Beam Enumeration: Probabilistic Explainability For Sample Efficient Self-conditioned Molecular Design

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Generative molecular design has moved from proof-of-concept to real-world applicability, as marked by the surge in very recent papers reporting experimental validation. Key challenges in explainability and sample efficiency present opportunities to enhance generative design to directly optimize expensive high-fidelity oracles and provide actionable insights to domain experts. Here, we propose Beam Enumeration to exhaustively enumerate the most probable sub-sequences from language-based molecular generative models and show that molecular substructures can be extracted. When coupled with reinforcement learning, extracted substructures become meaningful, providing a source of explainability and improving sample efficiency through self-conditioned generation. Beam Enumeration is generally applicable to any language-based molecular generative model and notably further improves the performance of the recently reported Augmented Memory algorithm, which achieved the new state-of-the-art on the Practical Molecular Optimization benchmark for sample efficiency. The combined algorithm generates more high reward molecules and faster, given a fixed oracle budget. Beam Enumeration is the first method to jointly address explainability and sample efficiency for molecular design. The code is available at `https://figshare.com/s/d0cd53fc14027accd7b0`.

1 Introduction

Molecular discovery requires identifying candidate molecules possessing desired properties amidst an enormous chemical space[1]. Generative molecular design has become a popular paradigm in drug discovery, offering the potential to navigate chemical space more efficiently with promise for accelerated discovery. Very recently, efforts have come to fruition and a large number of works have reported experimental validation of generated inhibitors, notably for both distribution learning[2–14] and goal-directed generation[15–20] approaches. Perhaps now more than ever, existing challenges in explainability and sample efficiency offer an avenue to propel generative molecular design towards outcomes that are not yet possible. Specifically, if one can elucidate *why* certain substructures or molecules satisfy a target objective, the model's *knowledge* can be made actionable, for example, in an interplay with domain experts. Moreover, sample efficiency concerns with how many experiments, i.e., oracle calls, are required for a model to optimize the target objective. This is a pressing problem as the most informative high-fidelity oracles are computationally expensive, e.g., molecular dynamics (MD) for binding energy prediction[21,22]. If a generative model can *directly* optimize these expensive oracles, the capabilities of generative design can be vastly advanced.

Submitted to NeurIPS 2021 AI for Science Workshop.

In this work, we propose Beam Enumeration to exhaustively enumerate the most probable token subsequences in language-based molecular generative models and show that valid molecular substructures can be extracted from these partial trajectories. We demonstrate that the extracted substructures are informative when coupled with reinforcement learning (RL) and show that this information can be made *actionable* to self-condition the model's generation by only evaluating sampled molecules containing these substructures with the oracle. The results show significantly enhanced sample efficiency with an expected small trade-off in diversity. Beam Enumeration is the first method to jointly address explainability and sample efficiency. Our contribution is as follows:

1. We propose Beam Enumeration as a task-agnostic method to exhaustively enumerate subsequences and show that molecular substructures can be extracted. When coupled with RL

2. During the course of RL, extracted substructures provide structural insights and are on track to yield high rewards, which, in turn, enables self-conditioned molecular generation.

3. We perform exhaustive hyperparameter investigations (2,224 experiments and 144 with molecular docking) and provide insights on the predictable behavior of Beam Enumeration and recommend default hyperparameters for out-of-the-box applications.

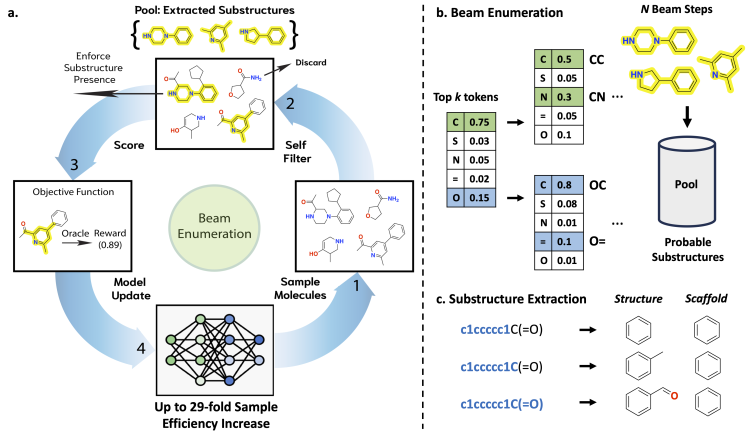
4. We combine Beam Enumeration with the recently reported Augmented Memory[23] optimization algorithm and show that the sample efficiency becomes sufficient (up to a 29-fold increase on the most challenging task) to find high reward molecules that satisfy a docking objective with only 2,000 oracle calls in three drug discovery case studies.

2 Related Work

Sample Efficiency in Molecular Design. Tailored molecular generation is vital for practical applications as every use case requires optimizing for a bespoke property profile. Over the past several years, so-called goal-directed generation has been achieved using a variety of architectures, including Simplified molecular-input line-entry system (SMILES)[24]-based recurrent neural networks (RNNs)[25–28], generative adversarial networks (GANs)[29–31], variational autoencoders (VAEs)[17,32,33], graph-based models[34–37], GFlowNets[38], and genetic algorithms[39]. However, while all methods can be successful in optimizing for various properties, the oracle budget, i.e., how many oracle calls (computational calculations) were required to do so, is rarely reported. To address this, Gao et al.[40] proposed the Practical Molecular Optimization (PMO)[40] benchmark, which assesses 25 models across 23 tasks and enforces a budget of 10,000 oracle calls. Recently, Guo et al. proposed Augmented Memory[23], which uses a language-based molecular generative model and achieves the new state-of-the-art on the PMO benchmark.

Explainability for Molecules. Explainable AI (XAI)[41] to interpret and explain model predictions is a vital component for decision-making. Existing methods include Gradient-weighted Class Activation Mapping (Grad-CAM)[42], which uses gradient-based heat maps for convolutional layers and Local Interpretable Model-agnostic Explanations (LIME)[43], which uses a locally interpretable model. Other methods include permutation importance[44] and SHAP values[45], which are model-agnostic. For molecules, the Molecular Model Agnostic Counterfactual Explanations (MMACE)[46] method was proposed to search for the most similar counterfactual (model predicts the opposite label) molecule. Recently, the pBRICS[47] algorithm was proposed to combine functional group decomposition with Grad-CAM to explain matched molecular pairs. While existing XAI methods can work well provided a dataset, making the explanations actionable *during* a generative design experiment that relies on an interplay between chemical space exploration and oracle feedback is difficult.

To address this limitation, we introduce *Beam Enumeration*, which extracts molecular substructures directly from the model's token sampling probabilities and derives explainability from a generative probabilistic perspective that is modulated by reward feedback. Moreover, when coupled with Augmented Memory[23], sample efficiency drastically improves.



C	0.75
S	0.03
N	0.05
=	0.02
O	0.15

C	0.5
S	0.05
N	0.3
=	0.05
O	0.1

C	0.8
S	0.08
N	0.01
=	0.1
O	0.01

Figure 1: Beam Enumeration overview. **a.** The proposed method proceeds via 4 steps: **1.** generate batch of molecules. **2.** filter molecules based on pool to enforce substructure presence, discarding the rest. **3.** compute reward **4.** update the model. After updating the model, if the reward has improved for consecutive epochs, execute Beam Enumeration. **b.** Beam Enumeration sequentially enumerates the top k tokens by probability for N beam steps, resulting in an exhaustive set of token sub-sequences. **c.** All valid substructures (either by the *Structure* or *Scaffold* criterion) are extracted from the sub-sequences. The most frequent substructures are used for self-conditioned generation.

3 Proposed Method: Beam Enumeration

In this section, each component of Beam Enumeration (Fig. 1) is described: the base molecular generative model, the Beam Enumeration algorithm, and how Beam Enumeration harnesses the model's built-in explainability which can be used to improve sample efficiency through self-conditioned generation (further details on Beam Enumeration are presented in Appendix A).

Autoregressive Language-based Molecular Generative Model. The starting point of Beam Enumeration is any autoregressive language-based molecular generative model. The specific model used in this work is Augmented Memory[23] which recently achieved the new state-of-the-art performance on the PMO[40] benchmark for sample efficiency, outperforming modern graph neural network-based approaches[48,49] and GFlowNets[50]. Augmented Memory builds on REINVENT[25,51] which is a SMILES-based[24] RNN using long-short-term memory (LSTM) cells[52]. The optimization process is cast as an on-policy RL problem. We define the state space, S_t, as all intermediate token sequences and the action space, $A_t(s_t)$, as the token sampling probabilities (conditioned on a given a state). $A_t(s_t)$ is given by the policy, π_θ, which is parameterized by the RNN. The objective is to iteratively update the policy such that token sampling, $A_t(s_t)$, yields trajectories (SMILES) with increasing reward. Formally, sampling a SMILES, x, is given by the product of conditional state probabilities (Equation 1), and the token sampling is Markovian:

$$P(x) = \prod_{t=1}^{T} P(s_t \mid s_{t-1}, s_{t-2}, \ldots, s_1) \tag{1}$$

Goal-directed generation proceeds by defining the Augmented Likelihood (Equation 2), where the Prior is the pre-trained model and S is the objective function returning a reward, given a SMILES, x.

$$\log \pi_{\theta_{\text{Augmented}}} = \log \pi_{\theta_{\text{Prior}}} + \sigma S(x) \tag{2}$$

The policy is directly optimized by minimizing the squared difference between the Augmented Likelihood and the Agent Likelihood given a sampled batch, B, of SMILES constructed following the actions, $a \in A^*$ (Equation 3):

$$L(\theta) = \frac{1}{|B|}\left[\sum_{a \in A^*}(\log \pi_{\theta_{\text{Augmented}}} - \log \pi_{\theta_{\text{Agent}}})\right]^2 \tag{3}$$

Minimizing $L(\theta)$ is equivalent to maximizing the expected reward as shown previously[23,53].

Beam Enumeration. Beam Enumeration is proposed based on the fact that on a successful optimization trajectory, it must become increasingly likely to generate high reward molecules. It is therefore reasonable to assume that the highest probability trajectories are more likely to yield high reward. Correspondingly, Beam Enumeration (Fig. 1) exhaustively enumerates the top k tokens (by probability) sequentially for N beam steps. Molecular substructures can be extracted from the set of *sub-sequences*, and we show how this information can be made *actionable*.

Probabilistic Explainability. Here, we describe how probabilistic explainability can be extracted from the exhaustive set of token sub-sequences. We hypothesized that molecular substructures can be extracted from a given sub-sequence by iteratively considering every (sub)-sub-sequence (Fig. 1). For example, given the sub-sequence "ABC", the set of (sub)-sub-sequences are: "A", "AB", and "ABC". It is expected that not every sub-sequence possesses (sub)-sub-sequences mapping to valid molecular substructures. Still, we show that a sufficient signal can be extracted (Appendix C). We implement two types of substructures: *Scaffold*, which extracts the Bemis-Murcko[54] scaffold and *Structure*, which extracts *any* valid substructure. The closest work to ours is the application of Beam Search[55,56] for molecular design[3]. Our work differs as the objective is to *exhaustively* enumerate the highest probability *sub-sequences* to extract molecular substructures for self-conditioned generation.

Self-conditioned Generation. The sub-sequences were enumerated by taking the most probable k tokens, and generating high reward molecules should be increasingly likely. Correspondingly, it is reasonable to posit that the most frequent molecular substructures are on track to becoming high reward full molecules and that the substructures themselves possess properties aligned with the target objective. The generative process can be self-conditioned to filter sampled batches for the presence of these molecular substructures and discard those that do not (Fig. 1).

Sample Efficiency Metrics. We define two metrics to assess sample efficiency: Generative Yield (referred to as Yield from now on) and Oracle Burden. Yield (Equation 4) is defined as the number of *unique* generated molecules above a reward threshold, where $g \in G$ are the molecules in the generated set, $\mathbb{I}$ is the indicator function which returns 1 if the reward, $R(g)$, is above a threshold, T. Yield is a useful metric for drug discovery as the generated set is usually triaged to prioritize molecules, e.g., based on synthetic feasibility, for experimental validation or more expensive computational oracles.

$$Generative\ Yield = \sum_{g=1}^{G} \mathbb{I}[R(g) > T] \tag{4}$$

Oracle Burden (Equation 5) is defined as the number of oracle calls (c) required to generate N *unique* molecules above a reward threshold. This is a direct measure of sample efficiency as high reward molecules satisfy the target objective, and the metric becomes increasingly important with expensive high-fidelity oracles.

$$Oracle\ Burden = c \mid \sum_{g=1}^{G} \mathbb{I}[R(g) > T] = N \tag{5}$$

4 Results and Discussion

We first design an illustrative experiment to demonstrate the feasibility of Beam Enumeration to extract meaningful substructures and, in turn, enable self-conditioned generation. Next, three drug discovery case studies to design prospective inhibitors were performed to demonstrate real-world

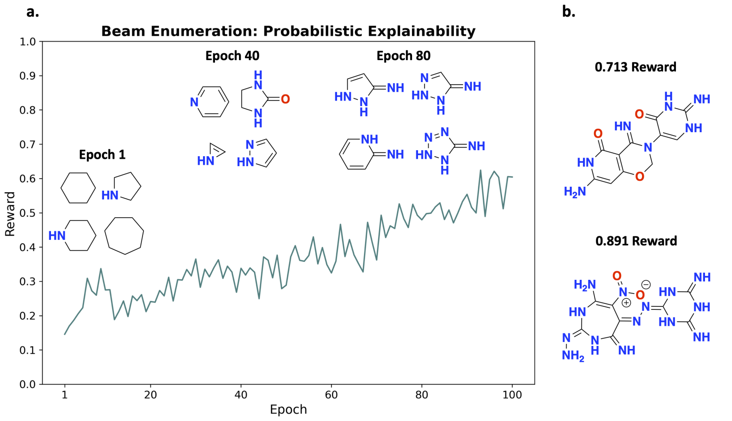


Figure 2: Illustrative experiment with the following multi-parameter optimization objective: maximize tPSA, molecular weight < 350 Da, number of rings ≥ 2. **a**. Augmented Memory[23] reward trajectory with annotated top-4 (excluding benzene) most frequent molecular substructure scaffolds at varying epochs using Beam Enumeration. **b.** Examples of molecules with high reward.

Table 1: Illustrative experiment: Beam Enumeration improves the sample efficiency of Augmented Memory. All experiments were run for 100 replicates with an oracle budget of 5,000 calls, and reported values are the mean and standard deviation. *Scaffold* and *Structure* indicate the type of substructure, and the number after is the *Structure Minimum Size*. Parentheses after Oracle Burden denote the cut-off number of molecules. Parentheses after values represent the number of unsuccessful replicates (for achieving the metric).

Metric	**Augmented Memory**				
	Beam Scaffold 15	**Beam Structure 15**	**Beam Scaffold**	**Beam Structure**	**Baseline**
Generative Yield$_{>0.7}$ (↑)	**1757 ± 305**	1669 ± 389	1117 ± 278	864 ± 202	496 ± 108
Generative Yield$_{>0.8}$ (↑)	**819 ± 291**	700 ± 389	425 ± 256	199 ± 122	85 ± 56
Oracle Burden$_{>0.7}$ (1) (↓)	**577 ± 310**	616 ± 230	1037 ± 414	897 ± 347	1085 ± 483
Oracle Burden$_{>0.7}$ (10) (↓)	947 ± 350	**926 ± 332**	1881 ± 259	1745 ± 292	2392 ± 216
Oracle Burden$_{>0.7}$ (100) (↓)	**1530 ± 468**	1547 ± 513	2736 ± 335	2713 ± 402	3672 ± 197
Oracle Burden$_{>0.8}$ (1) (↓)	**1311 ± 628**	1401 ± 695	2423 ± 487	2295 ± 482	3164 ± 492
Oracle Burden$_{>0.8}$ (10) (↓)	**1794 ± 617 (1)**	2009 ± 804 (1)	3124 ± 497	3241 ± 492	4146 ± 326
Oracle Burden$_{>0.8}$ (100) (↓)	**2704 ± 689 (1)**	2943 ± 811 (6)	3973 ± 592 (6)	4415 ± 437 (20)	4827 ± 170 (69)

application. The key result we convey is that Beam Enumeration can be added directly to existing algorithms, and it both provides structural insights and improves sample efficiency to not only generate more high reward molecules, but also faster, given a fixed oracle budget.

4.1 Illustrative Experiment

Extracted Substructures are Meaningful. The illustrative experiment aims to optimize the following multi-parameter optimization (MPO) objective: maximize topological polar surface area (tPSA), molecular weight (MW) < 350 Da, and number of rings ≥ 2. This specific MPO was chosen because satisfying the objective *requires* generating rings saturated with heteroatoms. Augmented Memory[23] was used to optimize the MPO objective. The reward trajectory tends towards 1, indicating the model gradually learns to satisfy the target objective, as desired (Fig. 2) Next, we investigate the top k and N beam steps parameters for Beam Enumeration and show that while the majority of sub-sequences do not possess valid substructures, a meaningful signal can still be extracted (Appendix C). We hypothesize that the optimal parameters are using a low top k as we are interested in the most probable sub-sequences and large N beam steps, which would enable extracting larger (and potentially more meaningful) substructures. Fig. 2 shows the top-4 substructures from Beam Enumeration at varying epochs. The substructures are informative when considering the MPO objective: the most frequent substructures gradually become rings saturated with heteroatoms, which possess a high tPSA.

