# PMO-DOCK: BENCHMARKING DOCKING, SPECIFICITY, AND GENERALIZATION IN MOLECULAR OPTIMIZATION

**Gor Simonyan***
YerevaNN
Yerevan State University

**Tatevik Abrahamyan***
YerevaNN
Yerevan State University

**Narek Abelyan**
YerevaNN
Russian-Armenian University

**Tigran Fahradyan**
University of Cambridge

**Hrant Khachatrian**
YerevaNN
Yerevan State University

## ABSTRACT

The Practical Molecular Optimization (PMO) benchmark standardized evaluation in the field, but its objectives are largely limited to simple property-based oracles. Recent methods evaluating molecules with docking-based objectives move closer to real drug-discovery settings, but unlike PMO, they are evaluated under fragmented protocols that make comparisons inconsistent. We introduce PMO-Dock, a new benchmark and evaluation protocol for docking-based molecular optimization that enforces strict generalization: algorithms must tune hyperparameters on a validation set of diverse protein targets and freeze them for a distinct test set. PMO-Dock contains 23 tasks covering hit generation, lead optimization, and novel "docking with specificity" tasks that require strong on-target binding while penalizing off-target interactions. We benchmark four diverse high-performing methods spanning different optimization paradigms: Saturn (reinforcement learning), GenMol (discrete diffusion), Genetic-guided GFlowNet, and Chemlactica (LLM- based). Our extensive analysis reveals that hyperparameters maximizing performance on validation targets rarely transfer to test tasks, highlighting a critical fragility in current state-of-the-art methods. Our PMO-Dock supports task-aware hyperparameter selection without test-set overfitting, providing a robust foundation for the next generation of generalizable molecular optimizers.

## 1 INTRODUCTION AND RELATED WORK

Small molecule optimization is a central challenge in modern drug discovery, requiring algorithms to efficiently search vast chemical spaces under complex, often computationally expensive objectives. Over the past decade, numerous optimization methods have been proposed, but their evaluation has been fragmented across different objectives and protocols, making direct comparison difficult.

The Practical Molecular Optimization (PMO) benchmark Gao et al. (2022) addressed this by establishing rigorous evaluation standards. PMO compiled 23 optimization tasks with well-specified oracles and evaluated 25 methods using unified protocols: strict oracle budgets, extensive hyperparameter sweeps, and fixed metrics. This standardization enabled meaningful progress, with methods such as Augmented Memory Guo & Schwaller (2024a), Genetic-guided GFlowNets Kim et al. (2024), Chemlactica Guevorguian et al. (2024), and MOLLEO Wang et al. (2025) advancing state-of-the-art on the benchmark. However, PMO's objectives are limited to simple combinations of RDKit-based properties, such as molecular similarity, isomer counts, and drug-likeness metrics. While computationally efficient, these objectives lack the practical relevance of real-world drug discovery, such as docking-based evaluations against protein targets that PMO does not cover.

Recent work has attempted to bridge this gap by incorporating docking-based objectives. Methods including Augmented Memory, Beam Enumeration Guo & Schwaller (2024b), Saturn Guo & Schwaller

*Denotes equal first authorship. Correspondence to Gor Simonyan, Tatevik Abrahamyan (gor@yerevann.com, tatevikabrahamyan@yerevann.com).

(2024c), GenMol Lee et al. (2025), and Chemlactica Guevorguian et al. (2024) have explored docking score optimization. While this line of work broadens applicability to real-world settings, it departs from PMO's rigorous evaluation standards. In this work we bridge this gap.

Our contributions are as follows:

1. We introduce PMO-Dock, a new benchmark that brings PMO-style clarity to docking-based objectives. It comprises 23 tasks split into 11 validation and 12 test tasks with a strict separation between model development and evaluation, and covers hit generation, lead optimization, and new binding specificity tasks.
2. We introduce a new task category for binding specificity, where molecules must bind strongly to a target protein or pocket while avoiding off-target interactions, a critical but underexplored aspect of drug discovery. We define three specificity tasks in this category, addressing both intra-protein and inter-protein selectivity challenges.
3. We systematically analyze how hyperparameters transfer across tasks, protein targets, and objective types by evaluating four diverse methods spanning different optimization paradigms: Saturn (reinforcement learning), GenMol (discrete diffusion), Genetic-guided GFlowNet, and Chemlactica (LLM-based). Our analysis reveals fundamental differences in generalization behavior across optimization methods and identifies optimal model selection strategies for each approach.
4. We establish a leaderboard evaluating the four methods using robust model selection protocols, providing a standardized assessment of current state-of-the-art performance on 12 held-out test tasks.

## 2 THE PMO-DOCK BENCHMARK

Building upon prior work, we define three categories of molecular optimization tasks: **hit generation**, where the goal is to find as many molecules as possible that pass the given thresholds of docking scores; **lead optimization**, where the goal is to find molecules with the highest docking scores with an additional condition to be close to the given seed molecule; and a novel category of **docking with specificity** tasks, where the goal is to find molecules that have high docking score with one given target and low docking score with another given target (or another pocket of the same target protein).

### 2.1 HIT GENERATION

We adopt the *hit generation* task from Guo & Schwaller (2024c): algorithms propose candidate molecules for protein targets under a fixed oracle evaluation budget.

**Targets.** PARP1, Factor VII (FA7), 5-hydroxytryptamine receptor 1B (5HT1B), BRAF, JAK2.

**Objective.** Given an oracle budget of $B = 3000$ molecule evaluations, an algorithm $\pi$ produces a sequence of evaluated molecules $\{x_i\}_{i=1}^{B}$. The objective is to maximize the expected **hit rate**,

$$\max_{\pi} \ \mathbb{E}_{\text{seeds}} \left[ \frac{1}{B} \sum_{i=1}^{B} h(x_i) \right],$$

where the expectation is over random seeds, and $h(x)$ is the validity indicator function.

$$h(x) = \begin{cases} 1 & \text{if QED}(x) \in [0.5, 1.0] \wedge \text{SA}(x) \in [1.0, 5.0] \wedge s^{(t)}(x) \geq s^{(t)}_{\min} \\ 0 & \text{otherwise} \end{cases}$$

$s^{(t)}(x)$ denotes the docking score for target $t$ computed using QuickVina Alhossary et al. (2015) with exhaustiveness set to 1, and $s^{(t)}_{\min}$ are target-specific thresholds provided in Appendix D.2. Each algorithm is evaluated using ten random seeds, and performance is reported as the average hit rate across all seeds.

### 2.2 LEAD OPTIMIZATION

Following Lee et al. (2025), we consider *lead optimization*: algorithms start from a seed molecule and optimize docking scores while maintaining local chemical similarity under a fixed oracle budget.

**Targets.** PARP1, Factor VII (FA7), 5-hydroxytryptamine receptor 1B (5HT1B), BRAF, JAK2.

**Objective.** Given a seed molecule $x_0$, target $t$, and oracle budget $B = 1000$, an algorithm $\pi$ produces a sequence of evaluated molecules $\{x_i\}_{i=1}^{B}$. The objective is to maximize the expected **Top-1 docking score**,

$$\max_{\pi} \mathbb{E}_{\text{seeds}}\left[\max_{1\le i\le B} s^{(t)}(x_i)h(x_i)\right],$$

where the expectation is over random seeds, and $h(x)$ is the validity indicator function.

$$h(x) = \begin{cases} 1 & \text{if } \mathrm{QED}(x) \in [0.5, 1.0] \wedge \mathrm{SA}(x) \in [1.0, 5.0] \wedge s^{(t)}(x) \ge s^{(t)}_{\min} \wedge \mathrm{Sim}(x, x_0) \ge 0.6 \\ 0 & \text{otherwise} \end{cases}$$

$\mathrm{Sim}(x, x_0)$ denotes Tanimoto similarity to the seed molecule $x_0$ and $s^{(t)}(x)$ denotes the docking score for target $t$ computed using QuickVina Alhossary et al. (2015) with exhaustiveness set to 1, and $s^{(t)}_{\min}$ are target-specific thresholds provided in Appendix D.2. Threshold sensitivity is analyzed in Section 5.1.

