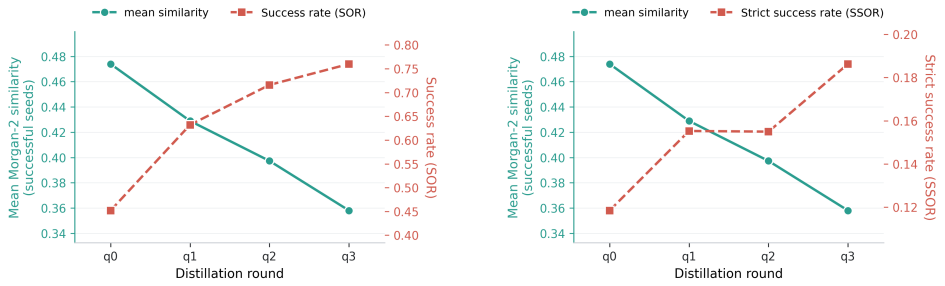


Figure 4: SOR versus candidate budget on 100 GeLLMO-Mistral instances spanning ten C-MuMOInstruct tasks. At matched candidate budgets, MH achieves the highest observed SOR. Legend values report mean lead similarity.

Choosing the distillation round. Figure 5 shows that the first distillation round captures the largest useful gain from search-derived supervision. From q_0 to q_1, SOR increases from approximately 0.45 to 0.63 and SSOR from 0.12 to 0.16, with only a modest decrease in similarity from 0.47 to 0.43. Additional rounds provide diminishing returns: q_2 improves SOR but not SSOR, while q_3 raises SSOR further only as similarity falls to approximately 0.36. We therefore use q_1 as our primary proposal, as it captures the largest simultaneous improvement in SOR and SSOR before later distillation rounds increasingly trade locality for additional optimization.



(a) SOR and similarity.

(b) SSOR and similarity.

Figure 5: Effect of repeated proposal distillation. Red dashed lines show SOR/SSOR; blue lines show mean lead similarity. q_0 is the original proposal and q_1–q_3 are successive distillation rounds.

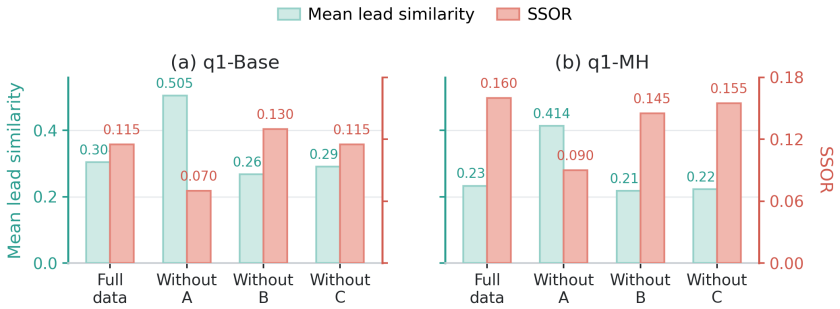


Figure 6: Effect of distillation sources on q_1-Base and q_1-MH. −A, −B, and −C remove successful endpoints, intermediate states, and improving transitions, respectively. Blue shows mean similarity; red dashed shows SSOR.

Effect of distillation sources. Figure 6 ablates the three supervision sources used for proposal distillation: (A) *lead-conditioned successful endpoints*, which teach complete solutions from the original lead; (B) *intermediate-state examples*, which train the proposal on states encountered during refinement; and (C) *accepted transitions*, which teach useful local edits that do not yet satisfy SOR. Removing (A) causes the largest drop in SSOR, showing that endpoint supervision is essential for learning successful optimization. Ablation (A) makes the proposal much more conservative: for q1-Base, similarity rises from 0.30 to 0.505 while SSOR drops from 0.115 to 0.070; after MH refinement, similarity rises from 0.23 to 0.414 while SSOR falls from 0.160 to 0.090. Thus, lead-conditioned successful endpoints appear to provide the main supervision for learning edits that are large enough to satisfy the optimization targets. Removing (B) or (C) lowers refined SSOR while also reducing similarity. The full mixture gives the highest refined SSOR (0.160), supporting trajectory-level distillation and providing the strongest operating point for subsequent MH refinement.