Self-conditioned Generation Improves Sample Efficiency. Thus far, the results only show that Beam Enumeration can extract meaningful molecular substructures. To enable self-conditioned generation, we consider *when* extracted substructures would be meaningful and propose to execute Beam Enumeration when the reward improves for *Patience* number of successive epochs (to mitigate sampling stochasticity). We combine Beam Enumeration with Augmented Memory[23] and perform an exhaustive hyperparameter grid search (with replicates) using Yield and Oracle Burden as the performance metrics (Appendix A). The results elucidate the behavior of Beam Enumeration with three key observations: firstly, *Structure* extraction is permissive compared to *Scaffold* and often leads to small functional groups being the most frequent substructures which diminish the sample efficiency benefits (Appendix C). Secondly, enforcing larger substructures to be extracted (*Structure Minimum Size*) improves performance across all hyperparameter combinations. This reinforces that extracted substructures are meaningful as larger substructures heavily bias molecular generation during self-conditioning. If they were not meaningful, sample efficiency would not improve (and would likely be detrimental). Thirdly, *Structure* extraction while enforcing a higher *Structure Minimum Size* prevents small functional group extraction which significantly enhances performance. Subsequently, we perform five experiments (N=100 replicates each) based on the optimal hyperparameters identified: Augmented Memory[23] (baseline) and Augmented Memory with Beam Enumeration (*Scaffold* and *Structure* with and without *Structure Minimum Size* = 15). Table 1 shows that Beam Enumeration drastically improves the Yield and Oracle Burden compared to the baseline at both the > 0.7 and > 0.8 reward thresholds, especially when *Structure Minimum Size* = 15 is enforced. We highlight that the improved sample efficiency is significant as baseline Augmented Memory could not find 100 molecules > 0.8 reward in 69/100 replicates.

4.2 Drug Discovery Case Studies

Next, we apply Beam Enumeration to drug discovery case studies to design inhibitors against DRD2 which is implicated in neurodegenerative diseases[57], MK2 kinase which is involved in pro-inflammatory responses[58], and AChE which is a target of interest against Alzheimer's disease[59]. Following Guo et al.[23,60], we formulate the following MPO objective: minimize the AutoDock Vina[61] docking score, maximize the QED[62] score, and MW < 500 Da. The QED and MW objectives prevent the generative model from exploiting the weaknesses of docking algorithms to give inflated docking scores to large, lipophilic molecules, which can be promiscuous binders[63]. Moreover, an oracle budget of 5,000 Vina calls was enforced which is almost half the budget of the original Augmented Memory[23] work (9,600). Since the observations made from the previous hyperparameter grid search may not be generalizable to docking tasks, we perform an additional hyperparameter grid search (with replicates). The results (Appendix D) show that the optimal hyperparameters across all drug discovery case studies are the same as the illustrative experiment. We designate these the default hyperparameters and demonstrate the applicability of Beam Enumeration to both Augmented Memory[23] and REINVENT[25,51] which is the second most (behind Augmented Memory) sample efficient model in the PMO[40] benchmark.

Qualitative Inspection: Explainability. We first show that Augmented Memory with Beam Enumeration generates molecules that satisfy the MPO objective (Fig. 3). We emphasize that results were not cherry-picked and the three generated examples shown are the top 1 (by reward) across triplicate experiments. All molecules possess better Vina scores and higher QED than the reference molecules, as desired. Fig. 3 shows the highlighted substructures extracted using *Structure* extraction with *Structure Minimum Size* = 15 with three key observations: firstly, "uncommon" molecular substructures may be extracted such as the bridged cycle against DRD2. The exact substructure extracted was an amide bond with a long carbon chain which implicitly enforces the bridged cycle, and the Vina pose shows that it fits in the binding cavity with no clashes, despite being a bulky group. Secondly, bicylic or double-ring systems are often extracted, forming central scaffolds of the full molecule. Thirdly, scaffolds with branch points, i.e., a central ring with single carbon bond extensions, are often extracted. These substructures are particularly interesting as they heavily bias what can be generated in the remaining portion of the full molecule. An exemplary example of this is in the first generated molecule against MK2, where the branch points are effectively a part of

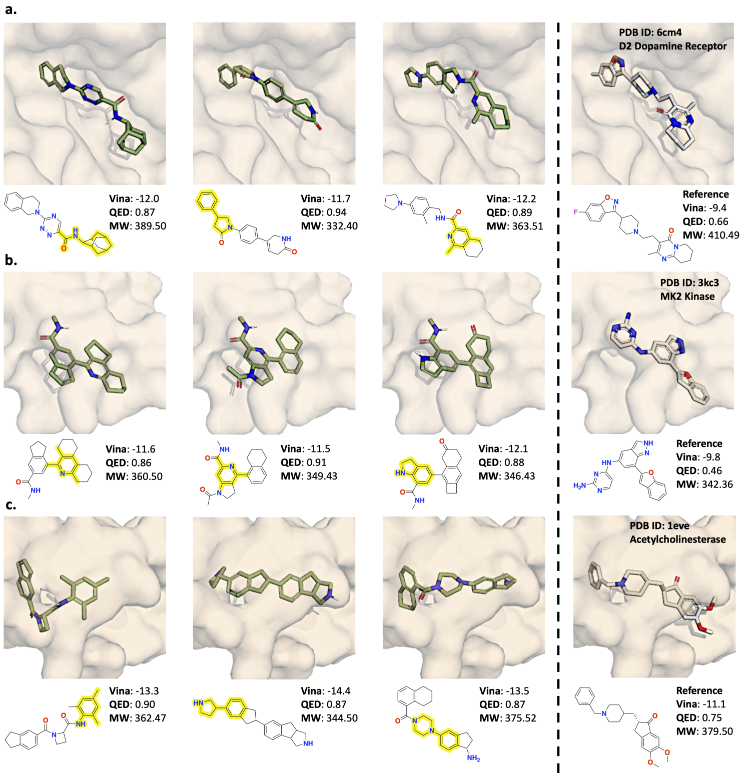


Figure 3: Three drug discovery case studies showing the top generated molecule (triplicate experiments) using Augmented Memory[23] with Beam Enumeration *Structure* Minimum Structure Size = 15 and the reference ligand. Extracted substructures from Beam Enumeration are highlighted. The multi-parameter optimization objective is: Minimize Vina score, maximize QED, and molecular weight < 500 Da. **a.** Dopamine type 2 receptor[57]. **b.** MK2 kinase[58]. **c.** Acetylcholinesterase[59].

two other ring systems (Fig. 3). Beam Enumeration can provide insights into the tolerability and suitability of certain substructures in the context of the full molecules (see Appendix D for more examples of substructures). Overall, the extracted substructures are meaningful and act both as a source of generative explainability and can self-direct the generative model into specific regions of chemical space with high reward.

Quantitative Analysis: Sample Efficiency. Next, we reinforce results from previous work showing that Augmented Memory[23] is significantly more sample efficient than REINVENT[25,64] (Table 2). Notably, the Yield of Augmented Memory is much greater than REINVENT at both the > 0.7 and > 0.8 reward thresholds, indicating that more high reward molecules are generated. Moreover, Augmented Memory has a lower Oracle Burden than REINVENT in all cases, except for Oracle Burden$_{>0.8}$ (1) for DRD2 and AChE where there is essentially no difference. The reason for this is because molecules with > 0.8 reward were already generated at epoch 1, indicating the pre-trained model (trained on ChEMBL[65]) is a good Prior for these case studies. By contrast, the MK2 case study is considerably more challenging as extremely few > 0.8 reward molecules are generated under a 5,000 oracle calls budget. Augmented Memory significantly outperforms REINVENT as the latter could not find 10 molecules with reward > 0.8 (Table 2).

Subsequently, we demonstrate that Beam Enumeration can be applied out-of-the-box on top of Augmented Memory and REINVENT. Firstly, the addition of Beam Enumeration improves the sample efficiency of both base algorithms, as evidenced by the Yield and Oracle Burden metrics in Table 2 with a small trade-off in diversity (Appendix D). The benefits are more pronounced in Augmented Memory as observed by the Yield$_{>0.8}$ improving by > 4x in all cases (MK2 improves by 29x) and the Oracle Burden $_{>0.8}$ (10 and 100) over halved in most cases. Notably, for MK2

Oracle Burden $_{>0.8}$ (100), baseline Augmented Memory could not accomplish the task while Beam Enumeration is successful in almost under 2,000 oracle calls (Table 2). We further verify that a large number of unique scaffolds are generated despite the Beam Enumeration bias (Appendix D), demonstrating that the combined algorithm with Augmented Memory achieves both exploration and exploitation. Overall, the results show that Beam Enumeration is task-agnostic and can be applied on top of existing algorithms to improve sample efficiency. The combined algorithm generates more high reward molecules and faster, even in challenging (MK2) scenarios under a limited oracle budget. Furthermore, in reference to all the Oracle Burden metrics (Table 2), Augmented Memory with Beam Enumeration can identify a small set of *excellent* (high reward) candidate molecules in under 2,000 oracle calls and in some cases, even under 1,000 oracle calls.

Table 2: Drug discovery case studies: Beam Enumeration improves sample efficiency. All experiments were run in triplicate with an oracle budget of 5,000 calls and reported values are the mean and standard deviation. *Scaffold* and *Structure* indicate the type of substructure (*Structure Minimum Size* = 15) extracted. The Generative Yield and Oracle Burden are reported at varying reward thresholds. Parentheses after Oracle Burden denote the cut-off number of molecules. Best performance is bolded with the exception of Oracle Burden (1) (DRD2/AChE) which have essentially identical performance due to the pre-trained model. * and ** denote one and two replicates were unsuccessful, respectively.

Metric	**Target**	**Augmented Memory**			**REINVENT**		
		Beam Structure 15	**Beam Scaffold 15**	**Baseline**	**Beam Structure 15**	**Beam Scaffold 15**	**Baseline**
Generative Yield$_{>0.7}$ (↑)	DRD2	**3474 ± 158**	3412 ± 95	2513 ± 442	2392 ± 699	2686 ± 235	1879 ± 16
	MK2	**3127 ± 138**	2584 ± 443	1446 ± 173	1822 ± 444	1553 ± 391	879 ± 10
	AChE	3824 ± 162	**3902 ± 189**	3288 ± 85	2511 ± 369	2684 ± 242	2437 ± 53
Generative Yield$_{>0.8}$ (↑)	DRD2	**1780 ± 439**	1607 ± 379	363 ± 195	417 ± 275	687 ± 366	102 ± 6
	MK2	**987 ± 211**	523 ± 438	34 ± 13	179 ± 241	19 ± 7	2 ± 0
	AChE	2059 ± 327	**2124 ± 326**	556 ± 47	323 ± 58	310 ± 207	147 ± 11
Oracle Burden$_{>0.8}$ (1) (↓)	DRD2	126 ± 90	83 ± 29	187 ± 51	63 ± 0	127 ± 52	168 ± 149
	MK2	**736 ± 166**	1221 ± 564	1360 ± 543	1110 ± 268	808 ± 524	1724 ± 802
	AChE	105 ± 29	63 ± 0	62 ± 0	62 ± 0	84 ± 29	83 ± 29
Oracle Burden$_{>0.8}$ (10) (↓)	DRD2	582 ± 83	**571 ± 104**	711 ± 120	1099 ± 930	604 ± 71	883 ± 105
	MK2	**1122 ± 154**	2426 ± 1525	3833 ± 394	1778 ± 0**	3891 ± 631	Failed
	AChE	462 ± 25	418 ± 27	**380 ± 0**	441 ± 132	421 ± 120	481 ± 108
Oracle Burden$_{>0.8}$ (100) (↓)	DRD2	**1120 ± 194**	1056 ± 146	2558 ± 30*	1928 ± 117	2109 ± 1090	4595 ± 0**
	MK2	**2189 ± 181**	2676 ± 403	Failed	3208 ± 0**	Failed	Failed
	AChE	1110 ± 265	**884 ± 162**	2021 ± 89	3073 ± 427	3596 ± 678	3931 ± 286

5 Conclusion

In this work, we propose Beam Enumeration to exhaustively enumerate sub-sequences from a language-based molecular generative model based on the top k most probable tokens and for N beam steps. We show that molecular substructures can be extracted from the sub-sequences, which enables self-conditioned generation by only evaluating (by the oracle) molecules possessing these substructures and discarding the rest. We show that Beam Enumeration can be coupled with existing RL-based algorithms including Augmented Memory[23] and REINVENT[25,64]. In three drug discovery case studies involving docking, the addition of Beam Enumeration improves sample efficiency as assessed by the Yield and Oracle Burden metrics with a small trade-off in diversity (which is expected). The extracted substructures themselves provide valuable structural insights, often enforcing the generation of specific cyclic systems and scaffolds with branch points which impose an overall molecular geometry, thus serving as a source of explainability. Beam Enumeration is the first proposed method to jointly address explainability and sample efficiency in molecular generative models. The improvements in the latter will enable more expensive high-fidelity oracles to be explicitly optimized. We note, however, that sparse reward environments[15] remain a difficult optimization task. Finally, Beam Enumeration is a task-agnostic method and can be combined with recent work integrating active learning with molecular generation to further improve sample efficiency[66,67]. If the benefits can be synergistic, we may approach sufficient sample efficiency to directly optimize expensive state-of-the-art (in predictive accuracy) physics-based oracles such as MD simulations[21,22]. Excitingly, this would in turn enhance explainability as high-fidelity oracles are inherently more informative.

The Appendix contains further experiments, ablation studies, experiment hyperparameters, and algorithmic details.

A Beam Enumeration

This section contains full details on Beam Enumeration including hyperparameters, design decisions, and pseudo-code.

A.1 Algorithm Overview

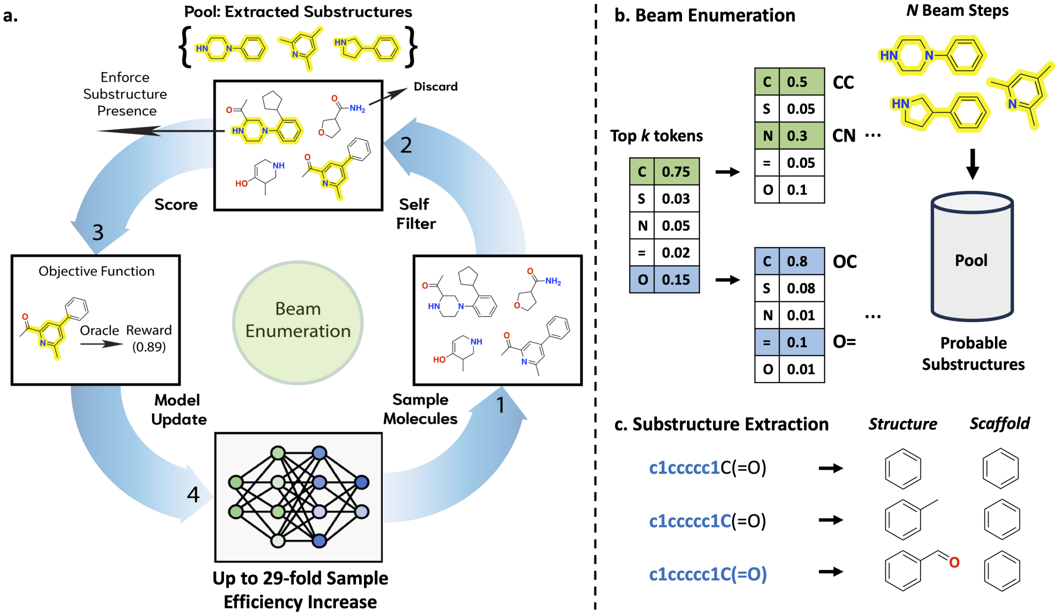


Figure A4: Beam Enumeration overview. **a.** The proposed method proceeds via 4 steps: **1.** generate batch of molecules. **2.** filter molecules based on pool to enforce substructure presence, discarding the rest. **3.** compute reward **4.** update the model. After updating the model, if the reward has improved for consecutive epochs, execute Beam Enumeration. **b.** Beam Enumeration sequentially enumerates the top k tokens by probability for N beam steps, resulting in an exhaustive set of token sub-sequences. **c.** All valid substructures (either by the *Structure* or *Scaffold* criterion) are extracted from the sub-sequences. The most frequent substructures are used for self-conditioned generation. This overview figure is the same as in the main text.

Beam Enumeration (Fig. A4) is an algorithm that extracts molecular substructures from a generative model's weights for self-conditioned generation. The problem set-up is any molecular design task to optimize for a target property profile, e.g., high predicted solubility and binding affinity. When molecular generative models are coupled with an optimization algorithm, it should be increasingly likely to generate desirable molecules, i.e., molecules that possess the target property profile.

Beam Enumeration is proposed based on two facts:

1. On a successful optimization trajectory, the model's weights must change such that desirable molecules are more likely to be generated, on average.
2. The act of generating molecules in an autoregressive manner involves sequentially sampling from conditional probability distributions.

In this work, Beam Enumeration is applied to a language-based autoregressive generative model operating on the simplified molecular-input line-entry system (SMILES)[24] representation. The optimization algorithm is Augmented Memory[23] which builds on REINVENT[25,51] and casts the optimization process as an on-policy reinforcement learning (RL) problem. Following RL terminology, sampling from the generative model involves sampling *trajectories*, which in this case, are SMILES, and the desirability of the corresponding molecule is given by the *reward*.

The underlying hypothesis of Beam Enumeration is that during the RL optimization process, *partial trajectories* provide a source of signal that can be exploited. Usually, full trajectories are sampled which map to a complete SMILES sequence that can be translated to a molecule. Our assumption is that partial trajectories (partial SMILES sequence) can be mapped to molecular substructures (a part of the full molecule). This statement is not guaranteed as SMILES and molecules are discrete and small perturbations often leads to invalid SMILES. We prove this assumption in Section C by showing that although the vast number of partial trajectories do not map to valid SMILES, the raw number is sufficient to extract a meaningful signal. Correspondingly, Beam Enumeration leverages partial trajectories on the assumption that molecular substructures are *on track* to becoming full molecules that *would* receive high reward.