### 2.3 SPECIFICITY

We introduce **docking specificity** tasks to evaluate selective binding: molecules must bind strongly to a target pocket while avoiding off-target interactions, a critical aspect of drug discovery.

**Targets.** We selected two closely related kinases from the JAK family, TYK2 and JAK2, which share high sequence and structural similarity. Each protein contains two domains: the active kinase domain (JH1, ATP-binding pocket) and the pseudokinase domain (JH2, allosteric pocket). The following experimentally resolved structures were used: TYK2 JH1 (PDB ID: 7UYT), TYK2 JH2 (PDB ID: 6NZP), JAK2 JH1 (PDB ID: 7UYW), and JAK2 JH2 (PDB ID: 5UT5).

Three tasks were defined, each aiming to maximize binding to 6NZP while minimizing off-target interactions:

1. **Intra-protein domain specificity:** molecules are optimized for strong binding to 6NZP while reducing binding to 7UYT.
2. **Inter-protein pseudokinase specificity:** molecules are optimized for strong binding to 6NZP while reducing binding to 5UT5.
3. **Cross-domain inter-protein specificity:** molecules are optimized for strong binding to 6NZP while minimizing binding to 7UYW.

**Objective.** Given an oracle budget of $B = 1000$ molecule evaluations, an algorithm $\pi$ produces a sequence of evaluated molecules $\{x_i\}_{i=1}^{B}$. The objective is to maximize the expected **specificity score**,

$$\max_{\pi} \mathbb{E}_{\text{seeds}}\left[\max_{1\le i\le B} \mathcal{L}(x_i)h(x_i)\right],$$

where the expectation is over random seeds, $\mathcal{L}(x)$ and $h(x)$ are defined below.

$$h(x) = \begin{cases} 1 & \text{if } \mathrm{QED}(x) \in [0.4, 1.0] \wedge \mathrm{SA}(x) \in [1.0, 4.0] \wedge s^{(\text{6NZP})}(x) \ge s^{(\text{spec})}_{\text{ref}} \\ 0 & \text{otherwise} \end{cases}$$

Here $s^{(\text{6NZP})}(x)$ is the QuickVina docking score of molecule $x$ against the 6NZP pocket (same software as hit/lead; specificity docking uses a higher exhaustiveness setting than the hit/lead oracle;

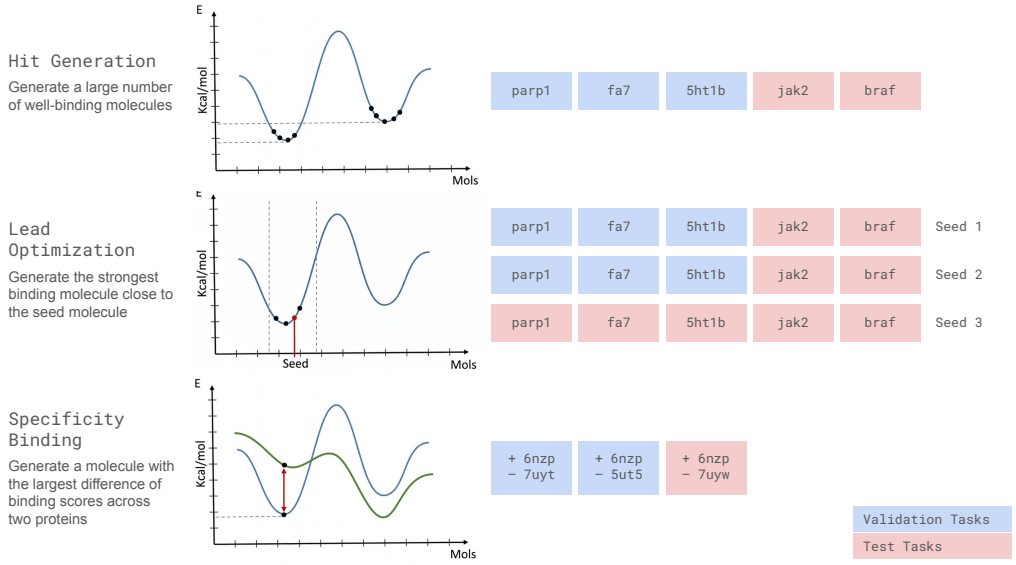


Figure 1: Overview of all 23 PMO-Dock tasks organized by oracle category. Each task is shown as a separate cell, color-coded by split: validation tasks (light blue) are used for hyperparameter tuning with unlimited oracle calls, while test tasks (light red) are strictly held out with limited evaluation budgets. Specificity Binding tasks focus on selective target engagement, Hit Generation tasks maximize the fraction of molecules meeting docking thresholds, and Lead Optimization tasks optimize docking scores while maintaining similarity to seed molecules.

see Appendix C), and $s_{\text{ref}}^{(\text{spec})} := s^{(6\text{NZP})}(\text{Deucravacitinib})$ is the corresponding score for the reference ligand Deucravacitinib under that pipeline (Appendix D.2).

$$\mathcal{L}(x) = \max(0,\ \text{Dock}(x, P_{\text{target}}) - \text{Dock}(x, P_{\text{off}}))$$

where $\text{Dock}(\cdot,\cdot)$ denotes the same QuickVina score as $s^{(\cdot)}(\cdot)$ above, and the score is left-trimmed at 0, so negative values are clipped to 0. For *all* specificity tasks, $P_{\text{target}}$ is the **same** on-target pocket, 6NZP. Each task uses a different off-target pocket $P_{\text{off}}$ (7UYT, 5UT5, or 7UYW; see below). This formulation promotes ligands with selective binding toward 6NZP while penalizing off-target interactions. Each algorithm is evaluated using three random seeds, and performance is reported as the average best specificity score across all seeds.

The docking procedure is described in detail in Appendix C.

### 2.4 Validation/Test Split and Model Selection

We define 23 tasks varying across objectives, targets, and metrics, split into **11 validation tasks** and **12 test tasks**. Unlimited optimization runs are permitted on validation tasks; oracle calls on test tasks are strictly limited. Figure 1 shows the complete task composition across all three categories.

**Model Selection Protocol.** Algorithm developers can perform unlimited hyperparameter sweeps on validation tasks. We consider three selection strategies: (a) **task-agnostic averaging**, selecting hyperparameters that maximize average performance across all validation tasks; (b) **category-specific selection**, choosing hyperparameters separately for each task category (hit generation, lead optimization, specificity); and (c) **adaptive selection**, prioritizing target protein similarity regardless of task constraints.

We hypothesize that the optimal strategy varies across methods (tested in Section 5). We support **adaptive hyperparameter selection** provided it is automatically determined solely from task semantics, target protein(s), and objective functions, without accessing test task performance. For example, if hyperparameters are selected based on protein similarity to validation targets, similarity metrics

must be defined and tuned exclusively on validation data, ensuring no information leakage from test tasks. Additional experimental methodology (hyperparameter sweeps, thresholds) is summarized in Appendix D.

# 3 METHODS EVALUATED

We evaluate four diverse methods spanning different optimization paradigms: Saturn Guo & Schwaller (2024c) (reinforcement learning with language models), GenMol Lee et al. (2025) (discrete diffusion), Genetic-guided GFlowNets Kim et al. (2024) (GFlowNet with genetic operators), and Chemlactica Guevorguian et al. (2024) (LLM-based population optimization). Detailed descriptions of each method and their hyperparameter configurations are provided in Appendix A.

