
RGFN: Synthesizable Molecular Generation Using GFlowNets

Michał Koziarski^{*1,2}, Andrei Rekes^{*3}, Dmytro Shevchuk^{*3}, Almer van der Sloot^{1,2},
Piotr Gaiński^{4,1,2}, Yoshua Bengio^{1,2}, Cheng-Hao Liu^{1,5}, Mike Tyers^{6,3}, Robert A. Batey^{3,7}

¹ Mila – Québec AI Institute, ² Université de Montréal, ³ University of Toronto,
⁴ Jagiellonian University, ⁵ McGill University, ⁶ The Hospital for Sick Children Research Institute,
⁷ Acceleration Consortium, * Equal contribution
michal.koziarski@mila.quebec, {a.rekes, dmytro.shevchuk}@mail.utoronto.ca

Abstract

Generative models hold great promise for small molecule discovery, significantly increasing the size of search space compared to traditional in silico screening libraries. However, most existing machine learning methods for small molecule generation suffer from poor synthesizability of candidate compounds, making experimental validation difficult. In this paper we propose Reaction-GFlowNet (RGFN), an extension of the GFlowNet framework that operates directly in the space of chemical reactions, thereby allowing out-of-the-box synthesizability while maintaining comparable quality of generated candidates. We demonstrate that with the proposed set of reactions and building blocks, it is possible to obtain a search space of molecules orders of magnitude larger than existing screening libraries coupled with low cost of synthesis. We also show that the approach scales to very large fragment libraries, further increasing the number of potential molecules. We demonstrate the effectiveness of the proposed approach across a range of oracle models, including pretrained proxy models and GPU-accelerated docking.

1 Introduction

Traditionally, machine learning has been applied to drug discovery for screening existing libraries of compounds, whether actual physical collections or pre-configured in silico collections of readily synthesizable compounds, in a supervised fashion. However, supervised screening of the whole drug-like space, often estimated to contain approximately 10^{60} [41] different compounds, is infeasible in practice. Generative methods offer the potential to circumvent this issue by sampling directly from a distribution over desirable chemical properties without the need to evaluate every possible molecular structure. Despite these advances, existing generative approaches tend not to explicitly enforce synthesizability [21], generating samples that might be either very costly or altogether impossible to chemically synthesize. Ensuring that generative methods operate in the space of synthesizable compounds, yet at a much larger and more diverse scale than existing chemical libraries, remains an open challenge.

In this paper, we propose Reaction-GFlowNet (RGFN), an extension of the GFlowNet framework [5] that generates molecules by combining basic chemical fragments using a chain of reactions. We propose a relatively small collection of cheap and accessible chemical building blocks (reactants), as well as established high-yield chemical transformations, that together can still produce a search space orders of magnitude larger than existing chemical libraries. We additionally propose several domain-specific extensions of the GFlowNet framework for action representation and scaling to a larger space of possible actions.

Source code available at <https://github.com/koziarskilab/RGFN>.

We experimentally evaluate RGFN on a set of diverse screening tasks, including docking score approximation with a trained proxy model for soluble epoxide hydrolase (sEH), GPU-accelerated direct docking score calculations for multiple protein targets (Mpro, ClpP, TBLR1 and sEH), and biological activity estimation with a trained proxy model for dopamine receptor type 2 (DRD2) receptor activity and senolytic activity [74]. We demonstrate that RGFN produces similar optimization quality and diversity to existing fragment-based approaches while ensuring straightforward synthetic routes for predicted hit compounds.

2 Related work

Generative models for molecular discovery. A plethora of methods have been developed for molecular generation [48, 7] using machine learning. These methods can be categorized depending on the molecular representation used, including textual representations such as SMILES [34, 3, 39], molecular graphs [33, 46, 54] or 3D atom coordinate representations [52], as well as the underlying methodology, for example, variational autoencoders [33, 46], reinforcement learning [54, 37] or diffusion models [60]. Recently, Generative Flow Networks (GFlowNets) [4, 50, 59, 64, 72, 19] have emerged as a promising paradigm for molecular generation due to their ability to sample large and diverse candidate small molecule space, which is crucial in the drug discovery process. Traditionally, GFlowNets for molecular generation operated on the graph representation level, and candidate molecules were generated as a sequence of actions in which either individual atoms or small molecular fragments were combined to form a final molecule. While using graph representations, as opposed to textual or 3D representations, allows the enforcement of the validity of the generated molecules, it does not guarantee a straightforward route for chemical synthesis. Here, we expand on the GFlowNet framework by modifying the space of actions to consist of choosing molecular fragments and executing compatible chemical reactions/transformations, in turn guaranteeing both physical-chemical validity and synthesizability.

Synthesizability in generative models. One approach to ensuring the synthesizability of generated molecules is by using a scoring function, either utilizing it as one of the optimization criteria [37], or as a post-processing step for filtering generated molecules. Multiple scoring approaches, both heuristic [18, 23] and ML-based [42], exist in the literature. Another branch of research focuses on using reaction models and traversing predicted synthesis graph [8, 12, 38, 55]. However, reaction and synthesizability estimation is difficult in practice, and can fail to generalize out-of-distribution in the case of ML models. Furthermore, theoretical synthesizability does not necessarily account for the cost of synthesis. Because of this, a preferable approach might be to constrain the space of possible molecules to those easily synthesized by operating in a predefined space of chemical reactions and fragments. Several recent strategies employ this approach [22, 49, 67], including reinforcement learning-based methods [24, 27] and a concurrent work utilizing GFlowNets [15]. We extend this line of investigation not only by translating the concept to the GFlowNet framework but also by proposing a curated set of robust chemical reactions and fragments that ensure efficient synthesis at low total costs.

3 Method

3.1 Generative Flow Networks

GFlowNets are amortized variational inference algorithms that are trained to sample from an unnormalized target distribution over compositional objects. GFlowNets aim to sample objects from a set of terminal states \mathcal{X} proportionally to a reward function $\mathcal{R} : X \rightarrow \mathbb{R}^+$. GFlowNets are defined on a pointed directed acyclic graph (DAG), $G = (S, A)$, where:

- $s \in S$ are the nodes, referred to as states in our setting, with the special starting state s_0 being the only state with no incoming edges, and the terminal states \mathcal{X} have no outgoing edges,
- $a = s \rightarrow s' \in A$ are the edges, referred to as actions in our setting, and correspond to applying an action while in a state s and landing in state s' .

We can define a non-negative flow function on the edges $F(s \rightarrow s')$ and on the states $F(s)$ of the DAG such that $\forall x \in \mathcal{X} F(x) = \mathcal{R}(x)$. A perfectly trained GFlowNet should satisfy the following flow-matching constraint:

$$\forall s \in S \quad F(s) = \sum_{(s'' \rightarrow s) \in A} F(s'' \rightarrow s) = \sum_{(s \rightarrow s') \in A} F(s \rightarrow s'). \quad (1)$$

A state sequence $\tau = (s_0 \rightarrow s_1 \rightarrow \dots \rightarrow s_n = x)$, with $s_n = x \in \mathcal{X}$ and $a_i = (s_i \rightarrow s_{i+1}) \in A$ for all i , is called a complete trajectory. We denote the set of trajectories as \mathcal{T} .

Another way to rephrase the flow-matching constraints is to learn a forward policy $P_F(s_{i+1}|s_i)$ such that trajectories starting at s_0 and taking actions sampled by P_F terminate at $x \in \mathcal{X}$ proportional to the reward.

Trajectory balance. Several training losses have been explored to train GFlowNets. Among these, trajectory balance [45] has been shown to improve credit assignment. In addition to learning a forward policy P_F , we also learn a backward policy P_B and a scalar Z_θ , such that, for every trajectory $\tau = (s_0 \rightarrow s_1 \rightarrow \dots \rightarrow s_n = x)$, they satisfy:

$$Z_\theta \prod_{t=1}^n P_F(s_t|s_{t-1}) = R(x) \prod_{t=1}^n P_B(s_{t-1}|s_t) \quad (2)$$

3.2 Reaction-GFlowNet

Reaction-GFlowNet generates molecules by combining basic chemical fragments using a chain of reactions. The generation process comprises the following steps (illustrated in Figure 1):

1. Select an initial building block (reactant or surrogate reactant; see Appendix C.2 for more details about surrogate reactants).
2. Select the reaction template (a graph transformation describing the reaction).
3. Select another reactant.
4. Perform the in silico reaction and select one of the resulting molecules.
5. Repeat steps 2-4 until the stop action is selected.

In the rest of this section, we describe the design of the Reaction-GFlowNet in detail.

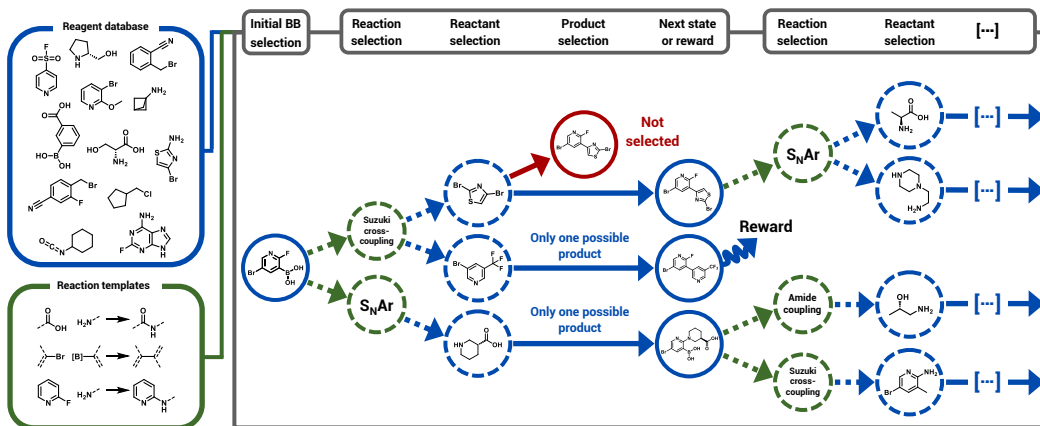


Figure 1: Illustration of RGFN sampling process. At the beginning, the RGFN selects an initial molecular building block. In the next two steps, a reaction and a proper reactant are chosen. Then the in silico reaction is simulated with RDKit's RunReactants functionality and one of the resulting molecules is selected. The process is repeated until the stop action is chosen. The obtained molecule is then evaluated using the reward function.

Preliminaries. Reaction-GFlowNet uses a predefined set of reaction patterns and molecules introduced in Section 3.3. We denote these as R and M respectively. As a backbone for our forward policy P_F , we use a graph transformer model f from [77]. The graph transformer takes as an input a molecular graph m and outputs the embedding $f(m) \in \mathbb{R}^D$, where D is the embedding dimension.

In particular, f can embed an empty graph \emptyset . It can additionally be conditioned on the reaction $r \in R$ which we denote as $f(m, r)$. The reaction in this context is represented as its index in the reaction set R .

Select an initial building block. At the beginning of each trajectory, Reaction-GFlowNet selects an initial fragment from the collection of building blocks M . The probability of choosing i -th fragment m_i is equal to:

$$p(m_i|\emptyset) = \sigma^{|M|}(\mathbf{s})_i, \quad s_i = \text{MLP}_M(f(\emptyset))_i, \quad (3)$$

where $\text{MLP}_M : \mathbb{R}^D \rightarrow \mathbb{R}^{|M|}$ is a multi-layer perceptron (MLP). The σ^k is a standard softmax over the logits vector $\mathbf{s} \in \mathbb{R}^k$ of the length k :

$$\sigma^k(\mathbf{s})_i = \frac{\exp(s_i)}{\sum_{j=1}^k \exp(s_j)}.$$

Select the reaction template. The next step is to select a reaction that can be applied to the molecule m . The probability of choosing i -th reaction from R is described as:

$$p(r_i|m) = \sigma^{|R|+1}(\mathbf{s})_i, \quad s_i = \text{MLP}_R(f(m))_i, \quad (4)$$

where $\text{MLP}_R : \mathbb{R}^D \rightarrow \mathbb{R}^{|R|+1}$ is an MLP that outputs logits for reactions from R and an additional stop action with index $|R| + 1$. Choosing the stop action in this phase ends the generation process. Note that not all the reactions may be applied to the molecule m . We appropriately filter such reactions and assume that the score s_i for non-feasible reactions is equal to $-\infty$.

Select another reactant. We want to find a molecule $m_i \in M$ that will react with m in the reaction r . The probability for selecting m_i is defined as:

$$p(m_i|m, r) = \sigma^{|M|}(\mathbf{s})_i, \quad s_i = \text{MLP}_M(f(m, r))_i \quad (5)$$

where MLP_M is shared with the initial fragment selection phase. As in the previous phase, not all the fragments can be used with the reaction r , so we filter these out.

Perform the reaction and select one of the resulting molecules. In this step, we apply the reaction r to the two fragment molecules chosen in previous steps. As the reaction pattern can be matched to multiple parts of the molecules, the result of this operation is a set of possible outcomes M' . We choose the molecule $m'_i \in M'$ by sampling from the following distribution:

$$p(m'_i) = \sigma^{|M'|}(\mathbf{s})_i, \quad s_i = \text{MLP}_{M'}(f(m'_i)), \quad (6)$$

where $\text{MLP}_{M'} : \mathbb{R}^D \rightarrow \mathbb{R}$ scores the embedded m'_i molecule.

Backward Policy. A backward policy in RGFN is only non-deterministic in states corresponding to a molecule m which is a result of performing some reaction $r \in R$ on molecule m' and reactant $m'' \in R$. We denote the set of such tuples (r, m', m'') that may result in m as T . We override the indexing and let (r_i, m'_i, m''_i) be the i -th tuple from T . The probability of choosing the i -th tuple is:

$$p((r_i, m'_i, m''_i)|m) = \sigma^{|T|}(\mathbf{s})_i, \quad s_i = \text{MLP}_B(f(m'_i, r_i)), \quad (7)$$

where $\text{MLP}_B : \mathbb{R}^D \rightarrow \mathbb{R}$ and f is a backbone transformer model similar to the one used in the forward policy. To properly define T , we need to implicitly keep track of the number of reactions performed to obtain m (denoted as k). Only those tuples (r, m', m'') are contained in the T for which we can recursively obtain m' in $k - 1$ reactions.

Action Embedding. While the MLP_M used to predict the probabilities of selecting a molecule $m_i \in M$ works well for our predefined M , it underperforms when the size of possible chemical building block library is increased. Such an MLP_M likely struggles to reconstruct the relationship between the molecules. Intuitively, when a molecule m_i is chosen in some trajectory, the training signal from the loss function should also influence the probability of choosing a structurally similar m_j . However, the MLP_M disregards the structural similarity by construction and it intertwines the probabilities of choosing m_i and m_j only with the softmax function. To incorporate the relationship between molecules into the model, we embed the molecular building blocks with a simple machine learning model g and reformulate the probability of choosing a particular building block m_i :

$$p(m_i|m, r) = \sigma^{|M|}(\mathbf{s})_i, \quad s_i = \phi(Wf(m, r))^T g(m_i), \quad (8)$$

where ϕ is some activation function (we use GELU) and $W \in \mathbb{R}^{D \times D}$ is a learnable linear layer. Note that if we define $g(m_i)$ as an index embedding function that simply returns a distinct embedding for every m_i , we will obtain a formulation equivalent to Equation (5). To leverage the structure of molecules during the training, we use g that linearly embeds a MACSS fingerprint [40] of an input molecule m_i along with the index i . Note that this approach does not add any additional computational costs during the inference as the embeddings $g(m_i)$ can be cached. In Section 4.3, we show that this method greatly improves the performance when scaling to larger sets of fragments.

3.3 Chemical language

We select seventeen reactions and 350 building blocks for our model. These include amide bond formation, nucleophilic aromatic substitution, Michael addition, isocyanate-based urea synthesis, sulfur fluoride exchange (SuFEx), sulfonyl chloride substitution, alkyne-azide and nitrile-azide cycloadditions, esterification reactions, urea synthesis using carbonyl surrogates, Suzuki-Miyaura, Buchwald-Hartwig, and Sonogashira cross-couplings, amide reduction, and peptide terminal thiourea cyclization reactions to produce iminohydantoin and tetrazoles. The chosen reactions are known to be typically quite robust [9] and are often high-yielding (75-100%), thus enforcing reliable synthesis pathways when sampling molecules from our model. To simulate couplings in Python, reactions are encoded as SMARTS templates. To ensure compatible building blocks yielding specific, chemically valid products and to enable parent state computation, we introduce multiple variants corresponding to differing reagent types for most of the proposed reactions. In some cases SMARTS templates encode reactions where one of the reagents is not specified. We describe these transformations as *implicit reactions* (Appendix E). Additionally, we introduce *surrogate reactions* where the SMARTS templates and SMILES strings encode for alternate building blocks (see Appendix C.2 for more details). Finally, once again for the sake of specificity, reactions are duplicated by swapping the order of the building block reactants. In total, 132 different SMARTS templates are used.

During the construction of the curated building block database, only affordable reagents (i.e., building blocks) are considered. For the purposes of this study we define affordable reagents to be those priced at less than or equal to \$200 per gram. The mean cost per gram of reagents selected for this study is \$22.52, the lowest cost \$0.023 per gram, and the highest cost \$190 per gram (see Appendix N for more details on cost estimation).

A crucial consideration when choosing the set of reactions and fragments used is the state space size (the number of possible molecules that can be generated using our framework). This is difficult to compute precisely since a different set of reactions or building blocks is valid for every state in a given trajectory. We estimate this based on 1,000 random trajectories instead (details can be found in Appendix A). In addition to our 350 low-cost fragments, we also perform this analysis with 8,000 additional random Enamine building blocks. Comparison for different numbers of maximum reactions is presented in Figure 2. We demonstrate that even with curated low-cost reactants and limiting the number to a maximum of four reactions, state space size is an order of magnitude greater than the number of molecules contained in Enamine REAL [17]. This size can increase significantly with the addition of more fragments and/or an increase in the maximum number of reactions. Additional discussions regarding scaling can be found in Section 4.3.

4 Experimental study

In the conducted experiments we compare oracle scores and synthesizability scores of RGFN with several state-of-the-art reference methods. Secondly, we examine the capabilities of RGFN to scale to larger fragment libraries, in particular when using the proposed action embedding mechanism. Finally, we perform an in-depth examination of generated ligands across several biologically relevant targets.