A.2 Enumerating Partial Trajectories

In order to extract molecular substructures, a set of partial trajectories must be sampled from the generative model. Recalling the fact that on a successful optimization trajectory, it must become increasingly likely to generate desirable molecules, partial trajectories are sampled by enumerating the top k tokens, based on the conditional probability. Therefore, the process of enumerating partial trajectories involves sequentially extending each token sequence by their next top k probable tokens, resulting in the total number of partial trajectories as 2^N where N is the number of beam steps, i.e., how many tokens in the partial trajectory. We note that taking the top k most probable tokens does not guarantee that the partial trajectories are indeed the most probable, as paying a probability penalty early can lead to higher probabilities later. However, our assumption is that on average, this leads to a set of partial trajectories that are at the very least, amongst the most probable. Moreover, there is a practical limit to how many partial trajectories are sampled due to exponential growth which makes scaling quickly computationally prohibitive. In the later section, we discuss this thoroughly. Finally, from here, partial trajectories will be referred to as token sub-sequences.

A.3 Extracting Molecular Substructures

Given a set of token sub-sequences, the goal is to extract out the most frequent molecular substructures. This is done by taking each sequence, considering every (sub)-sub-sequence, and counting the number of valid substructures (Fig. A). For example, given the sub-sequence “ABCD”, the set of (sub)-sub-sequences are: “A”, “AB”, “ABC”, and “ABCD”. In practice, we only consider (sub)-sub-sequences with at least three characters (“ABC” and “ABCD”) since each character loosely maps to one atom and three is approximately the minimum for meaningful functional groups, e.g., “C=O”, a carbonyl. The set of most frequent substructures is assumed to be on track to receive a high reward.

A.4 Defining Molecular Substructures: *Scaffold* vs. *Structure*

As shown in Fig. A, molecular substructures can be defined on the *Scaffold* or *Structure* level. The former extracts the Bemis-Murcko[54] scaffold while the latter extracts *any* valid structure. The *any* valid structure is an important distinction as our experiments find that extracting by *Structure* leads to the most frequent molecular substructures being small functional groups that do not have corresponding scaffolds. By contrast, extracting the scaffold always leads to ring structures. Moreover, extracting specifically the Bemis-Murcko scaffold is important as heavy atoms, e.g., nitrogen, are important for biological activity. Consequently, extracted substructures are also enforced to contain at least one heavy atom as we find that benzene, perhaps unsurprisingly, is commonly the most frequent substructure. See Section B for more details on the differing behavior of ‘Scaffold’ vs. ‘Structure’.

A.5 Self-conditioned Generation

Self-conditioned generation is achieved by filtering sampled batches of molecules from the generative model to only keep the ones that possess at least one of the most frequent substructures. The effect is that the generative process is self-biased to focus on a narrower chemical space which we show can drastically improve sample efficiency at the expense of some diversity, which is acceptable when expensive high-fidelity oracles are used: we want to identify a small set of *excellent* candidate molecules under minimal oracle calls.

A.6 Probabilistic Explainability

The set of most frequent molecular substructures should be meaningful as otherwise, the model's weights would not have been updated such that these substructures have become increasingly likely to be generated. We verify this statement in the illustrative experiment in the main text and in Section C. In the drug discovery case studies (Appendix D), the extracted substructures are more subtle in why they satisfy the target objective but certainly must possess meaning, however subtle, as otherwise, they would not receive a high reward. In the main text, we show that extracted substructures form core scaffolds and structural motifs in the generated molecules that complement the protein binding cavity. Finally, we emphasize that the *correctness* and *usefulness* of this explainability deeply depends on the oracle(s) being optimized for. The extracted substructures do not explain why the generated molecules satisfy the target objective. Rather, they explain why the generated molecules satisfy the oracle. The assumption in a generative design task is that optimizing the oracle is a good proxy for the target objective, e.g., generating molecules that dock well increases the likelihood of the molecules being true binders. This observation directly provides additional commentary on why sample efficiency is so important: the ability to directly optimize expensive high-fidelity oracles would inherently enhance the *correctness* of the extracted substructures.

B Beam Enumeration: Findings from Hyperparameter Screening

In this section, we introduce all seven hyperparameters of Beam Enumeration and then present results on an exhaustive hyperparameter search which elucidates the behavior and interactions of all the hyperparameters. In the end, we present our analyses and provide hyperparameter recommendations for Beam Enumeration which can serve as default values to promote out-of-the-box application.

B.1 Beam Enumeration Hyperparameters

Beam k. This hyperparameter denotes how many tokens to enumerate at each step. Given that our hypothesis is that the most probable sub-sequences yield meaningful substructures, we fix Beam k to 2. A larger value would also decrease the number of Beam Steps possible as the total number of sub-sequences is k^N and the exponential growth quickly leads to computational infeasibility.

Beam Steps N. This hyperparameter denotes how many token enumeration steps to execute and is the final token length of the enumerate sub-sequences. This parameter leads to exponential growth in the number of sequences which can quickly become computationally prohibitive. An important implication of this hyperparameter is that larger Beam Steps means that larger substructures *can* be extracted. In our experiments, we find that enforcing size in the extracted substructures can drastically improve sample efficiency with decreased diversity as the trade-off. We thoroughly discuss this in a later sub-section. Finally, in our experiments, the upper-limit investigated is 18 Beam Steps.

Substructure Type. This hyperparameter has two possible values: *Scaffold* or *Structure*. *Scaffold* extracts Bemis-Murcko[54] scaffolds while *Structure* extracts *any* valid substructure.

Structure Structure Minimum Size. This hyperparameter enforces the partial SMILES to contain at least a certain number of characters. In effect, this enforces extracted molecular substructures to be larger than a Structure Minimum Size. From the illustrative experiment in the main text and Section C, *Structure* extraction often leads to small functional groups being the most frequent in the sub-sequences. By enforcing a minimum structure size, *Structure* extraction leads to partial structures which may carry more meaning. We find that this hyperparameter greatly impacts sample efficiency and we present all our findings in a later sub-section.

Pool Size. This hyperparameter controls how many molecular substructures to keep track of. These *pooled* substructures are what is used to perform self-conditioning. The hypothesis is that the most frequent ones carry the most meaning and thus, a very large pool size may not be desired.

Patience. This hyperparameter controls how many successive reward improvements are required before Beam Enumeration executes and molecular substructures are extracted. Recalling the first fact in which Beam Enumeration was proposed on: On a successful optimization trajectory, the model's weights must change such that desirable molecules are more likely to be generated, on average. Patience is effectively an answer to "when would extracted substructures be meaningful?" Too low a patience and stochasticity can lead to negative effects while too high a patience diminishes the benefits of Beam Enumeration on sample efficiency.

Token Sampling Method. This hyperparameter has two possible values: "topk" or "sample" and denotes *how* tokens sub-sequences are enumerated. "topk" takes the top k most probable tokens at each Beam Step while "sample" samples from the distribution just like during batch generation. Our results show interesting observations surrounding this hyperparameter as "sample" can work just as well and *sometimes* even better than taking the "topk". These results were unexpected as the underlying hypothesis is that the most probable sub-sequences lead to the most useful substructures being extracted. However, our findings are not in contradiction as sampling the conditional probability distributions would still lead to sampling the top k tokens, on average. Moreover, after extensive experiments, we find that "sample" leads to more variance in performance across replicates which is in agreement with the assumption that sampling the distributions can lead to more improbable structures. We thoroughly discuss our findings in a later sub-section where we provide hyperparameter recommendations and analyses to the effects of tuning each hyperparameter.

B.2 Hyperparameters: Grid Search

We performed two exhaustive hyperparameter grid searches on the illustrative experiment which has the following multi-parameter optimization (MPO) objective: maximize topological polar surface area (tPSA), molecular weight < 350 Da, number of rings ≥ 2 with an oracle budget of 5,000. The first grid search investigated the following hyperparameter combinations:

- Beam K = 2
- Beam Steps = [15, 16, 17, 18]
- Substructure Type = [*Scaffold*, *Structure*]
- Pool Size = [3, 4, 5]
- Patience = [3, 4, 5]
- Token Sampling Method = ['topk', 'sample']

All hyperparameter combinations (144) were tried and run for 10 replicates each for statistical reproducibility, total of 1,440 experiments. Next, an additional grid search was performed with the following hyperparameter combinations:

- Beam K = 2
- Beam Steps = [17, 18]
- Substructure Type = [*Scaffold*, *Structure*]
- Structure Structure Minimum Size = [10, 15]
- Pool Size = [4, 5]
- Patience = [4, 5]

- Token Sampling Method = ['topk', 'sample']

We take the general trends from the first grid search and narrow down the most optimal hyperparameters to further investigate Substructure Type and structure Structure Minimum Size. **As from before, all hyperparameter combinations (64) were tried and run for 10 replicates each for statistical reproducibility, total of 640 experiments.**

The following heatmaps performance by the Generative Yield and Oracle Burden (10) metrics at the > 0.8 reward threshold and under a 5,000 oracle budget. The Generative Yield measures how many unique molecules above 0.8 reward were generated. The Oracle Burden (10) measures how few oracle calls were required to generate 10 molecules above 0.8 reward. We note that all Oracle Burden metrics are computed by not allowing more than 10 molecules to possess the same Bemis-Murcko[54] scaffold, thus also explicitly considering diversity in the generated set.

B.3 Analysis of Grid Search Results

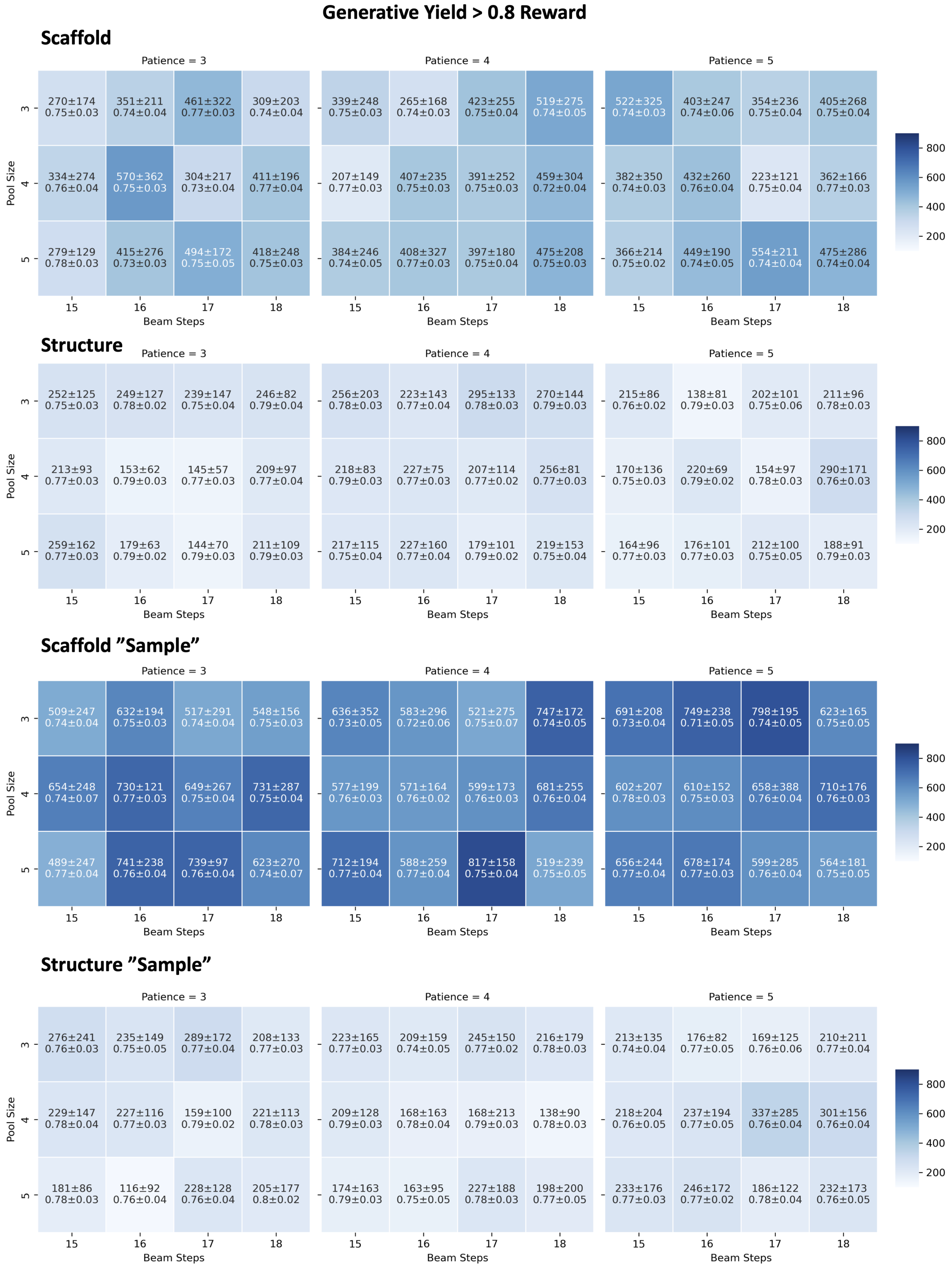
In this section, we summarize our analysis on the grid search experiments. Unless stated, each bullet point means the observation was observed for both Generative Yield and Oracle Burden (10). For example the point: *Scaffold* > *Structure* means *Scaffold* is generally more performant than *Structure* across all hyperparameters on both the Generative Yield and Oracle Burden (10).

- For *Scaffold*, higher Pool, higher Patience, and higher Beam Steps improves performance
- For *Structure*, lower pool and lower patience improves performance
- *Scaffold* > *Structure*
- *Scaffold* and *Structure* become more performant with increasing Structure Minimum Size
- *Scaffold* and *Structure* with Structure Minimum Size: "sample" sampling *can* be better than "topk" sampling but with more variance

Based on the above analysis, we propose the optimal hyperparameters for the illustrative experiment as:

- *Scaffold*
- "topk" sampling ("sample" sampling can be more performant but exhibits higher variance)
- Patience = 5
- Pool Size = 4
- Beam Steps = 18

Finally, we provide more commentary on interesting observations from the grid search results. *Structure* without Structure Minimum Size enforcing often leads to small functional groups being the most frequent molecular substructures extracted with Beam Enumeration. Enforcing Structure Minimum Size puts it almost on par with *Scaffold*, suggesting (perhaps not surprisingly) that larger substructures can carry more meaningful information. Moreover, when using "sample" sampling, the generative model undergoes more "filter rounds". Specifically, at each epoch, the sampled batch is filtered to contain the extracted substructures. When using "sample" sampling, the model is more prone to some epochs containing no molecules with the substructures. In practice, this is inconsequential as sampling is computationally inexpensive and a next batch of molecules can easily be sampled. However, specifically in the *Structure* with "sample" sampling and Structure Minimum Size = 15 experiment, "filter round" can be quite extensive, taking up to 100,000 epochs (maximum observed) for an oracle budget of 5,000 (adding about an hour to the wall time which is minor when the oracle is expensive). This means that many epochs contained molecules without the extracted substructures. There are two observations here: firstly, "sample" sampling can lead to more improbable substructures which are hence less likely to be sampled and secondly, *Structure* with Structure Minimum Size enforcement leads to extreme biasing (which improves sample efficiency).