To ensure fair cross-method comparison on PMO-Dock, we integrated method-specific inference pipelines into a unified benchmarking framework with task-aligned oracles, standardized evaluation budgets, and consistent metric computation across all benchmark tasks. Our implementation, including benchmark wrappers and evaluation code, is available in this repository and in PMO-Dock(`https://github.com/YerevaNN/PMO-Dock`).

# 4 HYPERPARAMETER TRANSFER ANALYSIS PROTOCOL

A central question in evaluating molecular optimization methods is whether hyperparameters tuned on one set of tasks generalize to new tasks. Robust algorithms should exhibit hyperparameters that work reasonably well across different protein targets and task types. We investigate this question systematically by measuring cross-task performance transfer.

## 4.1 TRANSFER MATRIX METHODOLOGY

We performed independent hyperparameter sweeps for each validation task and algorithm. For each task $i$, we identified the hyperparameter configuration maximizing performance on that task, then measured its performance on all other tasks $j$. Each matrix entry reports the *change* in task-$j$ performance when using task-$i$ hyperparameters versus task-$j$-optimal settings (after per-task normalization; Appendix B), so off-diagonal structure summarizes cross-task transfer rather than raw scores. This protocol quantifies whether tuning signals learned on one objective or target are predictive for others and motivates model selection strategies that avoid test-set leakage.

## 4.2 HYPERPARAMETER SELECTION STRATEGIES

To operationalize this analysis, we compare two selection strategies across all validation tasks. For each target task, hyperparameters are selected by maximizing average performance on either (a) Same-category tasks: all other tasks within the same category (hit, lead, or specificity); or (b) All-but-me tasks (*): all other validation tasks across all categories. Comparative outcomes under these strategies are reported in Section 5.

# 5 RESULTS

## 5.1 HYPERPARAMETER TRANSFERABILITY

Figure 3 compares the two selection strategies on validation tasks.

The results suggest algorithm-specific preferences. Using per-task normalized differences (same-category minus all-but-me), Saturn and Chemlactica favor same-category selection on average ($0.033$, $n{=}11$ tasks; and $0.067$, $n{=}11$ tasks, respectively), whereas GenMol favors global all-but-me selection ($-0.028$, $n{=}11$ tasks). Genetic GFN is closer to indifferent ($-0.002$, $n{=}11$ tasks). These task-level comparisons are consistent with the cross-category transfer patterns in the transfer matrices (Figure 4).

**Within-category transfer.** Within-category transfer analysis (Appendix B.4) suggests stronger cross-target alignment for Saturn than for GenMol. For parp1 vs 5ht1b hit generation, Saturn shows

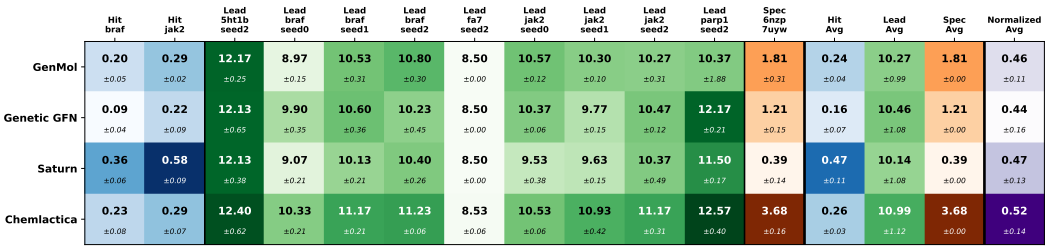


Figure 2: Test task leaderboard. Rows: methods; columns: test tasks by category. Hyperparameters selected per Section 4. Best per task highlighted. Colors are normalized separately by category, so intensities should only be compared within category. More intense colors indicate higher performance relative to other entries in the same category.

$r$=0.91 ($n$=16), whereas GenMol is near zero ($r$=$-0.03$, $n$=120), consistent with GenMol's weaker sensitivity to category-specific selection.

**Cross-category transfer.** Cross-category transfer analysis (Figure 9) suggests positive alignment for Saturn between average hit rate and average lead score at $\delta$=0.6 ($r$=0.85, $n$=16), whereas GenMol remains near zero ($r$= $-$ 0.05, $n$=120). The transfer matrices reinforce this difference: Saturn's mean within-category degradation is $-0.050$ ($n$=38 off-diagonal cells) versus $-0.177$ across categories ($n$=72), while GenMol shows a smaller separation ($-0.043$ within category, $n$=38; $-0.070$ across category, $n$=72). Within-protein analysis still indicates that task-type boundaries matter.

We also examined constraint sensitivity (similarity threshold $\delta$ variations, Appendix B.6) and specificity task transfer (Appendix B.7), finding incomplete transfer across constraints and across specificity pairs, especially for GenMol and Genetic GFN. Held-out target analysis (fa7+parp1 $\rightarrow$ 5ht1b, Appendix B.8) shows method-dependent behavior: Saturn exhibits substantially stronger alignment between validation-side aggregates and 5ht1b hit rate than the other methods, underscoring that validation performance does not uniformly predict held-out targets. Overall, current methods remain sensitive to hyperparameter choice across protein targets, task types, and constraints.

### 5.2 Main Leaderboard

Using hyperparameter selection strategies from Section 4, we present results on 12 test tasks (Figure 2). By the table's normalized aggregate (last column), Chemlactica ranks first, followed by Saturn, then GenMol, and Genetic GFN. Note that none of the methods were able to find a significantly better binding molecule for the lead optimization task with fa7 protein and seed #2.

The unusually high scores on certain instances of specificity binding tasks require further investigation of potential reward hacking.

## Acknowledgments

The research was supported by the Higher Education and Science Committee of MESCS RA (Research project No 24FP-1A058).

## REFERENCES


Amr Alhossary, Stephanus Daniel Handoko, Yuguang Mu, and Chee-Keong Kwoh. Fast, accurate, and reliable molecular docking with quickvina 2. *Bioinformatics*, 31(13):2214–2216, 2015. doi: 10.1093/bioinformatics/btv082. URL https://doi.org/10.1093/bioinformatics/btv082.

Guy W Bemis and Mark A Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of medicinal chemistry*, 39(15):2887–2893, 1996. doi: 10.1021/jm9602928. URL https://doi.org/10.1021/jm9602928.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems*, 34:27381–27394, 2021.

Jens Degen, Corina Wegscheid-Gerlach, Andrea Zaliani, and Matthias Rarey. On the art of compiling and using 'drug-like' chemical fragment spaces. *ChemMedChem: Chemistry Enabling Drug Discovery*, 3(10):1503–1507, 2008. doi: 10.1002/cmdc.200800178. URL https://doi.org/10.1002/cmdc.200800178.

Wenhao Gao, Tianfan Fu, Jimeng Sun, and Connor Coley. Sample efficiency matters: a benchmark for practical molecular optimization. *Advances in Neural Information Processing Systems*, 35: 21342–21357, 2022.

Philipp Guevorguian, Menua Bedrosian, Tigran Fahradyan, Gayane Chilingaryan, Hrant Khachatrian, and Armen Aghajanyan. Small molecule optimization with large language models, 2024. URL https://arxiv.org/abs/2407.18897.

Jeff Guo and Philippe Schwaller. Augmented memory: Sample-efficient generative molecular design with reinforcement learning. *JACS Au*, 4(4):1401–1413, 2024a. doi: 10.1021/jacsau.4c00066. URL https://doi.org/10.1021/jacsau.4c00066.