4.1 Set-up

Throughout the course of the conducted computational experimental study, we aim to evaluate the performance of the proposed approach across several diverse biological oracles of interest. This includes proxy models (machine learning oracles, pretrained on the existing data and used for higher computational efficiency): first, the commonly used sEH proxy as described in [4]. Second, a graph

neural network trained on the biological activity classification task of senolytic [74] recognition. Third, the Dopamine Receptor D2 (DRD2) oracle [53] from Therapeutics Data Commons [28]. Proxy model details are provided in Appendix B.1.

Per the GFlowNet training algorithm, the reward is calculated for a batch of dozens to hundreds of molecules at each training step, rendering traditional computational docking score algorithms like AutoDock Vina [70] infeasible for very large training runs. As a result, previous applications of GFlowNets to biological design [4, 64] employed a fast pre-trained proxy model trained on docking scores instead. These proxies, while lightweight, present potential issues should the GFlowNet generate molecules outside their training data distributions and require receptor-specific datasets. To circumvent this, we use the GPU-accelerated Vina-GPU 2.1 [68] implementation of the QuickVina 2 [2] docking algorithm to calculate docking scores directly in the training loop of RGFN. This approach allows for drastically increased flexibility in protein target selection while eliminating proxy generalization failure. We selected X-ray crystal structures of human soluble epoxy hydrolase (sEH), ATP-dependent Clp protease proteolytic subunit (ClpP), SARS-CoV-2 main protease (Mpro), and transducin β -like-related protein 1 as targets for evaluating RGFN using a docking reward (detailed motivation for specific target selection is provided in Appendix G).

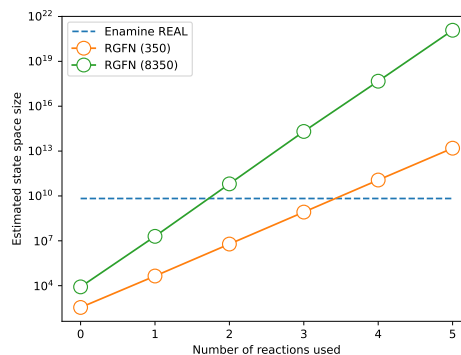


Figure 2: Estimation of the state space size of RGFN as a function of the maximum number of allowed reactions. RGFN (350) indicates a variant using 350 hand-picked inexpensive building blocks, while RGFN (8350) also uses 8,000 randomly selected Enamine building blocks. Enamine REAL (6.5B compounds) is shown as a reference.

4.2 Comparison with existing methods

We begin experimental evaluation with a comparison to several state-of-the-art methods for molecular discovery. Specifically, we consider a genetic algorithm operating on molecular graphs (GraphGA) [32] as implemented in [10], which has been demonstrated to be a very strong baseline for molecular discovery [21], Monte Carlo tree search-based SyntheMol [67], cascade variational autoencoder (casVAE) [49], and a fragment-based GFlowNet (FGFN) [4] as implemented in [57]. For FGFN, we additionally considered its variant that had a SAScore as one of the reward terms (FGFN+SA). Training details can be found in Appendix B.2. It is worth noting that besides SyntheMol, which also operates in the space of chemical reactions and building blocks derived from the Enamine database, and casVAE, which used the set of reaction trees obtained from the USPTO database [43], our remaining benchmarks do not explicitly enforce synthesizability when generating molecules. Because of this, in this section, we will examine not only the quality of generated molecules in terms of optimized properties but also their synthesizability. We consider only two reaction-based approaches, as other existing methods employing this paradigm [27, 24] do not share code or curated reactions and building blocks, making reproduction difficult.

We first examine the distributions of rewards found by each method across four different oracles used for training: sEH proxy, senolytic proxy, DRD2 proxy, and GPU-accelerated docking for ClpP. The results are presented in Figure 3. As can be seen, while RGFN underperforms in terms of average reward when compared to the method not enforcing synthesizability (GraphGA), it outperforms SyntheMol's and casVAE's reaction-based sampling. Interestingly, when compared to standard FGFN, RGFN either performs similarly (ClpP docking) or achieves higher average rewards. This is most striking in the case of the challenging senolytic discovery task, in which a proxy is trained on a severely imbalanced dataset with less than 100 actives, resulting in a sparse reward function. We suspect that this, possibly combined with a lack of compatibility between the FGFN fragments and known senolytics, led to the failure to discover any high-reward molecules. However, RGFN succeeds in the task and finds a wide range of senolytic candidates. Finally, the gap in performance is

even larger between RGFN and FGFN+SA, indicating that introducing synthesizability constraints reduces the ability of FGFN to discover high-reward molecules.

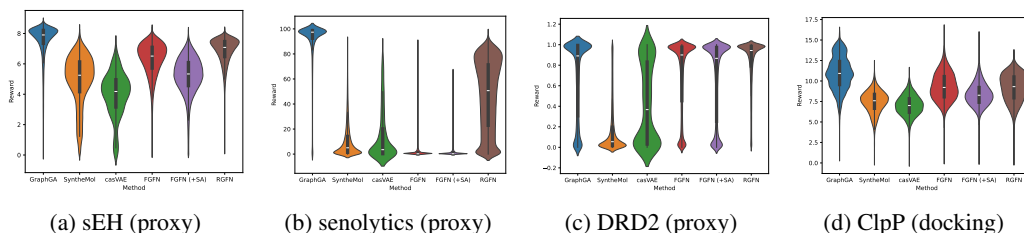


Figure 3: Distributions of rewards across different tasks.

Secondly, we examine the number of discovered modes for each method, with a mode defined as a molecule with computed reward above a threshold (sEH: 7, senolytics: 50, DRD2: 0.95, ClpP docking: 10), and Tanimoto similarity to every other mode < 0.5. We use Leader algorithm for mode computation. The number of discovered modes across tasks as a function of normalized iterations is presented in Figure 4. Note that in the case of GraphGA, FGFN, FGFN+SA, and RGFN this simply translates to the number of oracle calls, but for SyntheMol and casVAE, due to large computational overhead, we impose a maximum number of oracle calls such that training time was comparable to RGFN (see Appendix B.2 for details). Note that in the case of casVAE, this resulted in a very small number of molecules being visited in the allotted time. As can be seen, despite slightly worse average rewards, FGFN still outperforms other methods in terms of the number of discovered modes (with the exception of senolytic discovery task, where it fails to discover any high-reward molecules). This includes FGFN+SA, despite its generally worse performance than FGFN. This suggests that RGFN samples are less diverse, possibly due to the relatively small number of fragments and reactions used. However, RGFN still outperforms remaining methods across all tasks, suggesting that it preserves some of the benefits of the diversity-focused GFlowNet framework.

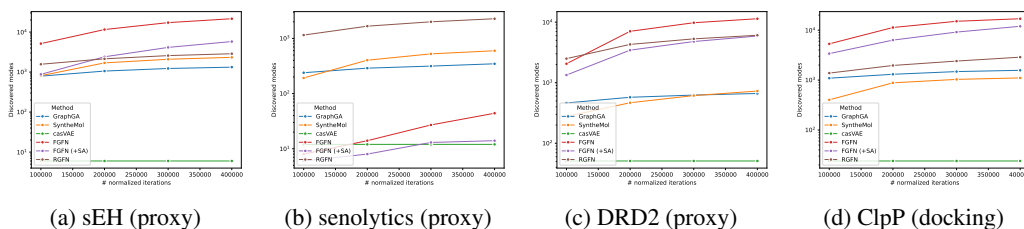


Figure 4: Number of discovered modes as a function of normalized iterations. Log scale used.

Finally, we evaluate the synthesizability of the generated compounds as a key output. We present average values of several synthesizability-related metrics, computed over top-k modes generated for each method, in Table 1. We include measures indicating average molecular weight and drug-likeness (QED) to gauge the size of generated compounds. Furthermore, for completeness, we also include SAScores [18], but note that they are only a rough approximation of ease of synthesis. For a better estimate of synthesizability we perform retrosynthesis using AiZynthFinder [23] and count the average number of molecules for which a valid retrosynthesis pathway was found. However, it is important to note that both SAScores and AiZynthFinder scores are inherently noisy metrics. While we evaluate all methods using them for the sake of rigorousness, ultimately molecules generated by RGFN (as well as SyntheMol and casVAE) are guaranteed to be highly likely synthesizable. Note that to reduce variance, we compute molecular weight, QED, and SAScores over the top-500 modes, but due to high computational cost, AiZynthFinder scores are computed only over top-100 modes. As can be seen, while there is some variance across tasks, RGFN performs similarly to SyntheMol and casVAE in terms of both synthesizability scores, and significantly outperforms GraphGA and FGFN. Crucially, including SAScore as a reward does improve the performance of FGFN in terms of that metric, but does not drastically change the AiZynthFinder scores, demonstrating that it is insufficient to guarantee synthesizability. All RGFN modes were additionally inspected manually by an expert chemist and confirmed as synthesizable, which indicates that AiZynth scores are likely underestimated.

Table 1: Average values of synthesizability-related metrics for top-k modes.

Task	Method	Mol. weight ↓	QED ↑	SAScore ↓	AiZynth ↑
sEH	GraphGA	528.6 ± 42.3	0.21 ± 0.06	3.87 ± 0.24	0.04
	SyntheMol	411.1 ± 66.7	0.57 ± 0.18	<u>2.85 ± 0.55</u>	0.80
	casVAE	<u>421.6 ± 103.4</u>	<u>0.52 ± 0.23</u>	2.41 ± 0.47	0.82
	FGFN	473.4 ± 58.9	0.39 ± 0.13	3.43 ± 0.48	0.14
	FGFN+SA	473.7 ± 62.2	0.36 ± 0.12	3.01 ± 0.50	0.27
	RGFN	495.2 ± 49.6	0.29 ± 0.10	3.09 ± 0.39	0.56
Seno.	GraphGA	485.7 ± 75.6	0.09 ± 0.05	2.92 ± 0.26	0.05
	SyntheMol	<u>441.4 ± 83.5</u>	<u>0.48 ± 0.19</u>	2.77 ± 0.40	0.53
	casVAE	431.5 ± 100.9	0.50 ± 0.19	<u>2.82 ± 0.46</u>	0.65
	FGFN	468.9 ± 47.7	0.42 ± 0.13	3.55 ± 0.52	0.02
	FGFN+SA	451.8 ± 54.5	0.32 ± 0.12	2.83 ± 0.44	0.13
	RGFN	558.7 ± 62.8	0.21 ± 0.09	3.24 ± 0.32	<u>0.58</u>
ClpP	GraphGA	521.0 ± 31.8	0.32 ± 0.07	4.14 ± 0.51	0.00
	SyntheMol	<u>458.2 ± 60.7</u>	<u>0.45 ± 0.16</u>	2.86 ± 0.56	0.56
	casVAE	423.0 ± 61.7	0.47 ± 0.17	2.44 ± 0.41	0.84
	FGFN	548.6 ± 42.9	0.22 ± 0.03	2.94 ± 0.54	0.25
	FGFN+SA	509.2 ± 52.4	0.24 ± 0.04	<u>2.61 ± 0.49</u>	0.33
	RGFN	526.2 ± 37.6	0.23 ± 0.04	2.83 ± 0.22	<u>0.65</u>
DRD2	GraphGA	475.4 ± 53.2	0.42 ± 0.12	2.50 ± 0.23	0.41
	SyntheMol	365.6 ± 54.3	0.72 ± 0.14	2.78 ± 0.43	0.66
	casVAE	404.8 ± 83.5	0.59 ± 0.20	<u>2.42 ± 0.38</u>	0.87
	FGFN	386.5 ± 45.0	0.63 ± 0.11	2.58 ± 0.54	0.76
	FGFN+SA	<u>381.1 ± 35.1</u>	<u>0.64 ± 0.10</u>	2.37 ± 0.37	<u>0.78</u>
	RGFN	447.1 ± 45.7	0.44 ± 0.10	2.79 ± 0.34	0.87

4.3 Scaling to larger sets of fragments

Next we investigate the influence of a fragment embedding scheme proposed in Section 3.2. In the standard implementation of the GFlowNet policy, actions are represented as independent embeddings in the MLP. These encode actions as indices, effectively disregarding their respective structures and all information contained therein. The model must thus select from a library of reagents without any knowledge as to their chemical makeup or properties. While finding similarities between actions may be a relatively easy task for small action spaces, it becomes more difficult when the size of the action space increases. To scale RGFN to a larger size of the building block library, we proposed to encode building block selection actions using molecular fingerprints, allowing the model to leverage their internal structures without any additional computational overhead during inference. In Figure 5, we observe that our fingerprint embedding scheme allows for drastically faster convergence compared to the standard independent action embedding, especially for large library sizes. The details on how the larger fragment libraries were created can be found in Appendix F.

4.4 Examination of the produced ligands

In the final stage of experiments we examine the capabilities of RGFN to produce high quality ligands across multiple diverse docking targets (see Appendix G for more details). The aim is to evaluate whether 1) the chemical language used is expressive enough to produce structurally diverse molecules for different targets, and 2) whether the generated ligands form realistic poses in the binding pockets. We first demonstrate the diversity of ligands across targets on a UMAP plot of extended-connectivity fingerprints (Figure 6). Ligands assigned to specific targets form very distinct clusters, showcasing their diversity. Interestingly, we also observe structural differences between sEH proxy and sEH docking, possibly indicating poor approximation of docking scores by the proxy model. Secondly, we examine the docking poses of the highest scoring generated ligands (Figure 7). As can be seen, the generated molecules produce realistic docking poses, closely resembling the poses of known ligands (Appendix K), despite being diverse in terms of structural similarity (Appendix M). We further conduct a cost analysis and synthesis planning for top modes in Appendices N and O. Overall, this demonstrates the usefulness of the proposed RGFN approach in the docking-based screens.

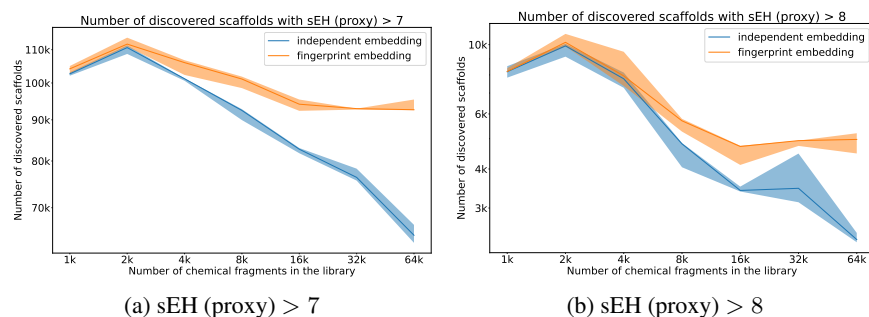


Figure 5: The number of discovered Murcko scaffolds with sEH proxy value above 7 (a) and 8 (b) as a function of fragment library size. We compare standard independent embeddings of fragment selection actions (blue) with our fingerprint-based embeddings (orange) that account for the fragments' chemical structure. The number of scaffolds is reported after 2k training iterations for 3 random seeds (the solid line is the median, while the shaded area spans from minimum to maximum values). We observe that our approach greatly outperforms independent embedding when scaling to a larger action space.

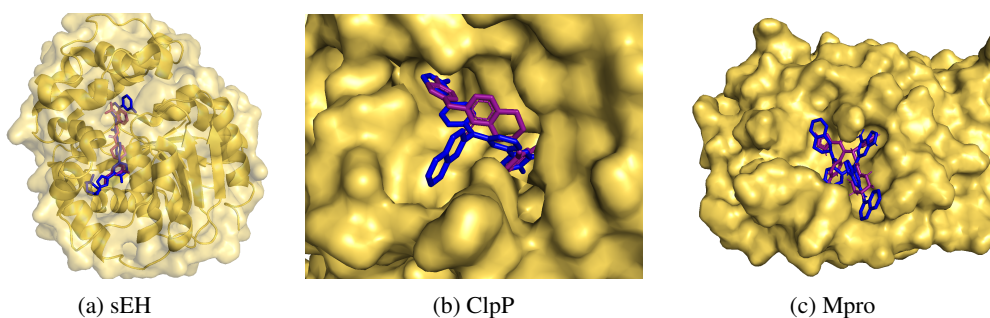


Figure 7: Top docked RGFN ligands after filtering steps (blue) overlaid with the PDB-derived ligand (purple) for each of sEH, ClpP, and Mpro.

5 Limitations

The current proof-of-principle implementation of RGFN uses only 17 reaction types and 350 building blocks. Although these limited inputs already generate a vast chemical space, this represents only a small fraction of possible drug-like space, which in turn limits the quality and potency of the generated molecular structures. The scaling experiment demonstrates that the number of building blocks can readily be increased, and increasing the number and diversity of building blocks is a straightforward way to enhance and survey the accessible chemical space.

The current set of building blocks and reactions tends to generate linear and flat-shaped molecules. Adding a small set of cyclization reactions (such as peptide macrocyclization and ring-closing metathesis), along with more complex-shaped scaffold building blocks, as well as reactions that introduce sp^3 hybridized atoms and stereochemical complexity will therefore allow for greater shape diversity and the

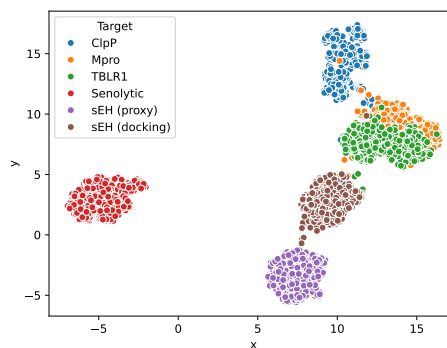


Figure 6: UMAP plot of chemical structures of top-500 modes generated for each target. RGFN generates sufficient chemical diversity to produce distinct clusters of compounds. See Appendix G for description of each target protein.

generation of more potent molecules [51, 20]. It is also important to recognize that RGFN does not explicitly generate synthetic routes to the molecules, at least not in a strict chemical sense, which in addition to a sequence of reactions transforming sets of reactants into products (which RGFN does provide), would also include choice of reaction conditions, external reagents, catalysts, protection group strategies, etc.