Scaffold

	Patience = 3				Patience = 4				Patience = 5			
Pool Size	15	16	17	18	15	16	17	18	15	16	17	18
3	270±174 0.75±0.03	351±211 0.74±0.04	461±322 0.77±0.03	309±203 0.74±0.04	339±248 0.75±0.03	265±168 0.74±0.03	423±255 0.75±0.04	519±275 0.74±0.05	522±325 0.74±0.03	403±247 0.74±0.06	354±236 0.75±0.04	405±268 0.75±0.04
4	334±274 0.76±0.04	570±362 0.75±0.03	304±217 0.73±0.04	411±196 0.77±0.04	207±149 0.77±0.03	407±235 0.75±0.03	391±252 0.75±0.03	459±304 0.72±0.04	382±350 0.74±0.03	432±260 0.76±0.04	223±121 0.75±0.04	362±166 0.77±0.03
5	279±129 0.78±0.03	415±276 0.73±0.03	494±172 0.75±0.05	418±248 0.75±0.03	384±246 0.74±0.05	408±327 0.77±0.03	397±180 0.75±0.04	475±208 0.75±0.03	366±214 0.75±0.02	449±190 0.74±0.05	554±211 0.74±0.04	475±286 0.74±0.04

Structure

	Patience = 3				Patience = 4				Patience = 5			
Pool Size	15	16	17	18	15	16	17	18	15	16	17	18
3	252±125 0.75±0.03	249±127 0.78±0.02	239±147 0.75±0.04	246±82 0.79±0.04	256±203 0.78±0.03	223±143 0.77±0.04	295±133 0.78±0.03	270±144 0.79±0.03	215±86 0.76±0.02	138±81 0.79±0.03	202±101 0.75±0.06	211±96 0.78±0.03
4	213±93 0.77±0.03	153±62 0.79±0.03	145±57 0.77±0.03	209±97 0.77±0.04	218±83 0.79±0.03	227±75 0.77±0.03	207±114 0.77±0.02	256±81 0.77±0.03	170±136 0.75±0.03	220±69 0.79±0.02	154±97 0.78±0.03	290±171 0.76±0.03
5	259±162 0.77±0.03	179±63 0.79±0.02	144±70 0.79±0.03	211±109 0.79±0.03	217±115 0.75±0.04	227±160 0.77±0.04	179±101 0.79±0.02	219±153 0.75±0.04	164±96 0.77±0.03	176±101 0.77±0.03	212±100 0.75±0.05	188±91 0.79±0.03

Scaffold ”Sample”

	Patience = 3				Patience = 4				Patience = 5			
Pool Size	15	16	17	18	15	16	17	18	15	16	17	18
3	509±247 0.74±0.04	632±194 0.75±0.03	517±291 0.74±0.04	548±156 0.75±0.03	636±352 0.73±0.05	583±296 0.72±0.06	521±275 0.75±0.07	747±172 0.74±0.05	691±208 0.73±0.04	749±238 0.71±0.05	798±195 0.74±0.05	623±165 0.75±0.05
4	654±248 0.74±0.07	730±121 0.77±0.03	649±267 0.75±0.04	731±287 0.75±0.04	577±199 0.76±0.03	571±164 0.76±0.02	599±173 0.76±0.03	681±255 0.76±0.04	602±207 0.78±0.03	610±152 0.75±0.03	658±388 0.76±0.04	710±176 0.76±0.03
5	489±247 0.77±0.04	741±238 0.76±0.04	739±97 0.76±0.04	623±270 0.74±0.07	712±194 0.77±0.04	588±259 0.77±0.04	817±158 0.75±0.04	519±239 0.75±0.05	656±244 0.77±0.04	678±174 0.77±0.03	599±285 0.76±0.04	564±181 0.75±0.05

Structure ”Sample”

	Patience = 3				Patience = 4				Patience = 5			
Pool Size	15	16	17	18	15	16	17	18	15	16	17	18
3	276±241 0.76±0.03	235±149 0.75±0.05	289±172 0.77±0.04	208±133 0.77±0.03	223±165 0.77±0.03	209±159 0.74±0.05	245±150 0.77±0.02	216±179 0.78±0.03	213±135 0.74±0.04	176±82 0.77±0.05	169±125 0.76±0.06	210±211 0.77±0.04
4	229±147 0.78±0.04	227±116 0.77±0.03	159±100 0.79±0.02	221±113 0.78±0.03	209±128 0.79±0.03	168±163 0.78±0.04	168±213 0.79±0.03	138±90 0.78±0.03	218±204 0.76±0.05	237±194 0.77±0.05	337±285 0.76±0.04	301±156 0.76±0.04
5	181±86 0.78±0.03	116±92 0.76±0.04	228±128 0.76±0.04	205±177 0.8±0.02	174±163 0.79±0.03	163±95 0.75±0.05	227±188 0.78±0.03	198±200 0.77±0.05	233±176 0.77±0.03	246±172 0.77±0.02	186±122 0.78±0.04	232±173 0.76±0.05

Figure B5: illustrative experiment Generative Yield > 0.8. The IntDiv1 [68] is annotated.

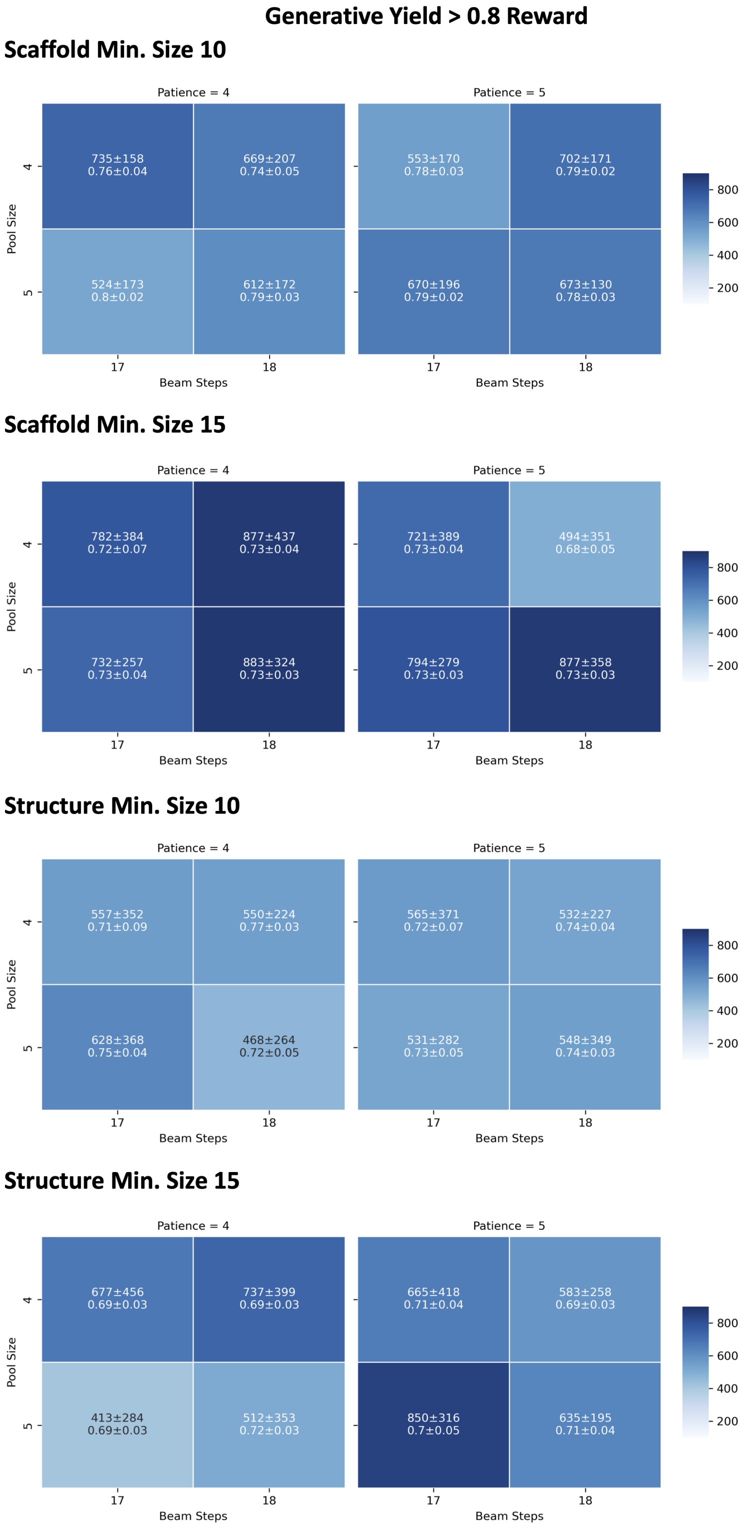
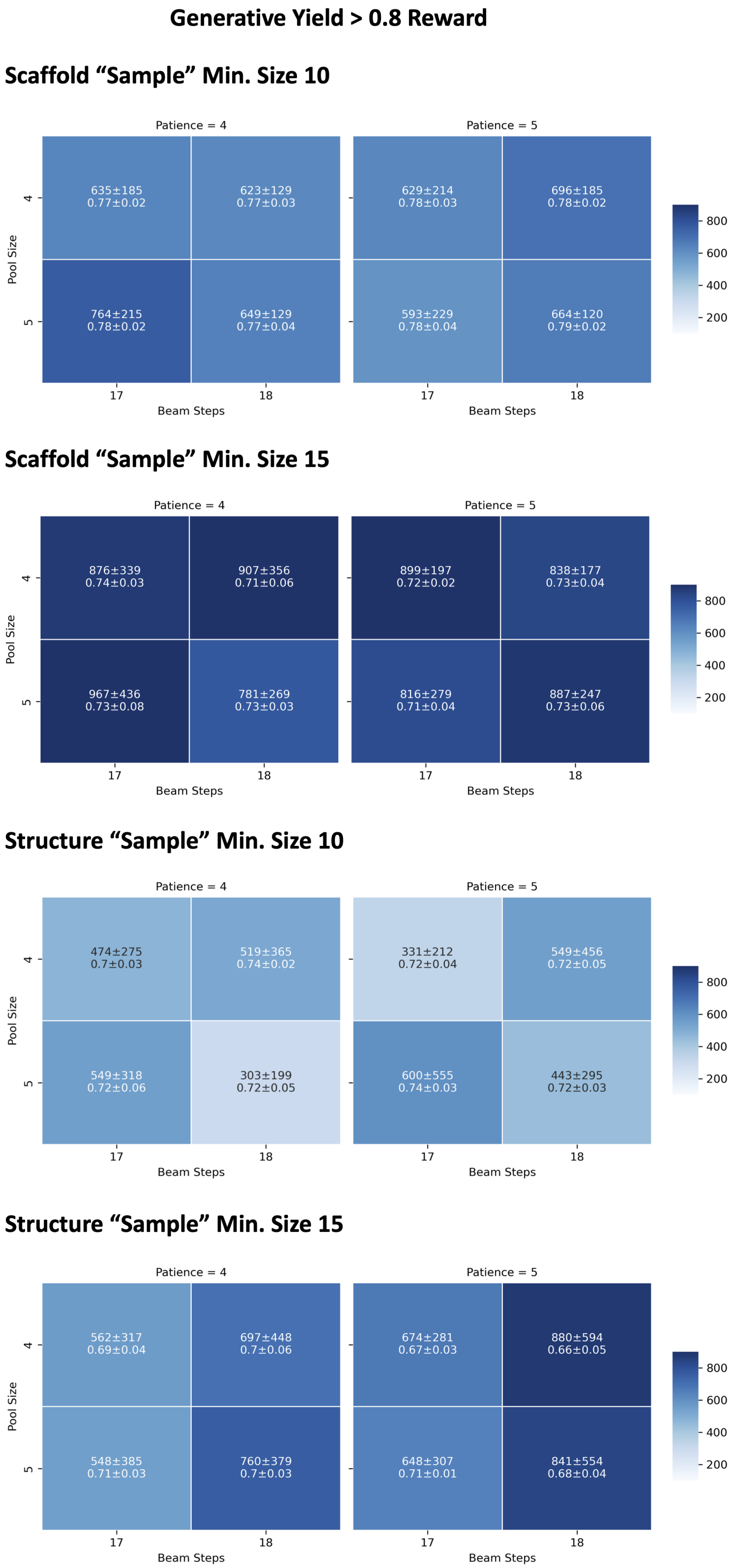


Figure B6: illustrative experiment Generative Yield > 0.8 with Structure Minimum Size. The IntDiv1[68] is annotated.



Generative Yield > 0.8 Reward

Scaffold “Sample” Min. Size 10

Scaffold “Sample” Min. Size 15

Structure “Sample” Min. Size 10

Structure “Sample” Min. Size 15

Figure B7: illustrative experiment Generative Yield > 0.8 with Structure Minimum Size and "Sample" token sampling. The IntDiv1[68] is annotated.

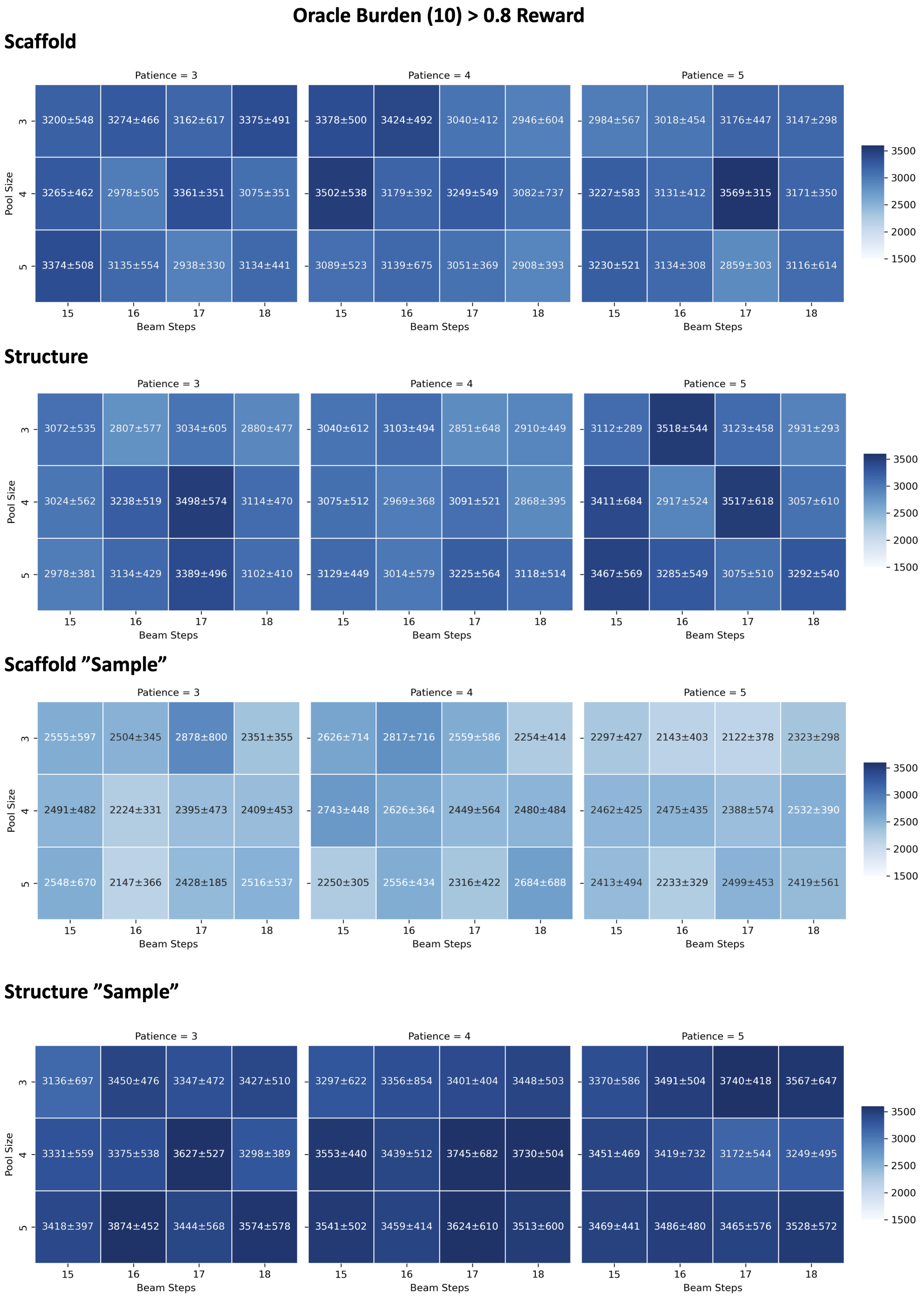


Figure B8: illustrative experiment Oracle Burden (10) > 0.8

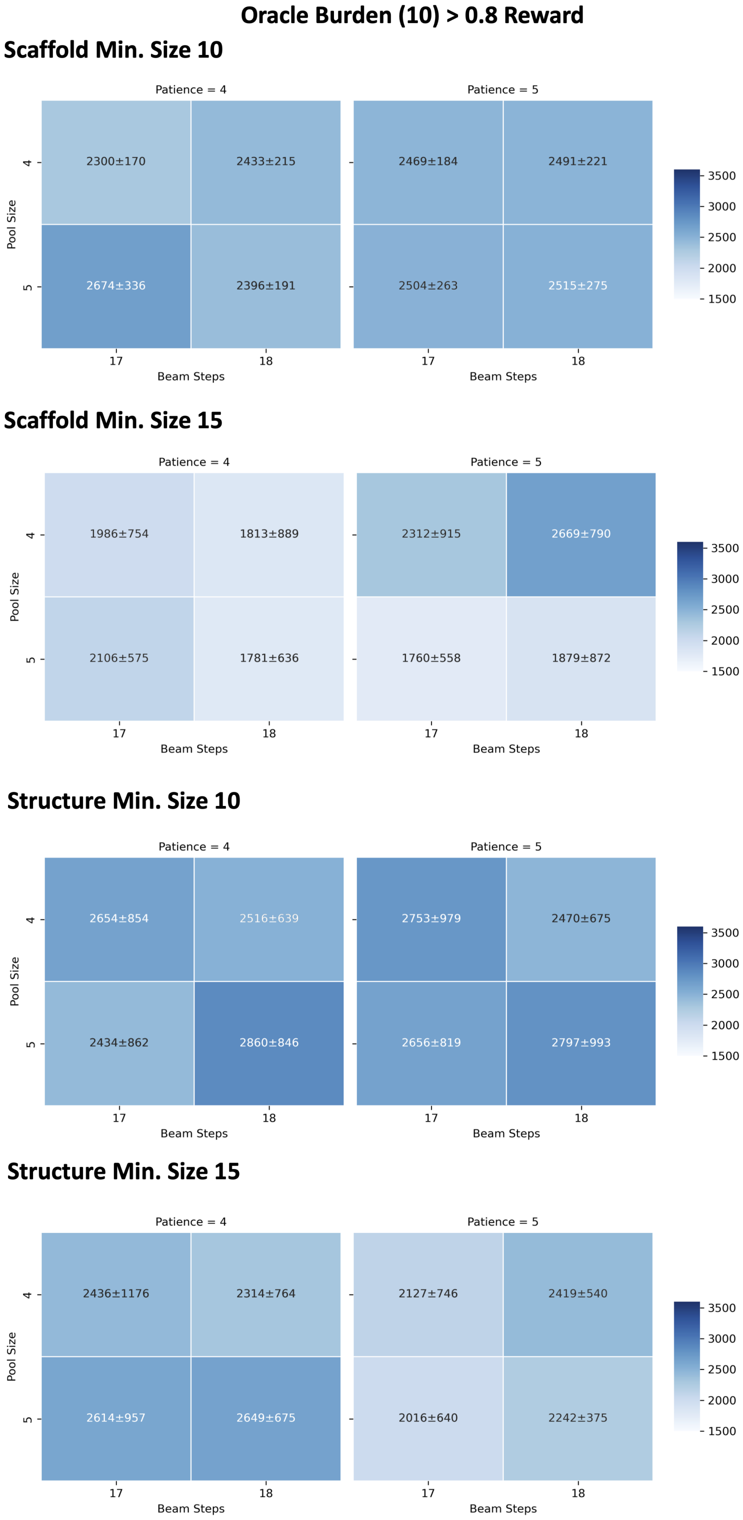


Figure B9: illustrative experiment Oracle Burden (10) > 0.8 with Structure Minimum Size