Jeff Guo and Philippe Schwaller. Beam enumeration: Probabilistic explainability for sample efficient self-conditioned molecular design, 2024b. URL https://arxiv.org/abs/2309.13957.

Jeff Guo and Philippe Schwaller. Saturn: Sample-efficient generative molecular design using memory manipulation, 2024c. URL https://arxiv.org/abs/2405.17066.

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

Hyeonah Kim, Minsu Kim, Sanghyeok Choi, and Jinkyoo Park. Genetic-guided gflownets for sample efficient molecular optimization, 2024. URL https://arxiv.org/abs/2402.05961.

Seul Lee, Karsten Kreis, Srimukh Prasad Veccham, Meng Liu, Danny Reidenbach, Yuxing Peng, Saee Paliwal, Weili Nie, and Arash Vahdat. Genmol: A drug discovery generalist with discrete diffusion, 2025. URL https://arxiv.org/abs/2501.06158.

Garrett M Morris, Ruth Huey, William Lindstrom, Michel F Sanner, Richard K Belew, David S Goodsell, and Arthur J Olson. Autodock4 and autodocktools4: Automated docking with selective receptor flexibility. *Journal of computational chemistry*, 30(16):2785–2791, 2009. doi: 10.1002/jcc.21256. URL https://doi.org/10.1002/jcc.21256.

Emmanuel Noutahi, Cristian Gabellini, Michael Craig, Jonathan S. C. Lim, and Prudencio Tossou. Gotta be safe: A new framework for molecular design, 2023. URL https://arxiv.org/abs/2310.10773.

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

# A Detailed Method Descriptions and Hyperparameter Configurations

This section provides detailed descriptions of each method evaluated in the benchmark, followed by complete hyperparameter search space specifications.

## A.1 Saturn

Saturn Guo & Schwaller (2024c) is a language-model-based molecular generator trained with reinforcement learning using an autoregressive policy that generates SMILES representations of molecules. It employs an *Augmented Memory* Guo & Schwaller (2024a) replay buffer that stores the top-$K$ high-reward molecules encountered during search (e.g., $K = 100$ in the original work). Buffer entries are repeatedly replayed by training on randomized SMILES augmentations to improve sample efficiency. A scaffold-based diversity filter (e.g., Bemis–Murcko Bemis & Murcko (1996) scaffold tracking) penalizes repeated scaffolds to mitigate mode collapse.

The authors proposed an additional feature, graph-based genetic operators (crossover and mutation) applied to the buffer population to produce new candidate molecules that can refresh the buffer. We kept this genetic algorithm component as a binary hyperparameter. The other hyperparameters we tested are buffer size $\sigma \in \{64, 128, 256, 512\}$ and two reward types: the one borrowed from GEAM algorithm and a hit-based reward. This resulted in 16 hyperparameter combinations.

**Reward Functions.** Saturn uses multiplicative reward functions that differ from the additive formulation used by Genetic-guided GFlowNets. For **hit optimization** tasks, the reward is:

$$R(x) = \frac{\text{Dock}_{\text{target}}(x)}{20} \cdot \text{QED}(x) \cdot \frac{10 - \text{SA}(x)}{9},$$

where all components are normalized to $[0, 1]$.

For **lead optimization** tasks, the reward is:

$$R(x) = \text{Sim}(x, x_0) \cdot \frac{\text{Dock}_{\text{target}}(x)}{20} \cdot \text{QED}(x) \cdot \frac{10 - \text{SA}(x)}{9},$$

where the first term encourages high similarity to the seed molecule $x_0$.

For **specificity optimization** tasks, Saturn's reward function explicitly penalizes anti-target binding:

$$R(x) = \left(1 - \frac{\text{Dock}_{\text{off}}(x)}{20}\right) \cdot \frac{\text{Dock}_{\text{target}}(x)}{20} \cdot \text{QED}(x) \cdot \frac{10 - \text{SA}(x)}{9},$$

where the first term $(1 - \text{Dock}_{\text{off}}(x)/20)$ encourages low anti-target binding.

## A.2 GenMol

GenMol Lee et al. (2025) is a masked discrete-diffusion model operating on a fragment-level representation SAFE Noutahi et al. (2023), where molecules are decomposed into fragment tokens (e.g., BRICS Degen et al. (2008) fragments). The model uses a bidirectional (BERT-like) denoising network and performs non-autoregressive iterative unmasking: starting from an all-masked fragment string, it repeatedly predicts token logits, locks in confident tokens, and repeats until decoding completes. For optimization, GenMol applies *fragment remasking*: entire fragments in a molecule are masked and resampled by the diffusion model, enabling structured, chemically meaningful mutations suitable for lead optimization. The architecture supports de novo generation, fragment-constrained sampling, hit finding, and lead optimization within the same framework.

We performed an extensive hyperparameter sweep of 120 combinations for GenMol. We tested masking parameter $\gamma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$, randomness $r \in \{1, 1.2, 2, 3\}$, and softmax temperature $\tau \in \{0.5, 0.8, 1, 1.2, 1.5\}$.

**Reward Functions.** GenMol uses a binary reward structure. For **hit** and **lead optimization** tasks, if molecule $x$ satisfies all hit constraints (as defined in Section 2), the reward is the docking score:

$$R(x) = \begin{cases} \text{Dock}_{\text{target}}(x) & \text{if } h(x) = 1 \\ 0 & \text{otherwise} \end{cases},$$

For **specificity optimization** tasks, the reward is:

$$R(x) = \begin{cases} (\text{Dock}_{\text{target}}(x) - \text{Dock}_{\text{off}}(x)) & \text{if } h(x) = 1 \\ 0 & \text{otherwise} \end{cases},$$

where $h(x)$ is the validity indicator function (equal to 1 if all constraints are satisfied, 0 otherwise).

### A.3 GENETIC-GUIDED GFLOWNETS

This method Kim et al. (2024) pretrains a SMILES generative policy (e.g., an RNN) on large chemical corpora to serve as a prior. During optimization, it fine-tunes the policy with the GFlowNet Bengio et al. (2021) trajectory-balance objective while integrating genetic operators: parent selection, Graph-GA Jensen (2019) crossover and mutation to produce offspring, and replay-buffered off-policy TB updates. Training uses rank-weighted sampling from the buffer and a combined loss consisting of the TB-loss plus a KL penalty to the pretrained prior: $\mathcal{L} = \text{TB-loss} + \lambda\,\text{KL}(\pi_{\text{pre}} \,\|\, \pi_{\text{new}})$. This hybrid leverages GA to produce high-reward candidates and GFlowNet TB training to amortize sampling from promising regions while maintaining diversity.

**Reward Functions.** While Genetic-GFN is primarily introduced for scalar-valued objectives, the original paper applies it to multi-objective molecular optimization by defining a scalar score as a linear combination of multiple objectives with specified coefficients; we follow the same approach. We use scalar scores $s(x)$ aligned with the PMO-Dock task objectives (Section 2) and set $R(x) = \exp(-\beta s(x))$ as in the GFlowNet literature. $\beta$ is an inverse temperature parameter.

The range for $\beta$ is $\{10, 25, 50, 70, 100\}$. Motivated by the observation in the original Genetic-GFN paper that reward scaling in multi-objective settings affects the effective inverse temperature, we performed $\beta$ selection using one representative task per task category. Within each category, the number of objectives is fixed, so the selected $\beta$ was transferred to all tasks in that category. We found that $\beta = 50$ performed best in all categories and fixed it for all tasks.