Another significant limitation in the quality of the generated molecular structures is the reliance on molecular docking as a scoring oracle. Although molecular docking has been successfully used in large-scale virtual screening efforts [44, 61, 35], it has well-known shortcomings in its predictive power. First, docking scores correlate strongly with molecular weight (MW) [13] and do not account for drug-likeness requirements like optimal MW or ClogP. In this work, molecule size was constrained only by the number of reaction steps, encouraging RGFN to generate large molecules within the building block limit. This can be somewhat rectified by augmenting reward with a drug-likeness or ligand efficiency term. Second, its binding affinity predictions and rankings often correlate weakly with experimental values and are highly dependent on the nature of the target protein's binding site [73, 71]. This is further illustrated by the fact that known ligands are not necessarily characterized by highest possible docking scores (Appendix L). This limitation impacts the learning of the chemical structure-activity relationship space, leading to the generation of sub-optimal molecules. One solution to this limitation is to incorporate more accurate but computationally expensive methods (such as ensemble docking, MM-PBSA, and FEP) within a multi-fidelity framework [26]. However, since we focus on robust, affordable, and facile synthesis methods, we ultimately aim to extend our approach beyond computational scoring methods by directly conducting experimental evaluation of synthesized compound batches within an active learning loop.

6 Conclusions

In this paper, we present RGFN, an extension of the GFlowNet framework that operates in the action space of chemical reactions. We propose a curated set of high-yield chemical reactions and low-cost molecular building blocks that can be used with the method. We demonstrate that even with a small set of reactions and building blocks, the proposed approach produces a state space with a size orders of magnitude larger than typical experimental screening libraries while ensuring high synthesizability of the generated compounds. We also show that the size of the search space can be further increased by including additional building blocks and that the proposed action embedding mechanism improves scalability to very large building block spaces.

In the course of our experiments, we show that RGFN achieves roughly comparable average rewards to state-of-the-art methods, and it outperforms another approach operating directly in the space of chemical reactions and, crucially, standard fragment-based GFlowNets. At the same time, it significantly improves the synthesizability of generated compounds when compared to a fragment-based GFlowNet. Analysis of ligands produced across the set of diverse tasks demonstrates sufficient diversity of proposed chemical space to generalize to various targets. While not yet demonstrated experimentally, ease of synthesis (due to the small stock of cheap fragments and high-yield chemical reactions used) combined with reasonably high optimization quality of bespoke ligands offer a promising alternative to standard high-throughput screening applications. In particular, it can be beneficial for active learning-based pipelines with significant wet lab component, reducing the reliance on inaccurate docking oracles. Facilitating the drug discovery process through the generation of novel small molecules can eventually lead to the discovery of novel medications leading to significant societal benefits.

Acknowledgments and Disclosure of Funding

This work was supported by funding from CQDM Fonds d'Accélération des Collaborations en Santé (FACS) / Acuité Québec and the National Research Council (NRC) Canada, the Canadian Institutes for Health Research (CIHR), grant no. FDN-167277, Samsung and Microsoft. D. Shevchuk is grateful for support from the Mitacs Globalink Graduate Fellowship, grant no. FR121160/FR121161. The research of P. Gaiński was supported by the National Science Centre (Poland), grant no. 2022/45/B/ST6/01117. Additional thanks for funding provided to the University of Toronto's Acceleration Consortium from the Canada First Research Excellence Fund, grant no. CFREF-2022-00042. Computational resources were provided by the Digital Research Alliance of

Canada (<https://alliancecan.ca>) and Mila (<https://mila.quebec>). We gratefully acknowledge Poland's high-performance Infrastructure PLGrid (ACK Cyfronet Athena, HPC) for providing computer facilities and support within computational grant no PLG/2023/016550.

References

- [1] Beatrice Alexander-Howden, Li Zhang, Almer M van der Sloot, Sylvain Tollis, Daniel J St-Cyr, Frank Sicheri, Adrian P Bird, Mike Tyers, and Matthew J Lyst. A screen for MeCP2-TBL1 interaction inhibitors using a luminescence-based assay. *Sci. Rep.*, 13(1):3868, March 2023.
- [2] Amr Alhossary, Stephanus Daniel Handoko, Yuguang Mu, and Chee-Keong Kwoh. Fast, accurate, and reliable molecular docking with QuickVina 2. *Bioinformatics*, 31(13):2214–2216, 02 2015. ISSN 1367-4803. doi: 10.1093/bioinformatics/btv082. URL <https://doi.org/10.1093/bioinformatics/btv082>.
- [3] Josep Arús-Pous, Atanas Patronov, Esben Jannik Bjerrum, Christian Tyrchan, Jean-Louis Reymond, Hongming Chen, and Ola Engkvist. SMILES-based deep generative scaffold decorator for de-novo drug design. *Journal of cheminformatics*, 12:1–18, 2020.
- [4] 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.
- [5] Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. GFlowNet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023.
- [6] Vaibhav Bhandari, Keith S Wong, Jin Lin Zhou, Mark F Mabanglo, Robert A Batey, and Walid A Houry. The role of ClpP protease in bacterial pathogenesis and human diseases. *ACS Chem. Biol.*, 13(6):1413–1425, June 2018.
- [7] Camille Bilodeau, Wengong Jin, Tommi Jaakkola, Regina Barzilay, and Klavs F Jensen. Generative models for molecular discovery: Recent advances and challenges. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 12(5):e1608, 2022.
- [8] John Bradshaw, Brooks Paige, Matt J Kusner, Marwin Segler, and José Miguel Hernández-Lobato. A model to search for synthesizable molecules. *Advances in Neural Information Processing Systems*, 32, 2019.
- [9] Dean G. Brown and Jonas Boström. Analysis of past and present synthetic methodologies on medicinal chemistry: Where have all the new reactions gone? Miniperspective. *Journal of Medicinal Chemistry*, 59(10):4443–4458, 2016.
- [10] Nathan Brown, Marco Fiscato, Marwin HS Segler, and Alain C Vaucher. GuacaMol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108, 2019.
- [11] Martin Buttenschoen, Garrett M. Morris, and Charlotte M. Deane. PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences, 2023.
- [12] Alexander Button, Daniel Merk, Jan A Hiss, and Gisbert Schneider. Automated de novo molecular design by hybrid machine intelligence and rule-driven chemical synthesis. *Nature machine intelligence*, 1(7):307–315, 2019.
- [13] Giorgio Carta, Andrew JS Knox, and David G Lloyd. Unbiasing scoring functions: a new normalization and rescoring strategy. *Journal of chemical information and modeling*, 47(4): 1564–1571, 2007.
- [14] COVID Moonshot Consortium, Hagit Achdout, Anthony Aimon, Dominic S Alonzi, Robert Arbon, Elad Bar-David, Haim Barr, Amir Ben-Shmuel, James Bennett, Vitaliy A Bilenko, et al. Open science discovery of potent non-covalent SARS-CoV-2 main protease inhibitors. *BioRxiv*, pages 2020–10, 2020.

- [15] Miruna Cretu, Charles Harris, Julien Roy, Emmanuel Bengio, and Pietro Lio. SynFlowNet: Towards molecule design with guaranteed synthesis pathways. In *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2024.
- [16] Thomas Duflot, Clothilde Roche, Fabien Lamoureux, Dominique Guerrot, and Jeremy Bellien. Design and discovery of soluble epoxide hydrolase inhibitors for the treatment of cardiovascular diseases. *Expert Opin. Drug Discov.*, 9(3):229–243, March 2014.
- [17] Enamine. Enamine REAL database. <http://enamine.net/compound-collections/real-compounds/real-database>, 2024. [Online; Accessed 22 May 2024].
- [18] Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1:1–11, 2009.
- [19] Piotr Gaiński, Michał Koziarski, Krzysztof Maziarz, Marwin Segler, Jacek Tabor, and Marek Śmieja. RetroGFN: Diverse and feasible retrosynthesis using GFlowNets. *arXiv preprint arXiv:2406.18739*, 2024.
- [20] Warren R.J.D. Galloway, Albert Isidro-Llobet, and David R. Spring. Diversity-oriented synthesis as a tool for the discovery of novel biologically active small molecules. *Nature Communications*, 1(1):80, Sep 2010. ISSN 2041-1723. doi: 10.1038/ncomms1081. URL <https://doi.org/10.1038/ncomms1081>.
- [21] Wenhao Gao and Connor W Coley. The synthesizability of molecules proposed by generative models. *Journal of chemical information and modeling*, 60(12):5714–5723, 2020.
- [22] Wenhao Gao, Rocío Mercado, and Connor W Coley. Amortized tree generation for bottom-up synthesis planning and synthesizable molecular design. *arXiv preprint arXiv:2110.06389*, 2021.
- [23] Samuel Genheden, Amol Thakkar, Veronika Chadimová, Jean-Louis Reymond, Ola Engkvist, and Esben Bjerrum. AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning. *Journal of cheminformatics*, 12(1):70, 2020.
- [24] Sai Krishna Gottipati, Boris Sattarov, Sufeng Niu, Yashaswi Pathak, Haoran Wei, Shengchao Liu, Simon Blackburn, Karam Thomas, Connor Coley, Jian Tang, et al. Learning to navigate the synthetically accessible chemical space using reinforcement learning. In *International conference on machine learning*, pages 3668–3679. PMLR, 2020.
- [25] Paul R Graves, Lucas J Aponte-Collazo, Emily M J Fennell, Adam C Graves, Andrew E Hale, Nedyalka Dicheva, Laura E Herring, Thomas S K Gilbert, Michael P East, Ian M McDonald, Matthew R Lockett, Hani Ashamalla, Nathaniel J Moorman, Donald S Karanewsky, Edwin J Iwanowicz, Ekhsan Holmuhamedov, and Lee M Graves. Mitochondrial protease ClpP is a target for the anticancer compounds ONC201 and related analogues. *ACS Chem. Biol.*, 14(5):1020–1029, May 2019.
- [26] Alex Hernandez-Garcia, Nikita Saxena, Moksh Jain, Cheng-Hao Liu, and Yoshua Bengio. Multi-fidelity active learning with GFlowNets. *arXiv preprint arXiv:2306.11715*, 2023.
- [27] Julien Horwood and Emmanuel Noutahi. Molecular design in synthetically accessible chemical space via deep reinforcement learning. *ACS omega*, 5(51):32984–32994, 2020.
- [28] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *arXiv preprint arXiv:2102.09548*, 2021.
- [29] John J. Irwin, Da Duan, Hayarpi Torosyan, Allison K. Doak, Kristin T. Ziebart, Teague Sterling, Gurgen Tumanian, and Brian K. Shoichet. An aggregation advisor for ligand discovery. *Journal of Medicinal Chemistry*, 58(17):7076–7087, 2015. doi: 10.1021/acs.jmedchem.5b01105. URL <https://doi.org/10.1021/acs.jmedchem.5b01105>. PMID: 26295373.

- [30] Jo Ishizawa, Sarah F Zarabi, R Eric Davis, Ondrej Halgas, Takenobu Nii, Yulia Jitkova, Ran Zhao, Jonathan St-Germain, Lauren E Heese, Grace Egan, Vivian R Ruvolo, Samir H Barghout, Yuki Nishida, Rose Hurren, Wencai Ma, Marcela Gronda, Todd Link, Keith Wong, Mark Mabanglo, Kensuke Kojima, Gautam Borthakur, Neil MacLean, Man Chun John Ma, Andrew B Leber, Mark D Minden, Walid Houry, Hagop Kantarjian, Martin Stogniew, Brian Raught, Emil F Pai, Aaron D Schimmer, and Michael Andreeff. Mitochondrial ClpP-mediated proteolysis induces selective cancer cell lethality. *Cancer Cell*, 35(5):721–737.e9, May 2019.
- [31] Samuel Jacques, Almer M van der Sloot, Caroline C Huard, Jasmin Coulombe-Huntington, Sarah Tsao, Sylvain Tollis, Thierry Bertomeu, Elizabeth J Culp, Daniel Pallant, Michael A Cook, Eric Bonneil, Pierre Thibault, Gerard D Wright, and Mike Tyers. Imipridone anticancer compounds ectopically activate the ClpP protease and represent a new scaffold for antibiotic development. *Genetics*, 214(4):1103–1120, April 2020.
- [32] Jan H Jensen. A graph-based genetic algorithm and generative model/Monte Carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572, 2019.
- [33] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pages 2323–2332. PMLR, 2018.
- [34] Seokho Kang and Kyunghyun Cho. Conditional molecular design with deep generative models. *Journal of chemical information and modeling*, 59(1):43–52, 2018.
- [35] Anat Levit Kaplan, Danielle N. Confair, Kuglae Kim, Ximena Barros-Álvarez, Ramona M. Rodriguiz, Ying Yang, Oh Sang Kweon, Tao Che, John D. McCorvy, David N. Kamber, James P. Phelan, Luan Carvalho Martins, Vladimir M. Pogorelov, Jeffrey F. DiBerto, Samuel T. Slocum, Xi-Ping Huang, Jain Manish Kumar, Michael J. Robertson, Ouliana Panova, Alpay B. Seven, Autumn Q. Wetsel, William C. Wetsel, John J. Irwin, Georgios Skiniotis, Brian K. Shoichet, Bryan L. Roth, and Jonathan A. Ellman. Bespoke library docking for 5-HT_{2A} receptor agonists with antidepressant activity. *Nature*, 610(7932):582–591, Oct 2022. ISSN 1476-4687. doi: 10.1038/s41586-022-05258-z. URL <https://doi.org/10.1038/s41586-022-05258-z>.
- [36] Daniel W Kneller, Gwyndalyn Phillips, Hugh M O’Neill, Robert Jedrzejczak, Lucy Stols, Paul Langan, Andrzej Joachimiak, Leighton Coates, and Andrey Kovalevsky. Structural plasticity of SARS-CoV-2 3CL mpro active site cavity revealed by room temperature x-ray crystallography. *Nat. Commun.*, 11(1):3202, June 2020.
- [37] Maksym Korablyov, Cheng-Hao Liu, Moksh Jain, Almer M van der Sloot, Eric Jolicoeur, Edward Ruediger, Andrei Cristian Nica, Emmanuel Bengio, Kostiantyn Lapchevskyi, Daniel St-Cyr, et al. Generative active learning for the search of small-molecule protein binders. *arXiv preprint arXiv:2405.01616*, 2024.
- [38] Ksenia Korovina, Sailun Xu, Kirthevasan Kandasamy, Willie Neiswanger, Barnabas Poczos, Jeff Schneider, and Eric Xing. ChemBO: Bayesian optimization of small organic molecules with synthesizable recommendations. In *International Conference on Artificial Intelligence and Statistics*, pages 3393–3403. PMLR, 2020.
- [39] Panagiotis-Christos Kotsias, Josep Arús-Pous, Hongming Chen, Ola Engkvist, Christian Tyrchan, and Esben Jannik Bjerrum. Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks. *Nature Machine Intelligence*, 2(5):254–265, 2020.
- [40] Hiroyuki Kuwahara and Xin Gao. Analysis of the effects of related fingerprints on molecular similarity using an eigenvalue entropy approach. *Journal of Cheminformatics*, 13:1–12, 2021.
- [41] Christopher A Lipinski, Franco Lombardo, Beryl W Dominy, and Paul J Feeney. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced drug delivery reviews*, 23(1-3):3–25, 1997.
- [42] Cheng-Hao Liu, Maksym Korablyov, Stanisław Jastrzebski, Paweł Włodarczyk-Pruszynski, Yoshua Bengio, and Marwin Segler. RetroGNN: fast estimation of synthesizability for virtual screening and de novo design by learning from slow retrosynthesis software. *Journal of Chemical Information and Modeling*, 62(10):2293–2300, 2022.

- [43] Daniel Mark Lowe. *Extraction of chemical structures and reactions from the literature*. PhD thesis, 2012.
- [44] Jiankun Lyu, Sheng Wang, Trent E. Balius, Isha Singh, Anat Levit, Yurii S. Moroz, Matthew J. O'Meara, Tao Che, Enkhjargal Algaa, Kateryna Tolmachova, Andrey A. Tolmachev, Brian K. Shoichet, Bryan L. Roth, and John J. Irwin. Ultra-large library docking for discovering new chemotypes. *Nature*, 566(7743):224–229, Feb 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-0917-9. URL <https://doi.org/10.1038/s41586-019-0917-9>.
- [45] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in GFlowNets. *Advances in Neural Information Processing Systems*, 35:5955–5967, 2022.
- [46] Łukasz Maziarka, Agnieszka Pocha, Jan Kaczmarczyk, Krzysztof Rataj, Tomasz Danel, and Michał Warchoń. Mol-CycleGAN: a generative model for molecular optimization. *Journal of Cheminformatics*, 12(1):1–18, 2020.
- [47] Kevin McCloskey, Eric A Sigel, Steven Kearnes, Ling Xue, Xia Tian, Dennis Moccia, Diana Gikunju, Sana Bazzaz, Betty Chan, Matthew A Clark, John W Cuozzo, Marie-Aude Gué, John P Guilinger, Christelle Huguet, Christopher D Hupp, Anthony D Keefe, Christopher J Mulhern, Ying Zhang, and Patrick Riley. Machine learning on DNA-encoded libraries: A new paradigm for hit finding. *J. Med. Chem.*, 63(16):8857–8866, August 2020.
- [48] Joshua Meyers, Benedek Fabian, and Nathan Brown. De novo molecular design and generative models. *Drug Discovery Today*, 26(11):2707–2715, 2021.
- [49] Dai Hai Nguyen and Koji Tsuda. Generating reaction trees with cascaded variational autoencoders. *The Journal of Chemical Physics*, 156(4), 2022.
- [50] Andrei Cristian Nica, Moksh Jain, Emmanuel Bengio, Cheng-Hao Liu, Maksym Korablyov, Michael M Bronstein, and Yoshua Bengio. Evaluating generalization in GFlowNets for molecule design. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.
- [51] Anthony Nicholls, Georgia B. McGaughey, Robert P. Sheridan, Andrew C. Good, Gregory Warren, Magali Mathieu, Steven W. Muchmore, Scott P. Brown, J. Andrew Grant, James A. Haigh, Neysa Nevins, Ajay N. Jain, and Brian Kelley. Molecular shape and medicinal chemistry: A perspective. *Journal of Medicinal Chemistry*, 53(10):3862–3886, 2010. doi: 10.1021/jm900818s. URL <https://doi.org/10.1021/jm900818s>. PMID: 20158188.
- [52] Pedro O O Pinheiro, Joshua Rackers, Joseph Kleinhenz, Michael Maser, Omar Mahmood, Andrew Watkins, Stephen Ra, Vishnu Sresht, and Saeed Saremi. 3D molecule generation by denoising voxel grids. *Advances in Neural Information Processing Systems*, 36, 2024.
- [53] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9:1–14, 2017.
- [54] Aryan Pedawi, Pawel Gniewek, Chaoyi Chang, Brandon Anderson, and Henry van den Bedem. An efficient graph generative model for navigating ultra-large combinatorial synthesis libraries. *Advances in Neural Information Processing Systems*, 35:8731–8745, 2022.
- [55] Bo Qiang, Yiran Zhou, Yuheng Ding, Ningfeng Liu, Song Song, Liangren Zhang, Bo Huang, and Zhenming Liu. Bridging the gap between chemical reaction pretraining and conditional molecule generation with a unified model. *Nature Machine Intelligence*, 5(12):1476–1485, 2023.
- [56] A. K. Rappe, C. J. Casewit, K. S. Colwell, W. A. III Goddard, and W. M. 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. doi: 10.1021/ja00051a040. URL <https://doi.org/10.1021/ja00051a040>.
- [57] Recursion. Recursion GFlowNet repository. <https://github.com/recursionpharma/gflownet>, 2024. [Online; Accessed 22 May 2024].