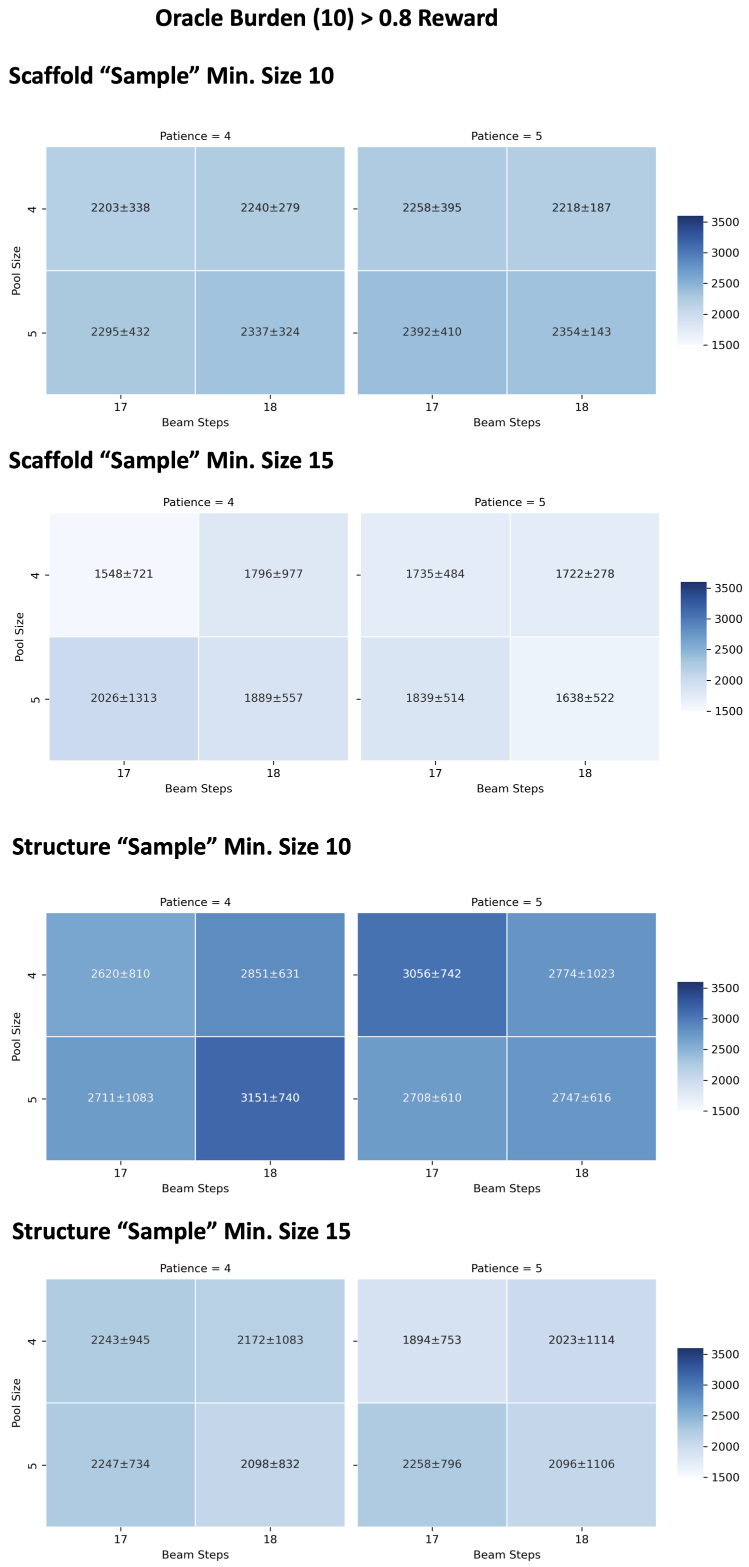


Figure B10: illustrative experiment Oracle Burden (10) > 0.8 with Structure Minimum Size and "Sample" token sampling

We believe the remarkable tolerability of the generative model sampling to such bias is an interesting observation. By contrast, *Scaffold* with Structure Minimum Size enforcement is not as prone to "filter rounds" because *Scaffold* "truncates" the substructure to its central shape (scaffold). For example, toluene (benzene with a methyl group) has a Bemis-Murcko[54] scaffold of just benzene. The consequence is that *Structure* leads to more extreme biasing (it is more likely for a molecule to contain benzene than specifically toluene) which is in agreement with the general observation that the diversity of the generated set decreases when using *Structure*. Overall, both *Scaffold* and *Structure* with Structure Minimum Size enforcing exhibits the best performance and "sample" sampling *can* be more performant than "topk" sampling but exhibits notably higher variance.

The set of optimal hyperparameters found here were used in drug discovery case studies. In order to be rigorous with our investigation, we only fix the following hyperparameters:

- Patience = 5 (lower variance)
- Pool Size = 4 (lower variance, higher Yield, lower Oracle Burden)
- Beam Steps = 18 (lower variance, higher Yield, lower Oracle Burden)

with these hyperparameters, we do a small grid search (on the drug discovery case studies) by changing the Structure Type, Token Sampling Method, and Structure Minimum Size hyperparameters as the optimal hyperparameters in the illustrative experiment are not necessarily the optimal ones in the drug discovery experiments. **The purpose of this is not to necessarily report the best performance on the drug discovery case studies but to gain insights into the optimal general parameters such that Beam Enumeration can be used out-of-the-box. In real-world expensive oracle settings, tuning hyperparameters is infeasible.**

All results from the drug discovery case studies are shown in Section D.

B.4 Beam Enumeration: Recommended default Hyperparameters

Taking into consideration all grid search experiments for the illustrative experiment and Drug Discovery case studies, the following optimal hyperparameters are recommended: Patience = 5, Pool Size = 4, Beam Steps = 18, *Structure*, Structure Minimum Size = 15, "topk" sampling.

Notable differences between the final recommended hyperparameters compared to those found from the illustrative experiment is that *Structure* and "topk" sampling are more performant than *Scaffold* and "sample" sampling. In the illustrative experiment, "sample" sampling was sometimes more performant than "topk" sampling. We rationalize these observations as follows: in MPO objectives that include physics-based oracles, structure specificity becomes increasingly important, e.g., specific chemical motifs dock well because they form interactions with the protein. Therefore, "topk" sampling is more robust as there is less variance in the extract substructures compared to "sample" sampling. We empirically observe the increased variance when using "sample" sampling measured by the standard deviation between replicate experiments (Appendix D). In the illustrative experiment where the oracle was more permissive, i.e., *any* rings saturated with heteroatoms would satisfy the MPO objective, small deviations in the extracted structure do not have as prominent an effect as physics-based oracles which require specificity. Another observation is that *Structure* sampling often extracts scaffolds with "branch points" which enforces extreme bias that can lead to more focused chemical space exploration. We discuss this in detail in Section D and believe the insights are generally interesting in the context of molecular optimization landscape.

Finally, we end this section by stating that we cannot try every single hyperparameter combination and the recommended values are from our grid search results which we make an effort to be robust, given that we perform 10 replicates of each experiment. We find that the optimal hyperparameters in the drug discovery case studies are generally the same as in the illustrative experiment.

B.5 Pseudo-code

The pseudo-code for Beam Enumeration is presented here. The $\oplus$ operator denotes every element on the left is being extended by every element on the right.

```
Algorithm 1: Beam Enumeration
Input: Generative Agent $\pi_{\theta_{\text{Agent}}}$, Top $k$, $N$ Beam Steps
Output: Enumerated Token Sub-sequences $S$
Initialization:
Hidden State = None;
Sub-sequences = [Top $k$ <START> Tokens];
Input Vector = top $k$ number of start tokens;
for $i = 1$ to $N$ do
    Logits, New Hidden State ← $\pi_{\theta_{\text{Agent}}}$(Input Vector, Hidden State);
    Tokens$_K$ ← top $k$ tokens from Softmax(Logits);
    if $i = 1$ then
        Sub-sequences ← Tokens$_K$;
        Input Vector ← Tokens$_K$;
        Hidden State = New Hidden State;
    else
        Create empty list $temp$;
        for each seq in Sub-sequences do
            $seq$ ← $seq \oplus$ Tokens$_K$;
            Append $seq$ to $temp$;
        Sub-sequences ← $temp$;
        Clear $temp$;
        Input Vector ← Flatten Tokens$_K$;
        Hidden State ← $(\text{New Hidden State}[i].\text{repeat\_interleave}(\text{top } k, \text{dim} = 1))_{i=0,1}$;
return Sub-sequences
```

C Illustrative Experiment

This section contains additional results from initial investigations into the feasibility of Beam Enumeration. The illustrative experiment was performed with the following multi-parameter optimization (MPO) objective: maximize topological polar surface area (tPSA), molecular weight (MW) < 350 Da, number of rings $\geq$ 2.

C.1 Substructure Extraction

The first experiments investigated whether a sufficient substructures signal could be extracted from enumerated sub-sequences. The two parameters of Beam Enumeration (without self-conditioning) are top k denoting the top k number of highest probability tokens to enumerate and N number of beam steps denoting how many steps to perform token expansion for (which is also the length of the final sub-sequence). Our hypothesis is that a lower top k is desirable as we are interested in the most probably substructures. Thus, the initial experiments were a grid-search with a top k of 2 and N beam steps of [15, 16, 17, 18]. The illustrative experiment was run for 100 epochs (6,400 oracle calls which is different from the 5,000 used in the main text experiments as this set of results is only to demonstrate that meaningful substructures can be extracted) and Beam Enumeration was applied at epochs 1, 20, 40, 60, 80, and 100.

Table 3 shows the absolute counts and percentage of sub-sequences containing valid substructures. While the percentage may appear low, we note the absolute counts is more than enough to extract some notion of most probable substructures. We use N beam steps of 18 for all experiments as we hypothesize that larger substructures can carry more information. The reason the max beam steps investigated was 18 is because of the memory overhead required for sequence expansion.

Table 3: Feasibility of Beam Enumeration to extract valid substructures. Top-k = 2.

N **Beam Steps**	**Epoch 1**	**Epoch 20**	**Epoch 40**	**Epoch 60**	**Epoch 80**	**Epoch 100**
15	2294/32768 (7.00%)	3123/32768 (9.53%)	5843/32768 (17.83%)	5538/32768 (16.90%)	5674/32768 (17.32%)	8004/32768 (24.43%)
16	4789/65536 (7.31%)	5890/65536 (8.99%)	5771/65536 (8.81%)	11159/65536 (17.03%)	7657/65536 (11.68%)	9771/65536 (14.91%)
17	9998/131072 (7.63%)	15266/131072 (11.65%)	26163/131072 (19.96%)	24352/131072 (18.58%)	21442/131072 (16.36%)	31160/131072 (23.77%)
18	20747/262144 (7.91%)	33969/262144 (12.96%)	72126/262144 (27.51%)	48417/262144 (18.47%)	45349/262144 (17.30%)	46994/262144 (17.93%)

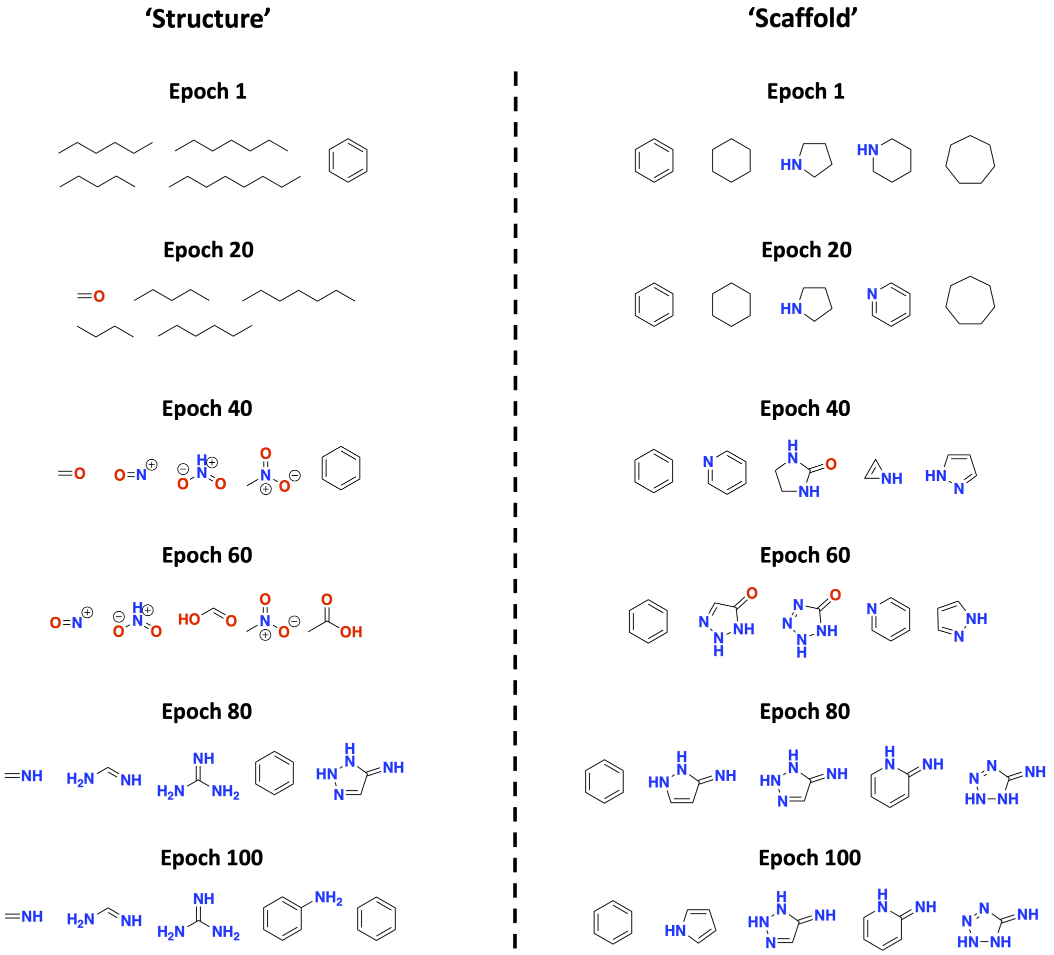


Figure C11: Substructures extracted in the illustrative example at varying epochs based on *Structure* and *Scaffold*.

C.2 Extracted Substructures

To illustrate the capability of Beam Enumeration to extract meaningful substructures, Fig. C11 shows the top 5 most probable substructures at epochs 1, 20, 40, 60, 80, and 100 based on *Structure* (extract any valid structure) and *Scaffold* (extract valid Bemis-Murcko[54] scaffold) using a top k of 2 and 18 beam steps. We make two crucial observations here. Firstly, *Structure* often extracts small functional groups which makes the self-conditioned filtering much more permissive as it is more likely for a molecule to possess a specific functional group than a specific scaffold. Secondly, benzene appears often and perhaps unsurprisingly as it is ubiquitous in nature. Based on these observations, we design

Beam Enumeration to only extract substructures containing at least one heteroatom on the assumption that heteroatoms are much more informative in forming polar interactions in drug molecules, e.g., a hydrogen-bond cannot form from benzene. Finally, the general observation is that the most probable substructures gradually contain more heteroatoms, as desired.

C.3 Supplementary Main Text Results

In this section, we present the same table as the main text illustrative experiment. The only difference is that the IntDiv1[69] is also annotated in the table here to show that the sample efficiency improvements of Beam Enumeration come only at a small trade-off in diversity (Table 4). In agreement with our observations in the hyperparameters grid search (Appendix A), *Structure* extraction with 'Structure Minimum Size' enforcement leads to highly specific substructures which decrease diversity relative to *Scaffold* extraction but with potential gains in sample efficiency as evidenced in the drug discovery case studies (Appendix D). We further perform statistical testing using Welch's t-test to compare all metrics for *Scaffold* with 'Structure Minimum Size' = 15 and Baseline Augmented Memory[23]. For the experiments that had unsuccessful replicates, we use the total number of successful experiments, e.g., Oracle Burden$_{>10}$ (100), the Baseline was unsuccessful in 69/100 replicates so a 31 sample size was used. Overall, all p-values are significant at the 95% confidence level.

Table 4: Illustrative experiment: Beam Enumeration improves the sample efficiency of Augmented Memory. All experiments were run for 100 replicates with an oracle budget of 5,000 calls and reported values are the mean and standard deviation. *Scaffold* and *Structure* indicate the type of substructure and the number after is the 'Structure Minimum Size'. Parentheses after Oracle Burden denote the cut-off number of molecules. Parentheses after values represent the number of unsuccessful replicates (for achieving the metric). The IntDiv1[68] is annotated under each Generative Yield. Welch's t-test is used to compare the difference between *Scaffold* with 'Structure Minimum Size' = 15 and Baseline Augmented Memory[23]. All p-values are significant.