7 Limitations & Conclusion

ReMedI reframes molecular optimization as inference-time refinement: a CLM proposes sequential edits, MH selects them under an explicit lead-relative objective, and successful search trajectories are distilled back into the proposal. Across C-MuMOInstruct and SMDD-Bench, this improves frozen proposals and yields a stronger strict-success–similarity tradeoff, while one round of distillation further improves both direct generation and subsequent search. The ablations further support the role of each component: sequential editing provides the main search gain, MH improves the success–similarity tradeoff, and the three distillation sources contribute complementary supervision for direct generation and subsequent refinement. A limitation of our method is all discoveries remains limited *in silico* predictions, which do not yet establish *in vitro* or **in vivo** efficacy.

AI USE STATEMENT

Generative AI tools assisted with prose editing and routine code and analysis. The authors are responsible for verifying claims, figures, and references before submission.

REFERENCES

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search, 2017. URL https://arxiv.org/abs/1705.08439.

D Bajusz, A Rácz, and K Héberger. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of cheminformatics*, 7:20, 2015. doi: 10.1186/s13321-015-0069-3. URL https://pubmed.ncbi.nlm.nih.gov/26052348/.

Vishal Dey, Xiao Hu, and Xia Ning. Large language models for controllable multi-property multi-objective molecule optimization, 2025a. URL https://arxiv.org/abs/2505.23987.

Vishal Dey, Xiao Hu, and Xia Ning. Gellmo: Generalizing large language models for multi-property molecule optimization, 2025b. URL https://arxiv.org/abs/2502.13398.

Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. Translation between molecules and natural language, 2022. URL https://arxiv.org/abs/2204.11817.

Carl Edwards, Chi Han, Gawon Lee, Thao Nguyen, Sara Szymkuć, Chetan Kumar Prasad, Bowen Jin, Jiawei Han, Ying Diao, Ge Liu, Hao Peng, Bartosz A. Grzybowski, Martin D. Burke, and Heng Ji. mclm: A modular chemical language model that generates functional and makeable molecules, 2026. URL https://arxiv.org/abs/2505.12565.

Rui Gao, Youngseung Jeon, Swastik Roy, Morteza Ziyadi, and Xiang 'Anthony' Chen. C-moral: Controllable multi-objective molecular optimization with reinforcement alignment for llms, 2026. URL https://arxiv.org/abs/2604.23061.

Kevin Han, Renfei Zhang, Kathy Wei, Hamed Mahdavi, Niloofar Mireshghallah, and Amir Barati Farimani. Smdd-bench: Can llms solve real-world small molecule drug design tasks?, 2026. URL https://arxiv.org/abs/2605.21740.

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. doi: 10.1093/biomet/57.1.97.

Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Artificial intelligence foundation for therapeutic science. *Nature Chemical Biology*, 2022.

Greg Landrum. RDKit: Open-source cheminformatics. https://www.rdkit.org, 2026. Accessed: [Insert Date Here].

Xuan Li, Zhanke Zhou, Zongze Li, Jiangchao Yao, Yu Rong, Lu Zhang, and Bo Han. Reference-guided policy optimization for molecular optimization via llm reasoning, 2026. URL https://arxiv.org/abs/2603.05900.

Xuan Lin, Long Chen, Yile Wang, Yangyang Chen, and Xiangxiang Zeng. Property enhanced instruction tuning for multi-task molecule generation with large language models, 2026. URL https://arxiv.org/abs/2412.18084.

Shengchao Liu, Weili Nie, Chengpeng Wang, Jiarui Lu, Zhuoran Qiao, Ling Liu, Jian Tang, Chaowei Xiao, and Anima Anandkumar. Multi-modal molecule structure–text model for text-based retrieval and editing. *Nature Machine Intelligence*, 5:1447–1457, 2023a. doi: 10.1038/s42256-023-00759-6.

Shengchao Liu, Jiongxiao Wang, Yijin Yang, Chengpeng Wang, Ling Liu, Hongyu Guo, and Chaowei Xiao. Chatgpt-powered conversational drug editing using retrieval and domain feedback, 2023b. URL https://arxiv.org/abs/2305.18090.

G Maggiora, M Vogt, D Stumpfe, and J Bajorath. Molecular similarity in medicinal chemistry. *Journal of medicinal chemistry*, 57(8):3186–204, 2014. doi: 10.1021/jm401411z. URL https://pubmed.ncbi.nlm.nih.gov/24151987/.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953. doi: 10.1063/1.1699114.

Harry L. Morgan. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, 5:107–113, 1965. URL https://api.semanticscholar.org/CorpusID:62164893.

John F. Nash and Jr. The bargaining problem. *Econometrica*, 18(2):155–162, 1950. URL https://onlinelibrary.wiley.com/doi/abs/0012-9682(195004)18:2<155:TBP>2.0.CO;2-H.