- [58] Sereina Riniker and Gregory A. Landrum. Better informed distance geometry: Using what we know to improve conformation generation. *Journal of Chemical Information and Modeling*, 55(12):2562–2574, 2015. doi: 10.1021/acs.jcim.5b00654. URL <https://doi.org/10.1021/acs.jcim.5b00654>. PMID: 26575315.
- [59] Julien Roy, Pierre-Luc Bacon, Christopher Pal, and Emmanuel Bengio. Goal-conditioned GFlowNets for controllable multi-objective molecular design. *arXiv preprint arXiv:2306.04620*, 2023.
- [60] Nicholas T Runcie and Antonia SJS Mey. SILVR: Guided diffusion for molecule generation. *Journal of Chemical Information and Modeling*, 63(19):5996–6005, 2023.
- [61] Arman A. Sadybekov, Anastasiia V. Sadybekov, Yongfeng Liu, Christos Iliopoulos-Tsoutsouvas, Xi-Ping Huang, Julie Pickett, Blake Houser, Nilkanth Patel, Ngan K. Tran, Fei Tong, Nikolai Zvonok, Manish K. Jain, Olena Savych, Dmytro S. Radchenko, Spyros P. Nikas, Nicos A. Petasis, Yurii S. Moroz, Bryan L. Roth, Alexandros Makriyannis, and Vsevolod Katritch. Synthon-based ligand discovery in virtual libraries of over 11 billion compounds. *Nature*, 601(7893):452–459, Jan 2022. ISSN 1476-4687. doi: 10.1038/s41586-021-04220-9. URL <https://doi.org/10.1038/s41586-021-04220-9>.
- [62] Gabriele Scalia, Steven T Rutherford, Ziqing Lu, Kerry R Buchholz, Nicholas Skelton, Kangway Chuang, Nathaniel Diamant, Jan-Christian Huetter, Jerome Luescher, Ahn Miu, et al. A high-throughput phenotypic screen combined with an ultra-large-scale deep learning-based virtual screening reveals novel scaffolds of antibacterial compounds. *bioRxiv*, pages 2024–09, 2024.
- [63] Ruth R Shah and Adrian P Bird. MeCP2 mutations: progress towards understanding and treating rett syndrome. *Genome Med.*, 9(1):17, February 2017.
- [64] Tony Shen, Mohit Pandey, and Martin Ester. TacoGFN: Target conditioned gflownet for structure-based drug design. *arXiv preprint arXiv:2310.03223*, 2023.
- [65] Vanessa Smer-Barreto, Andrea Quintanilla, Richard JR Elliott, John C Dawson, Jiugeng Sun, Víctor M Campa, Álvaro Lorente-Macías, Asier Unciti-Broceta, Neil O Carragher, Juan Carlos Acosta, et al. Discovery of senolytics using machine learning. *Nature communications*, 14(1):3445, 2023.
- [66] Teague Sterling and John J Irwin. ZINC 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11):2324–2337, 2015.
- [67] Kyle Swanson, Gary Liu, Denise B Catacutan, Autumn Arnold, James Zou, and Jonathan M Stokes. Generative AI for designing and validating easily synthesizable and structurally novel antibiotics. *Nature Machine Intelligence*, 6(3):338–353, 2024.
- [68] Shidi Tang, Ji Ding, Xiangyu Zhu, Zheng Wang, Haitao Zhao, and Jiansheng Wu. Vina-GPU 2.1: towards further optimizing docking speed and precision of AutoDock Vina and its derivatives. *bioRxiv*, 2023. doi: 10.1101/2023.11.04.565429. URL <https://www.biorxiv.org/content/early/2023/11/05/2023.11.04.565429>.
- [69] Paul C. Trippier and Christopher McGuigan. Boronic acids in medicinal chemistry: anticancer, antibacterial and antiviral applications. *Med. Chem. Commun.*, 1(3):183–198, 2010.
- [70] Oleg Trott and Arthur J Olson. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *J. Comput. Chem.*, 31(2):455–461, January 2010.
- [71] Tiziano Tuccinardi, Giulio Poli, Veronica Romboli, Antonio Giordano, and Adriano Martinelli. Extensive consensus docking evaluation for ligand pose prediction and virtual screening studies. *Journal of Chemical Information and Modeling*, 54(10):2980–2986, 2014. doi: 10.1021/ci500424n. URL <https://doi.org/10.1021/ci500424n>. PMID: 25211541.
- [72] Alexandra Volokhova, Michał Koziarski, Alex Hernández-García, Cheng-Hao Liu, Santiago Miret, Pablo Lemos, Luca Thiede, Zichao Yan, Alán Aspuru-Guzik, and Yoshua Bengio. Towards equilibrium molecular conformation generation with GFlowNets. *Digital Discovery*, 2024.

- [73] Zhe Wang, Huiyong Sun, Xiaojun Yao, Dan Li, Lei Xu, Youyong Li, Sheng Tian, and Tingjun Hou. Comprehensive evaluation of ten docking programs on a diverse set of protein–ligand complexes: the prediction accuracy of sampling power and scoring power. *Phys. Chem. Chem. Phys.*, 18:12964–12975, 2016. doi: 10.1039/C6CP01555G. URL <http://dx.doi.org/10.1039/C6CP01555G>.
- [74] Felix Wong, Satotaka Omori, Nina M Donghia, Erica J Zheng, and James J Collins. Discovering small-molecule senolytics with deep neural networks. *Nature Aging*, 3(6):734–750, 2023.
- [75] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.
- [76] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, 2018.
- [77] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. *Advances in neural information processing systems*, 32, 2019.
- [78] Xin-Min Zhang, Qing Chang, Lin Zeng, Judy Gu, Stuart Brown, and Ross S Basch. TBLR1 regulates the expression of nuclear hormone receptor co-repressors. *BMC Cell Biol.*, 7(1):31, August 2006.

A State space size estimation

We estimate the state space size by first sampling 1,000 random trajectories, masking out the end-of-sequence action unless the maximum trajectory length max is reached. Then, for every i -th reaction or fragment in the trajectory, we count the average number of valid fragments $frag_i$ and reactions $react_i$ from a given state in the trajectory, as well as the average number of unique trajectories $traj_i$ into which a state can be decomposed using the backward policy. We estimate the state space size as

$$\frac{(\prod_{i=0}^{max} frag_i)(\prod_{i=1}^{max} react_i)}{traj_{max}}. \quad (9)$$

Experimentally derived average values of these parameters can be found in Table 2. Note that in the second setting we randomly picked 8,000 fragments from the Enamine stock (with the same balancing procedure as in Section 4.3), which after merging with our own fragments, canonization and duplicate removal yielded a total of 8,317 fragments.

Table 2: Experimentally derived average values of valid fragments, valid reactions, and possible trajectories.

# reactions	350 fragments			8350 fragments		
	$frag_i$	$react_i$	$traj_i$	$frag_i$	$react_i$	$traj_i$
0	350.0	-	1.0	8317.0	-	1.0
1	37.5	11.8	3.5	835.8	12.0	4.2
2	39.9	16.5	16.8	822.1	15.9	17.0
3	40.7	15.4	76.7	832.6	17.0	75.2
4	40.0	15.8	349.3	814.0	18.0	480.8
5	42.1	16.8	1825.8	857.6	18.9	3058.1

B Training details

B.1 Proxy models

The sEH proxy is described in [4]. It is an MPNN trained on a normalized docking score data. We utilize the exact same model checkpoint as provided in [57].

The senolytic classification model is a graph neural network trained on the biological activity classification task of senolytic recognition [74]. Specifically, it was trained on two combined, publicly available senolytic datasets [74, 65]. Reward is given by the predicted probability of a compound being a senolytic. It is worth noting that due to the low amount of data and high imbalance (< 100 active compounds, a high proportion of which contained macrocycles and were infeasible to construct with fragment-based generative models), this is expected to be a difficult task with sparse reward.

The senolytic proxy model is GNEprop [62], which consisted of 5 GIN layers [75] with hidden dimensionality of 500, utilized Jumping Knowledge shortcuts [76], and had a single output MLP layer. Pretraining was done in an unsupervised fashion on the ZINC15 dataset [66]. The training was done for 30 epochs using the Adam optimizer with a learning rate of 5×10^{-5} and batch size of 50.

The DRD2 proxy is described in [53, 28]. It is a support vector machine classifier with a Gaussian kernel using ECFP6 fingerprints as a feature representation.

B.2 Generative models

Both RGFN and FGFN were trained with trajectory balance loss [45] using Adam optimizer with a learning rate of 1×10^{-3} , logZ learning rate of 1×10^{-1} , and batch size of 100. The training lasted 4,000 steps. A random action probability of 0.05 was used, and RGFN used a replay buffer of 20 samples per batch. Both methods use a graph transformer policy with 5 layers, 4 heads, and 64 hidden dimensions. Exponentiated reward $R(x) = \exp(\beta * score(x))$ was used, with β dependent on the task: 8 for sEH proxy, 0.5 for senolytic proxy, 48 for DRD2 proxy, and 4 for all docking runs. Note

that due to different ranges of score values, this resulted in a roughly comparable range of reward values.

For FGFN+SA, we used a modified reward function $R(x) = \exp(\beta * (0.5 * proxy(x) / max_proxy + (10 - SA_score) / 10))$, with β adjusted per proxy to match the original reward range.

All sampling algorithms were outfitted with the Vina GPU-2.1 docking, senolytic proxy, sEH proxy, and DRD2 proxy scoring functions. While model architecture hyperparameters and batch sizes were kept consistent between FGFN and RGFN, we allowed FGFN a maximum fragment count of 6 as opposed to RGFN's 5 due to RGFN's larger average building block sizes.

GuacaMol's Graph GA model was trained with a population size of 100, offspring size of 200, and a mutation rate of 0.01 for 2000 generations for a total of 400,000 visited molecules.

For SyntheMol experiments, we used the default building block library of 132,479 compatible molecules and pre-computed docking, senolytic, and sEH proxy scores for all prior to executing rollouts to follow the established methodology. Due to CPU constraints, sampling 500,000 molecules with SyntheMol was impractical. Instead, we executed 100,000 rollouts over approximately 72 hours to match the RGFN training time with docking, yielding 111,964 unique molecules. Additionally, we performed 50,000 rollouts each (approximately 24 hours) for sEH, senolytic, and DRD2 proxies, resulting in 73,941, 69,652, and 62,320 unique molecules, respectively.

casVAE was trained with Bayesian optimization (BO) using default parameters consisting of a hidden size of 200, latent size of 50, and message passing depth of 2. Again, due to time constraints imposed by BO latent space updates, we instead opted to approximately match RGFN training times, training for 72, 24, 24, and 24 hours on the docking, sEH, senolytic, and DRD2 tasks respectively for a total of 135, 41, 38, and 40 rollouts and 7708, 2097, 1521, and 2165 total generated molecules, respectively.

C GPU-accelerated docking

Our docking oracle first accepts canonized SMILES strings as input. These are then converted to RDKit Molecules, protonated, and a low-energy conformer is generated and minimized with the ETKDG [58] conformer generation method and UFF force field [56], respectively. For computational efficiency, we generate one initial conformer per ligand. Each conformer is converted to a pdbqt file and docked against a target with Vina-GPU 2.1 using model defaults: exhaustiveness (denoted by "thread" in the implementation) of 8000 and a heuristically determined search depth d given by

$$d = \max(1, \lfloor 0.36 \times N_{atom} + 0.44 \times N_{rot} - 5.11 \rfloor), \quad (10)$$

where N_{atom} and N_{rot} are the number of atoms and the number of rotatable bonds, respectively, in the generated molecule. Box sizes were determined individually to encompass each target binding site and centroids were calculated to be the average position of ligand atoms in the receptor template PDB structure file. A negative score is calculated and returned as a reward.

C.1 Target preprocessing

Each target was prepared by removing its complexed inhibitor and atoms of other solvent or solute molecules. We selectively prepared the ClpP 7UVU protein structure by retaining only two monomeric units to ensure the presence of a single active site available for ligand binding and similarly prepared the Mpro 6W63 protein structure by retaining only one monomeric unit.

C.2 Boron substitution for docking

Due to the lack of force field parameters for boron atoms, QuickVina2 is unable to process ligands with boronic acid or ester groups. Therefore, we chose to substitute the boronic acid group and its derivatives in the building blocks/reactants with a carboxylic acid, where the carbonyl carbon is a C13 isotope. This was done for two reasons: firstly, carboxylic acids are considered to be bioisosteric to boronic acids[69], and secondly, the C¹³ tag would allow us to distinguish actual carboxylic acids from boronic acid surrogates for SMARTS encoding purposes. This allowed us to dock all possible final products while maintaining structural similarity to the original group and specificity to compatible reaction SMARTS.

D Computational resources used

Evaluating fragment scaling took approximately 800 GPU hours on GeForce RTX 4090 in total. Remaining experiments took roughly 24 GPU hours per run on Quadro RTX 8000 for sEH, DRD2 and senolytic proxies, and roughly 72 GPU hours per run on an A100 for docking-based proxies.

E Implicit reactions

In this work, we define implicit reactions as SMARTS-encoded reactions, products of which contain atoms that are not included in the reactants and come from "implicit reagents". One example is urea synthesis using carbonyl surrogates: the SMARTS template uses two amines, but the product has more atoms than specified in the reactants. These come from the "implicit reagent", in our case — any phosgene surrogate (e.g., CDI) (Figure 8).

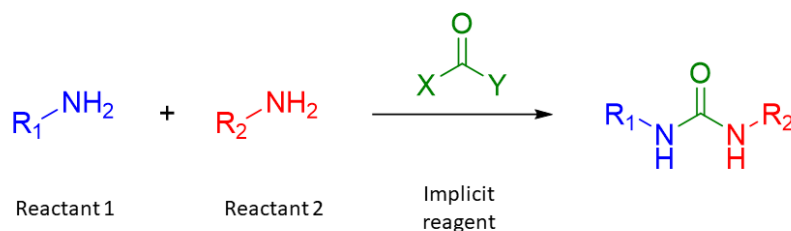


Figure 8: An example of an implicit reaction: urea synthesis reaction encoded in SMARTS.

In addition to urea formation, other implicit reactions encoded in SMARTS include azide-alkyne cycloaddition, azide-nitrile cycloadditions, and tetrazole synthesis using peptide terminal thiourea cyclizations.

F Sampling large fragment set from Enamine building blocks

In order to evaluate our approach at scale, our building block set had to be balanced to prevent introducing learning biases via building block selection. This was achieved by grouping reagents into twelve fragment classes, for which a specific weight was assigned based on the frequency of appearance of these structures in different reactions. For example, a carboxylic acid appearing in two distinct reactions was assigned a weight coefficient of two, which would later be used to calculate the normalized amount of building blocks per group for a specific database size using the following formula:

$$\text{Amount of BBs} = \frac{\text{Weight coef.}}{\text{Sum of weights}} * \text{database size} \quad (11)$$

The only exception is Michael acceptors, or group 3, which has a weight of 0.25 due to the low availability of these reagents in most public molecular databases and possible overlap with structures that might fit the corresponding SMARTS code (for example, 2-cyano or 2-carbonyl aromatic compounds).

The final set of fragments was constructed by randomly sampling fragments from publicly available Enamine building blocks in a way that roughly preserved the above grouping. This was done in a greedy round-robin fashion, randomly selecting one fragment at a time from the remaining fragments belonging to a given group, and iterating through all groups until the specified number of fragments was selected.

G Target selection

Our targets (sEH, Mpro, ClpP, and TBLR1) were chosen with diverse functions and binding sites in mind as test cases for RGFN. ClpP is a highly conserved compartmentalized protease that degrades

substrates in a signal-dependent fashion and is a promising target in cancer and infectious disease [6]. A number of structurally unrelated small molecules that allosterically activate ClpP, thereby deregulating its protease activity, inhibit the growth of cancer cells and bacterial pathogens [31, 25, 30].

The main protease of SARS-CoV-2, Mpro (also called 3CL_{pro}), is required for processing of the virally-encoded polyprotein and thus for viral replication, and is a well-validated and intensively studied anti-SARS-CoV-2 target [14]. The Mpro catalytic pocket contains 4 distinct sub-pockets that recognize amino acid motifs in substrate sites and is a challenging target due to its structural plasticity [36].

Soluble epoxide hydrolase (sEH) catalyzes a key step in the biosynthesis of eicosanoid inflammation mediators and is being pursued as potential target in cardiovascular and other diseases [16]. sEH has been used as a benchmark substrate for machine learning-based drug discovery and is particularly amenable to computational methods because of its deep hydrophobic pocket [47].

Finally, TBL1 and its paralog TBLR1 are components of the NCoR transcriptional repressor complex and contain WD40 domains that participate in protein interactions [78]. TBL1/TBLR1 mediates the transcriptional repression function of the MeCP2 methylCpG-binding protein, which is mutated in the neurodevelopmental disorder Rett syndrome. Loss of function mutations in Rett syndrome frequently disrupt the interaction of a disordered region in MeCP2 that binds to the WD40 domain of TBL1/TBLR1; conversely, MeCP2 gain-of-function by copy number mediated overexpression leads to X-linked intellectual disability [63].

Well-validated experimental assays are available for each of these 4 different targets [31, 14, 47, 1].

H Ligand post-processing

To ensure the diversity, specificity, and conformer validity of top-generated molecules for each target, we initially categorized our molecules into distinct modes, each representing any SMILES string with a Tanimoto similarity of 0.5 or lower with all other modes. Subsequently, we selected the top 100 modes based on their Vina-GPU 2.1 scores and filtered their docked poses using PoseBusters [11] in "mol" mode, where any pose failing any PoseBusters check was excluded from consideration. As a final precaution, we selected only modes with Tanimoto coefficients to known aggregators of 0.4 or lower using UCSF's Aggregation Advisor [29] dataset. This process resulted in 35, 68, 31, and 15 top modes for sEH, ClpP, Mpro, and TBLR1 binders, respectively Appendix J. Comparative analyses of docked top RGFN modes and confirmed sEH, ClpP, and Mpro ligand poses can be found in Appendices K to M.