Metric	**Augmented Memory**					**Welch's t-test (95%)**
	Beam Scaffold 15	**Beam Structure 15**	**Beam Scaffold**	**Beam Structure**	**Baseline**	**p-value (N=100)**
Generative Yield$_{>0.7}$ (↑)	**1757 ± 305**	1669 ± 389	1117 ± 278	864 ± 202	496 ± 108	2.60×10^{-75}
- Diversity	0.77 ± 0.03	0.73 ± 0.04	0.79 ± 0.03	0.83 ± 0.03	0.85 ± 0.02	
Generative Yield$_{>0.8}$ (↑)	**819 ± 291**	700 ± 389	425 ± 256	199 ± 122	85 ± 56	3.70×10^{-48}
- Diversity	0.73 ± 0.04	0.69 ± 0.05	0.75 ± 0.04	0.77 ± 0.04	0.78 ± 0.03	
Oracle Burden$_{>0.7}$ (1) (↓)	**577 ± 310**	616 ± 230	1037 ± 414	897 ± 347	1085 ± 483	3.06×10^{-19}
Oracle Burden$_{>0.7}$ (10) (↓)	947 ± 350	**926 ± 332**	1881 ± 259	1745 ± 292	2392 ± 216	4.99×10^{-87}
Oracle Burden$_{>0.7}$ (100) (↓)	**1530 ± 468**	1547 ± 513	2736 ± 335	2713 ± 402	3672 ± 197	2.34×10^{-86}
Oracle Burden$_{>0.8}$ (1) (↓)	**1311 ± 628**	1401 ± 695	2423 ± 487	2295 ± 482	3164 ± 492	6.07×10^{-65}
Oracle Burden$_{>0.8}$ (10) (↓)	**1794 ± 617 (1)**	2009 ± 804 (1)	3124 ± 497	3241 ± 492	4146 ± 326	6.48×10^{-79}
Oracle Burden$_{>0.8}$ (100) (↓)	**2704 ± 689 (1)**	2943 ± 811 (6)	3973 ± 592 (6)	4415 ± 437 (20)	4827 ± 170 (69)	6.17×10^{-21}

C.4 Beam Enumeration works in Exploitation Scenarios

In the main text illustrative experiment, Augmented Memory[23] was used with Selective Memory Purge activated which is the mechanism to promote chemical space exploration, as described in the original work. For completeness, we show that Beam Enumeration also works in pure exploitation scenarios where the goal is only to generate high reward molecules even if the same molecule is repeatedly sampled (Table 5). We perform statistical testing using Welch's t-test to compare all metrics for *Scaffold* with 'Structure Minimum Size' = 15 and Baseline Augmented Memory[23]. For the experiments that had unsuccessful replicates, we use the total number of successful experiments, e.g., Oracle Burden$_{>10}$ (100), the Baseline was unsuccessful in 69/100 replicates so a 31 sample size was used. Overall, all p-values are significant at the 95% confidence level.

C.5 Self-conditioned Filtering: *Structure* vs *Scaffold*

There is a clear discrepancy in the substructures extracted by *Structure* and *Scaffold*. In particular, *Structure* substructures contain small functional groups which is much more permissive when used as a filter criterion compared to full scaffolds. Therefore, one would expect that many molecules in the

Table 5: Beam Enumeration works in exploitation scenarios. all experiments were run for 100 replicates with an oracle budget of 5,000 calls and reported values are the mean and standard deviation. Parentheses after Oracle Burden denote the cut-off number of molecules. The IntDiv1[68] is annotated under each Generative Yield. Welch's t-test is used to compare the difference between *Scaffold* with 'Structure Minimum Size' = 15 and Baseline Augmented Memory[23]. All p-values are significant.

Metric	Augmented Memory		Welch's t-test (95%)
	Beam Scaffold 15	**Baseline**	**p-value (N=100)**
Generative Yield$_{>0.7}$ (↑) - Diversity	**1325 ± 468** 0.76 ± 0.04	496 ± 108 0.85 ± 0.02	1.54×10^{-29}
Generative Yield$_{>0.8}$ (↑) - Diversity	**601 ± 298** 0.70 ± 0.09	85 ± 56 0.78 ± 0.03	1.35×10^{-28}
Oracle Burden$_{>0.7}$ (1) (↓)	**626 ± 260**	1085 ± 483	4.52×10^{-15}
Oracle Burden$_{>0.7}$ (10) (↓)	**997 ± 326**	2392 ± 216	2.26×10^{-80}
Oracle Burden$_{>0.7}$ (100) (↓)	**1487 ± 352**	3672 ± 197	4.01×10^{-100}
Oracle Burden$_{>0.8}$ (1) (↓)	**1415 ± 645**	3164 ± 492	2.21×10^{-53}
Oracle Burden$_{>0.8}$ (10) (↓)	**1794 ± 553 (2)**	4146 ± 326	1.14×10^{-76}
Oracle Burden$_{>0.8}$ (100) (↓)	**2490 ± 576 (2)**	4827 ± 170 (69)	1.68×10^{-25}

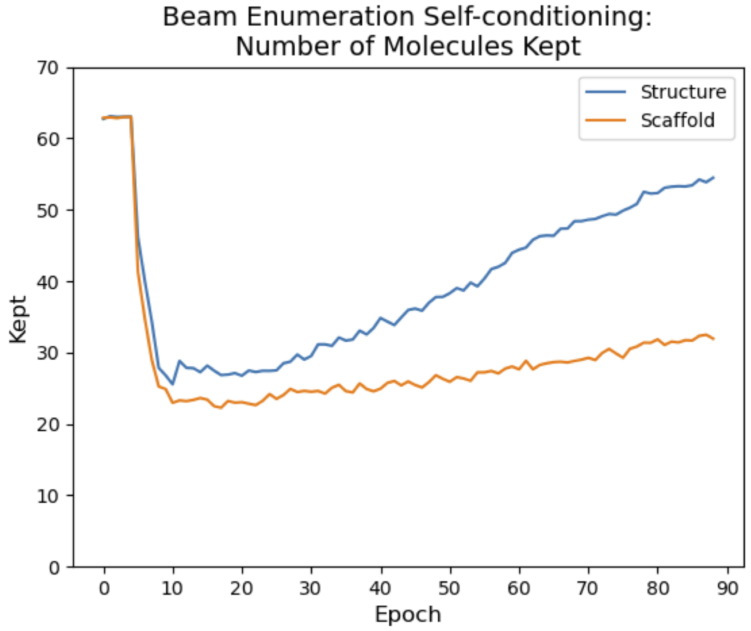


Figure C12: Behaviour of Beam Enumeration using *Structure* and *Scaffold* on self-conditioning.

sampled batches would be kept when using *Structure* Beam Enumeration. We plot the average number of molecules kept out of 64 (batch size) across the generative run when using Beam Enumeration. Note that the experiments ran for variable epochs due to the stochasticity of Beam Enumeration self-filtering. The number of epochs shown in C12 is the minimum number of epochs out of 100 replicates. Therefore, the average values shown are averaged over 100 replicates. It is evident that *Structure* is more lenient as many generated molecules make it through the filter compared to *Scaffold* which maintains a relatively strict filter. One interesting observation is that self-conditioning does not lead to obvious mode collapse. Self-conditioning is inherently biased and one would be concerned that the model gets stuck at generating the same molecules repeatedly. The fact that self-conditioning with *Scaffold* continues to filter throughout the entire generative run shows that the model is continually moving to new chemical space, supporting findings from the original Augmented Memory[23] work that Selective Memory Purge (built-in diversity mechanism) is capable of preventing mode collapse.

D Drug Discovery Case Studies

This section contains information on the Autodock Vina[61] docking protocol from receptor grid preparation to docking execution. The Beam Enumeration hyperparameters grid search results are presented for all three drug discovery case studies followed by analysis. Examples of extracted substructures are also shown and commentary provided to their significance and explainability. Finally, the wall times of all experiments are presented.

D.1 AutoDock Vina Receptor Preparation and Docking

All docking grids were prepared using DockStream[70] which uses PDBFixer[71] to refine receptor structures. The search box for all grids was 15Å x 15Å x 15Å. Docking was also performed through DockStream and followed a two step process: conformer generation using the RDKit Universal Force Field (UFF)[72] with the maximum convergence set to 600 iterations and then Vina docking was parallelized over 36 CPU cores (Intel(R) Xeon(R) Platinum 8360Y processors).

DRD2 - Dopamine Type 2 Receptor. The PDB ID is 6CM4[57] and the docking grid was centered at (x, y, z) = (9.93, 5.85, -9.58).

MK2 - MK2 Kinase. The PDB ID is 3KC3[58] and one monomer was extracted. The docking grid for the extracted monomer was centered at (x, y, z) = (-61.62, 30.31, -21.9).

AChE - Acetylcholinesterase. The PDB ID is 1EVE[59] and the docking grid was centered at (x, y, z) = (2.78, 64.38, 67.97).

D.2 Beam Enumeration Hyperparameters Grid Search Results

We performed an additional hyperparameter grid search on all three drug discovey case studies based on the insights drawn from the illustrative experiment grid search results. We fix the following hyperparameters:

- Beam K = 2
- Beam Steps = 18
- Pool Size = 4
- Patience = 5

and vary the following:

- Optimization Algorithm = [Augmented Memory[23], REINVENT[25,64]]
- Substructure Type = [*Scaffold*, *Structure*]
- Structure Minimum Size = [10, 15]
- Token Sampling Method = ["topk", "sample"]

All hyperparameter combinations (8) were tried and run for 3 replicates each for statistical reproducibility, total of 144 experiments. There are two main results we want to convey: firstly, the optimal hyperparameters are the same for all three drug discovery case studies and only the Substructure Type differs between the optimal hyperparameters here and the illustrative experiment. Secondly, Beam Enumeration is a task-agnostic general method that can be applied to existing algorithms including Augmented Memory[23] and REINVENT[25,51]. At the end of this section, we present these hyperparameters and designate these the default values. All grid search results are now presented in following tables:

Based on the results from the hyperparameters grid search in the drug discovery case studies, we make two key observations: firstly, *Structure* extraction with 'Structure Minimum Size' = 15 is now the most performant, on average (for both Augmented Memory[23] and REINVENT[25,51]). This is

Table 6: DRD2[57] case study hyperparameters grid search results for Augmented Memory[23]. All experiments were run in triplicate and the reported values are the mean and standard deviation. "Sample" denotes "sample" token sampling. All metrics are for the reward threshold > 0.8. The IntDiv1[68] is annotated under Generative Yield. * and ** denote one and two replicates were unsuccessful, respectively.

Experiment	**Generative**	**Unique**	**Oracle**	**Oracle**	**Oracle**
Augmented Memory DRD2	**Yield**	**Scaffolds**	**Burden (1)**	**Burden (10)**	**Burden (100)**
Baseline	363 ± 195	322 ± 166	187 ± 51	711 ± 120	2558 ± 30*
- Diversity	0.802 ± 0.019				
Scaffold	957 ± 75	749 ± 62	82 ± 29	668 ± 25	1818 ± 107
- Diversity	0.765 ± 0.006				
Scaffold Size 15	1607 ± 379	1023 ± 351	83 ± 29	571 ± 104	1056 ± 146
- Diversity	0.724 ± 0.027				
Scaffold Sample	948 ± 123	776 ± 128	126 ± 89	505 ± 17	1746 ± 20
- Diversity	0.734 ± 0.018				
Scaffold Sample Size 15	1552 ± 106	1274 ± 154	84 ± 29	598 ± 110	1511 ± 416
- Diversity	0.660 ± 0.041				
Structure	887 ± 112	711 ± 133	63 ± 0	595 ± 63	1862 ± 154
- Diversity	0.764 ± 0.008				
Structure Size 15	1780 ± 439	1323 ± 368	126 ± 90	582 ± 83	1120 ± 194
- Diversity	0.699 ± 0.020				
Structure Sample	912 ± 86	757 ± 30	63 ± 0	583 ± 37	2132 ± 148
- Diversity	0.767 ± 0.015				
Structure Sample Size 15	1752 ± 105	1352 ± 180	188 ± 103	776 ± 129	1289 ± 193
- Diversity	0.641 ± 0.059				

Table 7: DRD2[57] case study hyperparameters grid search results for REINVENT[25,51]. All experiments were run in triplicate and the reported values are the mean and standard deviation. "Sample" denotes "sample" token sampling. The IntDiv1[68] is annotated under Generative Yield. All metrics are for the reward threshold > 0.8. * and ** denote one and two replicates were unsuccessful, respectively.

Experiment	**Generative**	**Unique**	**Oracle**	**Oracle**	**Oracle**
REINVENT DRD2	**Yield**	**Scaffolds**	**Burden (1)**	**Burden (10)**	**Burden (100)**
Baseline	102 ± 6	101 ± 6	168 ± 149	883 ± 105	4595 ± 0**
- Diversity	0.833 ± 0.001				
Scaffold	190 ± 32	184 ± 32	63 ± 1	836 ± 178	3516 ± 575
- Diversity	0.814 ± 0.007				
Scaffold Size 15	687 ± 366	377 ± 204	127 ± 52	604 ± 71	2109 ± 1090
- Diversity	0.730 ± 0.013				
Scaffold Sample	176 ± 86	149 ± 49	105 ± 59	720 ± 121	3875 ± 883
- Diversity	0.801 ± 0.030				
Scaffold Sample Size 15	363 ± 249	225 ± 144	84 ± 30	754 ± 183	3170 ± 1188
- Diversity	0.704 ± 0.044				
Structure	184 ± 14	183 ± 14	104 ± 31	897 ± 100	3426 ± 282
- Diversity	0.817 ± 0.006				
Structure Size 15	417 ± 275	290 ± 178	63 ± 0	1099 ± 930	1928 ± 117*
- Diversity	0.730 ± 0.014				
Structure Sample	169 ± 24	167 ± 24	126 ± 52	711 ± 179	3568 ± 440
- Diversity	0.826 ± 0.003				
Structure Sample Size 15	261 ± 225	182 ± 132	209 ± 128	840 ± 107	3690 ± 1266*
- Diversity	0.734 ± 0.057				

in contrast to *Scaffold* extraction in the illustrative experiment which we rationalize through the permissive nature of the experiment compared to the docking experiments which require structure specificity. Previously, small deviations in the substructures may not have a significant impact on the reward. In physics-based oracles such as Vina[61] docking used here, small substructure differences can have an enormous impact on the outcome since the pose requires specific complementary to the protein binding site. The second observation we make which is in agreement with the illustrative experiment is that "sample" token sampling has more variance and does not perform better than "topk". The rationale is the same in that docking requires specificity and lower probability substructures exhibit more variable performance. Based on all the observations from the illustrative experiment and the drug discovery case studies, we designate the following default hyperparameter values:

- Beam K = 2
- Beam Steps = 18

Table 8: MK2[58] case study hyperparameters grid search results for Augmented Memory[23]. All experiments were run in triplicate and the reported values are the mean and standard deviation. "Sample" denotes "sample" token sampling. All metrics are for the reward threshold > 0.8. The IntDiv1[68] is annotated under Generative Yield. * and ** denote one and two replicates were unsuccessful, respectively.

Experiment **Augmented Memory MK2**	**Generative** **Yield**	**Unique** **Scaffolds**	**Oracle** **Burden (1)**	**Oracle** **Burden (10)**	**Oracle** **Burden (100)**
Baseline	34 ± 13	32 ± 12	1360 ± 543	3833 ± 394	Failed
- Diversity	0.794 ± 0.008				
Scaffold	179 ± 63	131 ± 16	1163 ± 457	2550 ± 148	4421 ± 344
- Diversity	0.743 ± 0.038				
Scaffold Size 15	523 ± 438	330 ± 269	1221 ± 564	2426 ± 1525	2676 ± 403*
- Diversity	0.676 ± 0.016				
Scaffold Sample	106 ± 71	87 ± 58	1005 ± 573	3296 ± 1181	4592 ± 334*
- Diversity	0.722 ± 0.017				
Scaffold Sample Size 15	379 ± 357	257 ± 227	983 ± 540	1846 ± 680	3244 ± 1133*
- Diversity	0.653 ± 0.026				
Structure	66 ± 18	59 ± 20	1246 ± 716	2708 ± 232	Failed
- Diversity	0.769 ± 0.029				
Structure Size 15	987 ± 211	610 ± 117	736 ± 166	1122 ± 154	2189 ± 181
- Diversity	0.704 ± 0.030				
Structure Sample	40 ± 15	34 ± 11	1119 ± 1183	3516 ± 506	Failed
- Diversity	0.784 ± 0.024				
Structure Sample Size 15	129 ± 52	117 ± 50	1208 ± 660	2799 ± 476	4037 ± 0**
- Diversity	0.671 ± 0.073				

Table 9: MK2[58] case study hyperparameters grid search results for REINVENT[25,51]. All experiments were run in triplicate and the reported values are the mean and standard deviation. "Sample" denotes "sample" token sampling. All metrics are for the reward threshold > 0.8. The IntDiv1[68] is annotated under Generative Yield. * and ** denote one and two replicates were unsuccessful, respectively.