In addition to $\beta$, we tuned the rank-based replay coefficient and the KL regularization coefficient on validation tasks. The rank coefficient was selected from $\{0.01, 0.03, 0.05\}$ and the KL coefficient from $\{0.0005, 0.001, 0.005\}$, following the official Genetic-GFN hyperparameter search space. All remaining components and hyperparameters were used without modification.

### A.4 CHEMLACTICA

Chemlactica Guevorguian et al. (2024) uses large SMILES language models as neural mutation/proposal operators inside a population-based optimization loop. A pool of top-performing molecules is maintained; the LLM is prompted to generate analogs of pool members (e.g., similarity prompts), generated candidates are scored by the oracle, and the pool is updated by retaining top unique molecules. When improvement stalls, the LLM is fine-tuned on the pool using molecule–score paired prompts (e.g., prompts that include property/score tokens) so the model learns explicit structure–property relationships; generation then resumes with the adapted model. The procedure cycles: LLM-based proposal $\rightarrow$ oracle evaluation $\rightarrow$ pool update $\rightarrow$ occasional fine-tuning.

We downloaded the most recent version of Chemlactica, trained on Llama 3-1B. We tested the following range of hyperparameters: pool size $P \in \{10, 20, 40, 80\}$, number of prompts $I \in \{25, 50, 100, 200\}$, and number of similar molecules in the prompt $S \in \{1, 2, 3, 4\}$.

**Reward Functions.** Chemlactica uses the following scalar reward construction:

$$d(m; [a, b]) = \begin{cases} 0 & \text{if } a \le m \le b, \\ \min\{|m - a|,\ |m - b|\} & \text{otherwise.} \end{cases}$$

Here, $[a, b]$ denotes the desired range for a given property, and $m$ is the value of that property.

$$G(d; \mu, \sigma) = \exp\left(-\frac{1}{2}\left(\frac{d - \mu}{\sigma}\right)^2\right).$$

$$R(m;[a,b],\mu,\sigma) = \begin{cases} 1 & \text{if } a \le m \le b, \\ \exp\left(-\frac{1}{2}\left(\frac{d(m;[a,b])-\mu}{\sigma}\right)^2\right) & \text{otherwise.} \end{cases}$$

$$\mu = 0.$$

$$\sigma_{\text{QED}} = 0.1, \quad \sigma_{\text{SAS}} = 0.7, \quad \sigma_{\text{DOCKING}} = 2.0, \quad \sigma_{\text{SIMILAR}} = 0.1.$$

$$R(x) = \prod_{i=1}^{n} R_i.$$

## B Additional Hyperparameter Transfer Analyses

This appendix supplements Section 4 and the transferability discussion in Section 5.1: validation-side selection-strategy comparison, full transfer matrices for all four methods, and follow-on analyses (cross-target hit, hit–lead, lead thresholds, specificity pairs, held-out 5ht1b).

### B.1 Hyperparameter selection strategy comparison (validation)

Figure 3 reports validation performance under category-specific hyperparameter selection versus global all-but-me (*) selection.

| | Hit 5ht1b | Hit fa7 | Hit parp1 | Lead 5ht1b seed0 | Lead 5ht1b seed1 | Lead fa7 seed0 | Lead fa7 seed1 | Lead parp1 seed0 | Lead parp1 seed1 | Spec 6nzp 5ut5 | Spec 6nzp 7uyt | Hit Avg | Lead Avg | Spec Avg | Normalized Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| GenMol * | 0.28 ±0.06 | 0.12 ±0.05 | 0.09 ±0.08 | 12.07 ±0.21 | 12.60 ±0.30 | 7.73 ±0.76 | 7.77 ±0.23 | 11.67 ±0.32 | 10.90 ±0.17 | 3.57 ±0.51 | 2.19 ±0.08 | 0.16 ±0.08 | 10.46 ±1.98 | 2.88 ±0.69 | 0.42 ±0.18 |
| GenMol | 0.26 ±0.07 | 0.10 ±0.06 | 0.08 ±0.05 | 12.10 ±0.17 | 9.17 ±2.71 | 7.47 ±0.76 | 7.53 ±0.35 | 11.57 ±0.21 | 11.03 ±0.23 | 3.20 ±1.34 | 2.23 ±0.20 | 0.15 ±0.08 | 9.81 ±1.87 | 2.71 ±0.48 | 0.39 ±0.18 |
| Genetic GFN * | 0.39 ±0.09 | 0.02 ±0.01 | 0.11 ±0.03 | 4.50 ±0.00 | 11.50 ±0.78 | 8.23 ±0.15 | 6.83 ±0.23 | 11.40 ±0.26 | 10.77 ±0.55 | 2.34 ±0.53 | 1.82 ±0.29 | 0.17 ±0.16 | 8.87 ±2.60 | 2.08 ±0.26 | 0.34 ±0.12 |
| Genetic GFN | 0.39 ±0.09 | 0.02 ±0.01 | 0.11 ±0.03 | 4.50 ±0.00 | 11.50 ±0.78 | 7.97 ±0.61 | 6.77 ±0.12 | 11.33 ±0.23 | 10.77 ±0.55 | 2.34 ±0.53 | 1.82 ±0.29 | 0.17 ±0.16 | 8.81 ±2.61 | 2.08 ±0.26 | 0.34 ±0.12 |
| Saturn * | 0.83 ±0.04 | 0.29 ±0.15 | 0.59 ±0.08 | 4.50 ±0.00 | 9.90 ±1.18 | 6.40 ±0.00 | 6.70 ±0.00 | 11.87 ±0.68 | 10.33 ±0.72 | 5.82 ±2.18 | 2.41 ±0.30 | 0.57 ±0.22 | 8.28 ±2.58 | 4.11 ±1.70 | 0.60 ±0.17 |
| Saturn | 0.90 ±0.02 | 0.29 ±0.15 | 0.72 ±0.05 | 4.50 ±0.00 | 11.10 ±0.80 | 8.50 ±0.00 | 6.70 ±0.00 | 12.27 ±0.23 | 10.10 ±1.28 | 5.82 ±2.18 | 2.41 ±0.30 | 0.63 ±0.25 | 8.86 ±2.64 | 4.11 ±1.70 | 0.63 ±0.15 |
| Chemlactica * | 0.47 ±0.07 | 0.37 ±0.08 | 0.46 ±0.08 | 12.83 ±0.06 | 10.23 ±0.97 | 8.70 ±0.26 | 7.27 ±0.49 | 8.87 ±0.76 | 12.37 ±0.06 | 4.67 ±0.12 | 4.03 ±0.62 | 0.43 ±0.04 | 10.04 ±2.00 | 4.35 ±0.32 | 0.60 ±0.19 |
| Chemlactica | 0.44 ±0.02 | 0.34 ±0.08 | 0.47 ±0.11 | 12.83 ±0.12 | 11.60 ±0.36 | 8.77 ±0.12 | 7.33 ±0.35 | 9.37 ±0.40 | 12.27 ±0.68 | 4.67 ±0.12 | 4.03 ±0.62 | 0.42 ±0.06 | 10.36 ±2.00 | 4.35 ±0.32 | 0.60 ±0.19 |

Figure 3: Hyperparameter selection strategy comparison on validation tasks. Asterisks (*): global selection; unmarked: category-specific. Saturn and Chemlactica tend to benefit from category-specific selection, whereas GenMol is better matched by global all-but-me selection and Genetic GFN is closer to indifferent. Colors are normalized separately by category, so intensities should only be compared within category. More intense colors indicate higher performance relative to other entries in the same category.

### B.2 Transfer Heatmaps for Saturn and GenMol

Figure 4 shows transfer matrices for Saturn and GenMol. Each cell $(i, j)$ reports the relative change in task-$j$ performance when using hyperparameters selected from source task $i$ instead of task-$j$-optimal hyperparameters, after task-type normalization. Thus the heatmaps summarize transfer loss relative to the target-task optimum rather than absolute task performance.