I Posebusters, Aggregation Advisor analysis of top 100 modes

Table 3: Proportion of top-100 generated sEH, ClpP, Mpro, and TBLR1 modes satisfying each of PoseBuster's Mol-mode checks, as well as the Aggregation Advisor Tanimoto similarity threshold of 0.4. Molecules were assessed in PoseBuster according to their ability to be loaded and sanitized in RDKit, reasonableness of bond lengths and angles, lack of steric clashes in the pose, aromatic ring and double bond flatness, and internal energy.

Target	Condition									
	Load	Sanitize	Connected	Bond Lengths	Bond Angles	Steric Clashes	Aromatic Ring Flatness	Double Bond Flatness	Internal Energy	Agg. Sim. < 0.4
sEH	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.9	0.38
ClpP	1.0	1.0	1.0	1.0	1.0	0.98	1.0	1.0	0.94	0.72
Mpro	1.0	1.0	1.0	1.0	1.0	0.89	1.0	1.0	0.62	0.65
TBLR1	1.0	1.0	1.0	1.0	1.0	0.96	1.0	1.0	0.34	0.53

J Top filtered molecules for all targets

Senolytics

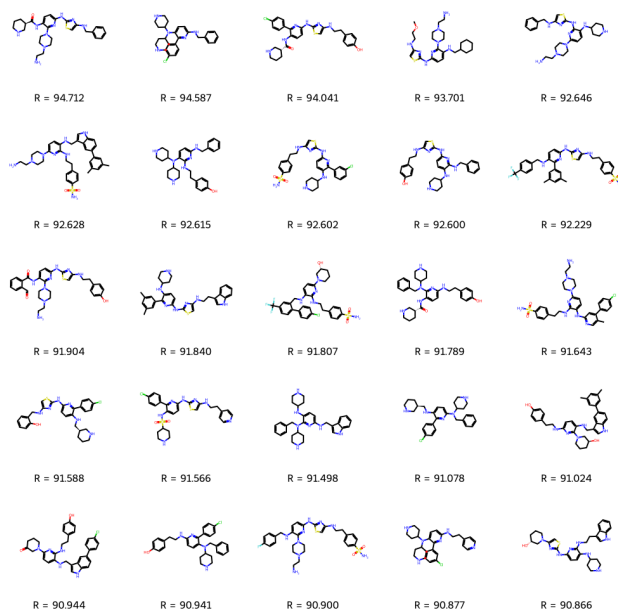


Figure 9: Top 25 senolytic compound modes generated by RGFN.

sEH

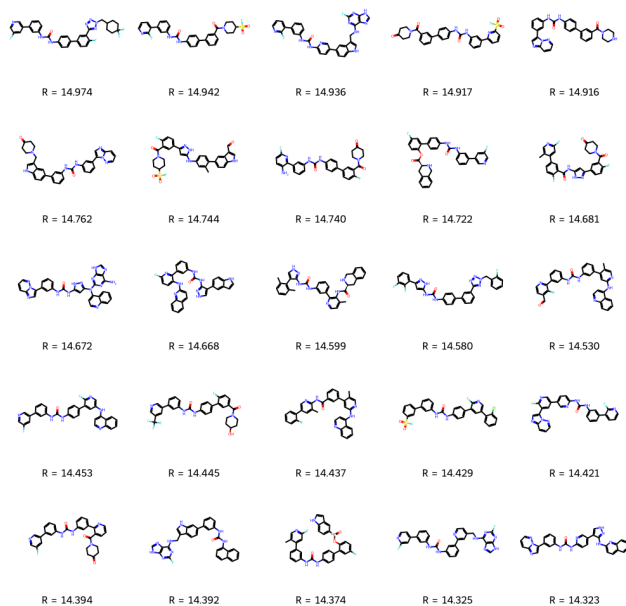


Figure 10: Top 25 filtered binders to sEH drawn from top 100 RGFN modes.

ClpP

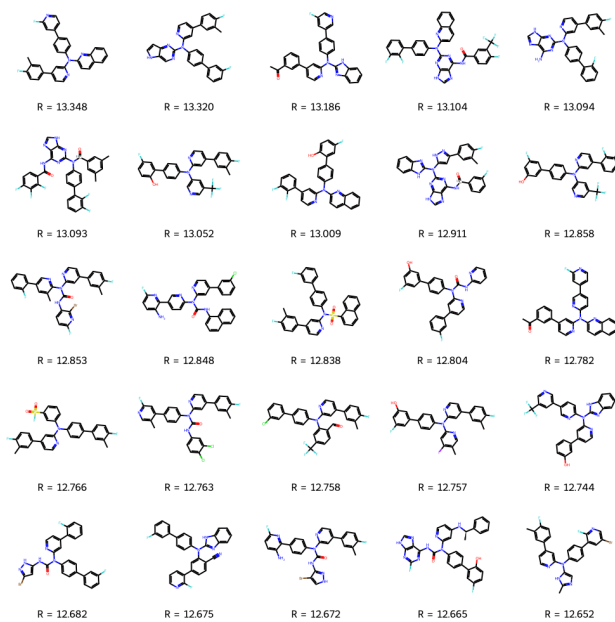


Figure 11: Top 25 filtered binders to ClpP drawn from top 100 RGFN modes.

Mpro

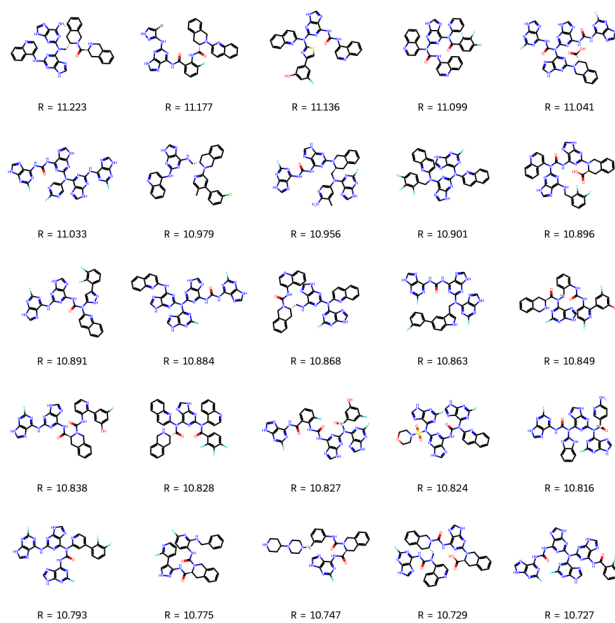


Figure 12: Top 25 filtered binders to Mpro drawn from top 100 RGFN modes.

TBLR1

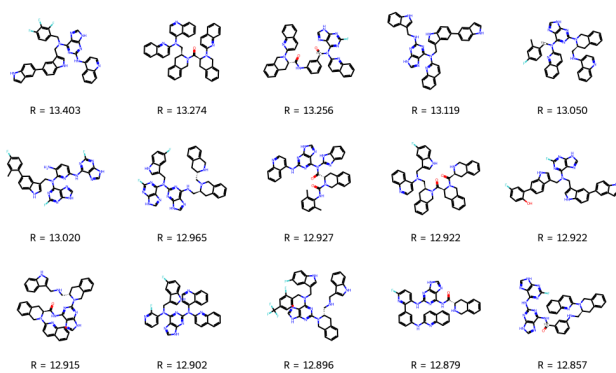
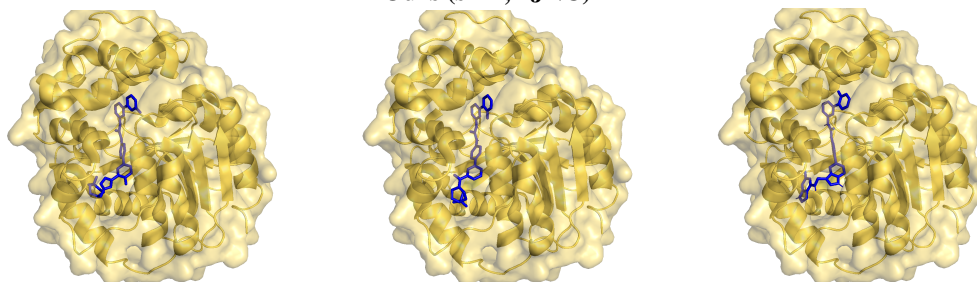


Figure 13: All 15 filtered binders to TBLR1 drawn from top 100 RGFN modes.

K Docked poses of top generated molecules

Ours (sEH, 4JNC)

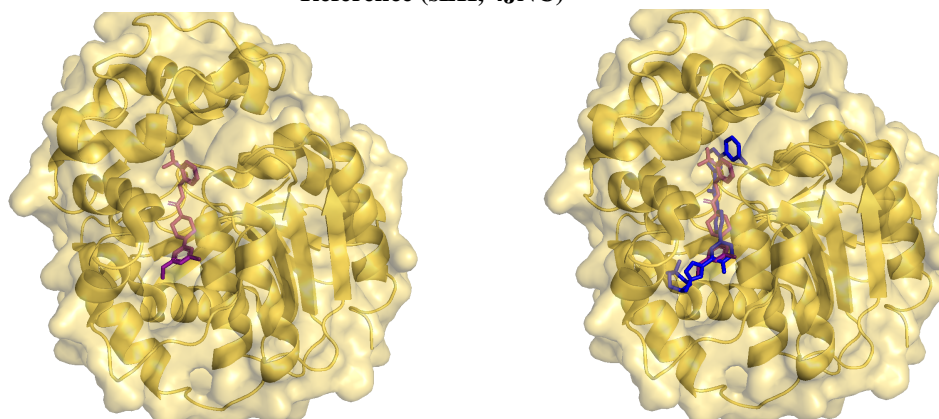


Vina-GPU 2.1 Score: -14.97

Vina-GPU 2.1 Score: -14.94

Vina-GPU 2.1 Score: -14.94

Reference (sEH, 4JNC)

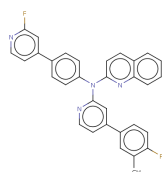
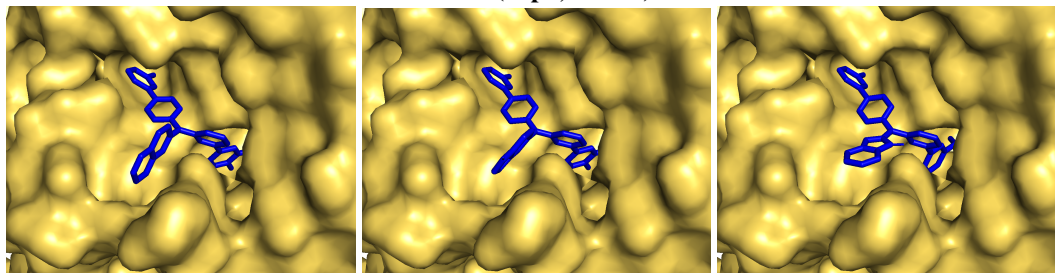


Vina-GPU 2.1 Score: -11.13

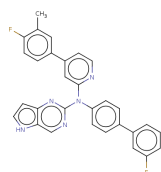
Vina-GPU 2.1 Score: -14.97

Figure 14: Top left to right: Top 3 generated ligand scaffolds for sEH (blue). Bottom left: Reference ligand pose (purple, PDB ID: 1LF). Bottom right: Reference ligand (purple) overlaid with top-scoring ligand (blue).

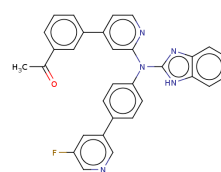
Ours (ClpP, 7UVU)



Vina-GPU 2.1 Score: -13.35

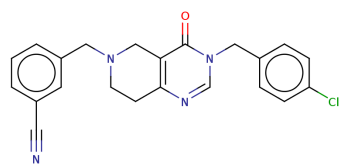
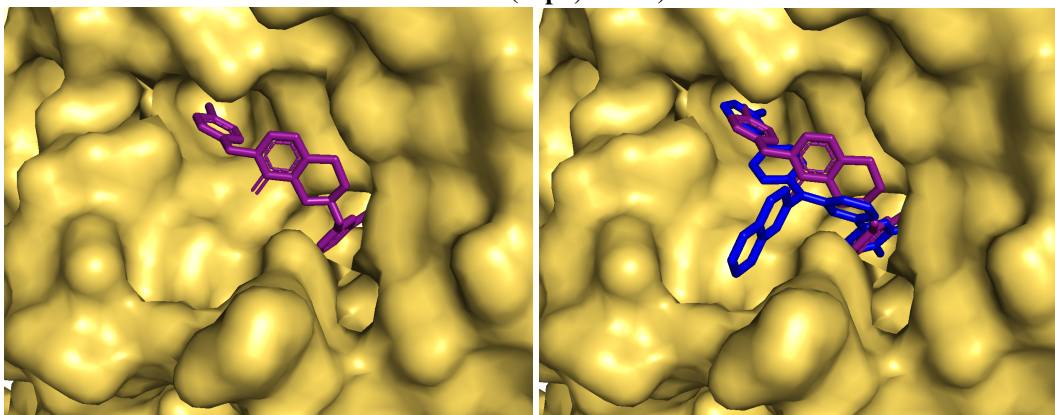


Vina-GPU 2.1 Score: -13.32

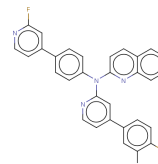


Vina-GPU 2.1 Score: -13.19

Reference (ClpP, 7UVU)



Vina-GPU 2.1 Score: -10.31



Vina-GPU 2.1 Score: -13.35

Figure 15: Top left to right: Top 3 generated ligand scaffolds for ClpP (blue). Bottom left: Reference ligand pose (purple, PDB ID: OY9). Bottom right: Reference ligand (purple) overlaid with top-scoring ligand (blue).

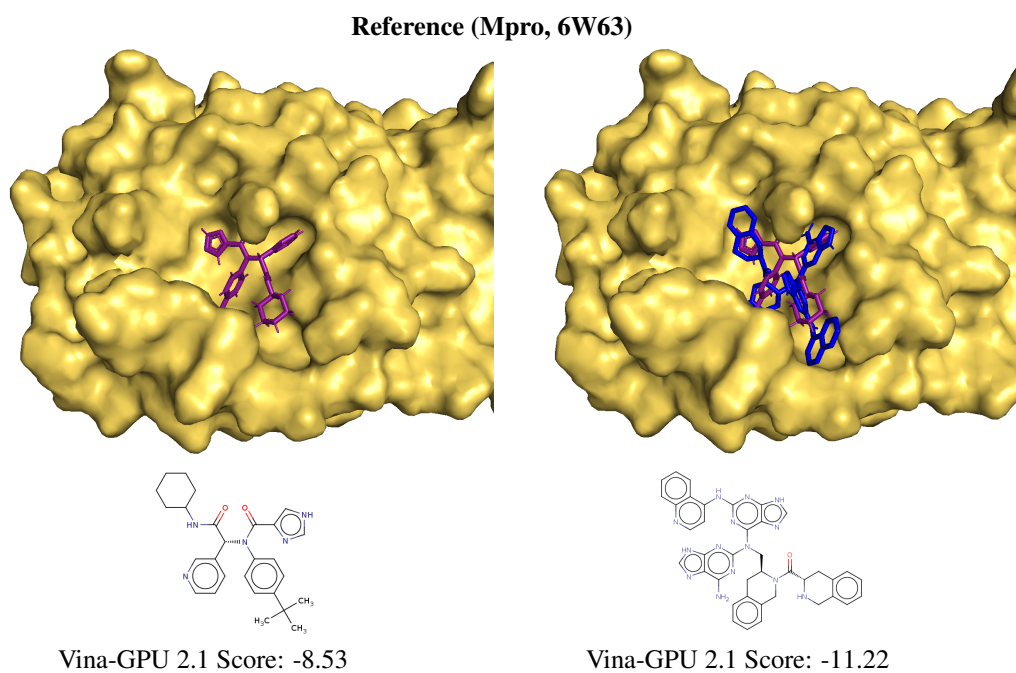
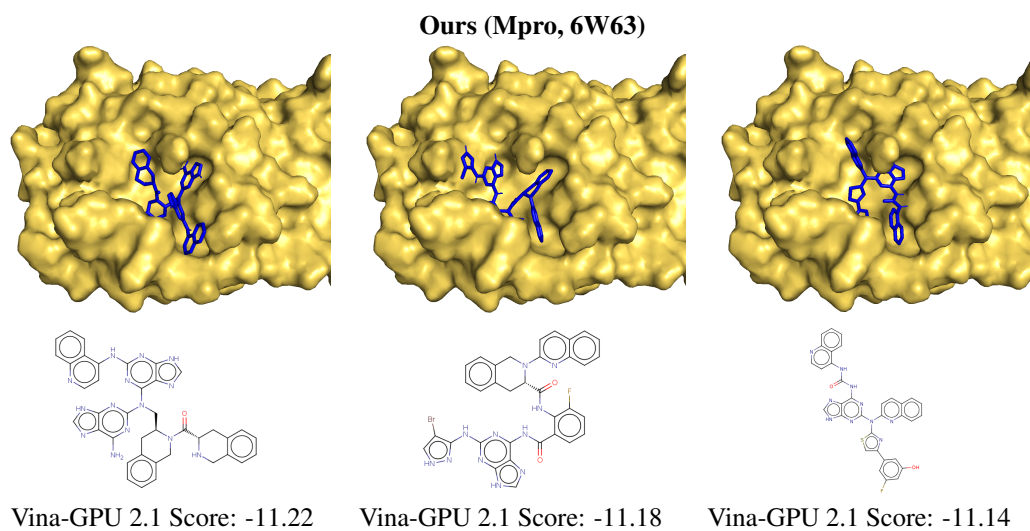


Figure 16: Top left to right: Top 3 generated ligand scaffolds for Mpro (blue). Bottom left: Reference ligand pose (purple, PDB ID: X77). Bottom right: Reference ligand (purple) overlaid with top-scoring ligand (blue).