Experiment **REINVENT MK2**	**Generative** **Yield**	**Unique** **Scaffolds**	**Oracle** **Burden (1)**	**Oracle** **Burden (10)**	**Oracle** **Burden (100)**
Baseline	2 ± 0	2 ± 0	1723 ± 802	Failed	Failed
- Diversity	0.424 ± 0.031				
Scaffold	7 ± 2	7 ± 2	1272 ± 884	4948 ± 0**	Failed
- Diversity	0.704 ± 0.051				
Scaffold Size 15	19 ± 7	18 ± 7	808 ± 524	3891 ± 631	Failed
- Diversity	0.674 ± 0.065				
Scaffold Sample	6 ± 2	6 ± 2	1427 ± 343	Failed	Failed
- Diversity	0.677 ± 0.075				
Scaffold Sample Size 15	4 ± 2	3 ± 1	2600 ± 1455	Failed	Failed
- Diversity	0.653 ± 0.026				
Structure	3 ± 1	3 ± 1	2571 ± 1155	Failed	Failed
- Diversity	0.571 ± 0.112				
Structure Size 15	179 ± 241	70 ± 87	1110 ± 268	1778 ± 0**	3208 ± 0**
- Diversity	0.670 ± 0.020				
Structure Sample	1 ± 0	1 ± 0	1737 ± 1595	Failed	Failed
- Diversity	0.192 ± 0.271				
Structure Sample Size 15	8 ± 5	7 ± 4	1943 ± 1153	4851 ± 0**	Failed
- Diversity	0.357 ± 0.255				

- Pool Size = 4
- Patience = 5
- Substructure Type = *Structure*
- Structure Minimum Size = 15
- Token Sampling Method = "topk"

D.3 Examples of Extracted Substructures: *Structure* Extraction with 'Structure Minimum Size' = 15

In this section, the top substructures at the end of the generative experiments (using Augmented Memory[23]) are shown for all three drug discovery case studies (3 replicates). All experiments are for *Structure* extraction with 'Structure Minimum Size' = 15. The extracted substructures are commonly scaffolds with "branch points", i.e., a central scaffold with single carbon bond extensions outward,

Table 10: AChE[59] case study hyperparameters grid search results for Augmented memory[23]. All experiments were run in triplicate and the reported values are the mean and standard deviation. "Sample" denotes "sample" token sampling. All metrics are for the reward threshold > 0.8. The IntDiv1[68] is annotated under Generative Yield. * and ** denote one and two replicates were unsuccessful, respectively.

Experiment **Augmented Memory AChE**	**Generative Yield**	**Unique Scaffolds**	**Oracle Burden (1)**	**Oracle Burden (10)**	**Oracle Burden (100)**
Baseline	556 ± 47	544 ± 50	62 ± 0	380 ± 0	2021 ± 89
- Diversity	0.838 ± 0.002				
Scaffold	1058 ± 102	1006 ± 113	62 ± 0	430 ± 90	1469 ± 56
- Diversity	0.823 ± 0.005				
Scaffold Size 15	2124 ± 326	1523 ± 260	63 ± 0	418 ± 27	884 ± 162
- Diversity	0.752 ± 0.029				
Scaffold Sample	1187 ± 48	1075 ± 39	84 ± 29	409 ± 77	1519 ± 141
- Diversity	0.806 ± 0.003				
Scaffold Sample Size 15	1295 ± 126	1168 ± 143	188 ± 103	602 ± 108	1440 ± 115
- Diversity	0.750 ± 0.021				
Structure	992 ± 64	946 ± 52	105 ± 59	558 ± 94	1635 ± 81
- Diversity	0.823 ± 0.005				
Structure Size 15	2059 ± 327	1552 ± 344	105 ± 29	462 ± 25	1110 ± 265
- Diversity	0.735 ± 0.017				
Structure Sample	831 ± 126	790 ± 130	62 ± 1	357 ± 29	1617 ± 220
- Diversity	0.841 ± 0.003				
Structure Sample Size 15	1277 ± 526	1031 ± 421	127 ± 52	800 ± 342	1879 ± 531
- Diversity	0.657 ± 0.070				

Table 11: AChE[59] case study hyperparameters grid search results for REINVENT[25,51]. All experiments were run in triplicate and the reported values are the mean and standard deviation. "Sample" denotes "sample" token sampling. The IntDiv1[68] is annotated under Generative Yield. All metrics are for the reward threshold > 0.8. * and ** denote one and two replicates were unsuccessful, respectively.

Experiment **REINVENT AChE**	**Generative Yield**	**Unique Scaffolds**	**Oracle Burden (1)**	**Oracle Burden (10)**	**Oracle Burden (100)**
Baseline	147 ± 11	146 ± 11	83 ± 29	481 ± 108	3931 ± 286
- Diversity	0.852 ± 0.004				
Scaffold	245 ± 50	244 ± 50	63 ± 0	566 ± 136	3360 ± 164
- Diversity	0.844 ± 0.003				
Scaffold Size 15	310 ± 207	227 ± 159	84 ± 29	421 ± 120	3596 ± 678
- Diversity	0.744 ± 0.038				
Scaffold Sample	257 ± 77	252 ± 76	63 ± 0	480 ± 60	2946 ± 460
- Diversity	0.847 ± 0.004				
Scaffold Sample Size 15	310 ± 92	271 ± 70	148 ± 28	673 ± 107	2881 ± 475
- Diversity	0.759 ± 0.039				
Structure	356 ± 22	351 ± 24	63 ± 0	294 ± 28	2284 ± 238
- Diversity	0.841 ± 0.002				
Structure Size 15	323 ± 58	284 ± 71	62 ± 0	441 ± 132	3073 ± 427
- Diversity	0.795 ± 0.009				
Structure Sample	213 ± 26	206 ± 22	84 ± 30	558 ± 222	3073 ± 279
- Diversity	0.844 ± 0.005				
Structure Sample Size 15	316 ± 253	190 ± 146	125 ± 50	561 ± 140	2683 ± 320
- Diversity	0.721 ± 0.111				

which heavily bias generation. We posit that this may be a reason why *Structure* extraction can be more performant than *Scaffold*, as observed in the hyperparameters grid search in the previous subsection.

D.4 Wall Times

The wall times for all drug discovery case studies with every algorithm is presented in Table 12. The reported values are averaged over 3 replicates. In general, adding Beam Enumeration to the base Augmented Memory[23] and REINVENT[25,51] algorithms increased wall times but only slightly and it is negligible when considering expensive oracles. An interesting observation is that "sample" token sampling increases wall time variance. This is because less probable substructures lead to more "filter rounds’, i.e., epochs where all the sampled molecules are discarded as none of them contain the Beam Enumeration extracted substructures. In addition, REINVENT generally has longer wall times even though the oracle budget is the same. The reason for this is because REINVENT

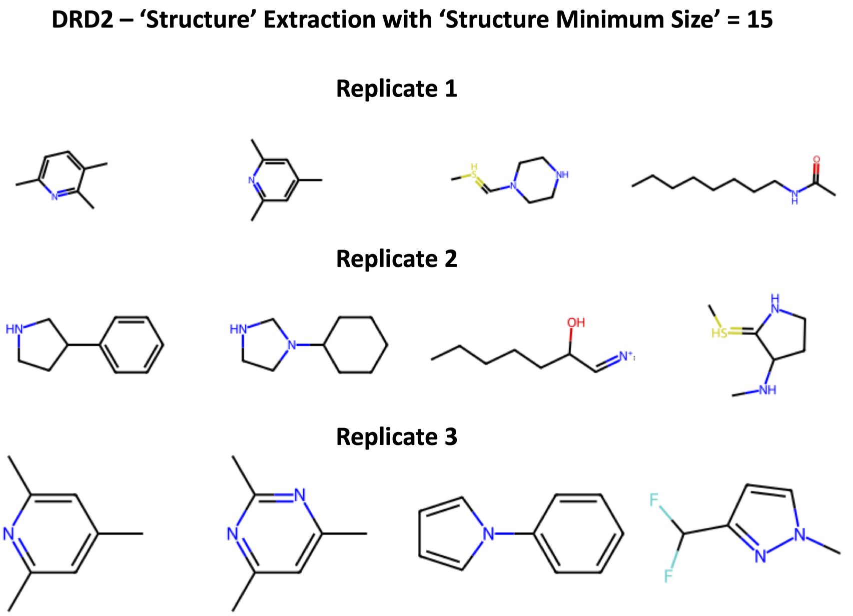


Figure D13: Augmented Memory[23] DRD2[57] substructures with *Structure* extraction and ’Structure Minimum Size’ = 15 after 5,000 oracle calls.

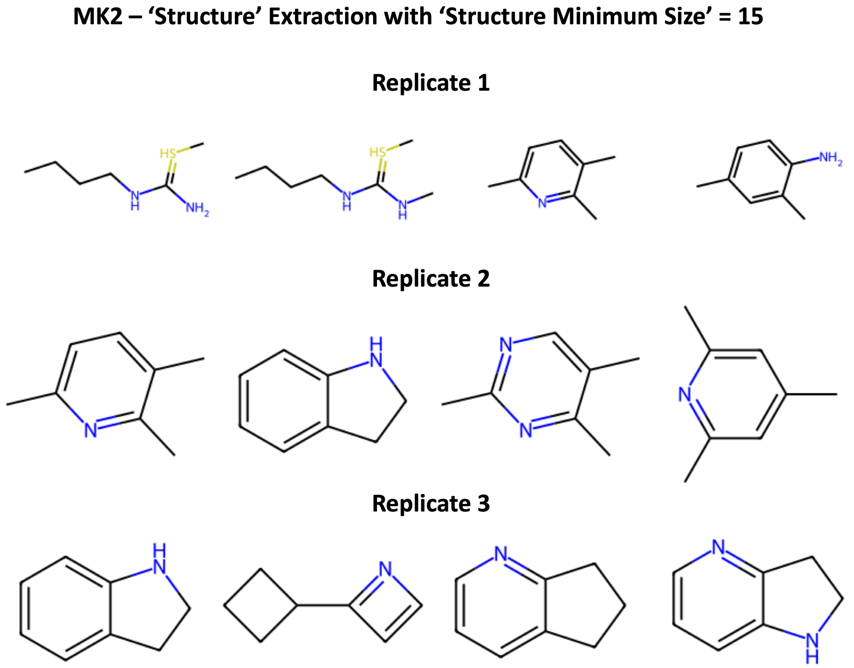

MK2 – ‘Structure’ Extraction with ‘Structure Minimum Size’ = 15
Replicate 1
HS
N
H
NH2
HS
N
H
N
H
N
NH2
Replicate 2
N
H
N
N
N
N
Replicate 3
H
N
N
N
N
N
H

Figure D14: Augmented Memory[23] MK2[58] substructures with *Structure* extraction and ’Structure Minimum Size’ = 15 after 5,000 oracle calls.

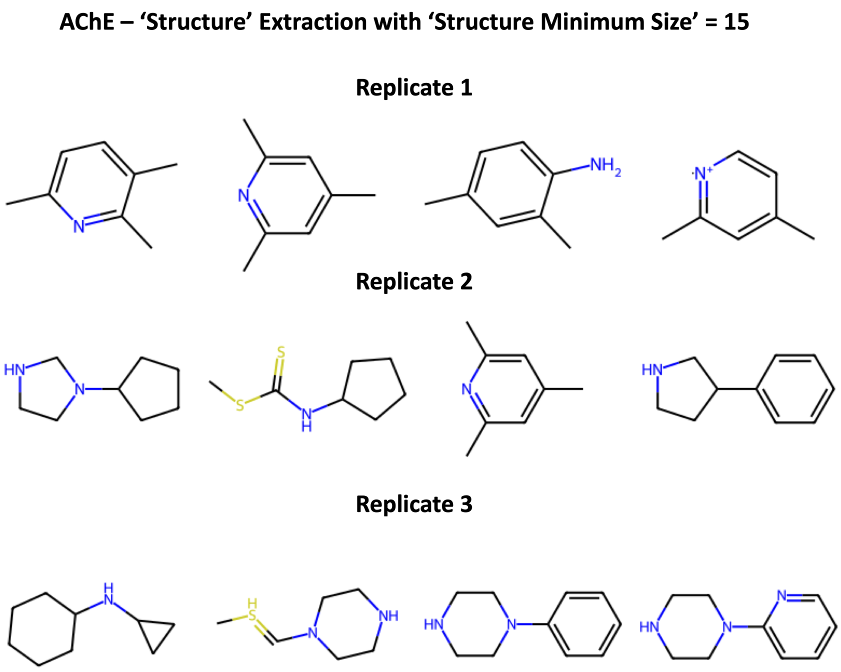

AChE – 'Structure' Extraction with 'Structure Minimum Size' = 15
Replicate 1
N
N
NH2
N+
Replicate 2
HN
N
S
S
N
H
N
HN
Replicate 3
H
N
H
S
N
NH
HN
N
HN
N
N

Figure D15: Augmented Memory[23] AChE[59] substructures with *Structure* extraction and 'Structure Minimum Size' = 15 after 5,000 oracle calls.

Table 12: Wall times for all drug discovery case studies hyperparameters grid search using Augmented Memory[23] and REINVENT[25,51]. "Sample" denotes "sample" token sampling. All experiments were run in triplicate and the values are the mean and standard deviation.

Target	Experiment	Augmented Memory Wall Time	REINVENT Wall Time
DRD2	Baseline	14h 0m ± 1h 26m	16h 36m ± 0h 55m
	Scaffold	12h 58m ± 1h 11m	17h 9m ± 1h 28m
	Scaffold Size 15	12h 56m ± 0h 46m	16h 51m ± 1h 58m
	Scaffold Sample	12h 11m ± 0h 24m	16h 32m ± 1h 3m
	Scaffold Sample Size 15	13h 32m ± 0h 50m	16h 26m ± 2h 58m
	Structure	14h 30m ± 0h 51m	22h 5m ± 1h 52m
	Structure Size 15	14h 54m ± 2h 24m	24h 33m ± 5h 8m
	Structure Sample	13h 58m ± 0h 51m	20h 5m ± 1h 42m
	Structure Sample Size 15	14h 52m ± 1h 32m	19h 52m ± 3h 22m
MK2	Baseline	10h 46m ± 0h 3m	15h 19m ± 0h 34m
	Scaffold	11h 0m ± 0h 28m	16h 21m ± 0h 53m
	Scaffold Size 15	11h 22m ± 2h 30m	16h 38m ± 1h 33m
	Scaffold Sample	12h 56m ± 0h 36m	15h 49m ± 0h 36m
	Scaffold Sample Size 15	11h 52m ± 1h 5m	16h 28m ± 0h 33m
	Structure	12h 29m ± 0h 19m	19h 40m ± 1h 55m
	Structure Size 15	11h 22m ± 1h 17m	18h 39m ± 1h 33m
	Structure Sample	12h 22m ± 0h 28m	18h 12m ± 0h 57m
	Structure Sample Size 15	12h 37m ± 0h 47m	16h 6m ± 1h 37m
AChE	Baseline	10h 6m ± 0h 39m	14h 12m ± 0h 59m
	Scaffold	11h 46m ± 0h 51m	15h 10m ± 1h 4m
	Scaffold Size 15	11h 10m ± 0h 44m	15h 52m ± 1h 4m
	Scaffold Sample	10h 55m ± 0h 44m	15h 27m ± 0h 57m
	Scaffold Sample Size 15	10h 24m ± 0h 17m	14h 53m ± 0h 53m
	Structure	13h 0m ± 0h 47m	19h 10m ± 0h 22m
	Structure Size 15	11h 26m ± 0h 51m	18h 30m ± 0h 20m
	Structure Sample	11h 23m ± 0h 22m	15h 36m ± 0h 20m
	Structure Sample Size 15	17h 56m ± 4h 27m	19h 16m ± 2h 43m

optimizes the structure components of the MPO objective: QED[62] and MW constraint to a lesser extent. Consequently, REINVENT generates larger molecules, on average, which take longer to dock with Vina[61]. This observation is in agreement with the original Augmented Memory work which compared to REINVENT.

E Augmented Memory and REINVENT Model Hyperparameters

Table 13: LSTM model hyperparameters for Augmented Memory[23] and REINVENT[25,51]

Cell Type	LSTM
Number of Layers	3
Embedding Layer Size	256
Dropout	0
Training Batch Size	128
Sampling Batch Size	64
Learning Rate	0.001

The same pre-trained prior on ChEMBL[73] was used for Augmented Memory[23] and REINVENT[25,51]. All shared hyperparameters (sampling batch size and learning rate) are the same. Default additional hyperparameters for Augmented Memory were used based on the original work[23]: two augmentation rounds and using Selective Memory Purge to prevent mode collapse. Experience replay[51,74] was kept default in REINVENT (randomly sample 10 molecules out of 100 from the replay buffer at each epoch).

References

1. Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, July 2018. doi: 10.1126/science.aat2663. URL `https://www.science.org/doi/10.1126/science.aat2663`. Publisher: American Association for the Advancement of Science.

2. Daniel Merk, Francesca Grisoni, Lukas Friedrich, and Gisbert Schneider. Tuning artificial intelligence on the de novo design of natural-product-inspired retinoid x receptor modulators. *Communications Chemistry*, 1(1):68, 2018.

3. Michael Moret, Moritz Helmstädter, Francesca Grisoni, Gisbert Schneider, and Daniel Merk. Beam search for automated design and scoring of novel ror ligands with machine intelligence. *Angewandte Chemie International Edition*, 60(35):19477–19482, 2021.

4. Francesca Grisoni, Berend JH Huisman, Alexander L Button, Michael Moret, Kenneth Atz, Daniel Merk, and Gisbert Schneider. Combining generative artificial intelligence and on-chip synthesis for de novo drug design. *Science Advances*, 7(24):eabg3338, 2021.

5. Yang Yu, Tingyang Xu, Jiawen Li, Yaping Qiu, Yu Rong, Zhen Gong, Xuemin Cheng, Liming Dong, Wei Liu, Jin Li, et al. A novel scalarized scaffold hopping algorithm with graph-based variational autoencoder for discovery of jak1 inhibitors. *ACS omega*, 6(35):22945–22954, 2021.

6. Merveille Eguida, Christel Schmitt-Valencia, Marcel Hibert, Pascal Villa, and Didier Rognan. Target-focused library design by pocket-applied computer vision and fragment deep generative linking. *Journal of Medicinal Chemistry*, 65(20):13771–13783, 2022.