Jiri Navratil, Jarret Ross, Payel Das, Youssef Mroueh, Samuel C Hoffman, Vijil Chenthamarakshan, and Brian Belgodere. Gp-molformer-sim: Test time molecular optimization through contextual similarity guidance, 2025. URL https://arxiv.org/abs/2506.05628.

Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de novo design through deep reinforcement learning, 2017. URL https://arxiv.org/abs/1704.07555.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science Advances*, 4(7), 2018. ISSN 2375-2548. doi: 10.1126/sciadv.aap7885. URL http://dx.doi.org/10.1126/sciadv.aap7885.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. doi: 10.1021/ci100050t.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

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

Kyle Swanson, Patrick Walther, Jeremy Leitz, Saurabh Mukherjee, Joseph C. Wu, Rohan V. Shivnaraine, and James Zou. ADMET-AI: A machine learning ADMET platform for evaluation of large-scale chemical libraries. *Bioinformatics*, 40(7):btae416, 2024. doi: 10.1093/bioinformatics/btae416.

Ziqing Wang, Kexin Zhang, Zihan Zhao, Yibo Wen, Abhishek Pandey, Han Liu, and Kaize Ding. A survey of large language models for text-guided molecular discovery: from molecule generation to optimization, 2025. URL https://arxiv.org/abs/2505.16094.

Zhenxing Wu, Odin Zhang, Xiaorui Wang, Li Fu, Huifeng Zhao, Jike Wang, Hongyan Du, Dejun Jiang, Yafeng Deng, Dongsheng Cao, Chang-Yu Hsieh, and Tingjun Hou. Leveraging language model for advanced multiproperty molecular optimization via prompt engineering. *Nature Machine Intelligence*, 6:1359–1369, 2024. doi: 10.1038/s42256-024-00916-5.

Yutong Xie, Chence Shi, Hao Zhou, Yuwei Yang, Weinan Zhang, Yong Yu, and Lei Li. MARS: Markov molecular sampling for multi-objective drug discovery. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=kHSu4ebxFXY.

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, 2023. URL https://arxiv.org/abs/2401.10334.

DS Yim, SH Bae, and S Choi. Predicting human pharmacokinetics from preclinical data: clearance. *Translational and clinical pharmacology*, 29(2):78–87, 2021. doi: 10.12793/tcp.2021.29.e12. URL https://pubmed.ncbi.nlm.nih.gov/34235120/.

Botao Yu, Frazier N. Baker, Ziqi Chen, Xia Ning, and Huan Sun. Llasmol: Advancing large language models for chemistry with a large-scale, comprehensive, high-quality instruction tuning dataset, 2024. URL https://arxiv.org/abs/2402.09391.

Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N. Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific Reports*, 9(1), 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-47148-x. URL http://dx.doi.org/10.1038/s41598-019-47148-x.

A Appendix

A.1 Benchmark Metrics

Let $f_r(x)$ denote property r of molecule x, and let $s_r \in \{+1, -1\}$ indicate whether higher or lower values are preferred. For lead $x_{0,i}$ and output x_i, SOR requires each targeted property to improve by its prescribed margin Δ_r and each protected property to remain within tolerance ϵ_r:

$$\mathrm{SOR} = \frac{1}{n}\sum_{i=1}^{n} \prod_{r\in\mathcal{P}_{\mathrm{improve}}} \mathbf{1}\{s_r(f_r(x_i) - f_r(x_{0,i})) \geq \Delta_r\} \prod_{r\in\mathcal{P}_{\mathrm{preserve}}} \mathbf{1}\{|f_r(x_i) - f_r(x_{0,i})| \leq \epsilon_r\}. \tag{8}$$

SSOR instead checks whether all relevant properties meet their prescribed near-optimal thresholds Θ_r (Gao et al., 2026):

$$\mathrm{SSOR} = \frac{1}{n}\sum_{i=1}^{n} \prod_{r\in\mathcal{P}} \mathbf{1}\left[s_r\left(f_r(x_i) - \Theta_r\right) \geq 0\right]. \tag{9}$$

For example, consider hypothetical normalized permeability and toxicity scores, with targets of permeability ≥ 0.8 and toxicity ≤ 0.2. Suppose the lead has permeability 0.4 and toxicity 0.1, and the request requires a permeability increase of at least 0.1 while preserving toxicity. An output with permeability 0.6 and unchanged toxicity passes SOR but fails SSOR: permeability improves sufficiently but remains below its absolute target. An output with permeability 0.85 and unchanged toxicity passes both.

A.2 Performance across backbones.

Table 3 reports the full C-MuMOInstruct results across GeLLMO-Llama, GeLLMO-Mistral, and mCLM-4B. Inference-time refinement consistently improves the frozen proposal: q_1-MH increases both SOR and SSOR over q_1-base for every backbone and split. Distillation further strengthens the proposal on the GeLLMO backbones, and combining one round of distillation with refinement (q_1-MH) gives the highest SOR and SSOR in nearly all settings. The gains also extend to OOD property combinations, indicating that trajectories distilled from IND tasks improve optimization beyond the combinations used for training. Similarity generally decreases as optimization becomes more aggressive, motivating the similarity-constrained coverage analysis in the main text rather than interpreting success and similarity independently. RI is included only as a within-method diagnostic because its scale is not comparable to the C-MORAL implementation.

A.3 Trajectory Depth Ablation

Table 4 compares trajectory depths at a fixed budget of 80 proposals per lead. Independent proposals preserve locality but rarely reach the stringent target. Replacing 80 independent proposals with 10 trajectories of eight steps doubles overall SSOR from 0.070 to 0.140. Because every configuration evaluates the same number of proposals, this gain cannot be attributed to additional sampling. Instead, sequential conditioning allows multiple edits to compose into solutions satisfying the stringent multi-property constraints.

Table 3: C-MuMOInstruct performance of C-MORAL and ReMedI across GeLLMO-C (Llama, Mistral) and mCLM-4B backbones. $n = 50$ held-out seeds/task, best-of-20. q_0 is the frozen proposal, q_1 one round of search distillation, *base* direct generation, and *MH* inference-time refinement. This 50-lead-per-task summary uses the candidate selection in the source draft (success, then strict success, then similarity), distinct from the success-then-maximum-similarity selector of the 25-lead coverage figures. SOR, SSOR, and Sim are reported across methods; RI is a within-method diagnostic and is not comparable across methods (C-MORAL's RI is computed on a different scale), so RI rows are not bolded. Bold marks the best comparable value in each row, excluding q_0-*base*.

Backbone	Metric	Split	C-MORAL	q_0-base	q_0-MH	q_1-base	q_1-MH
GeLLMO-Llama	SOR ↑	IND	0.320	0.340	0.632	0.532	**0.756**
		OOD	0.290	0.300	0.448	0.424	**0.524**
	SSOR ↑	IND	0.030	0.092	0.200	0.140	**0.264**
		OOD	0.140	0.180	0.276	0.240	**0.332**
	RI (diag.)	IND	0.61	0.23	0.64	0.49	1.14
		OOD	0.52	0.31	0.51	0.49	0.67
	Sim ↑	IND	0.460	0.686	0.491	**0.523**	0.369
		OOD	0.450	0.686	**0.596**	0.569	0.487
GeLLMO-Mistral	SOR ↑	IND	0.400	0.324	0.528	0.524	**0.692**
		OOD	0.322	0.268	0.400	0.372	**0.532**
	SSOR ↑	IND	0.056	0.068	0.124	0.120	**0.196**
		OOD	0.122	0.164	0.236	0.280	**0.356**
	RI (diag.)	IND	4.85	0.28	0.43	0.41	0.72
		OOD	4.61	0.19	0.38	0.33	0.64
	Sim ↑	IND	0.565	0.719	**0.595**	0.586	0.467
		OOD	0.582	0.722	**0.644**	**0.644**	0.533
mCLM-4B	SOR ↑	IND	0.420	0.168	0.432	0.220	**0.448**
		OOD	0.273	0.152	**0.356**	0.212	**0.356**
	SSOR ↑	IND	0.044	0.012	0.088	0.068	**0.156**
		OOD	0.116	0.080	0.204	0.096	**0.216**
	RI (diag.)	IND	5.78	0.06	0.39	0.13	0.47
		OOD	6.85	0.13	0.43	0.19	0.44
	Sim ↑	IND	0.235	0.421	0.363	**0.406**	0.363
		OOD	0.231	0.388	0.360	**0.409**	0.384

Table 4: Effect of trajectory depth on frozen GeLLMO-Mistral. Every configuration uses 80 proposals per lead. Results aggregate IND and OOD; bold marks the best value in each column.

Chains×steps	SOR ↑	SSOR ↑	Sim ↑
80×1	0.440	0.070	**0.463**
40×2	0.500	0.100	0.422
20×4	**0.560**	0.125	0.372
10×8	0.550	**0.140**	0.343