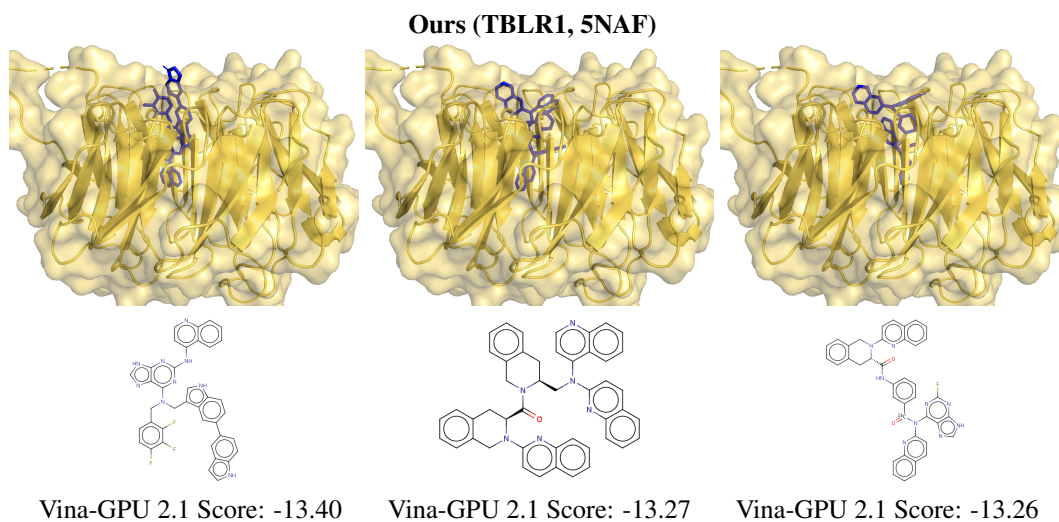


Figure 17: Left to right: Top 3 generated ligand scaffolds for the TBRL1 WD40 domain. TBRL1 has no known small molecule ligands.

L Docking analysis of known ligands

Table 4: Vina-GPU 2.1 docking scores of 10 PDB-available ligands per target.

Ligand	sEH		ClpP		Mpro	
	PDB ID	Score	PDB ID	Score	PDB ID	Score
1	5IV	-11.79	OX0	-11.10	J7O	-10.13
2	XDZ	-11.64	PJF	-10.67	KAE	-9.45
3	WJ5	-11.30	P4I	-10.40	7YY	-9.33
4	E3N	-11.14	P3O	-10.39	7XB	-8.6
5	1LF	-11.13	OY9	-10.31	XYV	-8.6
6	8S9	-11.07	ZLL	-10.18	X77	-8.54
7	TK9	-9.64	7SR	-10.17	XF1	-8.40
8	G3W	-9.60	OSR	-9.49	0EN	-8.19
9	G3Q	-9.14	ONC	-9.47	J7R	-8.00
10	J0U	-8.75	9DF	-9.39	4N0	-7.33

We calculate and report the Vina-GPU 2.1 docking scores of 10 true ligands per target assessed in the paper. Each PDB was prepared as described in Appendix C.1.

M Tanimoto similarity to known ligands

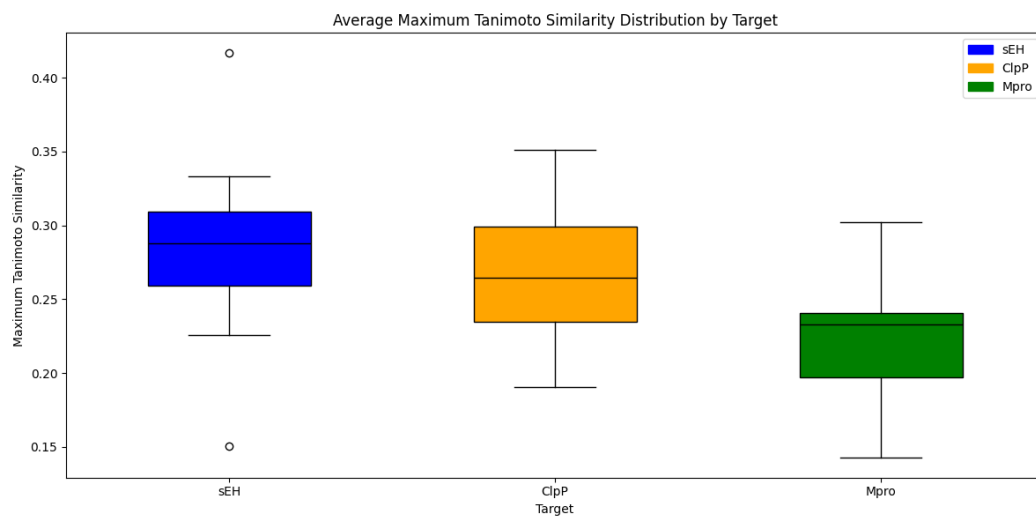


Figure 18: Average Maximum Tanimoto Similarity to known ligands. For each target, we plot the highest Tanimoto similarity score across all generated molecules to the ten corresponding PDB-derived ligands.

N Ligands cost analysis

To demonstrate the synthesizability and cost-effectiveness of the ligands produced by RGFN, we conducted a comprehensive cost analysis and proposed a synthesis plan for the top 10 scored ClpP hits generated by RGFN and SyntheMol (see Appendix O). We compared these two methods due to their similarity in chemical approach. For the cost calculation, we only considered the price of building blocks, which were directly sourced from EnamineStore for the case of Synthemol ligands. All prices are represented in US dollars per 0.1 mmol of product (Figures 19 and 20). Notably, for very common reagents only available in bulk, costs per gram were estimated by taking the cheapest or smallest available alternative and dividing its cost by its mass. All prices for individual building blocks were sourced from their respective vendors, which included Millipore Sigma, Oakwood Chemicals, Combi-Blocks, TCI, and AngeneSci.

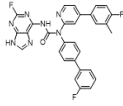
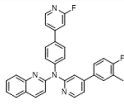
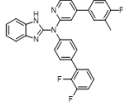
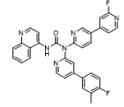
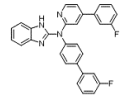
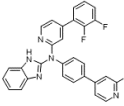
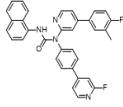
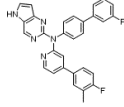
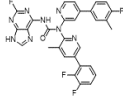
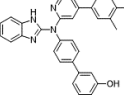
Position	Ligand	Cost per 0,1 mmol, \$	Position	Ligand	Cost per 0,1 mmol, \$
1		2.07	6		1.72
2		1.90	7		3.93
3		1.76	8		1.82
4		1.37	9		2.35
5		1.80	10		1.84

Figure 19: Synthesis cost for top-10 scoring ClpP ligands produced by RGFN.

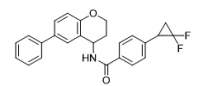
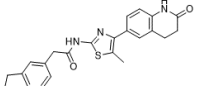
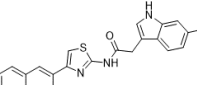
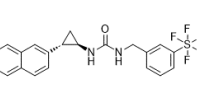
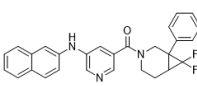
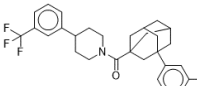
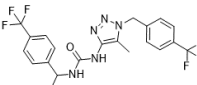
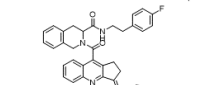
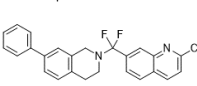
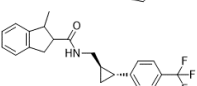
Position	Ligand	Cost per 0,1 mmol, \$	Position	Ligand	Cost per 0,1 mmol, \$
1		185.79	6		112.84
2		17.68	7		263.95
3		260.33	8		78.62
4		233.44	9		35.44
5		N/A	10		185.02

Figure 20: Synthesis cost for top-10 scoring ClpP ligands produced by Synthemol.

As expected, ligands generated by RGFN are significantly cheaper to produce compared to SyntheMol, despite having more synthesis steps and lower overall theoretical yields, with an average of 55-70% after 4 steps in the case of RGFN compared to 90-95% average yield after 1 step for SyntheMol. Noticeably, SyntheMol ligand 5 wasn't found to be synthesizable by using reactions proposed in their work. After conducting further analysis, it was found that such a compound could have been produced by Reaction 8 via nucleophilic substitution of the fluorine atom in the trifluoromethyl group; however, this particular reaction is not likely to be performed with sufficient yields, and therefore such a product cannot be considered synthesizable.

O Examples of plausible synthetic routes to molecules

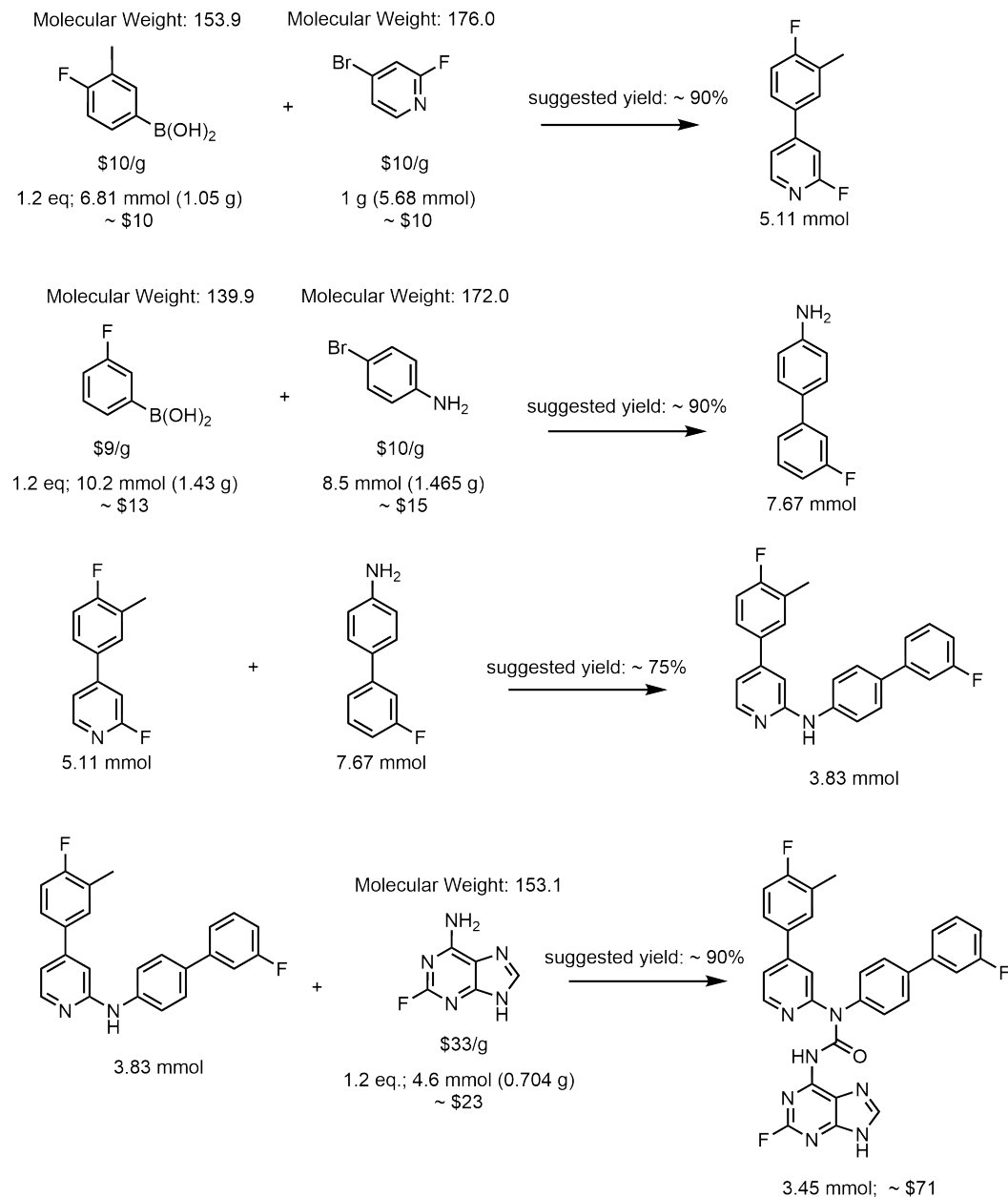


Figure 21: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 1.

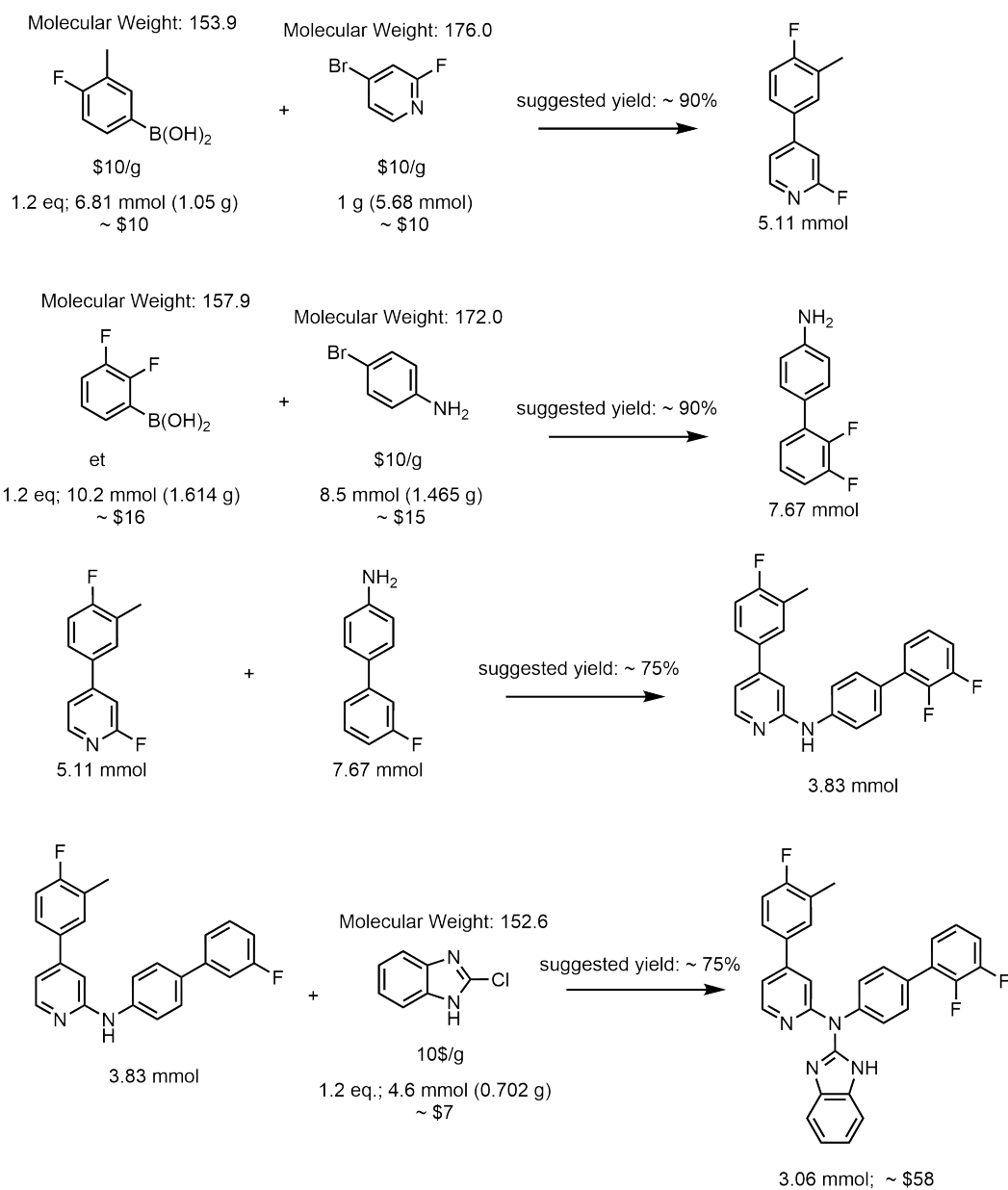


Figure 22: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 2.

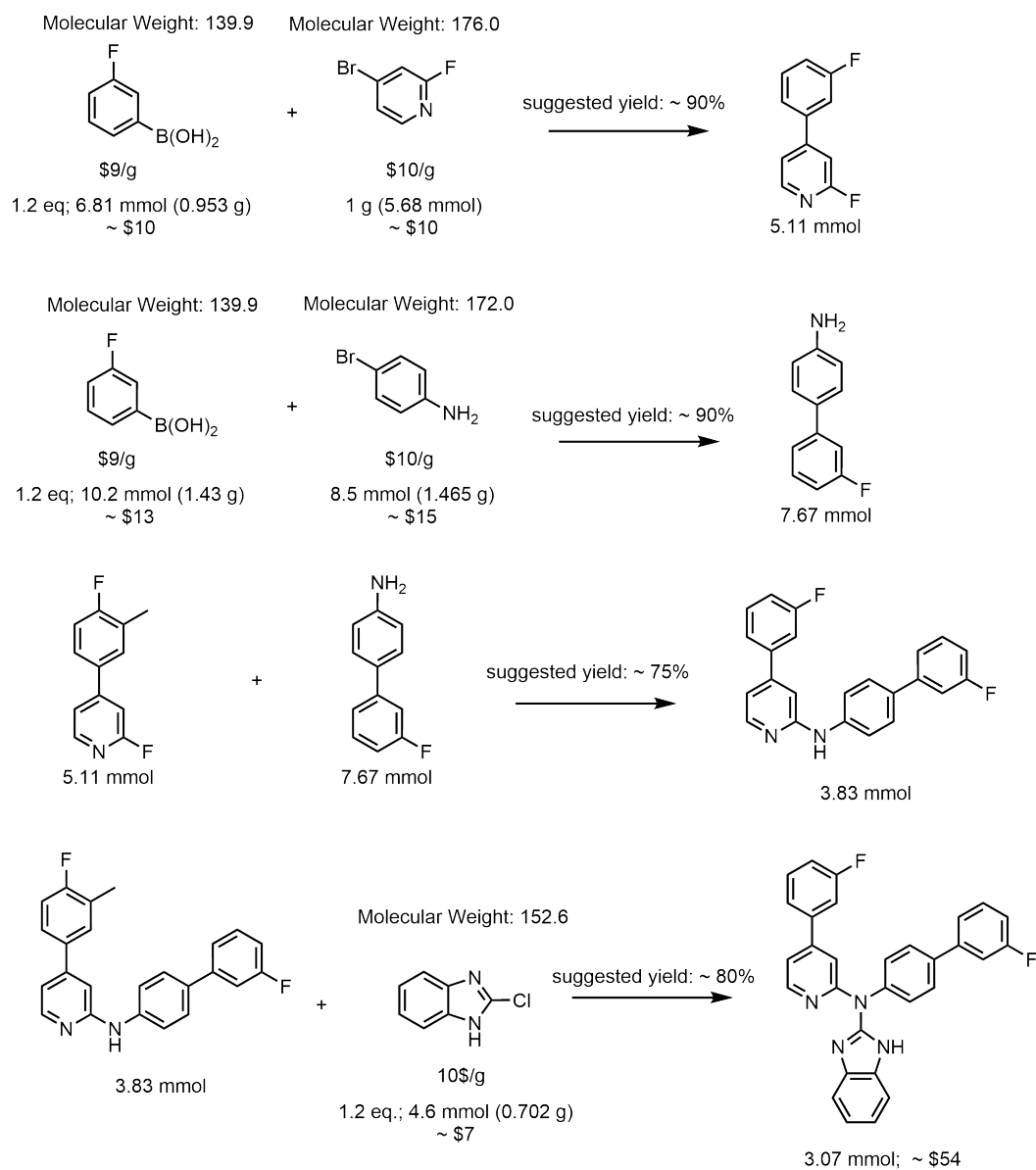


Figure 23: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 3.

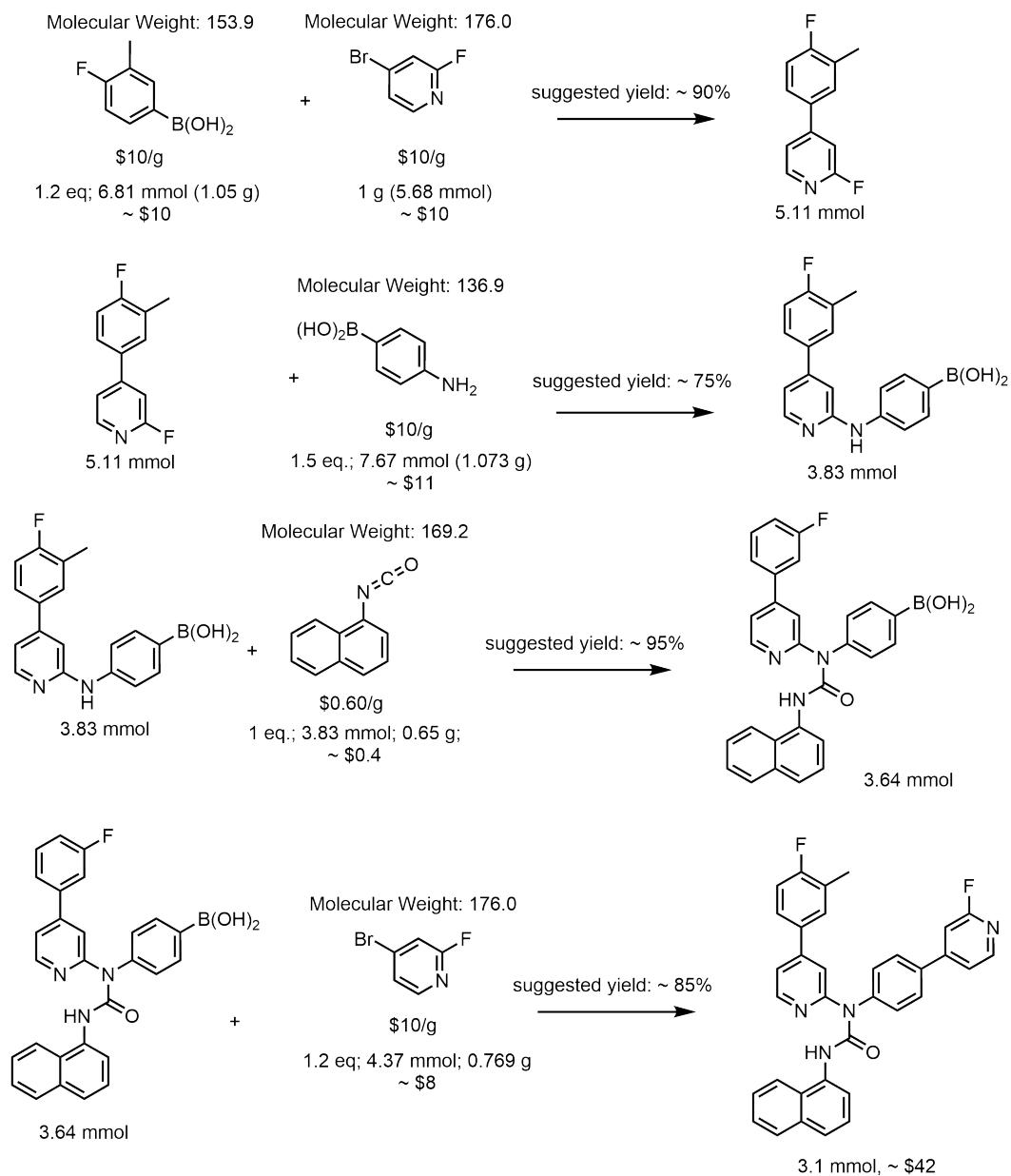


Figure 24: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 4.

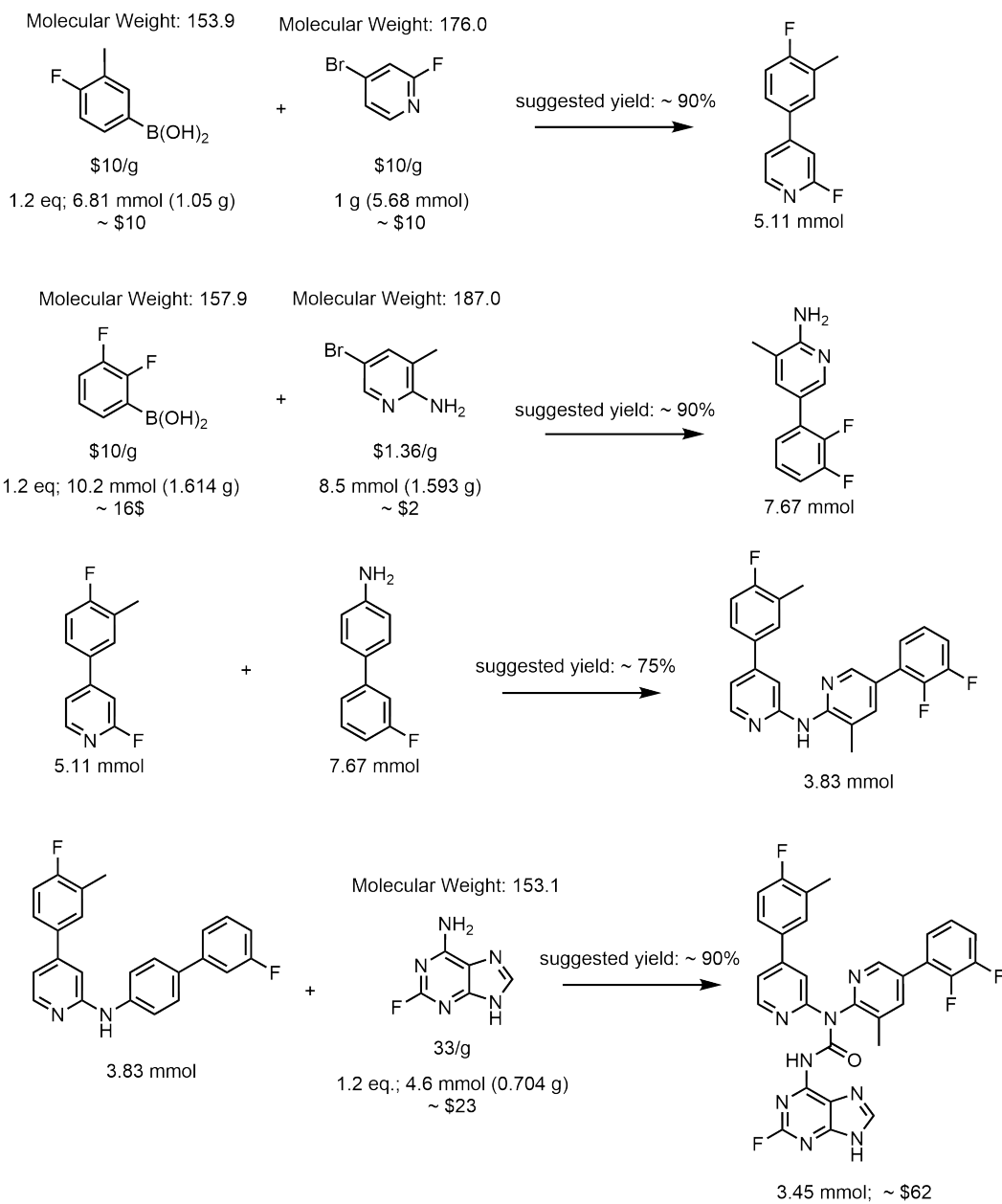


Figure 25: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 5.

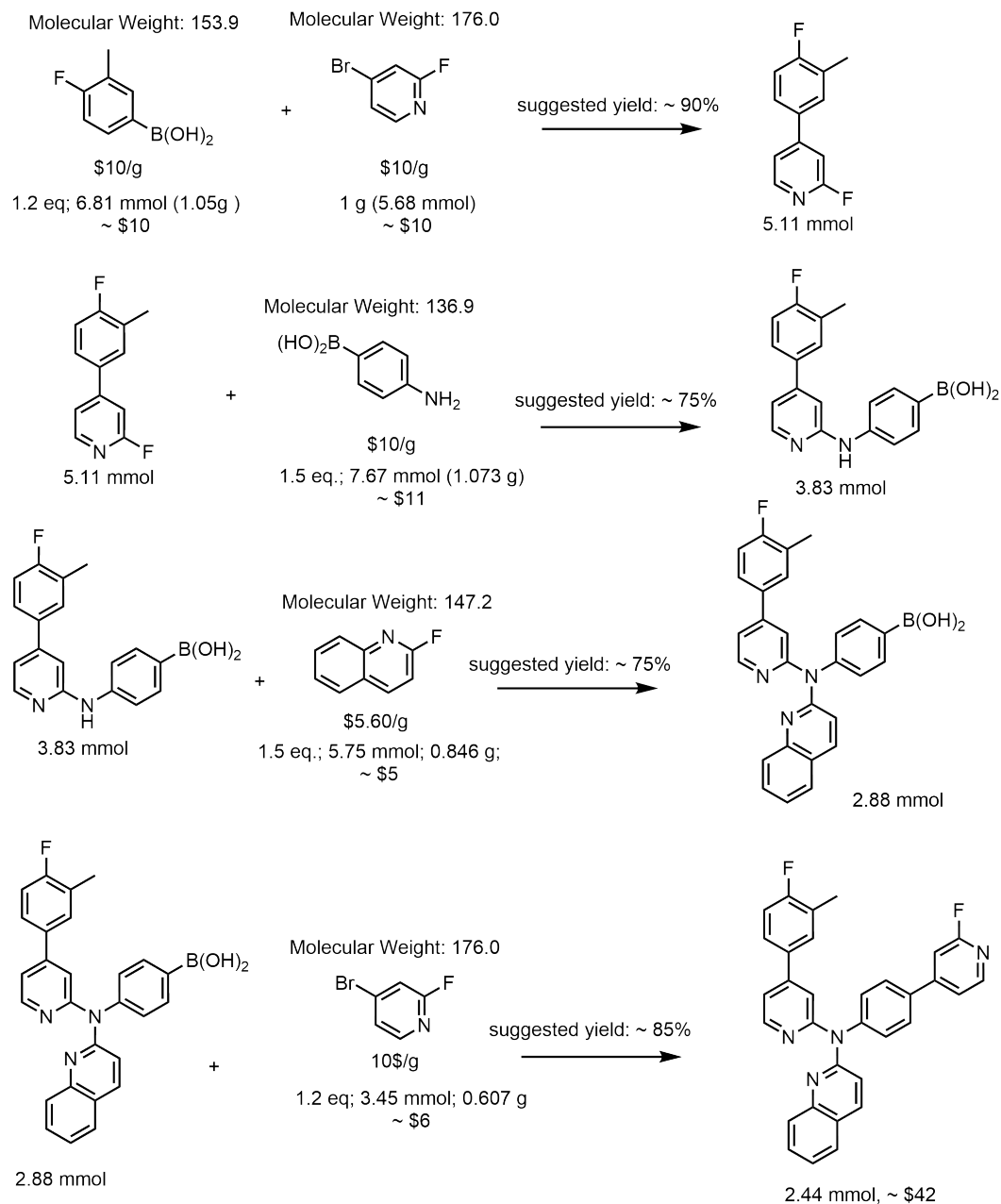


Figure 26: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 6.

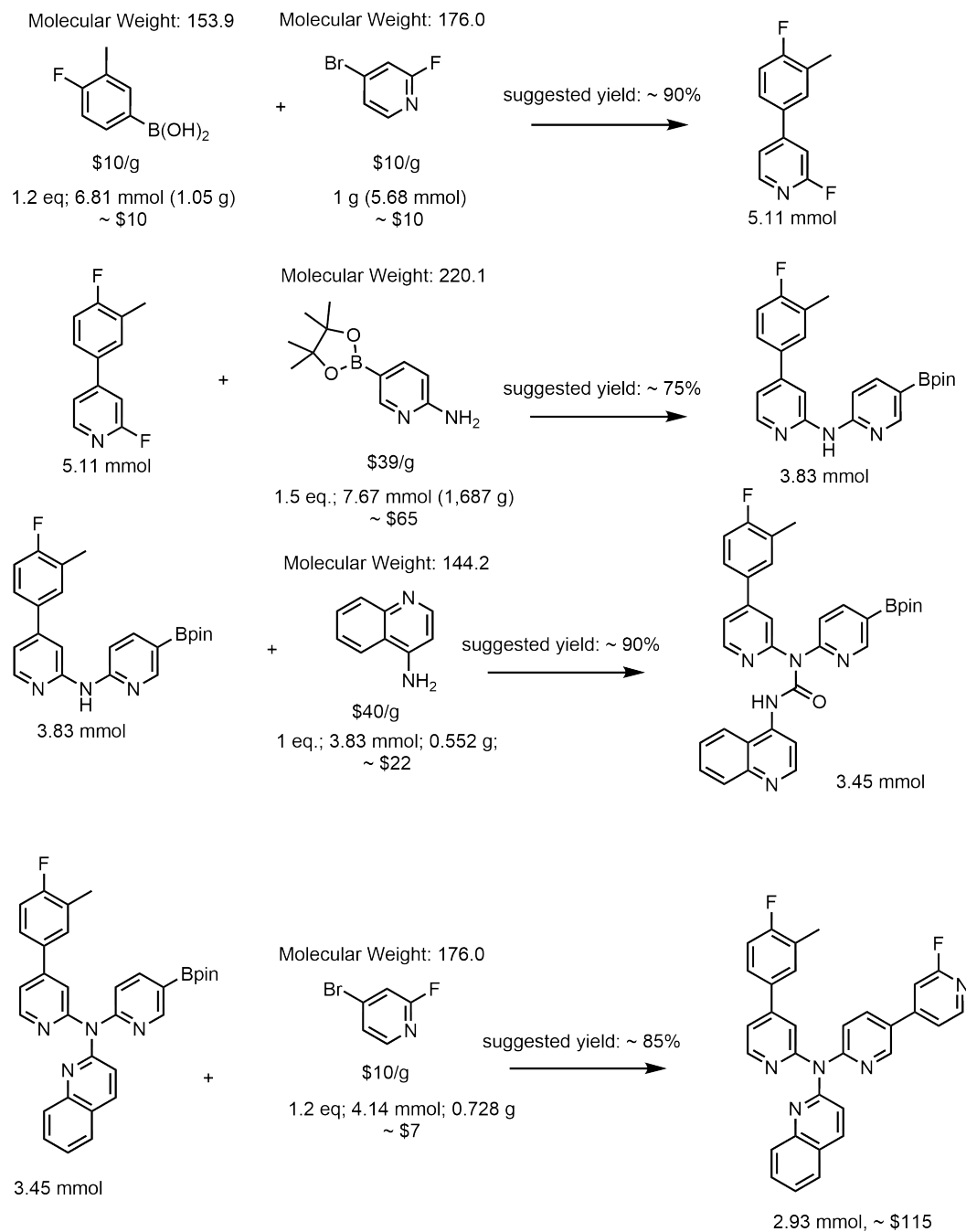


Figure 27: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 7.

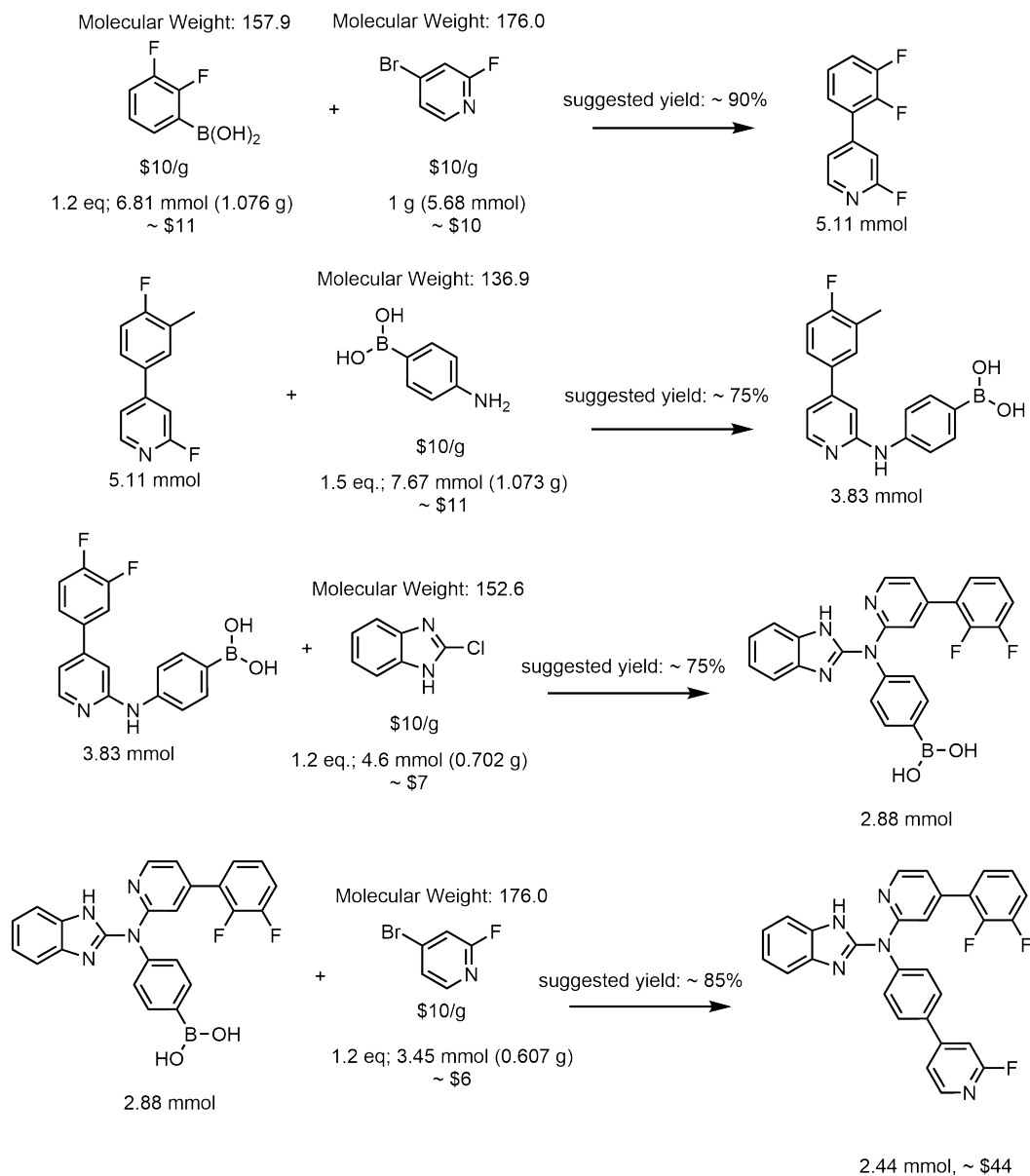


Figure 28: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 8.

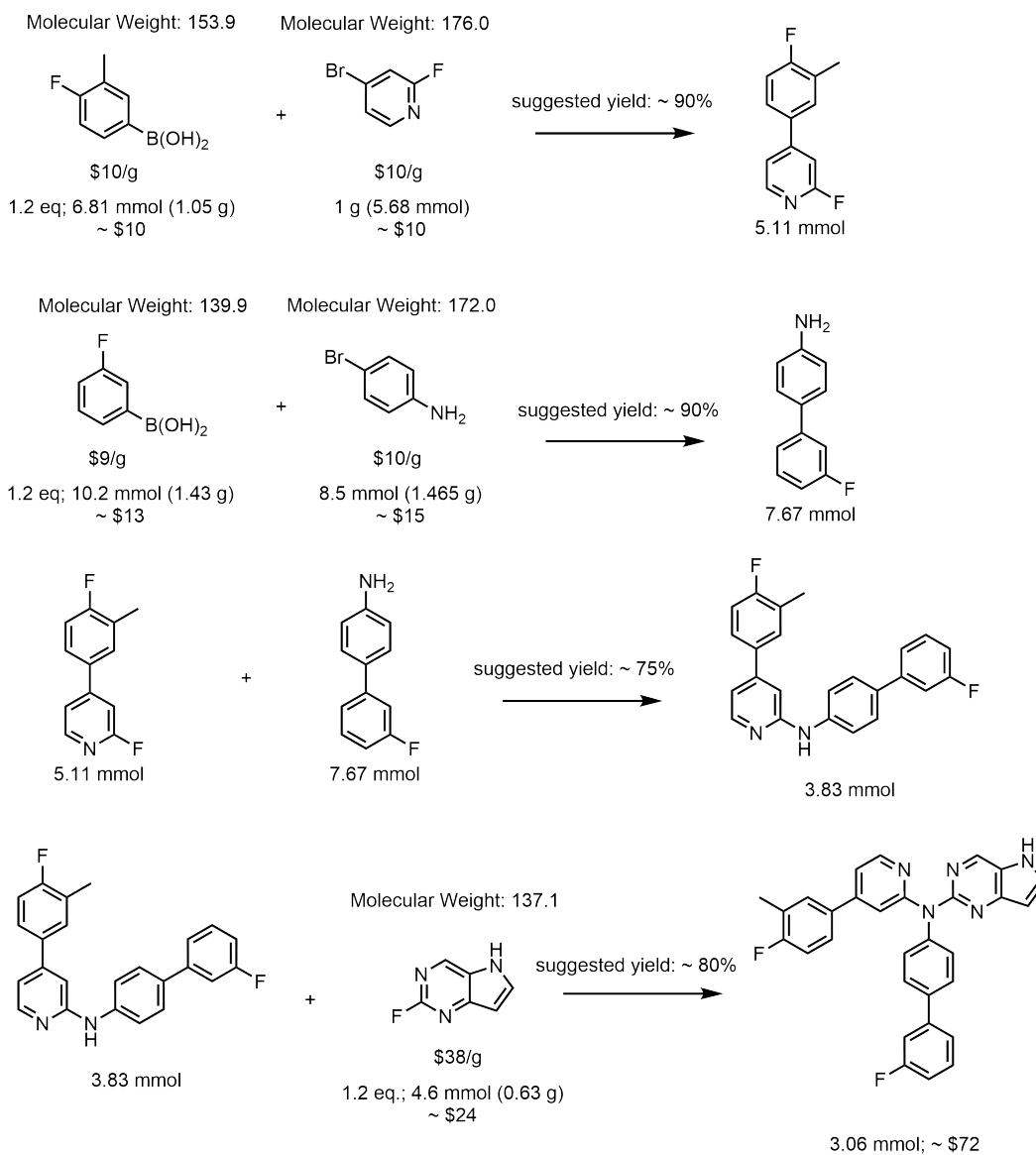


Figure 29: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 9.

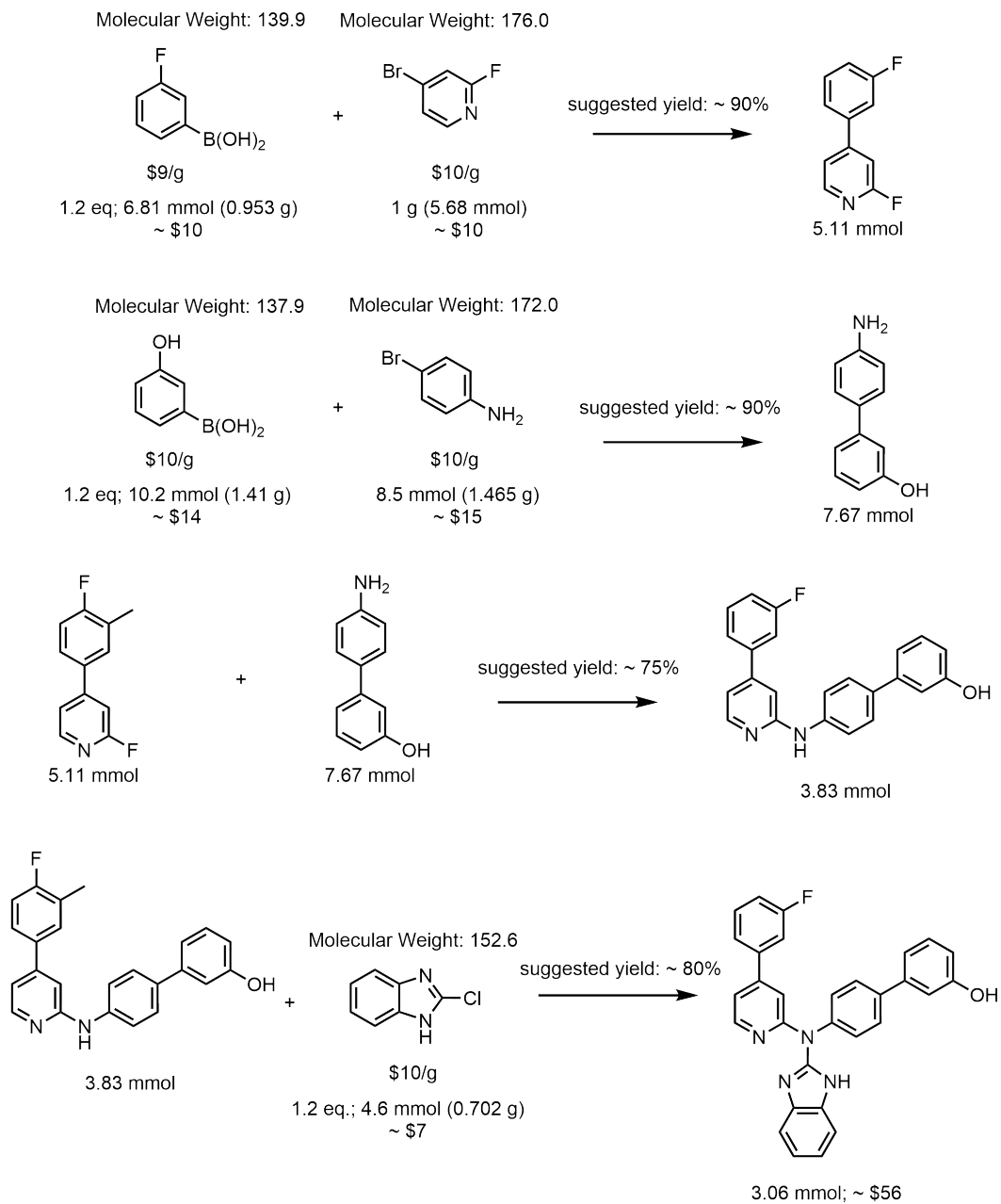


Figure 30: Plausible synthesis plan and estimated precursor cost for RGFN-produced ClpP ligand 10.

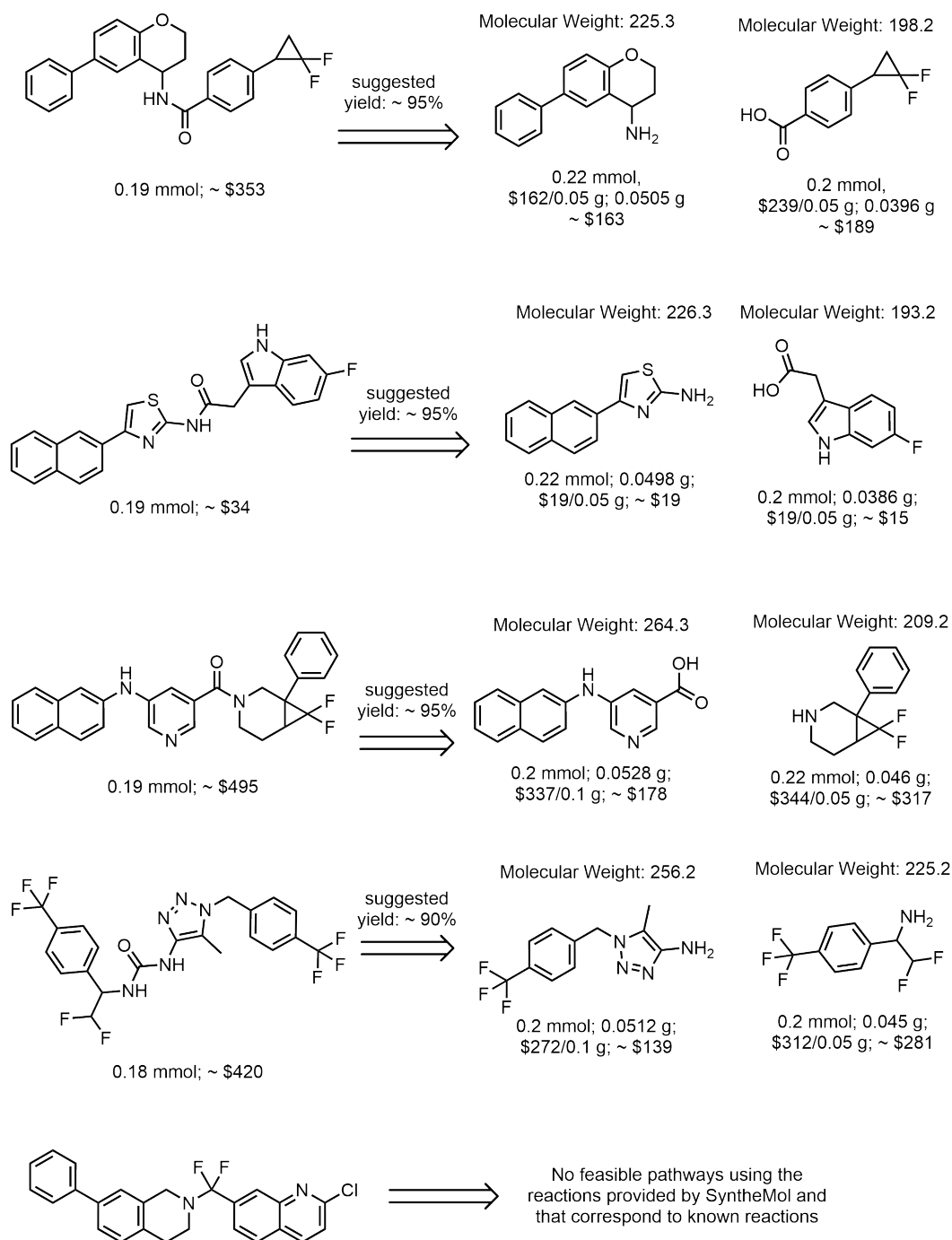


Figure 31: Plausible retrosynthesis plan and estimated precursor cost for SyntheMol-produced ClpP ligands 1-5.

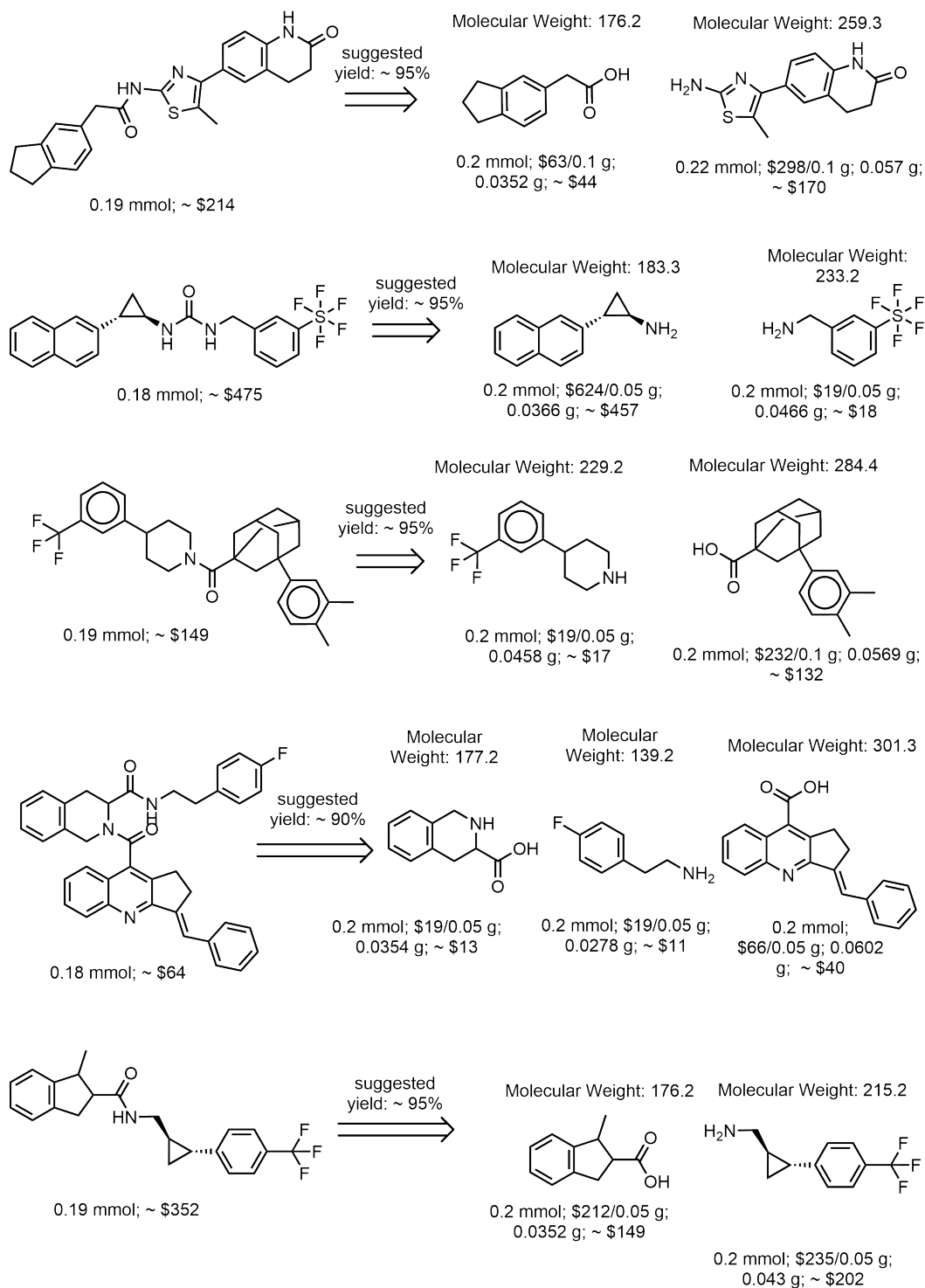


Figure 32: Plausible retrosynthesis plan and estimated precursor cost for SyntheMol-produced ClpP ligands 6-10.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: We provide extensively results in various tasks and compare against other state-of-the-art methods.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We discuss the practicality of the generation process being limited by types of reactions used, oracles employed, and auxillary pharmacological constraints that are not considered.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: For each task, we provide detailed information on experimental setup. Appendix provides further details which allow for full reproducibility of the results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All baselines are either derived from openly available repositories or described in sufficient detail to allow reproduction. Code related to the current manuscript is publicly accessible.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide training details including hyperparameters and optimizers, and other details in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Multiple seeds were run for ablation experiment, and we use the variance in these experiments to estimate and report error bars. For molecular docking, because of the high compute cost, we only run one experiment per algorithm. Errors bars are also included when applicable for metric computation across generated samples.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide details in the appendix for compute resources required.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines?>

Answer: [Yes]

Justification: The research adheres strictly to the NeurIPS Code of Ethics. There are no human subjects or participants involved, and we carefully considered the safety and security aspects of this research.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discussed the societal impact of the work in conclusions.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We do not foresee this paper to pose such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All previous baselines and tasks are properly referenced and licenses are used within the appropriate terms.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes] ,

Justification: Code of this manuscript is open sourced and contains documentation describing how to set up and run the code.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: No crowdsourcing or human subjects are involved.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No human subjects are involved.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.