Specificity tasks use a sparser experiment grid than hit and lead tasks. When a source-selected configuration is absent from the specificity grid, we evaluate transfer using the best overlapping configuration on the specificity grid under the same source-side selection criterion. This yields a valid proxy for source-task transfer on the subset of configurations available for specificity evaluation.

Summaries of the off-diagonal individual-task cells support the same pattern. Saturn's mean degradation is milder within category than across categories ($-0.050$, $n$=38 vs. $-0.177$, $n$=72). Chemlactica

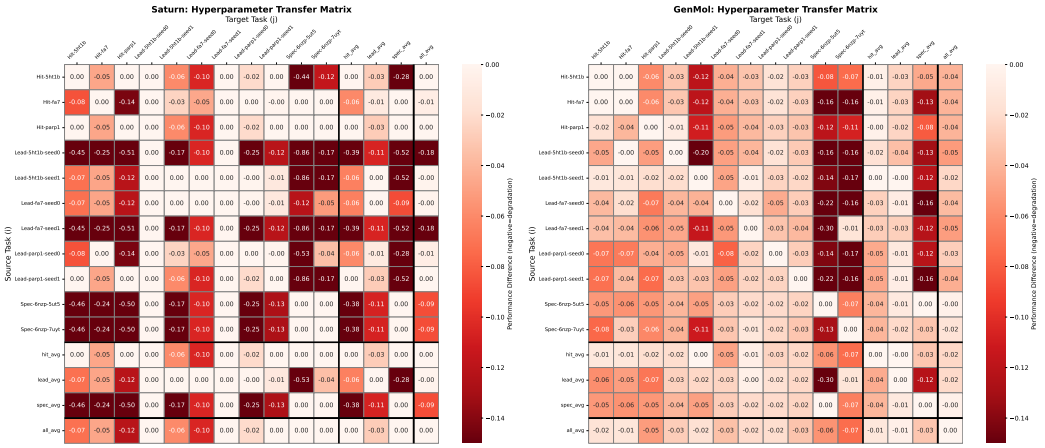


Figure 4: Hyperparameter transfer matrices for Saturn (left) and GenMol (right). Cell $(i, j)$ shows the change in task-$j$ performance when using hyperparameters selected from source task $i$ instead of task-$j$-optimal hyperparameters. Scores are normalized by task type before comparison (hit rates direct; lead scores divided by 20; specificity scores divided by 5). Because entries are measured relative to the target-task optimum, values are expected to be non-positive; values near 0 indicate little transfer loss, while more negative values indicate larger degradation. These matrices summarize relative transfer rather than absolute task performance. Thick separators distinguish individual tasks from aggregate rows and columns (‘hit$_a vg$‘, ‘$lead_a vg$‘, ‘$spec_a vg$‘, ‘$all_a vg$‘).

shows a similar separation ($-0.054$, $n$=38 vs. $-0.129$, $n$=72), whereas Genetic GFN and GenMol show much smaller within-versus-cross gaps.

### B.3 Transfer Heatmaps for Genetic GFlowNet and Chemlactica

Figure 5 shows the hyperparameter transfer matrices for Genetic GFlowNet and Chemlactica.

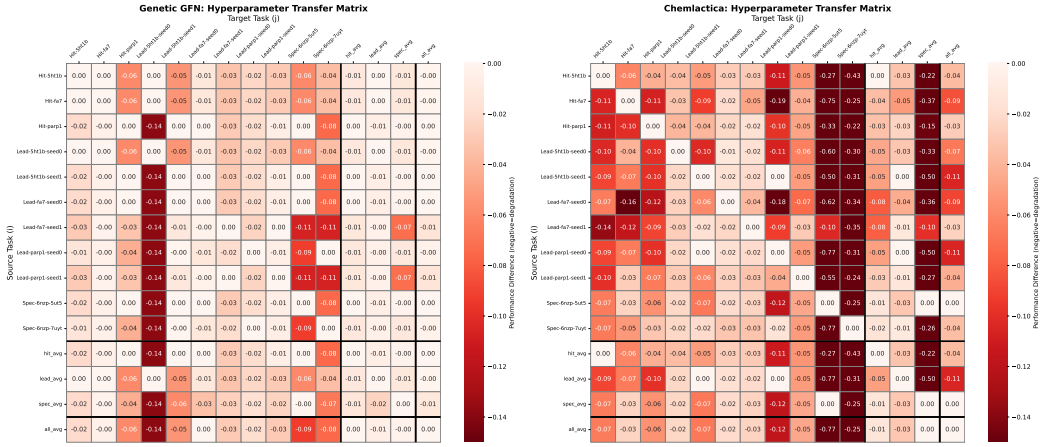


Figure 5: Hyperparameter transfer matrices for Genetic GFlowNet (left) and Chemlactica (right). As in Figure 4, each cell $(i, j)$ reports the change in task-$j$ performance when using hyperparameters selected from source task $i$ instead of task-$j$-optimal hyperparameters. Scores are normalized by task type before comparison (hit rates direct; lead scores divided by 20; specificity scores divided by 5). Values are expected to be non-positive because the reference is the target-task optimum; entries near 0 indicate little transfer loss and more negative entries indicate larger degradation.

### B.4 Cross-Target Hit Generation: Additional Protein Pairs

Figures 6, 7, and 8 show hit generation performance correlations across protein pairs. The patterns are consistent: Saturn shows substantially stronger positive alignment across targets within the hit category than the other methods, while GenMol remains near zero for parp1 vs 5ht1b.

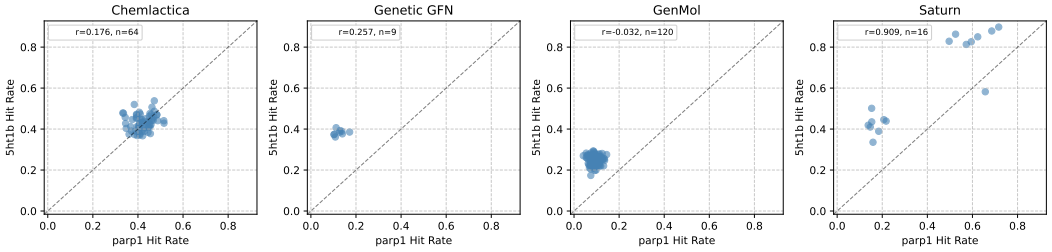


Figure 6: Cross-target hit generation performance: parp1 vs 5ht1b. Each point represents a hyperparameter configuration, averaged across random seeds. Legends report Pearson correlations and sample sizes. Saturn shows much tighter positive alignment than GenMol.

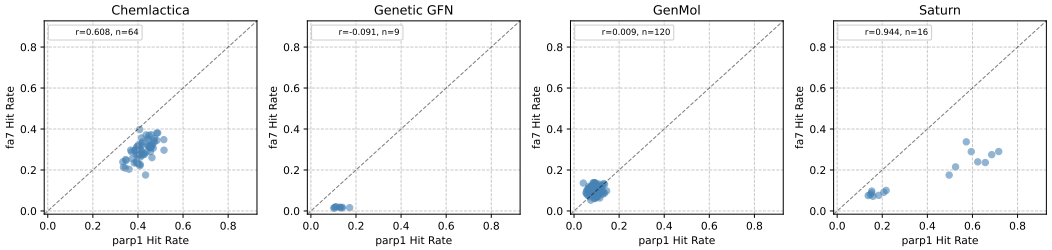


Figure 7: Cross-target hit generation performance: parp1 vs fa7. Each point represents a hyperparameter configuration, averaged across random seeds.

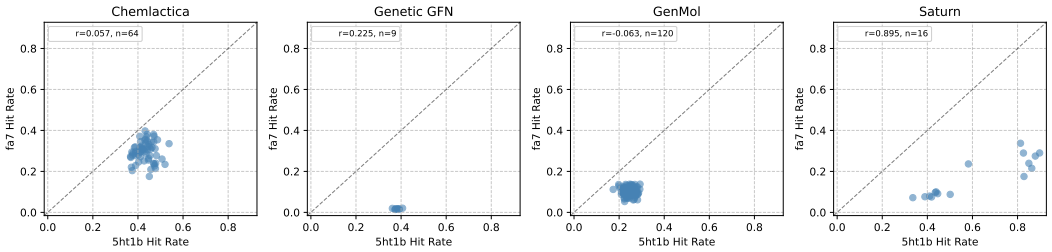


Figure 8: Cross-target hit generation performance: 5ht1b vs fa7. Each point represents a hyperparameter configuration, averaged across random seeds.

### B.5 Hit vs Lead: Per-Protein Breakdown

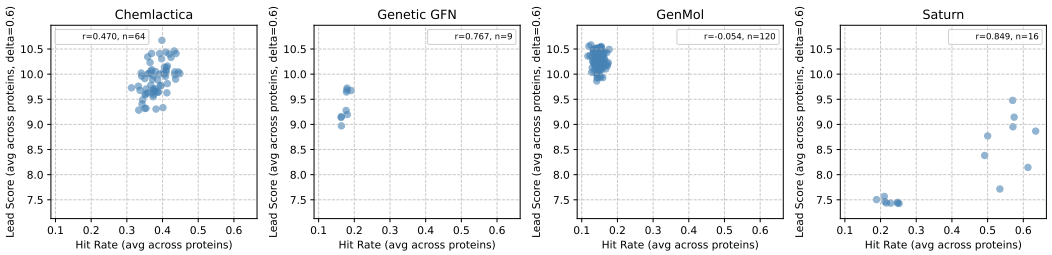


Figure 9: Hit vs lead (averaged across proteins; lead at $\delta$=0.6). Legends report Pearson correlations and sample sizes. Saturn shows clearer positive alignment than GenMol. Per-protein breakdown below.

Figure 10 shows lead optimization performance across protein targets. Different colors indicate different seed molecule pairs. The limited correlation reinforces that hyperparameters optimized for one protein rarely transfer effectively to another, even within the lead optimization category.

Figure 11 breaks down hit-to-lead transfer by individual protein targets. Each protein is shown in a different color. Even within the same protein, the best hyperparameters for lead optimization may underperform on hit generation tasks, demonstrating that task-type boundaries dominate over protein-specific effects.

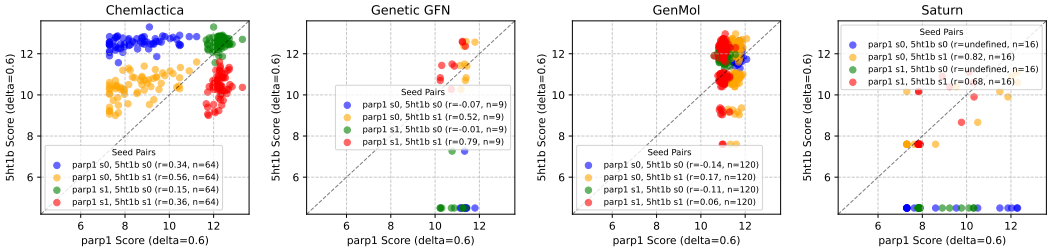


Figure 10: Cross-target lead optimization performance (parp1 vs 5ht1b, $\delta = 0.6$). Colors represent different seed molecule pairs. Points are aggregated across random seeds, and legends report Pearson correlations and sample sizes for each seed-pair subset.

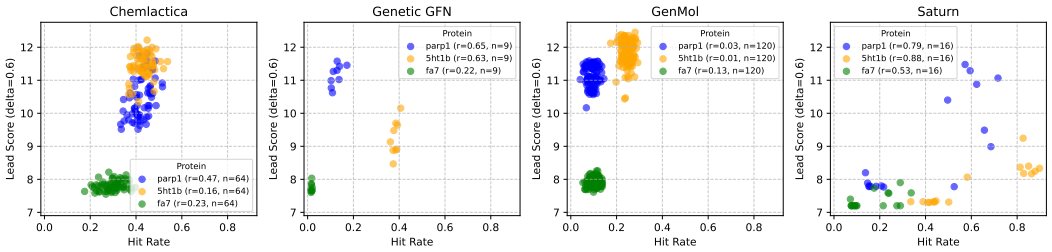


Figure 11: Hit rate versus lead optimization score ($\delta = 0.6$) by protein. Each protein is shown in a different color. Scores are aggregated across random seeds and seed molecules (for lead tasks), and the legends report Pearson correlations and sample sizes. The scatter suggests that even within-protein hit-to-lead transfer is imperfect, with best lead hyperparameters often degrading hit performance.

### B.6 Lead Optimization: Threshold Transfer

Figure 12 examines transfer when varying the similarity threshold $\delta$ in lead optimization tasks. Each color represents a different (target, seed molecule) combination. The scatter reveals that hyperparameters optimized for $\delta = 0.4$ do not reliably transfer to $\delta = 0.6$, indicating sensitivity to constraint tightness. In a few subsets the reported Pearson correlation is undefined because one threshold yields constant scores across experiments, so the variance needed for correlation is zero.

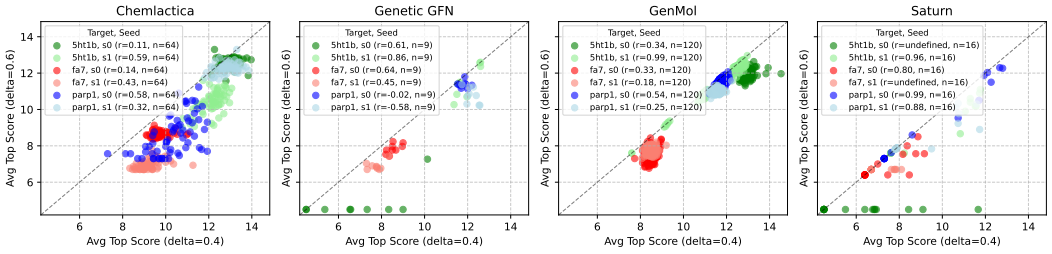


Figure 12: Lead optimization performance comparison across threshold values ($\delta = 0.4$ vs $\delta = 0.6$). Each color represents a specific (target, seed molecule) pair. Legends report Pearson correlations and sample sizes. Labels showing $r =$ undefined correspond to subsets where one threshold produces constant scores across experiments, making Pearson correlation undefined.

### B.7 Specificity Task Transfer

Figure 13 compares performance on two different specificity tasks (6nzp-5ut5 vs 6nzp-7uyt). The specificity score is computed as $\text{dock}_{\text{target}} - \text{dock}_{\text{off-target}}$. Correlation between the two pairs varies by method (more positive for Chemlactica and Saturn, weaker or more uncertain for GenMol and Genetic GFN), so strong performance on one pair does not guarantee strong performance on the other.

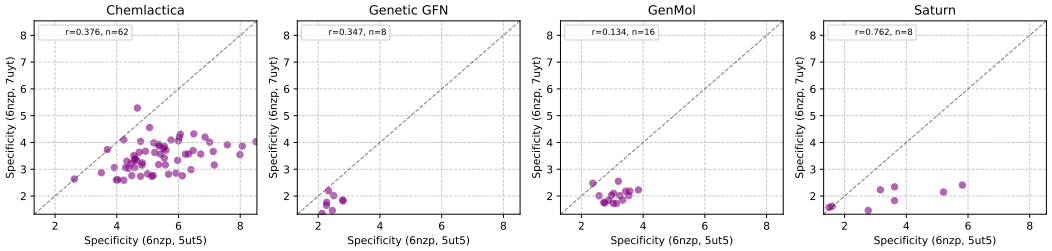


Figure 13: Comparison of specificity task performance (6nzp-5ut5 vs 6nzp-7uyt). Axes: mean top-10 specificity score (delta) for each pair. Legends report Pearson correlations and sample sizes. Correlation is method-dependent, indicating incomplete transfer across antitarget choices.

### B.8 Held-Out Target Generalization

Figure 14 examines whether hyperparameters optimized on fa7 and parp1 transfer to 5ht1b as a held-out target. The 2×4 grid shows 5ht1b hit performance on the Y-axis throughout. In the top row, the X-axis shows average hit performance on fa7 and parp1. In the bottom row, the X-axis shows combined hit and lead performance (normalized) on fa7 and parp1. This simulates the realistic scenario where validation tasks guide hyperparameter selection and test tasks measure true generalization. Saturn shows substantially stronger positive alignment between validation-side aggregates and 5ht1b hit rate than the other methods, so aggregate validation performance does not uniformly predict held-out 5ht1b hit rate.

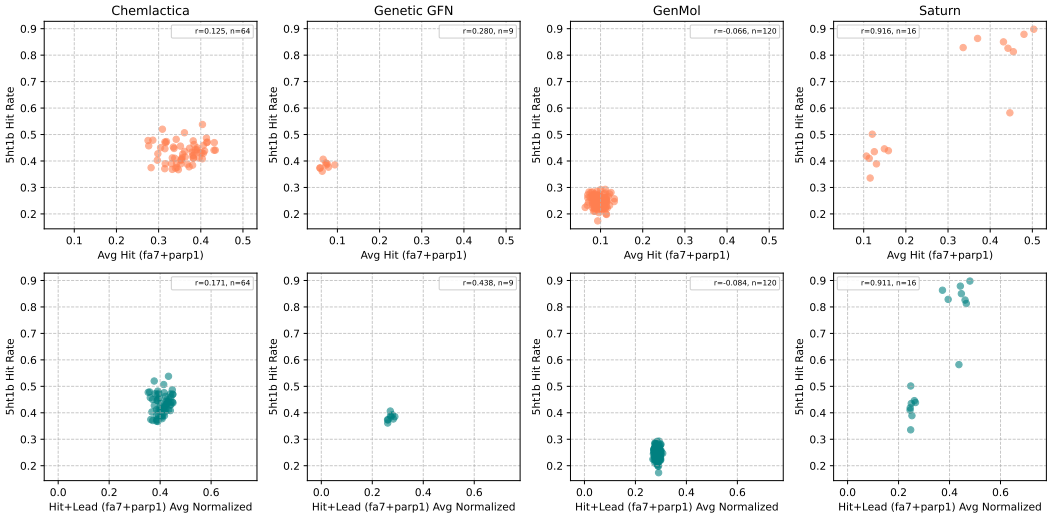


Figure 14: Held-out target transfer analysis: 5ht1b hit performance versus other validation tasks. Top row: 5ht1b hit vs average hit on fa7+parp1. Bottom row: 5ht1b hit vs combined hit+lead on fa7+parp1 (normalized). Each column represents one algorithm. Legends report Pearson correlations and sample sizes. Saturn shows substantially stronger positive alignment than the other methods.

## C Details of the Docking Pipeline for Specificity Tasks

Hit and lead tasks use QuickVina with exhaustiveness 1 as stated in Section 2. The specificity tasks use the same scoring function but a separate docking pipeline with higher exhaustiveness for the larger binding boxes described below.

Docking evaluations were performed using QuickVina Alhossary et al. (2015). For all three specificity tasks, the binding box was defined around the ligands observed in the crystallographic structures, ensuring that all amino acid residues participating in ligand interactions were included. The grid box

volume did not exceed 27,000 Å$^3$ in any case. Based on this volume, the exhaustiveness parameter was set to 8, according to the Vina manual. All remaining preprocessing steps were performed following a standard protocol. Non-essential molecules - including crystallographic water molecules and co-crystallized ligands were removed from the structures. Structures were then analyzed for the presence of alternative atom conformations (altlocs). In cases where alternative positions were observed, the conformation with the highest occupancy was selected. Structures were further checked for missing atoms, and all identified missing atoms were reconstructed (the files are available on GitHub). Hydrogen atoms were added using AutoDock Tools Morris et al. (2009), retaining only polar hydrogens to preserve hydrogen-bonding interactions, and Gasteiger partial charges were subsequently assigned to the protein.

To verify the reliability of our docking pipeline for the current specificity tasks, we tested it on a well-characterized reference compound, Deucravacitinib (BMS-986165) - a first-in-class, approved, highly selective TYK2 pseudokinase inhibitor. The docking results were consistent with experimental data: this molecule exhibited the highest docking score for TYK2 JH2 while showing lower docking scores for TYK2 JH1 as well as JAK2 JH1 and JH2. This validation confirms that our docking setup can reproduce known binding preferences and reliably assess target selectivity.

# D Experimental Methodology and Community Resources

This appendix provides hyperparameter sweep procedures, task-specific thresholds, and community resources. The validation/test split, task overview (Figure 1), and model selection protocol are described in Section 2.4.

## D.1 Hyperparameter Sweep Details

We ran independent hyperparameter sweeps on every validation task for Saturn, GenMol, Genetic-guided GFlowNet, and Chemlactica. Transfer matrices, same-category versus all-but-me aggregation, and test evaluation follow Sections 4 and 5; this subsection records that all tuning was confined to validation tasks before a single test-time evaluation per strategy. Complete search spaces and per-method training details are in Appendix A.

## D.2 Task-Specific Thresholds

### D.2.1 Hit Generation Thresholds

For hit generation tasks, the target-specific docking score thresholds $s_{\min}^{(t)}$ are:

$$s_{\min}^{(\mathrm{PARP1})} = 10.0, \quad s_{\min}^{(\mathrm{FA7})} = 8.5, \quad s_{\min}^{(\mathrm{5HT1B})} = 8.7845, \quad s_{\min}^{(\mathrm{BRAF})} = 9.20, \quad s_{\min}^{(\mathrm{JAK2})} = 10.30.$$

### D.2.2 Specificity Tasks: Reference Docking Score and Property Filters

For specificity tasks, the validity indicator $h(x)$ requires $\mathrm{QED}(x) \in [0.4, 1.0]$ and $\mathrm{SA}(x) \in [1.0, 4.0]$ in addition to an on-target docking floor. Let $s_{\mathrm{ref}}^{(\mathrm{spec})} := s^{(\text{TYK2 JH2})}(\text{Deucravacitinib})$ denote the QuickVina docking score of the reference inhibitor Deucravacitinib (BMS-986165) against TYK2 JH2 under the protocol in Appendix C. We use

$$s_{\mathrm{ref}}^{(\mathrm{spec})} = 10.67.$$

Molecules must satisfy $s^{(\text{TYK2 JH2})}(x) \geq s_{\mathrm{ref}}^{(\mathrm{spec})}$ to count as hits for the specificity objective.

## D.3 Code and Community Contributions

We invite the community to contribute novel methods and to extend the benchmark with additional objectives. Benchmark wrappers, evaluation code, and baselines accompany this submission. An extended public release with additional baselines and task definitions is available as PMO-Dock: `https://github.com/YerevaNN/PMO-Dock`.