7. Yueshan Li, Liting Zhang, Yifei Wang, Jun Zou, Ruicheng Yang, Xinling Luo, Chengyong Wu, Wei Yang, Chenyu Tian, Haixing Xu, et al. Generative deep learning enables the discovery of a potent and selective ripk1 inhibitor. *Nature Communications*, 13(1):6891, 2022.

8. Xiaoqin Tan, Chunpu Li, Ruirui Yang, Sen Zhao, Fei Li, Xutong Li, Lifan Chen, Xiaozhe Wan, Xiaohong Liu, Tianbiao Yang, et al. Discovery of pyrazolo [3, 4-d] pyridazinone derivatives as selective ddr1 inhibitors via deep learning based design, synthesis, and biological evaluation. *Journal of Medicinal Chemistry*, 65(1):103–119, 2021.

9. Seong Hun Jang, Dakshinamurthy Sivakumar, Sathish Kumar Mudedla, Jaehan Choi, Sungmin Lee, Minjun Jeon, Suneel Kumar Bvs, Jinha Hwang, Minsung Kang, Eun Gyeong Shin, et al. Pcw-a1001, ai-assisted de novo design approach to design a selective inhibitor for flt-3 (d835y) in acute myeloid leukemia. *Frontiers in Molecular Biosciences*, 9:1072028, 2022.

10. Nannan Chen, Lijuan Yang, Na Ding, Guiwen Li, Jiajing Cai, Xiaoli An, Zhijie Wang, Jie Qin, and Yuzhen Niu. Recurrent neural network (rnn) model accelerates the development of antibacterial metronidazole derivatives. *RSC advances*, 12(35):22893–22901, 2022.

11. Yi Hua, Xiaobao Fang, Guomeng Xing, Yuan Xu, Li Liang, Chenglong Deng, Xiaowen Dai, Haichun Liu, Tao Lu, Yanmin Zhang, et al. Effective reaction-based de novo strategy for kinase targets: a case study on mertk inhibitors. *Journal of Chemical Information and Modeling*, 62(7): 1654–1668, 2022.

12. Shukai Song, Haotian Tang, Ting Ran, Feng Fang, Linjiang Tong, Hongming Chen, Hua Xie, and Xiaoyun Lu. Application of deep generative model for design of pyrrolo [2, 3-d] pyrimidine derivatives as new selective tank binding kinase 1 (tbk1) inhibitors. *European Journal of Medicinal Chemistry*, 247:115034, 2023.

13. Michael Moret, Irene Pachon Angona, Leandro Cotos, Shen Yan, Kenneth Atz, Cyrill Brunner, Martin Baumgartner, Francesca Grisoni, and Gisbert Schneider. Leveraging molecular structure and bioactivity with chemical language models for de novo drug design. *Nature Communications*, 14(1):114, 2023.

14. Marco Ballarotto, Sabine Willems, Tanja Stiller, Felix Nawa, Julian A Marschner, Francesca Grisoni, and Daniel Merk. De novo design of nurr1 agonists via fragment-augmented generative deep learning in low-data regime. *Journal of Medicinal Chemistry*, 2023.

15. Maria Korshunova, Niles Huang, Stephen Capuzzi, Dmytro S. Radchenko, Olena Savych, Yuriy S. Moroz, Carrow I. Wells, Timothy M. Willson, Alexander Tropsha, and Olexandr Isayev. Generative and reinforcement learning approaches for the automated de novo design of bioactive compounds. *Commun Chem*, 5(1):1–11, October 2022. ISSN 2399-3669. doi: 10.1038/s42004-022-00733-0. URL `https://www.nature.com/articles/s42004-022-00733-0`. Number: 1 Publisher: Nature Publishing Group.

16. Atsushi Yoshimori, Yasunobu Asawa, Enzo Kawasaki, Tomohiko Tasaka, Seiji Matsuda, Toru Sekikawa, Satoshi Tanabe, Masahiro Neya, Hideaki Natsugari, and Chisato Kanai. Design and synthesis of ddr1 inhibitors with a desired pharmacophore using deep generative models. *ChemMedChem*, 16(6):955–958, 2021.

17. Alex Zhavoronkov, Yan A. Ivanenkov, Alex Aliper, Mark S. Veselov, Vladimir A. Aladinskiy, Anastasiya V. Aladinskaya, Victor A. Terentiev, Daniil A. Polykovskiy, Maksim D. Kuznetsov, Arip Asadulaev, Yury Volkov, Artem Zholus, Rim R. Shayakhmetov, Alexander Zhebrak, Lidiya I. Minaeva, Bogdan A. Zagribelnyy, Lennart H. Lee, Richard Soll, David Madge, Li Xing, Tao Guo, and Alán Aspuru-Guzik. Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nat Biotechnol*, 37(9):1038–1040, September 2019. ISSN 1546-1696. doi: 10.1038/s41587-019-0224-x. URL `https://www.nature.com/articles/s41587-019-0224-x`. Number: 9 Publisher: Nature Publishing Group.

18. Feng Ren, Xiao Ding, Min Zheng, Mikhail Korzinkin, Xin Cai, Wei Zhu, Alexey Mantsyzov, Alex Aliper, Vladimir Aladinskiy, Zhongying Cao, Shanshan Kong, Xi Long, Bonnie Hei Man Liu, Yingtao Liu, Vladimir Naumov, Anastasia Shneyderman, Ivan V. Ozerov, Ju Wang, Frank W. Pun, Daniil A. Polykovskiy, Chong Sun, Michael Levitt, Alán Aspuru-Guzik, and Alex Zhavoronkov. AlphaFold accelerates artificial intelligence powered drug discovery: efficient discovery of a novel CDK20 small molecule inhibitor. *Chemical Science*, 14(6):1443–1452, 2023. doi: 10.1039/D2SC05709C. URL `https://pubs.rsc.org/en/content/articlelanding/2023/sc/d2sc05709c`. Publisher: Royal Society of Chemistry.

19. Yangguang Li, Yingtao Liu, Jianping Wu, Xiaosong Liu, Lin Wang, Ju Wang, Jiaojiao Yu, Hongyun Qi, Luoheng Qin, Xiao Ding, et al. Discovery of potent, selective, and orally bioavailable small-molecule inhibitors of cdk8 for the treatment of cancer. *Journal of Medicinal Chemistry*, 2023.

20. Leslie Salas-Estrada, Davide Provasi, Xing Qiu, H Umit Kaniskan, Xi-Ping Huang, Jeffrey DiBerto, Joao Marcelo Lamim Ribeiro, Jian Jin, Bryan L Roth, and Marta Filizola. De novo design of κ-opioid receptor antagonists using a generative deep learning framework. *bioRxiv*, pages 2023–04, 2023.

21. Lingle Wang, Yujie Wu, Yuqing Deng, Byungchan Kim, Levi Pierce, Goran Krilov, Dmitry Lupyan, Shaughnessy Robinson, Markus K Dahlgren, Jeremy Greenwood, et al. Accurate and reliable prediction of relative ligand binding potency in prospective drug discovery by way of a modern free-energy calculation protocol and force field. *Journal of the American Chemical Society*, 137(7):2695–2703, 2015.

22. J Harry Moore, Christian Margreitter, Jon Paul Janet, Ola Engkvist, Bert L de Groot, and Vytautas Gapsys. Automated relative binding free energy calculations from smiles to $\delta\delta$g. *Communications Chemistry*, 6(1):82, 2023.

23. Jeff Guo and Philippe Schwaller. Augmented memory: Capitalizing on experience replay to accelerate de novo molecular design. *arXiv preprint arXiv:2305.16160*, 2023.

24. David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.*, 28(1):31–36, February 1988. ISSN 0095-2338. doi: 10.1021/ci00057a005. URL `https://doi.org/10.1021/ci00057a005`. Publisher: American Chemical Society.

25. Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of Cheminformatics*, 9(1):48, September 2017. ISSN 1758-2946. doi: 10.1186/s13321-017-0235-x. URL `https://doi.org/10.1186/s13321-017-0235-x`.

26. Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science Advances*, 4(7):eaap7885, July 2018. doi: 10.1126/sciadv.aap7885. URL `https://www.science.org/doi/10.1126/sciadv.aap7885`. Publisher: American Association for the Advancement of Science.

27. Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1): 120–131, 2018.

28. Manan Goel, Shampa Raghunathan, Siddhartha Laghuvarapu, and U Deva Priyakumar. Molegular: molecule generation using reinforcement learning with alternating rewards. *Journal of Chemical Information and Modeling*, 61(12):5815–5826, 2021.

29. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, June 2014. URL `http://arxiv.org/abs/1406.2661`. arXiv:1406.2661 [cs, stat].

30. Benjamin Sanchez-Lengeling, Carlos Outeiral, Gabriel L Guimaraes, and Alan Aspuru-Guzik. Optimizing distributions over molecular space. an objective-reinforced generative adversarial network for inverse-design chemistry (organic). 2017.

31. Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models, February 2018. URL `http://arxiv.org/abs/1705.10843`. arXiv:1705.10843 [cs, stat].

32. Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, December 2022. URL `http://arxiv.org/abs/1312.6114`. arXiv:1312.6114 [cs, stat].

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

34. Jiaxuan You, Bowen Liu, Rex Ying, Vijay Pande, and Jure Leskovec. Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation, February 2019. URL `http://arxiv.org/abs/1806.02473`. arXiv:1806.02473 [cs, stat].

35. Wengong Jin, Dr Regina Barzilay, and Tommi Jaakkola. Multi-Objective Molecule Generation using Interpretable Substructures. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4849–4859. PMLR, November 2020. URL `https://proceedings.mlr.press/v119/jin20b.html`. ISSN: 2640-3498.

36. Rocío Mercado, Tobias Rastemo, Edvard Lindelöf, Günter Klambauer, Ola Engkvist, Hongming Chen, and Esben Jannik Bjerrum. Graph networks for molecular design. *Mach. Learn.: Sci. Technol.*, 2(2):025023, March 2021. ISSN 2632-2153. doi: 10.1088/2632-2153/abcf91. URL `https://dx.doi.org/10.1088/2632-2153/abcf91`. Publisher: IOP Publishing.

37. Sara Romeo Atance, Juan Viguera Diez, Ola Engkvist, Simon Olsson, and Rocío Mercado. De novo drug design using reinforcement learning with graph-based deep generative models. *Journal of Chemical Information and Modeling*, 62(20):4863–4872, 2022.

38. 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.

39. Tianfan Fu, Wenhao Gao, Connor Coley, and Jimeng Sun. Reinforced genetic algorithm for structure-based drug design. *Advances in Neural Information Processing Systems*, 35:12325–12338, 2022.

40. Wenhao Gao, Tianfan Fu, Jimeng Sun, and Connor W. Coley. Sample Efficiency Matters: A Benchmark for Practical Molecular Optimization, October 2022. URL `http://arxiv.org/abs/2206.12411`. arXiv:2206.12411 [cs, q-bio].

41. Prashant Gohel, Priyanka Singh, and Manoranjan Mohanty. Explainable ai: current status and future directions. *arXiv preprint arXiv:2107.07045*, 2021.

42. Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

43. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

44. André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10):1340–1347, 2010.

45. Lloyd S Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10): 1095–1100, 1953.

46. Geemi P Wellawatte, Aditi Seshadri, and Andrew D White. Model agnostic generation of counterfactual explanations for molecules. *Chemical science*, 13(13):3697–3705, 2022.

47. Sarveswara Rao Vangala, Sowmya Ramaswamy Krishnan, Navneet Bung, Rajgopal Srinivasan, and Arijit Roy. pbrics: A novel fragmentation method for explainable property prediction of drug-like small molecules. *Journal of Chemical Information and Modeling*, 2023.

48. Tianfan Fu, Cao Xiao, Xinhao Li, Lucas M Glass, and Jimeng Sun. MIMOSA: Multi-constraint molecule sampling for molecule optimization. *AAAI*, 2020.

49. 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 *ICLR*, 2021.

50. Yoshua Bengio, Tristan Deleu, Edward J. Hu, Salem Lahlou, Mo Tiwari, and Emmanuel Bengio. GFlowNet foundations. *CoRR*, abs/2111.09266, 2021.

51. Thomas Blaschke, Josep Arús-Pous, Hongming Chen, Christian Margreitter, Christian Tyrchan, Ola Engkvist, Kostas Papadopoulos, and Atanas Patronov. REINVENT 2.0: An AI Tool for De Novo Drug Design. *J. Chem. Inf. Model.*, 60(12):5918–5922, December 2020. ISSN 1549-9596. doi: 10.1021/acs.jcim.0c00915. URL `https://doi.org/10.1021/acs.jcim.0c00915`. Publisher: American Chemical Society.

52. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

53. Vendy Fialková, Jiaxi Zhao, Kostas Papadopoulos, Ola Engkvist, Esben Jannik Bjerrum, Thierry Kogej, and Atanas Patronov. LibINVENT: Reaction-based Generative Scaffold Decoration for in Silico Library Design. *J. Chem. Inf. Model.*, 62(9):2046–2063, May 2022. ISSN 1549-9596. doi: 10.1021/acs.jcim.1c00469. URL `https://doi.org/10.1021/acs.jcim.1c00469`. Publisher: American Chemical Society.

54. Guy W Bemis and Mark A Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of medicinal chemistry*, 39(15):2887–2893, 1996.

55. Alex Graves. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*, 2012.

56. Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. High-dimensional sequence transduction. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 3178–3182. IEEE, 2013.

57. Sheng Wang, Tao Che, Anat Levit, Brian K Shoichet, Daniel Wacker, and Bryan L Roth. Structure of the d2 dopamine receptor bound to the atypical antipsychotic drug risperidone. *Nature*, 555 (7695):269–273, 2018.

58. Maria A Argiriadi, Anna M Ericsson, Christopher M Harris, David L Banach, David W Borhani, David J Calderwood, Megan D Demers, Jennifer DiMauro, Richard W Dixon, Jennifer Hardman, et al. 2, 4-diaminopyrimidine mk2 inhibitors. part i: observation of an unexpected inhibitor binding mode. *Bioorganic & medicinal chemistry letters*, 20(1):330–333, 2010.

59. Gitay Kryger, Israel Silman, and Joel L Sussman. Structure of acetylcholinesterase complexed with e2020 (aricept®): implications for the design of new anti-alzheimer drugs. *Structure*, 7(3): 297–307, 1999.

60. Jeff Guo, Jon Paul Janet, Matthias R. Bauer, Eva Nittinger, Kathryn A. Giblin, Kostas Papadopoulos, Alexey Voronov, Atanas Patronov, Ola Engkvist, and Christian Margreitter. DockStream: a docking wrapper to enhance de novo molecular design. *Journal of Cheminformatics*, 13(1):89, November 2021. ISSN 1758-2946. doi: 10.1186/s13321-021-00563-7. URL `https://doi.org/10.1186/s13321-021-00563-7`.

61. Oleg Trott and Arthur J Olson. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2):455–461, 2010.

62. G. Richard Bickerton, Gaia V. Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L. Hopkins. Quantifying the chemical beauty of drugs. *Nature Chem*, 4(2):90–98, February 2012. ISSN 1755-4349. doi: 10.1038/nchem.1243. URL `https://www.nature.com/articles/nchem.1243`. Number: 2 Publisher: Nature Publishing Group.

63. John A Arnott and Sonia Lobo Planey. The influence of lipophilicity in drug discovery and design. *Expert opinion on drug discovery*, 7(10):863–875, 2012.

64. Thomas Blaschke, Josep Arús-Pous, Hongming Chen, Christian Margreitter, Christian Tyrchan, Ola Engkvist, Kostas Papadopoulos, and Atanas Patronov. REINVENT 2.0: an ai tool for de novo drug design. *Journal of Chemical Information and Modeling*, 60(12):5918–5922, 2020.

65. Anna Gaulton, Louisa J. Bellis, A. Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, and John P. Overington. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Res*, 40 (Database issue):D1100–D1107, January 2012. ISSN 0305-1048. doi: 10.1093/nar/gkr777. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3245175/`.

66. Michael Dodds, Jeff Guo, Thomas Löhr, Alessandro Tibo, Ola Engkvist, and Jon Paul Janet. Sample efficient reinforcement learning with active learning for molecular design. 2023.

67. Gregory W Kyro, Anton Morgunov, Rafael I Brent, and Victor S Batista. Chemspaceal: An efficient active learning methodology applied to protein-specific molecular generation. *arXiv preprint arXiv:2309.05853*, 2023.

68. Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, et al. Molecular sets (moses): a benchmarking platform for molecular generation models. *Frontiers in pharmacology*, 11:565644, 2020.

69. Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Simon Johansson, Hongming Chen, Sergey Nikolenko, Alán Aspuru-Guzik, and Alex Zhavoronkov. Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. *Frontiers in Pharmacology*, 11, 2020. ISSN 1663-9812. URL `https://www.frontiersin.org/articles/10.3389/fphar.2020.565644`.

70. Jeff Guo, Jon Paul Janet, Matthias R Bauer, Eva Nittinger, Kathryn A Giblin, Kostas Papadopoulos, Alexey Voronov, Atanas Patronov, Ola Engkvist, and Christian Margreitter. Dockstream: a docking wrapper to enhance de novo molecular design. *Journal of cheminformatics*, 13(1):1–21, 2021.

71. Peter Eastman, Jason Swails, John D Chodera, Robert T McGibbon, Yutong Zhao, Kyle A Beauchamp, Lee-Ping Wang, Andrew C Simmonett, Matthew P Harrigan, Chaya D Stern, et al. Openmm 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS computational biology*, 13(7):e1005659, 2017.

72. Anthony K Rappé, Carla J Casewit, KS Colwell, William A Goddard III, and W Mason Skiff. Uff, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *Journal of the American chemical society*, 114(25):10024–10035, 1992.

73. Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. Chembl: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(D1):D1100–D1107, 2012.

74. Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching.