Diverse Mini-Batch Selection in Reinforcement Learning for Efficient Chemical Exploration in *de novo* Drug Design

Hampus Gummesson Svensson[1, 2], **Ola Engkvist**[1, 2], **Jon Paul Janet**[1], **Christian Tyrchan**[3], **Morteza Haghir Chehreghani**[2]

[1]Molecular AI, Discovery Sciences, R&D, AstraZeneca, Gothenburg, Sweden
[2]Department of Computer Science and Engineering, Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden
[3]Medicinal Chemistry, Research and Early Development, Respiratory and Immunology (R&I), BioPharmaceuticals R&D, AstraZeneca, Gothenburg, Sweden
hamsven@chalmers.se

Abstract

In many real-world applications, evaluating the quality of instances is costly and time-consuming, e.g., human feedback and physics simulations, in contrast to proposing new instances. In particular, this is even more critical in reinforcement learning, since it relies on interactions with the environment (i.e., new instances) that must be evaluated to provide a reward signal for learning. At the same time, performing sufficient exploration is crucial in reinforcement learning to find high-rewarding solutions, meaning that the agent should observe and learn from a diverse set of experiences to find different solutions. Thus, we argue that learning from a diverse mini-batch of experiences can have a large impact on the exploration and help mitigate mode collapse. In this paper, we introduce mini-batch diversification for reinforcement learning and study this framework in the context of a real-world problem, namely, drug discovery. We extensively evaluate how our proposed framework can enhance the effectiveness of chemical exploration in *de novo* drug design, where finding diverse and high-quality solutions is crucial. Our experiments demonstrate that our proposed diverse mini-batch selection framework can substantially enhance the diversity of solutions while maintaining high-quality solutions. In drug discovery, such an outcome can potentially lead to fulfilling unmet medical needs faster.

1 Introduction

In recent years, utilizing reinforcement learning (RL) for fine-tuning of pre-trained generative models has shown great success in various applications (Zhai et al. 2024; Fan et al. 2023), including *de novo* drug design (Olivecrona et al. 2017; Atance et al. 2022). *De novo* drug design is a computational problem that aims to identify novel molecular structures with specific properties without any starting template (Mouchlis et al. 2021), where generative models have shown great success (Tong et al. 2021; Pang et al. 2023). When fine-tuning a generative model, the goal is often to align the model's outputs with respect to human preferences or experiments. However, many practical applications require frequent assessment of data and experiences, e.g., via human expert evaluation, computer simulations, field testing, and laboratory experimentation. These assessment methods are often resource-intensive, demanding significant time and financial investment. In *de novo* drug design, resource-intensive computational methods are used to assess the fit of molecules into the binding site of a target protein to predict the strength of each protein-ligand interaction (Paggi, Pandit, and Dror 2024). Consequently, the volume of data that can undergo thorough evaluation is often constrained by budgetary limitations.

In this paper, we tackle this problem in reinforcement learning, where the training instances are provided solely from the agent's interaction with the environment. In particular, we study this problem in the context of *de novo* drug design, where RL techniques are commonly used to fine-tune a pre-trained generative model to produce molecules with desired properties (Patronov, Papadopoulos, and Engkvist 2021; Pitt et al. 2025). In general, many successful RL algorithms, e.g., (Schulman et al. 2017; Mnih et al. 2016), run many copies of the environment in parallel to synchronously or asynchronously learn from numerous interactions. For synchronous on-policy algorithms, the experiences are accumulated to compute an average loss to update the agent's policy. This is also true for *de novo* drug design (Olivecrona et al. 2017), where for each policy update, a batch of molecules is first generated in parallel. However, in many real-world applications, including *de novo* drug design, it is impractical to assess all interactions with the environment, where each assessment of an interaction provides a reward signal for the agent. Instead, it is preferable to evaluate a smaller, representative set and learn from it.

At the same time, to avoid mode collapse, exploration mechanisms play a vital role in agent performance, especially in tasks with delayed/sparse reward or for a reward landscape with a vast number of local optima to explore. In *de novo* drug design, a reward can only be obtained when the full molecular structure has been generated. Moreover, diversity among generated molecules is essential since a diverse molecular library increases the likelihood of identifying candidates with unique and favorable pharmacological profiles, thereby enhancing the overall efficiency and success rate of drug development pipelines. In drug design, the reward function is often complex and has many high-rewarding modes that should be found and subsequently exploited to obtain a diverse set of solutions. Thus, chemical

exploration and diversification are of integral importance in drug design. In real-world deployment of this *de novo* drug design, it is also often costly and time-consuming to evaluate an instance (i.e., a state-action episode) to obtain a reward. This creates a *reward bottleneck* which limits the policy updates, leading to the need for efficient exploration.

One popular approach to enhance exploration in RL is the addition of an exploration bonus to the reward function, commonly denoted as *intrinsic reward* (Burda et al. 2018; Badia et al. 2020; Seo et al. 2021; Tang et al. 2017). Another common approach is maximum entropy RL, where the agent tries to maximize both the reward and entropy simultaneously, i.e., succeeding at a task while still acting as randomly as possible (O'Donoghue et al. 2017; Haarnoja et al. 2017). Our work provides a consistent perspective where, while improving exploration by achieving diverse behaviors, it is important to make sure that the interactions with the environment are of high quality (i.e., receive high rewards). This becomes especially critical when the agent must account for safety considerations, exhibits sensitivity to noise, or operates in environments where numerous trajectories are infeasible. For example, in *de novo* drug design, a molecular representation may not correspond to a chemically viable compound, and minor modifications can readily compromise its validity. In this work, we accomplish this by considering mini-batch diversification in reinforcement learning, where a large number of interactions (obtained from running copies of the environment in parallel) are summarized in a smaller, diverse set of interactions used for updating the policy. This provides an effective way to impose additional exploration in the learning process, while overcoming the reward bottleneck by learning from a smaller set.

In this paper, we argue that providing a diverse mini-batch of interactions makes the agent's exploration more effective and increases the diversity of the forthcoming interactions, especially in *de novo* drug design. Thus, there are two key benefits for such mini-batch diversification: (1) computational aspects to address the reward bottleneck; (2) enhance exploration by diverse behaviors. Therefore, we introduce a framework for diverse mini-batch selection in reinforcement learning, which is illustrated in figure 1. To the best of our knowledge, this is the first effort to study the effects of diverse mini-batch selection in reinforcement learning to overcome the reward bottleneck and promote exploration. We study the use of determinantal point processes (DPP) (Kulesza and Taskar 2012), the MaxMin algorithm (Ashton et al. 2002) and k-medoids clustering (Rdusseeun and Kaufman 1987) for this task. DPPs provide an effective framework to sample a diverse set based on specified similarity information, while the MaxMin algorithm and k-medoids clustering seek to choose a subset to maximize the coverage of a larger set. Previous work has proposed a mini-batch diversification scheme based on DPPs for stochastic gradient descent and shown its effectiveness (Zhang, Kjellström, and Mandt 2017; Huang, Da Xu, and Oppermann 2019), but such a scheme has not been applied to reinforcement learning. Also, previous work has used DPPs in diverse sampling for batch Bayesian Optimization (Nava, Mutny, and Krause 2022). In this paper, we focus on mini-batch diversification

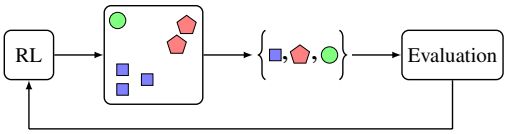


Figure 1: We propose a framework for diverse mini-batch selection in reinforcement learning. The RL agent generates a set of experiences in parallel, e.g., trajectories. A kernel measures the pairwise similarities between trajectories and is used to select a diverse set. The selected set is evaluated and, subsequently, is used to update the RL agent.

for improving exploration and reducing reward computations (i.e., addressing the reward bottleneck) in reinforcement learning. In reinforcement learning, DPPs have previously been used for unsupervised option discovery (Chen, Aggarwal, and Lan 2023), diverse recommendations for RL-based user preferences (Liu et al. 2021), and multi-agent RL (Sheikh, Frisbee, and Phielipp 2022; Yang et al. 2020; Osogami and Raymond 2019). All of these are different from our setting and can not be applied to our setting. The MaxMin algorithm is a popular method used in drug discovery to pick a diverse set (Dreiman et al. 2021; Tan et al. 2022), but has not been investigated in combination with reinforcement learning. Furthermore, k-medoids clustering is a widely known clustering technique for finding a good partition in non-Euclidean data and has only been used for cluster-based RL (Grua and Hoogendoorn 2018), which is different from our setting. To the best of our knowledge, our paper provides the first combinations of these methods with reinforcement learning to effectively fine-tune a generative model for *de novo* drug design (or any other application).

Thereby, the contribution of this paper is twofold:

- We propose a mini-batch diversification framework for RL to enhance exploration and, at the same time, to address the reward bottleneck issue.
- We extensively investigate the proposed framework on the *de novo* drug design application, and demonstrate its effectiveness via extensive experiments.

Due to the characteristics of the *de novo* drug design problem, it is a suitable problem to employ diverse mini-batch selection and study its effectiveness. We believe that this framework can also help to overcome the reward bottleneck and enhance exploration in other real-world applications of reinforcement learning, especially for fine-tuning a pre-trained generative model in other domains. Exploration is a key challenge in RL, and domain-specific information can easily be incorporated into the proposed framework.

2 Background

2.1 RL-based *de novo* Drug Design

The aim of *de novo* drug design is to design novel drug molecules given a set of predefined constraints, but without any known initial structure (Mouchlis et al. 2021). A popular approach for *de novo* drug design is to use chemical language models to generate string-based representations

of molecules (Arús-Pous et al. 2019; Segler et al. 2018). To steer the chemical language model to promising areas of the chemical space, reinforcement learning can be leveraged (Olivecrona et al. 2017). This paper focuses on promoting diversity in RL-based fine-tuning of a chemical language model via mini-batch diversification. An action a in this RL problem corresponds to adding one token to the string representation of the molecule, where $\mathcal{A}$ is the set of possible tokens that can be added, including a start token a^{start} and a stop token a^{stop}. The reward function assesses the quality of the molecule represented by the string, and the molecule can only obtain a reward when the full string representation has been generated, i.e., a stop token has been added. This *de novo* drug design problem can be modeled as a Markov decision process (MDP), e.g., see (Gummesson Svensson et al. 2024) for more details.

One popular string-based representation of chemical entities is Simplified Molecular Input Line Entry System (Weininger 1988), abbreviated SMILES. Evaluations by both Gao et al. (2022) and Thomas et al. (2022) have concluded good performance of the SMILES-based REINVENT (Segler et al. 2018; Olivecrona et al. 2017; Blaschke et al. 2020a; Loeffler et al. 2024) compared to both other RL-based and non-RL-based approaches for *de novo* drug design. REINVENT consists of a long short-term memory (LSTM) network (Hochreiter 1997) using SMILES to represent molecules as text strings. REINVENT utilizes an on-policy RL algorithm to perform online optimization of the policy π_θ to generate higher-rewarding molecules. Previous work has shown that minimizing its loss function is equivalent to maximizing the expected return, as for policy gradient algorithms (Guo and Schwaller 2024). Our work builds upon the success of REINVENT and focuses on improving its chemical exploration and avoiding mode collapse.

2.2 Diversity in *de novo* drug design

The drug-like chemical space is estimated to consist of 10^{33} synthesisable molecules (Polishchuk, Madzhidov, and Varnek 2013). To explore this space and improve the diversity of the generated molecules, several studies aim to improve the chemical exploration carried out by the RL agent. Without the use of any exploration technique, the policy easily collapses to generating only a few modes of the reward function, which leads to low diversity. To improve the diversity in RL-based *de novo* drug design, Blaschke et al. (2020b) therefore introduces a count-based method that reduces the reward for similar molecules based on their structure. The work of Park et al. (2024) and Wang and Zhu (2024) employs memory and learning-based intrinsic motivation to improve the reward of the generated molecules. Moreover, previous work shows that incorporating both structure- and learning-based information into the reward function can improve the overall diversity of *de novo* drug design (Gummesson Svensson et al. 2025). Our work takes on a fundamentally alternative perspective to enhance diversity. Rather than just encouraging diverse and explorative behavior via the reward signal, our work studies the effect of maximizing the diversity of the molecules that we evaluate and learn from.

To measure the diversity among a given set of molecules, several existing metrics have been proposed. Hu et al. (2024) divides these metrics into two main categories: reference-based and distance-based. A reference-based metric compares a molecular set with a reference set to find the intersection. Distance-based metrics use pairwise distances among the molecular set to determine the diversity. In this work, both metrics are applied. As the representative reference-based metric, the number of molecular scaffolds, also known as Bemis-Murcko scaffolds (Bemis and Murcko 1996), is used. As a distance-based metric, we utilize the number of *diverse actives*[1] metric by Renz, Luukkonen, and Klambauer (2024), which is based on #Circles metric proposed by Xie et al. (2023). Following the definition by Renz, Luukkonen, and Klambauer (2024) but using the terminology of predicted active molecules (rather than hit molecules), the number of diverse actives for distance threshold D is defined by

$$\mu\left(\mathcal{H}; D\right) = \max_{\mathcal{C}\in 2^{\mathcal{H}}} |\mathcal{C}| \text{ s.t. } \forall x \neq y \in \mathcal{C} : d(x,y) \geq D, \quad (1)$$

where $\mathcal{H}$ is a set of predicted active molecules, $2^{\mathcal{H}}$ is the power set, $d(x,y)$ is the distance between molecules x and y. As suggested by (Renz, Luukkonen, and Klambauer 2024), we use the MaxMin algorithm (Ashton et al. 2002) implemented in RDKit (Landrum 2006) to find an approximate maximal value of the cardinality of C.

3 Diverse Mini-Batch Selection For RL

Algorithm 1: Diverse Mini-Batch Selection

input: $G, B, k, \theta_0, T, p_0$
$\mathcal{M} \leftarrow \emptyset$
$\theta \leftarrow \theta_0$ ▷ Initial policy parameters
for $g = 1, \ldots, G$ **do**
  for $b = 1, \ldots, B$ **do** ▷ Generate in parallel
    $s_0 \sim p_0(\cdot)$ ▷ Sample first state
    for $t = 0, 1, \ldots, T-1$ **do**
      $a_t \sim \pi_\theta(s_t)$
      Observe next state $s_{t+1} \sim P(\cdot|s_t, a_t)$
    end for
    $\tau_b := s_0, a_0, \ldots, a_{T-1}, s_T$ ▷ Trajectory
  end for
  $\mathcal{B} \leftarrow \{\tau_1, \ldots, \tau_B\}$
  Compute kernel matrix L over $\mathcal{B}$
  Select k representative trajectories from $\mathcal{B}$
  $\forall \tau \in Y$, observe return $r(\tau)$ ▷ Evaluation
  $\mathcal{M} \leftarrow \mathcal{M} \cup (\cup_{\tau\in Y}\{\tau, r(\tau)\})$
  Update θ using RL algorithm
end for
output: $\theta, \mathcal{M}$

We propose a framework to enhance exploration in reinforcement learning while reducing the number of interactions evaluated. We seek to generate more diverse solutions through reinforcement learning-based fine-tuning of a pre-trained generative model. In this paper, we focus on fine-

[1]Diverse actives is termed diverse hits in previous work by Renz, Luukkonen, and Klambauer (2024).

tuning a chemical language model. We assume delayed rewards and that acquiring a sequence of states and actions is inexpensive compared to the evaluation, which is often true for real-world problems such as *de novo* drug design. Given a large set of interactions, we seek to select a smaller, representative set to use for updating the parameters of our policy. We hypothesize that this affects the agent's exploration of the solution space, which is of vast importance in RL-based *de novo* drug design, while overcoming the reward bottleneck by considering a fixed budget of evaluations. The intuition is that learning from diverse experiences helps the agent to explore more effectively.

Therefore, we suggest enforcing diversity among the selected interactions to improve the efficiency of the exploration. For this purpose, we propose a diverse mini-batch selection framework for reinforcement learning, which is illustrated in algorithm 1. Here, we focus on trajectories of actions, i.e., an interaction corresponds to a trajectory, which we use as a more general notion of an episode. However, the framework can easily be extended beyond trajectories/episodes. In the *de novo* drug design problem that we consider, an episode corresponds to a fully generated molecule, since each action in the episode corresponds to adding a character in the SMILES representation. Also, we consider policy-based RL, where we directly learn a policy π_θ with policy parameters θ, but the suggested framework can also be applied to value-based RL algorithms, e.g., by diverse mini-batch selection from the replay buffer.

Over G training/generative steps, using the agent's current policy π_θ, a batch $\mathcal{B}$ of B trajectories is sampled in parallel over copies of the same environment. Each trajectory has a maximum horizon of T steps, where the true length of each trajectory can depend on some stopping criteria or when the terminal state is reached. If B is chosen such that $B \gg k$ and the agent's policy is stochastic, this set will contain primarily unique items. We let the RL agent in each copy of the environment focus on maximizing the expected return of each trajectory with respect to the reward function, i.e., maximizing the return generated by the agent's policy

$$\mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)], \tag{2}$$

where τ is a state-action trajectory $S_0, A_0, S_1 \ldots, S_{T-1}, A_{T-1}, S_T$, $R(\tau)$ is the return of following τ and $\mathbb{E}_{\pi_\theta}[\cdot]$ denotes the expected value of a random variable given that the agent follows policy π_θ. This generates $\mathcal{B}$ under the belief that the agent tries to maximize each return, without explicitly considering the diversity among individual trajectories. This can be particularly important when the agent has to consider safety concerns, is sensitive to noise, or when many trajectories are not viable. For instance, in *de novo* drug design, a SMILES string is not necessarily chemically feasible, and small changes can easily break its validity. Therefore, it is important that the agent primarily focuses on generating chemically valid SMILES strings of high quality. Moreover, the proposed method can be combined with other exploration techniques, e.g., intrinsic motivation (Burda et al. 2018; Tang et al. 2017), to provide additional domain-specific exploration. Given a large batch of trajectories $\mathcal{B}$, to stay within the given budget of evaluations (per generative step), the next step is to obtain a smaller, diverse mini-batch Y that summarizes $\mathcal{B}$. We study the use of determinantal point processes (DPPs) (Kulesza and Taskar 2012), the MaxMin algorithm (Ashton et al. 2002) and k-medoids clustering (Rdusseeun and Kaufman 1987) for this task. After a set Y of k trajectories has been obtained, each trajectory in Y is evaluated to obtain the corresponding returns and/or state-action rewards. Using the returns and rewards, the policy parameters are updated by employing an arbitrary RL algorithm. The discussed framework is agnostic to the RL algorithm used to update the policy parameters, and yields both the policy parameters θ and a diverse set of trajectory-return pairs $\{\tau, R(\tau)\}$.

3.1 Determinantal Point Processes (DPPs)

We propose and study the use of determinantal point processes (Kulesza and Taskar 2012) to sample a diverse mini-batch for RL updates. DPPs provide an effective framework to sample a diverse set based on specified similarity information. To the best of our knowledge, our work is a novel combination of DPP and reinforcement learning to effectively fine-tune a generative model.

A point process $\mathcal{P}$ is a probability measure over finite subsets of a set $\mathcal{B}$. We consider the discrete case of $\mathcal{B} = \{1, 2, \ldots, B\}$, where B is the number of unique trajectories. In this case, a point process is a probability measure on the power set $2^{\mathcal{B}}$, i.e., the set of all subsets of $\mathcal{B}$. Determinantal point processes (DPPs) are a family of point processes characterized by the *repulsion* of items such that similar items are less likely to co-occur in the same sample. Given a kernel, providing a similarity measure between pairs of items, DPP places a high probability on subsets that are diverse with respect to the kernel. We consider a class of DPPs named L-ensembles (Borodin and Rains 2005), which is defined via a real, symmetric matrix L over the entire (finite) domain of $\mathcal{B}$. This matrix is often denoted as the *kernel matrix*. The probability of subset $Y \subseteq \mathcal{B}$ is given by

$$\mathcal{P}_L(Y) \propto \det(L_Y), \tag{3}$$

where $L_Y = [L_{ij}]_{i,j\in Y}$ denotes the restriction of L to the entries indexed by items of Y. Thus, the probability of sampling the set $Y \subseteq \mathcal{B}$ is proportional to the determinant of L_Y restricted to Y. The normalization constant is available in closed form since $\sum_{Y\subseteq\mathcal{B}} \det(L_Y) = \det(L + I)$, where I is the $N \times N$ identity matrix.

Given the larger set $\mathcal{B}$, we want the smaller set Y to contain a pre-defined number of items from $\mathcal{B}$. Thus, we are interested in sampling a subset Y with a fixed cardinality $|Y| = k$ to sample a mini-batch with a fixed size. k-DPPs (Kulesza and Taskar 2011) concern DPPs conditioned on the cardinality of the random subset. Formally, the probability of a k-DPP to sample a subset $Y \subseteq \mathcal{B}$ is given by

$$\mathcal{P}^k_L(Y) = \frac{\det(L_Y)}{\sum_{Y'\subseteq\mathcal{B}:|Y'|=k} \det(L_{Y'})}, \tag{4}$$

where $|Y| = k$. The k-DPP's inherent ability to promote diversity makes it an excellent choice for selecting diverse

and representative mini-batches in reinforcement learning. In this way, k-DPP provides a smaller and diverse set of items from a larger set of items.

How the k-DPP will summarize the larger set is determined by the kernel matrix L. Constructing the kernel matrix entails using domain knowledge, but other information can also be used. Let $q_i \in \mathbb{R}^+$ be a quality term and $\phi_i \in \mathbb{R}^D$, $||\phi_i|| = 1$, a vector of normalized diversity features of the i-th item in $\mathcal{B}$, e.g., the i-th generated SMILES string. Following the work of Kulesza and Taskar (2012), the entries of the kernel matrix can then be expressed

$$L_{ij} = q_i \phi_i^T \phi_j q_j, \tag{5}$$

where q_i is a quality term measuring the intrinsic "goodness" of the i-th item, and $\phi_i^T \phi_j \in [-1, 1]$ is a signed measure of similarity between i-th and j-th item. Therefore, utilizing k-DPPs allows for a flexible sampling procedure that behaves differently depending on the information incorporated in the kernel matrix L. It does not directly optimize the determinant of L, but instead includes randomness to encourage additional exploration. In the *de novo* drug design problem studied in this paper, we only consider the similarity between items and do not explore the effects of quality terms. The reason for this is that we focus on pure diversification, and we assume that the items generated by the policy have similar quality. Our preliminary studies on *de novo* drug design did not find any performance gain in incorporating a quality term provided by an oracle. However, we believe that it can be beneficial to include a quality term, but different terms need to be investigated to find a suitable one.

3.2 Maximum Coverage

As an alternative to selecting a representative set by sampling via k-DPPs, we also study mini-batch diversification by maximizing the coverage of the larger set for a fixed cardinality. While k-DPPs provide a sampling procedure to summarize a larger set given a kernel matrix, maximum coverage aims to directly cover as large a part of the space as possible. In this way, we seek to pick the most diverse items subject to the cardinality constraint of k. For a given set $\mathcal{B}$ of B candidate items, let $f(Y)$ be a function that measures the "coverage" of any given set Y of items. The goal is to choose a set Y of k items such that $f(Y)$ is maximized. Here we consider a fixed size of k, but a possible extension could be to choose the smallest set Y such that a sufficient coverage of $\mathcal{B}$ is obtained. Formally, we define this problem by

$$\max_{Y \in [\mathcal{B}]^k} f(Y), \tag{6}$$

where $[\mathcal{B}]^k \triangleq \{X \in 2^{\mathcal{B}} : |X| = k\}$ is the set of all subsets with cardinality k. In this work, we consider coverage functions $f(Y)$ based on dissimilarities between trajectories.

We investigate two algorithms to find an approximate maximum coverage of the large set: (1) the MaxMin algorithm (Ashton et al. 2002), implemented by RDKit (Landrum 2006); (2) k-medoids clustering (Rdusseeun and Kaufman 1987), using the FasterPAM algorithm (Schubert and Rousseeuw 2019, 2021) implemented by Schubert and Lenssen (2022). The MaxMin algorithm first picks a starting item, creating a picked set. Then the algorithm iteratively, from the items in the candidate pool, finds the item that has the maximum dissimilarity to molecules in the picked set and adds this item to the picked set. The MaxMin algorithm is widely used in drug discovery to pick a diverse set (Dreiman et al. 2021; Tan et al. 2022).

k-medoids clustering (Rdusseeun and Kaufman 1987) is a popular technique to cluster non-Euclidean data using arbitrary dissimilarities or input domains. The k-medoids problem aims to split B items into k ($\leq B$) clusters, where the number of clusters is assumed to be specified beforehand. The medoid of a cluster is defined as the item in the cluster with the minimum average of dissimilarity to all the other items in the cluster, i.e., the item that is most centrally located within the cluster. Unlike several other clustering algorithms, e.g., k-means (Arthur and Vassilvitskii 2007), the medoid is an actual item in the cluster. Thus, the objective is to find medoids $m_1, \ldots, m_k$ that minimizes

$$\underset{\{m_1,\ldots,m_k\} \subset Y}{\arg\min} \sum_{i=1}^{k} \sum_{x_c \in C_i} d(x_c, m_i), \tag{7}$$

where C_i is the cluster of medoid m_i and d is an arbitrary dissimilarity function. While the MaxMin algorithm sequentially adds items to the picked set in a greedy manner, k-medoids simultaneously seeks to optimize all medoids to find the best picks. Finding the global optimum of the k-medoid problem is NP-hard (Kariv and Hakimi 1979). Instead, the Partitioning Around Medoids (PAM) algorithm (Rdusseeun and Kaufman 1987), which is the standard algorithm for k-medoids clustering, improves the clustering towards a local optimum. In this paper, we use the FasterPAM algorithm (Schubert and Rousseeuw 2019, 2021), which achieves a speedup in runtime compared to the original PAM algorithm, to select k items (given by the medoids found by the algorithm).

4 Experimental Evaluation

We extensively evaluate our framework on *de novo* drug design. We run experiments on three reward functions based on well-established molecule binary bioactivity label optimization tasks: the Dopamine Receptor D2 (DRD2), c-Jun N-terminal Kinases-3 (JNK3), and Glycogen Synthase Kinase 3 Beta (GSK3β) predictive activity models (Olivecrona et al. 2017; Li, Zhang, and Liu 2018) provided by Therapeutics Data Commons (Velez-Arce et al. 2024). The final (extrinsic) reward also includes the quantitative estimation of drug-likeness (QED) (Bickerton et al. 2012), molecular weight, number of hydrogen bond donors, and structural constraints. For full specifications on the reward functions, we refer to appendix C.

To update the policy and generate SMILES, we use the REINVENT framework (Loeffler et al. 2024) with its pretrained policy on the ChEMBL database (Gaulton et al. 2017) to generate drug-like bioactive molecules. Previous benchmarks on *de novo* drug design have, for this framework, concluded among the best performances (Gao et al.

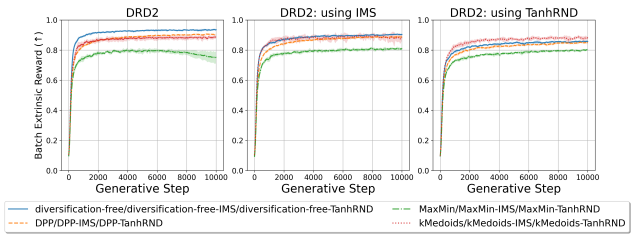
2022; Thomas et al. 2022), while it is also used in real-world applications (Pitt et al. 2025). The action space $\mathcal{A}$ consists of 34 tokens, including start and stop tokens. We evaluate the diversity of the generated set $\mathcal{M}$ by the number of molecular scaffolds and the number of diverse actives (see equation (1)), where the diverse actives are computed for every 250th generative step. For the diverse actives, we use Tanimoto dissimilarity to measure the distance between 2048-bit Morgan fingerprints (with radius 2 and computed by RDKit (Landrum 2006)) and the distance threshold $D = 0.7$ proposed by (Renz, Luukkonen, and Klambauer 2024). When computing the diversity in terms of both scaffolds and diverse actives, we only regard active molecules, defined as molecules with both QED and predicted activity larger than 0.5.

We compare the use of mini-batch diversification in combination with different techniques to modify the original reward for *de novo* drug design: (1) no modification of the reward, i.e., the agent observes the original (extrinsic) reward; (2) using the popular identical molecular scaffold (IMS) penalty (Blaschke et al. 2020b), which sets the reward to 0 when M molecules with the same molecular scaffold have been generated; (3) using the TanhRND technique (Gummesson Svensson et al. 2025), which shows promising empirical results in terms of diversity. No modification of the reward is included as a baseline to investigate if mini-batch diversification can act as an alternative approach to avoid mode collapse by modifying the original reward. We hereafter denote the original reward without any modification as the *extrinsic reward*. For mini-batch diversification with a mini-batch of $k = 64$ SMILES, we first generate $B = 640$ SMILES via multinomial sampling and then use k-DPP to select a diverse mini-batch. Without mini-batch diversification, we directly generate $k = 64$ SMILES via multinomial sampling, which is the standard procedure of the REINVENT framework. We denote these approaches without mini-batch diversification as *diversification-free*.

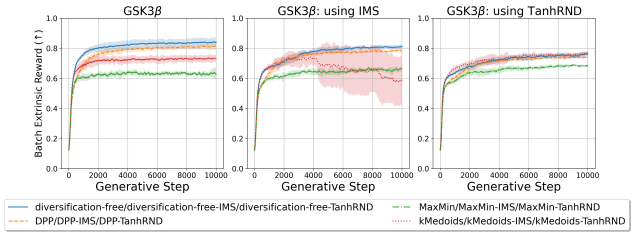
4.1 Construction of Kernel Matrix

All of the investigated methods for mini-batch diversification (i.e., DPP, the MaxMin algorithm and k-medoids clustering) rely on a kernel matrix L to encode the similarity between different molecules. We construct this kernel matrix based on two other kernel matrices L_T and L_D, which we denote as "base" kernel matrices. The first base kernel matrix L_T is constructed by the Tanimoto similarity between the corresponding 2048-bit Morgan fingerprints (with radius 2 using RDKit (Landrum 2006)) of the generated SMILES. To incorporate more scaffold-based information, we construct the base kernel matrix L_D by computing the Dice coefficients (Dice 1945; Sorensen 1948) between the scaffolds' atom pair fingerprints (Carhart, Smith, and Venkataraghavan 1985). Given these base kernels, we aggregate these base kernel matrices to define the kernel matrix L, which is used for selecting k molecules, by $L = L_T + L_D$. In appendices A and B, we provide a study on different combinations of the base matrices to define L and argue that the kernel matrix defined here provides the best balance between the different diversity metrics.

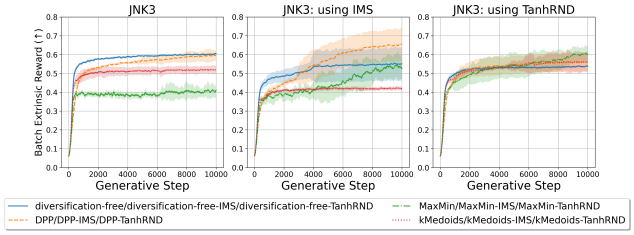
4.2 Effects on Quality of Diverse Mini-Batch Selection



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 2: Average extrinsic rewards per generative step across the mini-batch of SMILES evaluated on the DRD2-, GSK3β-, or JNK3-based reward functions. For clarity of presentation, we display the moving averages with a window size of 101. The average across 10 independent runs per generative step is plotted over 10 000 generative steps, where the shaded area shows standard deviations among the independent runs.

We first assess the quality (i.e., the reward) of the generated molecules to evaluate if our proposed framework can maintain high quality while enhancing the diversity. Therefore, we study the extrinsic reward of each configuration. The extrinsic reward is the original reward provided for each molecule that we want to maximize, but not the reward observed by the agent when using IMS or TanhRND.

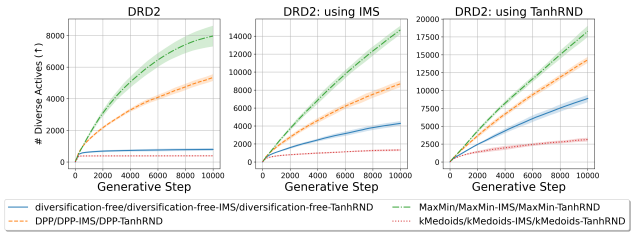
Figure 2 displays the average extrinsic rewards for each mini-batch Y of SMILES evaluated in each generative step. The average across 10 independent runs per generative step is plotted over 10 000 generative steps, where the shaded area shows the corresponding standard deviation across the independent runs. For clarity of presentation, we show the moving averages with a window size of 101 (i.e., the current step and upto 50 steps on each side). Each plot of figures 2a to 2c compares the use of diverse mini-batch selection using k-DPP, the MaxMin algorithm and k-medoids clustering in

combination with different techniques of modifying the extrinsic (original) reward for *de novo* drug design. The left plots compare the extrinsic rewards for both with and without mini-batch diversification when the extrinsic reward is not modified. The middle plots compare the extrinsic rewards when using the identical molecular scaffold (IMS) penalty proposed by (Blaschke et al. 2020b) and the right plots display the comparisons when utilizing the TanhRND technique (Gummesson Svensson et al. 2025).

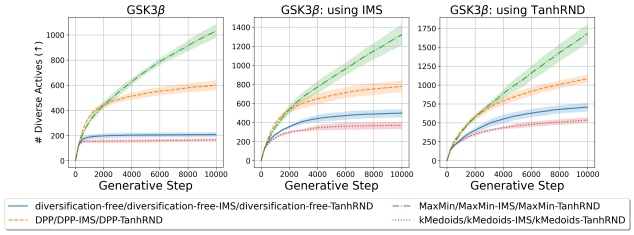
For the DPP and diversification-free methods on the DRD2- and GSK3β-based reward functions (see figures 2a and 2b), we observe similar trends in terms of extrinsic reward, especially when using IMS or TanhRND. Moreover, on the DRD2 reward, these experiments achieve a reward of 0.8 or higher, while rewards close to 0.8 are achieved on the GSK3β function. The diversification-free experiments converge faster, but the DPP experiments often converge to a similar average reward. Faster convergence tends to indicate that less exploration is performed, which is demonstrated in figures 3 and 4 below in terms of diversity of the generated molecules. k-medoids shows similar results on DRD2, but achieves more unstable and lower quality on GSK3β. For the MaxMin experiments on the DRD2 and GSK3β problems, we observe that extrinsic rewards are lower than for both the DPP and diversification-free experiments. This is possibly because more exploration is enforced, due to a more diverse mini-batch, at the cost of less exploitation. For the experiments on the JNK3-based reward function (see figure 2c), we observe similar trends as for DRD2 and GSK3β when not modifying the extrinsic reward (see left plot in figure 2c). On the other hand, when using the IMS or TanhRND technique to modify the extrinsic reward, all methods display similar extrinsic reward, but different convergence rates. Only k-medoids utilizing IMS performs differently, displaying an early convergence to a reward of around 0.4, which is lower than the other methods. This is likely due to insufficient exploration induced by this configuration. In general, the extrinsic rewards are significantly lower on JNK3, indicating that the JNK3-based reward function is more challenging to optimize. One possible explanation is that there are fewer active molecules for JNK3 in the ChEMBL database. When we evaluate molecules from ChEMBL on the DRD2, GSK3β, and JNK3 oracles, we observe that 2.4%, 1.8%, and 0.3% of the molecules, respectively, have an oracle score above 0.5 (we refer to appendix D for more details). Thus, there are fewer good solutions for JNK3. Since we use a model pre-trained on ChEMBL data, which limits the generation to molecules similar to those found in this data, the initial model is less likely to find sufficient solutions for JNK3.

4.3 Diverse Mini-Batch Selection Enhances Distance-Based Diversity

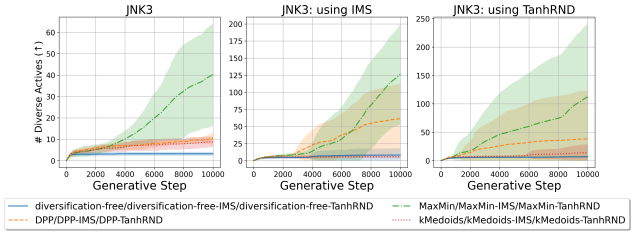
To evaluate the distance-based diversity among the generated molecules, we calculate the number of diverse actives. Figure 3 shows the total number of diverse actives for every 250th generative step in the *de novo* drug design task for the DRD2-, GSK3β- and JNK3-based reward functions. The lines and shaded area display the mean and standard deviation, respectively, across 10 independent reruns for each configuration. Each plot of figures 3a to 3c compares the use of DPP in combination with different techniques of modifying the extrinsic reward for *de novo* drug design.



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 3: Total number of diverse activities after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model. The total number of diverse actives is plotted for every 250th generative step. The average across 10 independent runs per generative step is plotted over 10 000 generative steps, where the shaded area shows standard deviations among the independent runs.

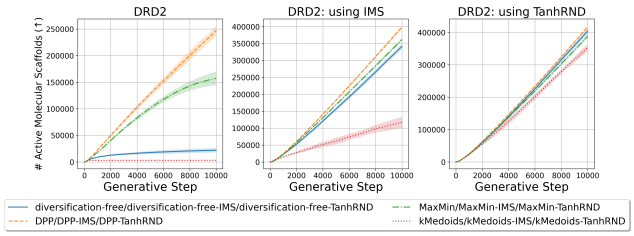
Dopamine Receptor D2 (DRD2) Figure 3a displays the cumulative number of diverse actives per generative step on the DRD2-based reward function. We observe that utilizing mini-batch diversification significantly improves the total number of diverse actives found over 10000 generative steps compared to the diversification-free experiments (blue lines). We observe a significant gain after just a few hundred generative steps. In particular, MaxMin consistently yields the best results in terms of diverse actives, compared to the

Interestingly, when not using IMS or TanhRND to modify the extrinsic reward (see left plot in figure 3a), DPP and MaxMin display a considerable increase in distance-based diversity after a few hundred generative steps compared to the diversification-free method, where diversity quickly stagnates. Without mini-batch diversification (and any extrinsic reward modification), it is expected that the diver-

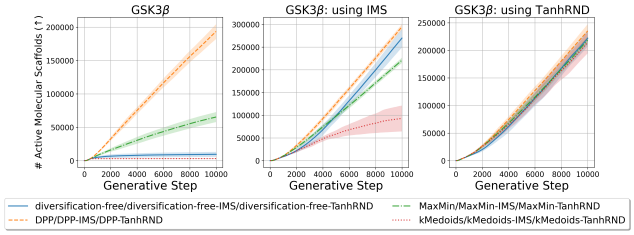
sity should stagnate since it has previously been observed that the agent can easily get stuck in a local optimum and will then generate similar molecules (Blaschke et al. 2020b). Using mini-batch diversification via DPP or MaxMin overcomes this issue even without modifying the extrinsic reward, which is the standard method for tackling this issue. In addition, we observe that mini-batch diversification in combination with a modification of the extrinsic reward (see the middle and right plot in figure 3a) yields the largest number of diverse actives, especially when utilizing TanhRND. However, using k-medoids for mini-batch diversification generates fewer diverse activities than the diversification-free methods, even when not modifying the rewards.

Glycogen Synthase Kinase 3 Beta (GSK3β) Figure 3b displays the cumulative number of diverse actives per generative step on the GSK3β-based reward function. We observe that utilizing mini-batch diversification via DPP or MaxMin generates significantly more diverse active after a few hundred generative steps. We see this behaviour no matter if we modify the extrinsic reward or not, meaning that mini-batch diversification can successfully be used as an exploration technique to overcome mode collapse and lead to diverse behaviors. Moreover, we notice that, after at most 4000 generative steps, MaxMin yields substantially more diverse actives than the other methods. Also, we note that, similar to the experiments on the DRD2-based reward functions, using k-medoids yields a substantially lower number of diverse actives than the other methods, including diversification-free methods.

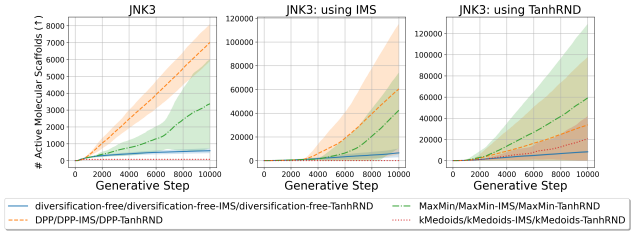
c-Jun N-terminal Kinases-3 (JNK3) Figure 3c shows the cumulative number of diverse actives per generative step on the JNK3-based reward function. Firstly, we observe a high standard deviation among all experiments, compared to the other reward functions. This is likely since the JNK3 oracle is more difficult to optimize than the other oracles, and therefore does not have a large margin to the activity threshold of 0.5 for diverse actives. Similar trends in terms of diversity have been observed by previous work (Gummesson Svensson et al. 2025). Most approaches using mini-batch diversification keep improving over a large number of generative steps, while the diversification-free experiments generally show a substantially lower number of average diverse actives. For no extrinsic reward modification (see left plot in figure 3c), MaxMin generates the highest average number of diverse actives, while DPP has lower variance but yields fewer diverse actives. When using the IMS or TanhRND strategy to modify the reward (see middle and right plot in figure 3c), MaxMin also yields the highest average number of diverse actives, but the runs overlap with DPP since both have high variance. For the experiments using TanhRND (see right plot in figure 3c), all MaxMin configurations display a larger increase in the average number of diverse actives over time. On this reward function, k-medoids can generate more diverse actives than the diversification-free method when not modifying the (extrinsic) reward, while these two methods display similar performance when modifying the reward.



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 4: Total number of molecular scaffolds after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model. The average across 10 independent runs per generative step is plotted over 10 000 generative steps, where the shaded area shows standard deviations among the independent runs.

4.4 Diverse Mini-Batch Selection Enhances Reference-Based Diversity

To obtain a more comprehensive evaluation of the diversity, we also investigate reference-based diversity (Hu et al. 2024). In particular, we consider the number of unique molecular scaffolds, also named Bemis-Murcko scaffolds (Bemis and Murcko 1996), computed by RDKit (Landrum 2006). We are only interested in the diversity of molecules suitable for our target and, therefore, only consider scaffolds of active molecules with both an oracle score and QED above 0.5. Figure 4 shows the cumulative number of unique active molecular scaffolds per generative step for the DRD2-, GSK3β- and JNK3-based reward functions. The lines and shaded area display the mean and standard deviation, respectively, across 10 independent reruns for each configuration.

Dopamine Receptor D2 (DRD2) Figure 4a displays the cumulative number of active molecular scaffolds, per generative step, evaluated on the DRD2-based reward function. When not modifying the extrinsic reward (see left plot in figure 4a), using mini-batch diversification via DPP or MaxMin leads to substantially more scaffolds, compared to

the diversification-free method, after less than 750 generative steps. In particular, our experiments demonstrate that DPP generates most scaffolds on average. When utilizing the identical molecular scaffold (IMS) filter (Blaschke et al. 2020b) for modifying the extrinsic reward (see middle plot in figure 4a), we observe that DPP generates more molecular scaffolds compared to the other methods. For the TanhRND technique (see right plot in figure 4a), the diversification-free, MaxMin and DPP methods show similar diversity in terms of molecular scaffolds and perform on par with the best methods when using IMS (see middle plot in figure 4a). In terms of molecular scaffolds, it is clear that the scaffold-based similarity that mini-batch diversification provides can be important, especially in combination with no or less effective exploration techniques. However, across all experiments, it is clear that k-medoids generates the least amount of scaffolds, and it is therefore important to choose an appropriate method for mini-batch diversification.

Glycogen Synthase Kinase 3 Beta (GSK3β) Figure 4b displays the cumulative number of molecular scaffolds for the evaluation on the GSK3β-based reward function. Without any modification of the extrinsic reward (see left plot in figure 4b), we observe that mini-batch diversification via DPP or MaxMin yields significantly more scaffolds compared to the diversification-free method (blue line). The DPP effectively generates more molecular scaffolds, while MaxMin is less effective. For reward modification (see the middle and right plots in figure 4b), we observe that using mini-batch diversification via DPP generates more scaffolds on average and has lower variance. However, the difference in effectiveness of using DPP is reduced in terms of diverse actives, but DPP can still consistently improve diversity. For MaxMin, which consistently generates the largest number of diverse actives (see figure 3b), we observe a lower number of scaffolds. Thus, when using the MaxMin algorithm to impose mini-batch diversity, we see that high distance-based diversity does not directly result in high reference-based diversity, and vice versa. When using mini-batch diversification via k-medoids, it generates significantly fewer scaffolds, except when using TanhRND, where it performs on par with the other methods.

c-Jun N-terminal Kinases-3 (JNK3) Figure 4c displays the scaffold diversity for the evaluation on the JNK3-based reward function. When not modifying the extrinsic reward (see left plot in figure 4c), all DPP-based methods are more effective after around 2000 generative steps. For DPP, we observe the largest average number of molecular scaffolds and notice a more consistent exploration, since the rate of diverse solutions is higher. The MaxMin algorithm does not display the same consistent improvement in the number of scaffolds. When modifying the extrinsic reward (see middle and right plots in figure 4c), both DPP and MaxMin obtain a higher average number of scaffolds, but they also display a high variability and are therefore not always more effective. This is likely because the agent is not able to effectively optimize the reward (see figure 2c). In general, as depicted in figure 2c, the JNK3-based reward is more difficult to optimize for the RL agent. Thus, we notice that the robustness of our proposed mini-batch diversification depends on how well the agent can optimize the given task. This is expected since the mini-batch selection depends on the given larger set $\mathcal{B}$ and, therefore, has limited capabilities to enhance the diversity if the RL agent itself cannot find sufficient solutions.

5 Conclusions

In this work, we present an easily applicable framework for enhancing mini-batch diversity in reinforcement learning algorithms. The framework seeks to tackle the problem of efficient exploration when it is costly to evaluate a reward function. In this paper, we apply our framework to *de novo* drug design, but the framework is problem-agnostic. We believe that the proposed framework can also be beneficial in other applications in reinforcement learning, where efficient exploration and diverse behaviors are crucial. To solve the problem of mini-batch diversification in RL, we study the use of determinantal point processes (DPPs) (Kulesza and Taskar 2012), the MaxMin algorithm (Ashton et al. 2002) and k-medoids clustering (Rdusseeun and Kaufman 1987) for the diversification process. In this way, we seek to summarize a larger set of molecules by selecting a smaller mini-batch of diverse molecules to evaluate, requiring fewer evaluations. DPP samples a diverse mini-batch given a kernel matrix, while the MaxMin algorithm and k-medoids clustering aim to find the maximum coverage of the larger set with respect to dissimilarities between molecules. We argue that this enhances the exploration by focusing on promising, more diverse molecules, while keeping the rewards high. We observe that our proposed framework for mini-batch diversification can substantially improve the diversity of *de novo* drug design, especially when combined with a domain-specific modification of the extrinsic reward, such as TanhRND (Gummesson Svensson et al. 2025). We demonstrate that DPP-based mini-batch diversification enhances both distance- and reference-based diversity, while the MaxMin algorithm primarily improves distance-based diversity. Therefore, we propose to use DPP for the diversification process, since it also allows for a more adaptable kernel matrix, e.g., by incorporating quality terms, and a natural way to introduce randomness in the diversification process. Moreover, we notice that if the agent alone provides sufficient solutions, our framework can substantially enhance the diversity of the generated solutions. Our experiments indicate that using diverse mini-batches in reinforcement learning improves exploration and provides a basis for the effectiveness of this approach.

Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The experimental evaluation was enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725. We thank

References

Anari, N.; Gharan, S. O.; and Rezaei, A. 2016. Monte Carlo Markov chain algorithms for sampling strongly Rayleigh distributions and determinantal point processes. In *Conference on Learning Theory*, 103–115. PMLR.

Arthur, D.; and Vassilvitskii, S. 2007. k-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 1027–1035.

Arús-Pous, J.; Blaschke, T.; Ulander, S.; Reymond, J.-L.; Chen, H.; and Engkvist, O. 2019. Exploring the GDB-13 chemical space using deep generative models. *Journal of cheminformatics*, 11: 1–14.

Ashton, M.; Barnard, J.; Casset, F.; Charlton, M.; Downs, G.; Gorse, D.; Holliday, J.; Lahana, R.; and Willett, P. 2002. Identification of diverse database subsets using property-based and fragment-based molecular descriptions. *Quantitative Structure-Activity Relationships*, 21(6): 598–604.

Atance, S. R.; Diez, J. V.; Engkvist, O.; Olsson, S.; and Mercado, R. 2022. De novo drug design using reinforcement learning with graph-based deep generative models. *Journal of chemical information and modeling*, 62(20): 4863–4872.

Badia, A. P.; Sprechmann, P.; Vitvitskyi, A.; Guo, D.; Piot, B.; Kapturowski, S.; Tieleman, O.; Arjovsky, M.; Pritzel, A.; Bolt, A.; and Blundell, C. 2020. Never Give Up: Learning Directed Exploration Strategies. In *International Conference on Learning Representations*.

Bemis, G. W.; and Murcko, M. A. 1996. The properties of known drugs. 1. Molecular frameworks. *Journal of medicinal chemistry*, 39(15): 2887–2893.

Bickerton, G. R.; Paolini, G. V.; Besnard, J.; Muresan, S.; and Hopkins, A. L. 2012. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2): 90–98.

Blaschke, T.; Arús-Pous, J.; Chen, H.; Margreitter, C.; Tyrchan, C.; Engkvist, O.; Papadopoulos, K.; and Patronov, A. 2020a. REINVENT 2.0: an AI tool for de novo drug design. *Journal of chemical information and modeling*, 60(12): 5918–5922.

Blaschke, T.; Engkvist, O.; Bajorath, J.; and Chen, H. 2020b. Memory-assisted reinforcement learning for diverse molecular de novo design. *Journal of cheminformatics*, 12(1): 68.

Borodin, A.; and Rains, E. M. 2005. Eynard–Mehta theorem, Schur process, and their Pfaffian analogs. *Journal of statistical physics*, 121: 291–317.

Buitinck, L.; Louppe, G.; Blondel, M.; Pedregosa, F.; Mueller, A.; Grisel, O.; Niculae, V.; Prettenhofer, P.; Gramfort, A.; Grobler, J.; Layton, R.; VanderPlas, J.; Joly, A.; Holt, B.; and Varoquaux, G. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 108–122.

Burda, Y.; Edwards, H.; Storkey, A.; and Klimov, O. 2018. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*.

Calandriello, D.; Derezinski, M.; and Valko, M. 2020. Sampling from a k-DPP without looking at all items. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 6889–6899. Curran Associates, Inc.

Carhart, R. E.; Smith, D. H.; and Venkataraghavan, R. 1985. Atom pairs as molecular features in structure-activity studies: definition and applications. *Journal of Chemical Information and Computer Sciences*, 25(2): 64–73.

Chen, J.; Aggarwal, V.; and Lan, T. 2023. A unified algorithm framework for unsupervised discovery of skills based on determinantal point process. *Advances in Neural Information Processing Systems*, 36: 67925–67947.

Derezinski, M.; Calandriello, D.; and Valko, M. 2019. Exact sampling of determinantal point processes with sublinear time preprocessing. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32, 11546–11558. Curran Associates, Inc.

Dice, L. R. 1945. Measures of the amount of ecologic association between species. *Ecology*, 26(3): 297–302.

Dreiman, G. H.; Bictash, M.; Fish, P. V.; Griffin, L.; and Svensson, F. 2021. Changing the HTS paradigm: AI-driven iterative screening for hit finding. *SLAS DISCOVERY: Advancing the Science of Drug Discovery*, 26(2): 257–262.

Fan, Y.; Watkins, O.; Du, Y.; Liu, H.; Ryu, M.; Boutilier, C.; Abbeel, P.; Ghavamzadeh, M.; Lee, K.; and Lee, K. 2023. DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36: 79858–79885.

Gao, W.; Fu, T.; Sun, J.; and Coley, C. W. 2022. Sample Efficiency Matters: A Benchmark for Practical Molecular Optimization. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 21342–21357. Curran Associates, Inc.

Gaulton, A.; Hersey, A.; Nowotka, M.; Bento, A. P.; Chambers, J.; Mendez, D.; Mutowo, P.; Atkinson, F.; Bellis, L. J.; Cibrián-Uhalte, E.; et al. 2017. The ChEMBL database in 2017. *Nucleic acids research*, 45(D1): D945–D954.

Gautier, G.; Polito, G.; Bardenet, R.; and Valko, M. 2019. DPPy: DPP Sampling with Python. *Journal of Machine Learning Research - Machine Learning Open Source Software (JMLR-MLOSS)*. Code at http://github.com/guilgautier/DPPy/ Documentation at http://dppy.readthedocs.io/.

Grosse, J.; Fischer, R.; Garnett, R.; and Hennig, P. 2024. A Greedy Approximation for k-Determinantal Point Processes. In *International Conference on Artificial Intelligence and Statistics*, 3052–3060. PMLR.

Grua, E. M.; and Hoogendoorn, M. 2018. Exploring clustering techniques for effective reinforcement learning based personalization for health and wellbeing. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, 813–820. IEEE.

Gummesson Svensson, H.; Tyrchan, C.; Engkvist, O.; and Haghir Chehreghani, M. 2024. Utilizing reinforcement

learning for de novo drug design. *Machine Learning*, 113(7): 4811–4843.

Gummesson Svensson, H.; Tyrchan, C.; Engkvist, O.; and Haghir Chehreghani, M. 2025. Diversity-Aware Reinforcement Learning for de novo Drug Design. In Kwok, J., ed., *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, 9194–9204. International Joint Conferences on Artificial Intelligence Organization. AI4Tech: AI Enabling Technologies.

Guo, J.; and Schwaller, P. 2024. Augmented Memory: Sample-Efficient Generative Molecular Design with Reinforcement Learning. *Jacs Au*, 4(6): 2160–2172.

Haarnoja, T.; Tang, H.; Abbeel, P.; and Levine, S. 2017. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, 1352–1361. PMLR.

Hochreiter, S. 1997. Long Short-term Memory. *Neural Computation MIT-Press*.

Hu, X.; Liu, G.; Yao, Q.; Zhao, Y.; and Zhang, H. 2024. Hamiltonian diversity: effectively measuring molecular diversity by shortest Hamiltonian circuits. *Journal of Cheminformatics*, 16(1): 94.

Huang, K.; Fu, T.; Gao, W.; Zhao, Y.; Roohani, Y.; Leskovec, J.; Coley, C. W.; Xiao, C.; Sun, J.; and Zitnik, M. 2021. Therapeutics Data Commons: Machine Learning Datasets and Tasks for Drug Discovery and Development. *Proceedings of Neural Information Processing Systems, NeurIPS Datasets and Benchmarks*.

Huang, W.; Da Xu, R. Y.; and Oppermann, I. 2019. Efficient diversified mini-batch selection using variable high-layer features. In *Asian Conference on Machine Learning*, 300–315. PMLR.

Jasial, S.; Hu, Y.; Vogt, M.; and Bajorath, J. 2016. Activity-relevant similarity values for fingerprints and implications for similarity searching. *F1000Research*, 5(591).

Kariv, O.; and Hakimi, S. L. 1979. An algorithmic approach to network location problems. I: The p-centers. *SIAM journal on applied mathematics*, 37(3): 513–538.

Kingma, D. P.; and Ba, J. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980.

Kulesza, A.; and Taskar, B. 2011. k-dpps: Fixed-size determinantal point processes. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 1193–1200.

Kulesza, A.; and Taskar, B. 2012. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3): 123–286.

Landrum, G. 2006. RDKit: Open-source cheminformatics. http://www.rdkit.org.

Li, C.; Jegelka, S.; and Sra, S. 2016. Efficient sampling for k-determinantal point processes. In *Artificial Intelligence and Statistics*, 1328–1337. PMLR.

Li, Y.; Zhang, L.; and Liu, Z. 2018. Multi-objective de novo drug design with conditional graph generative model. *Journal of cheminformatics*, 10: 1–24.

Liu, Y.; Shen, Z.; Zhang, Y.; and Cui, L. 2021. Diversity-promoting deep reinforcement learning for interactive recommendation. In *5th international conference on crowd science and engineering*, 132–139.

Loeffler, H. H.; He, J.; Tibo, A.; Janet, J. P.; Voronov, A.; Mervin, L. H.; and Engkvist, O. 2024. Reinvent 4: Modern AI–driven generative molecule design. *Journal of Cheminformatics*, 16(1): 20.

McInnes, L.; Healy, J.; and Melville, J. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 1928–1937. PmLR.

Mouchlis, V. D.; Afantitis, A.; Serra, A.; Fratello, M.; Papadiamantis, A. G.; Aidinis, V.; Lynch, I.; Greco, D.; and Melagraki, G. 2021. Advances in de novo drug design: from conventional to machine learning methods. *International journal of molecular sciences*, 22(4): 1676.

Nava, E.; Mutny, M.; and Krause, A. 2022. Diversified sampling for batched Bayesian optimization with determinantal point processes. In *International Conference on Artificial Intelligence and Statistics*, 7031–7054. PMLR.

O'Donoghue, B.; Munos, R.; Kavukcuoglu, K.; and Mnih, V. 2017. Combining policy gradient and Q-learning. In *International Conference on Learning Representations*.

Olivecrona, M.; Blaschke, T.; Engkvist, O.; and Chen, H. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9: 1–14.

Osogami, T.; and Raymond, R. 2019. Determinantal reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 4659–4666.

Paggi, J. M.; Pandit, A.; and Dror, R. O. 2024. The art and science of molecular docking. *Annual review of biochemistry*, 93(1): 389–410.

Pang, C.; Qiao, J.; Zeng, X.; Zou, Q.; and Wei, L. 2023. Deep generative models in de novo drug molecule generation. *Journal of Chemical Information and Modeling*, 64(7): 2174–2194.

Park, J.; Ahn, J.; Choi, J.; and Kim, J. 2024. MolAIR: Molecular Reinforcement Learning with Adaptive Intrinsic Rewards for Goal-directed Molecular Generation. arXiv:2403.20109.

Patronov, A.; Papadopoulos, K.; and Engkvist, O. 2021. Has artificial intelligence impacted drug discovery? In *Artificial Intelligence in Drug Design*, 153–176. Springer.

Pitt, W. R.; Bentley, J.; Boldron, C.; Colliandre, L.; Esposito, C.; Frush, E. H.; Kopec, J.; Labouille, S.; Meneyrol, J.; Pardoe, D. A.; Palazzesi, F.; Pozzan, A.; Remington, J. M.; Rex, R.; Southey, M.; Vishwakarma, S.; and Walker, P. 2025. Real-World Applications and Experiences of AI/ML Deployment for Drug Discovery. *Journal of Medicinal Chemistry*. PMID: 39772505.

Polishchuk, P. G.; Madzhidov, T. I.; and Varnek, A. 2013. Estimation of the size of drug-like chemical space based on GDB-17 data. *Journal of computer-aided molecular design*, 27: 675–679.

Rdusseeun, L.; and Kaufman, P. 1987. Clustering by means of medoids. In *Proceedings of the statistical data analysis based on the L1 norm conference, neuchatel, switzerland*, volume 31, 28.

Renz, P.; Luukkonen, S.; and Klambauer, G. 2024. Diverse Hits in De Novo Molecule Design: Diversity-Based Comparison of Goal-Directed Generators. *Journal of Chemical Information and Modeling*, 64(15): 5756–5761.

Rezaei, A.; and Gharan, S. O. 2019. A polynomial time MCMC method for sampling from continuous determinantal point processes. In *International Conference on Machine Learning*, 5438–5447. PMLR.

Schubert, E.; and Lenssen, L. 2022. Fast k-medoids Clustering in Rust and Python. *Journal of Open Source Software*, 7(75): 4183.

Schubert, E.; and Rousseeuw, P. J. 2019. Faster k-medoids clustering: improving the PAM, CLARA, and CLARANS algorithms. In *International conference on similarity search and applications*, 171–187. Springer.

Schubert, E.; and Rousseeuw, P. J. 2021. Fast and eager k-medoids clustering: O (k) runtime improvement of the PAM, CLARA, and CLARANS algorithms. *Information Systems*, 101: 101804.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Segler, M. H.; Kogej, T.; Tyrchan, C.; and Waller, M. P. 2018. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1): 120–131.

Seo, Y.; Chen, L.; Shin, J.; Lee, H.; Abbeel, P.; and Lee, K. 2021. State entropy maximization with random encoders for efficient exploration. In *International Conference on Machine Learning*, 9443–9454. PMLR.

Sheikh, H.; Frisbee, K.; and Phielipp, M. 2022. DNS: Determinantal point process based neural network sampler for ensemble reinforcement learning. In *International Conference on Machine Learning*, 19731–19746. PMLR.

Sorensen, T. 1948. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. *Biologiske skrifter*, 5: 1–34.

Tan, Y.; Dai, L.; Huang, W.; Guo, Y.; Zheng, S.; Lei, J.; Chen, H.; and Yang, Y. 2022. DRlinker: deep reinforcement learning for optimization in fragment linking design. *Journal of Chemical Information and Modeling*, 62(23): 5907–5917.

Tang, H.; Houthooft, R.; Foote, D.; Stooke, A.; Xi Chen, O.; Duan, Y.; Schulman, J.; DeTurck, F.; and Abbeel, P. 2017. #Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30, 2753–2762. Curran Associates, Inc.

Thomas, M.; O'Boyle, N. M.; Bender, A.; and Graaf, C. D. 2022. Re-evaluating sample efficiency in de novo molecule generation. arXiv:2212.01385.

Tong, X.; Liu, X.; Tan, X.; Li, X.; Jiang, J.; Xiong, Z.; Xu, T.; Jiang, H.; Qiao, N.; and Zheng, M. 2021. Generative models for de novo drug design. *Journal of Medicinal Chemistry*, 64(19): 14011–14027.

Velez-Arce, A.; Huang, K.; Li, M.; Lin, X.; Gao, W.; Fu, T.; Kellis, M.; Pentelute, B. L.; and Zitnik, M. 2024. TDC-2: Multimodal Foundation for Therapeutic Science. *bioRxiv*.

Wang, J.; and Zhu, F. 2024. ExSelfRL: An exploration-inspired self-supervised reinforcement learning approach to molecular generation. *Expert Systems with Applications*, 125410.

Weininger, D. 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1): 31–36.

Xie, Y.; Xu, Z.; Ma, J.; and Mei, Q. 2023. How Much Space Has Been Explored? Measuring the Chemical Space Covered by Databases and Machine-Generated Molecules. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Yang, Y.; Wen, Y.; Wang, J.; Chen, L.; Shao, K.; Mguni, D.; and Zhang, W. 2020. Multi-agent determinantal q-learning. In *International Conference on Machine Learning*, 10757–10766. PMLR.

Zhai, S.; Bai, H.; Lin, Z.; Pan, J.; Tong, P.; Zhou, Y.; Suhr, A.; Xie, S.; LeCun, Y.; Ma, Y.; et al. 2024. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *Advances in Neural Information Processing Systems*, 37: 110935–110971.

Zhang, C.; Kjellström, H.; and Mandt, S. 2017. Determinantal point processes for mini-batch diversification. In *Uncertainty in Artificial Intelligence-Proceedings of the 33rd Conference, UAI 2017*.

DPP-A	DPP-T	DPP-P	DPP-D
$L_T + L_D$	L_T	$L_T \odot L_D$	L_D

Table 1: Different kernel matrix L configurations. L_T consists of Tanimoto similarities between Morgan fingerprints and L_D consists of Dice similarities between atom-pair fingerprints. $\odot$ denotes element-wise multiplication.

A Kernel Matrix for DPP

To obtain a diverse mini-batch, we perform exact sampling from a k-DPP using the Gram-Schmidt sampler implemented in DPPy (Gautier et al. 2019). Performing exact sampling from the k-DPP typically requires an eigendecomposition of its kernel (Kulesza and Taskar 2011), typically requiring $\mathcal{O}(N^3)$ time. Given a decomposition, drawing a sample typically takes $\mathcal{O}(NK^3)$ time overall (Kulesza and Taskar 2012). For more details, we refer to (Derezinski, Calandriello, and Valko 2019; Calandriello, Derezinski, and Valko 2020) for more efficient exact sampling procedures, (Li, Jegelka, and Sra 2016; Grosse et al. 2024) for approximative methods and (Anari, Gharan, and Rezaei 2016; Rezaei and Gharan 2019) for Markov-Chain-Monte-Carlo (MCMC) procedures.

To perform sampling from a k-DPP, a kernel matrix L needs to be constructed at each generative step. We explore two different approaches to measure the similarity between molecules, resulting in two base kernel matrices that incorporate varying levels of information. The first base kernel matrix is constructed by the Tanimoto similarity between the corresponding 2048-bit Morgan fingerprints (with radius 2 using RDKit (Landrum 2006)) of the generated SMILES. We denote this base matrix by L_T. To incorporate more scaffold-based information, we also create a base kernel matrix by computing the Dice coefficients (Dice 1945; Sorensen 1948) between the scaffolds' atom pair fingerprints (Carhart, Smith, and Venkataraghavan 1985). We denote this base kernel matrix by L_D.

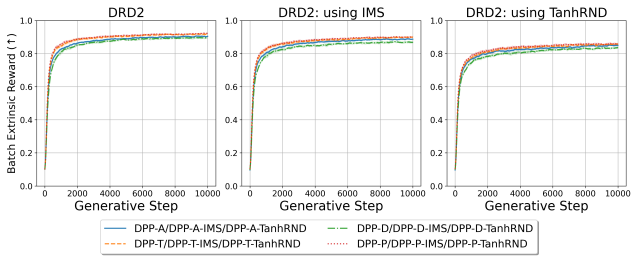
We investigate four combinations of L_T and L_D to create the kernel matrix L used for sampling from a k-DPP (see table 1). We obtain the first variant by element-wise summation of L_T and L_D, which we denote by DPP-A. Note that taking an element-wise arithmetic mean instead, i.e., multiplying a constant term $1/2$ with all items in L, does not change the probabilities and, therefore, would make no difference in practice for sampling. The second variant is obtained by only using L_T, which we denote by DPP-T. The third variant is obtained by the element-wise product of the two matrices, which we denote by DPP-P. The last variant is obtained only using L_D and is denoted by DPP-D. This results in four different configurations of DPP. For each kernel matrix in table 1 for DPP, we study how it affects the quality and diversity on the different reward functions. We investigate mini-batch diversification in combination with different techniques to modify the reward function (for enhancing exploration and diversity). Figure 5 displays the average extrinsic reward and standard deviation per generative step on the DRD2-, GSK3β-, or JNK3-based reward functions. For clarity of presentation, we display the moving averages with a window size of 101. Each line shows the average, while the shaded area shows the standard deviation. For all different configurations of DPP, we observe similar trends in terms of extrinsic rewards. Figure 6 displays the total number of diverse activities up to the current generative steps. The total number of diverse actives is plotted for every 250th generative step. For the DRD2-based reward functions, both DPP-A and DPP-T generate among the largest number of diverse actives. For the GSK3β-based reward function, DPP-T often generates the largest number of diverse actives, while DPP is the second-best configuration. For the JNK3-based reward function, the variability of all methods is high when they can generate more than around 10 diverse actives. We observe that DPP-A often displays the largest number of diverse actives when using the IMS and TanhRND technique to modify the reward. Figure 7 shows the total number of molecular scaffolds up to the current generative step. For the DRD2-based reward function, DPP-A consistently generates the largest number of scaffolds. On GSK3β, DPP-D generates the largest number of scaffolds, while DPP-A is the second-best method. There is a high variability for the JNK3 reward. DPP-A displays a large average number of scaffolds. Overall, we observe that DPP-A consistently displays a good balance between the different diversity metrics. Therefore, we use this method in the main paper to represent mini-batch diversification via DPP sampling.

B Kernel Matrix for Maximum Coverage

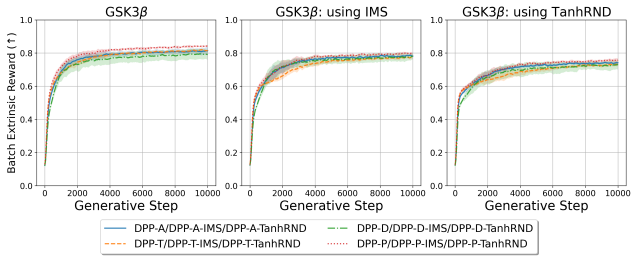
We also investigate three different dissimilarity functions for the MaxMin algorithm and k-medoids clustering. We also refer to kernel matrices for the MaxMin algorithm and k-medoids clustering. Thus, we explore the following configurations of the MaxMin algorithm and k-medoids clustering: (1) "MaxMin-T"/"kMedoids-T" using the Tanimoto similarity between the Morgan fingerprints described above, which corresponds to using kernel matrix L_T; (2) "MaxMin-D"/"kMedoids-D" using the Dice similarity with the atom pair fingerprints described above, which corresponds to L_D; (3) "MaxMin-A"/"kMedoids-A" using the average Tanimoto and Dice similarities, which corresponds to $\frac{L_D+L_T}{2}$. This results in 3 different configurations for the MaxMin algorithm and k-medoids clustering. We also denote these dissimilarity functions as kernel matrices. For each dissimilarity function for the MaxMin algorithm and k-medoids clustering, we study how it affects the quality and diversity on the different reward functions. We investigate mini-batch diversification in combination with different techniques to modify the reward function (for enhancing exploration and diversity).

B.1 MaxMin Algorithm

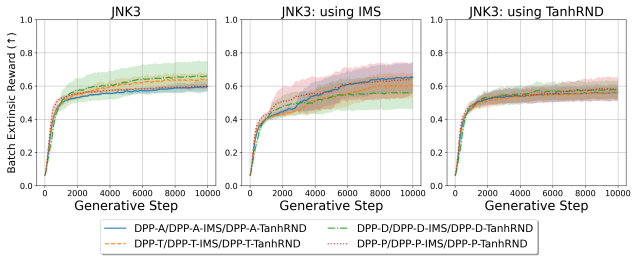
Figure 8 displays the extrinsic reward per generative step on the DRD2-, GSK3β-, and JNK3-based reward functions. For clarity of presentation, we display the moving averages with a window size of 101. Each line shows the average, while the shaded area shows the standard deviation. For all configurations of the MaxMin algorithm, we mostly observe similar



(a) DRD2

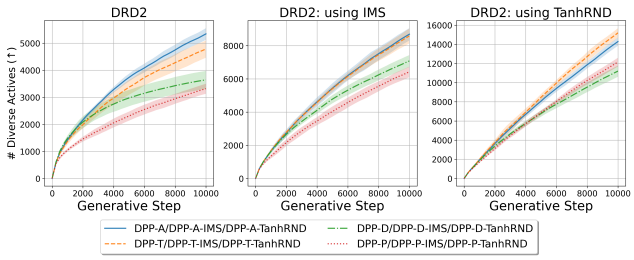


(b) GSK3β

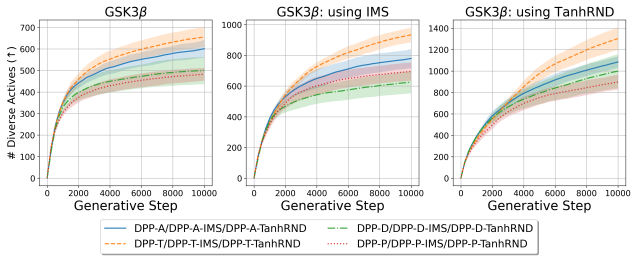


(c) JNK3

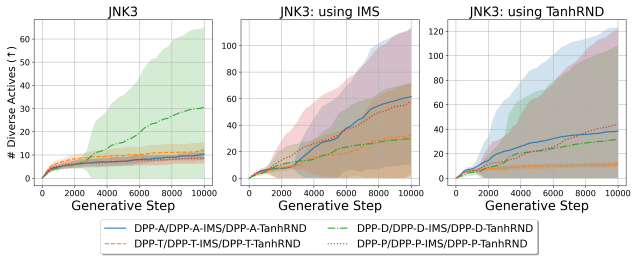
Figure 5: Average extrinsic rewards per generative step across the mini-batch of SMILES evaluated on the DRD2-, GSK3β-, or JNK3-based reward functions. For clarity of presentation, we display the moving averages with a window size of 101.



(a) DRD2



(b) GSK3β



(c) JNK3

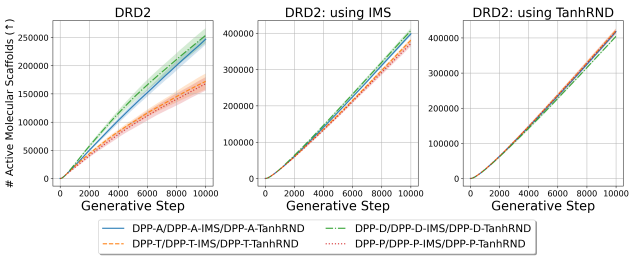
Figure 6: Total number of diverse activities after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model. The total number of diverse actives is plotted for every 250th generative step.

extrinsic rewards, but MaxMin-D sometimes displays lower and sometimes higher rewards. Figure 9 shows the total number of diverse actives up to the current generative step. For all experiments, except MaxMin-D-TanhRND on JNK3, MaxMin-T generated the largest number of diverse actives, while MaxMin-A is second-best. When using TanhRND on the JNK3-based reward function, all configurations display similar results, with high variability. Figure 10 shows the total number of molecular scaffolds up to the current generative step. For the DRD2-based reward function, all configurations show similar trends when using IMS or TanhRND to enhance exploration, where MaxMin-T generates the largest number of scaffolds across all experiments. For the GSK3β-based reward function, MaxMin-T generates the smallest number of scaffolds, while MaxMin-D and MaxMin-A yield the largest and second largest number of scaffolds, respectively. On the JNK3 problem, MaxMin-A and MaxMin-T display similar trends when using IMS or no reward modification, yielding a larger number of scaffolds compared to MaxMin-D, which stagnates after a few thousand steps. When using TanhRND, all configurations display a large variability, where MaxMin-D yields the largest average and MaxMin-T the smallest average.

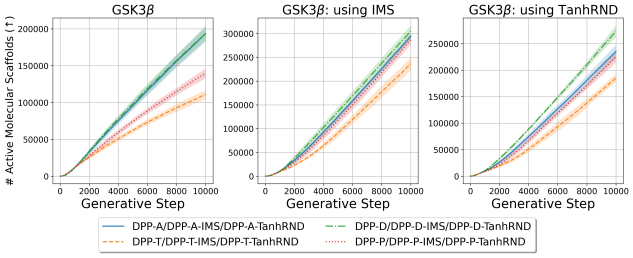
Overall, MaxMin-T generates the largest number of diverse actives, while MaxMin-A illustrates comparable diversity and better diversity in terms of scaffolds. MaxMin-A better balances the two different diversity metrics and, therefore, we use this configuration in the main paper.

B.2 k-Medoids Clustering

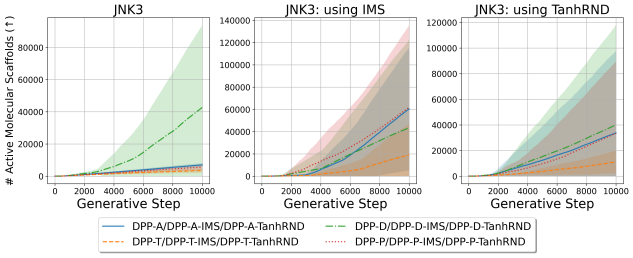
Figure 11 displays the extrinsic reward per generative step on the DRD2-, GSK3β-, and JNK3-based reward functions when using k-medoids clustering for mini-batch diversification. For clarity of presentation, we display the moving averages with a window size of 101. Each line shows the average, while the shaded area shows the standard deviation. For all different kernel matrices explored for k-medoids clustering, we observe similar trends. Rewards on the DRD2 problem are above 0.8, rewards on GSK3β are mostly between 0.8 and 0.6, and rewards on JNK3 are primarily below 0.6. Figure 12 shows the total number of diverse actives up to the



(a) DRD2

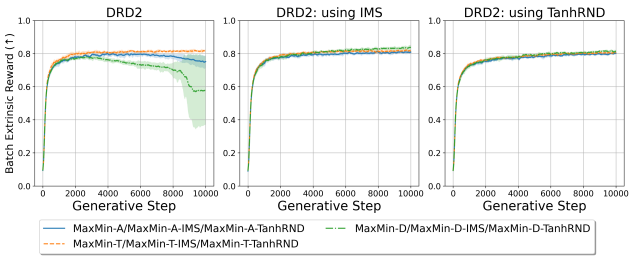


(b) GSK3β

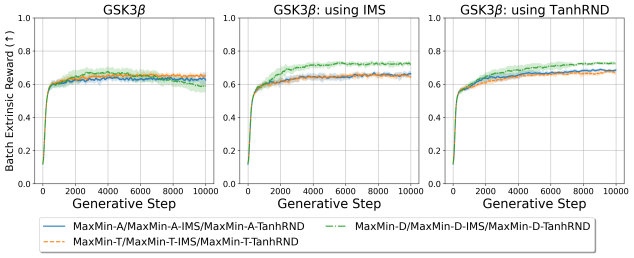


(c) JNK3

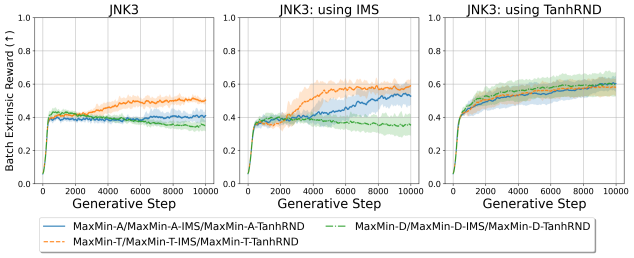
Figure 7: Total number of molecular scaffolds after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model.



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 8: Average extrinsic rewards per generative step across the mini-batch of SMILES evaluated on the DRD2-, GSK3β-, or JNK3-based reward functions. For clarity of presentation, we display the moving averages with a window size of 101.

current generative step. On the DRD2-based reward function (see figure 12a), kMedoids-D consistently yields the largest number of diverse actives, while kMedoids-A is slightly better than kMedoids-T. For the experiments on GSK3β (see figure 12b), kMedoids-T yields the largest number of diverse actives when using TanhRND, but otherwise generates a smaller number of diverse actives. kMedoids-A generates the second largest average number of diverse actives across all experiments, but its standard deviation overlaps with the other methods. For the JNK3-based reward function (see figure 12c), all methods generate a similar number of diverse actives. Figure 13 shows the total number of molecular scaffolds up to the current generative step. When modifying the reward (see middle and right plots in Figure 13), the experiments of kMedoids-A generate the largest number of scaffolds, but their standard deviations overlap with the ones of kMedoids-T. When not modifying the extrinsic reward (see left plots in Figure 13), fewer scaffolds are generated, where kMedoids-D performs the best.

Overall, both kMedoids-D and kMedoids-T generate the largest number of diverse actives (for different reward functions), while kMedoids-A or kMedoids-T yield the largest number of scaffolds. We argue that kMedoids-A best balances the benefits of kMedoids-T and kMedoids-D, since it is always the second-best or best method. Therefore, we use this configuration in the main paper.

C Experimental Detatils

The *de novo* drug design problem can be modeled as a Markov decision process (MDP). Then, $a_t \in \mathcal{A}$ is the action taken at state s_t. We can define the current state as the sequence of performed actions up to round t

$$s_t \coloneqq a_{0:t-1} = a_0, a_1, \ldots, a_{t-1}, \tag{8}$$

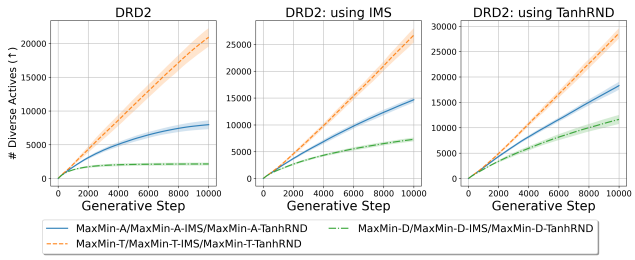
where the initial action is always the start token $a_0 = a^{\text{start}}$. This means that the distribution of the initial state s_0 is deterministic $p_0(s_0 = a^{\text{start}}) = 0$. The transition probabilities are deterministic

$$P(s_{t+1}|s_t, a_t) = \delta_{s_t + + a_t}, \tag{9}$$

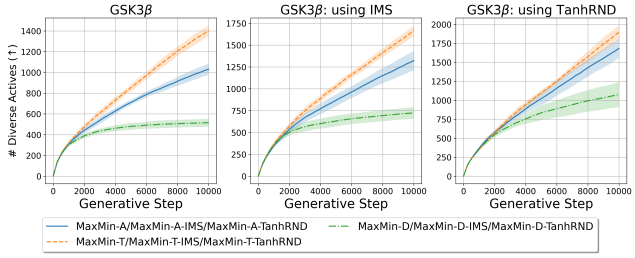
where $++$ denotes the concatenation of two sequences. If action a^{stop} is taken, the following state is terminal, stopping

Algorithm 2: Diverse Mini-Batch Selection for Drug Design

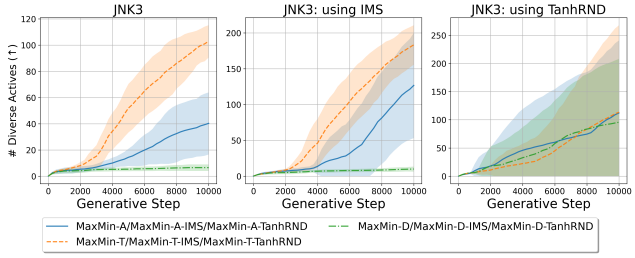
input: $G, B, k, \theta_{prior}, h$
$\mathcal{M} \leftarrow \emptyset$ ▷ Initialize memory
$\theta \leftarrow \theta_{\text{prior}}$ ▷ The prior policy is fine-tuned
for $g = 1, \ldots, G$ **do** ▷ Generative steps
  $\mathcal{L}(\theta) \leftarrow 0$
  $\mathcal{K} \leftarrow \emptyset$
  for $b = 1, \ldots, B$ **do** ▷ Large batch of SMILES
    $t \leftarrow 0$
    $a_t \leftarrow a^{(\text{start})}$ ▷ Start token is initial action
    $s_{t+1} \leftarrow a_t$
    while s_{t+1} is not terminal **do**
      $t \leftarrow t + 1$
      $a_t \sim \pi_\theta(s_t)$
      $s_{t+1} \leftarrow a_{0:t}$
    end while
    $\mathcal{B} \leftarrow \mathcal{B} \cup s_{t+1}$
    Observe property score $r(s_{t+1})$
    if $r(s_{t+1}) \geq h$ **then**
      $\mathcal{M} \leftarrow \mathcal{M} \cup \{s_{t+1}\}$
    end if
    Compute and store penalty $f(s_{t+1}; \mathcal{M})$
  end for
  Compute matrix kernel L over $\mathcal{B}$
  Select k SMILES from $\mathcal{B}$
  for $A \in Y$ **do**
    Compute intrinsic reward $R_I(A; \mathcal{M})$
    Computed modified reward $\hat{R}(A)$
    Compute loss $\mathcal{L}_A(\theta)$ wrt $\hat{R}(A)$
    $\mathcal{L}(\theta) \leftarrow \mathcal{L}(\theta) + \mathcal{L}_A(\theta)$
  end for
  Update θ by minimizing $\mathcal{L}(\theta)$ in equation (13)
end for
output: $\theta, \mathcal{M}$



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 9: Total number of diverse activities after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model. The total number of diverse actives is plotted for every 250th generative step.

the current generation process and subsequently evaluating the generated molecule,

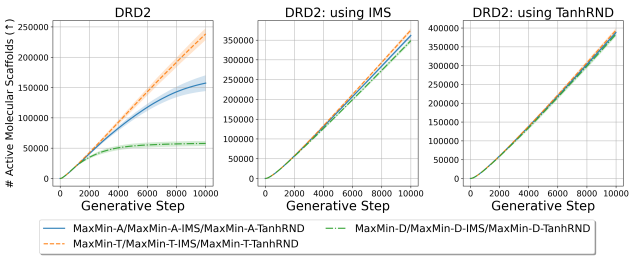
$$P(\text{terminal}|s_t, a^{\text{stop}}) = 1, \tag{10}$$

where δ_z denotes the Dirac distribution at z. The extrinsic reward epsidoci such that

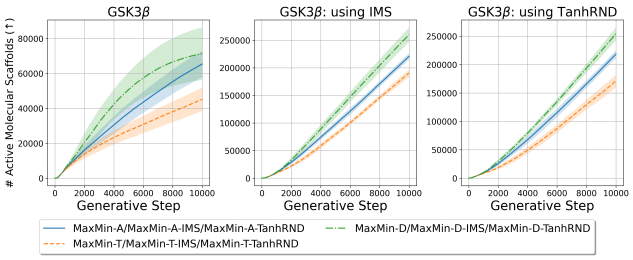
$$R(s_t, a_t) = R(a_{0:t}) = \begin{cases} r(s_{t+1}) & \text{if } a_t = a^{\text{stop}}, \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

where reward $r(s_T) \in [0, 1]$ (only observable at a terminal state) measures the desired property, which we want to optimize, of molecule $A = a_{1:T-2}$. We let T denote the round that a terminal state is visited, i.e., $a_{T-1} = a^{\text{stop}}$. Note that in practice, the string between the start and stop tokens encodes a molecule such that $a_{1:T-2}$ is equivalent to $a_{0:T-1}$ during evaluation. The objective is to fine-tune a policy π_θ, parameterized by θ, to generate a structurally diverse set of molecules optimizing the property score $r(\cdot)$.

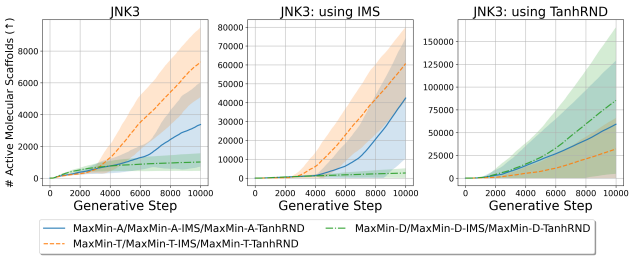
We use the REINVENT4 (Loeffler et al. 2024) framework to sequentially fine-tune the pre-trained (prior) policy. The algorithm is based on the *augmented log-likelihood* defined



(a) DRD2

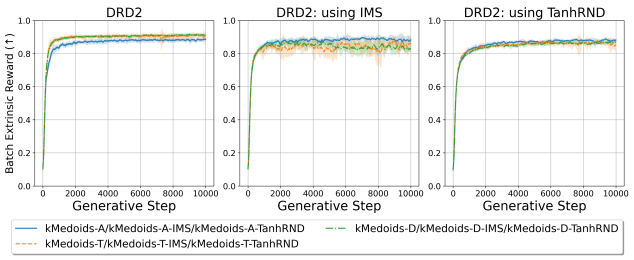


(b) GSK3β

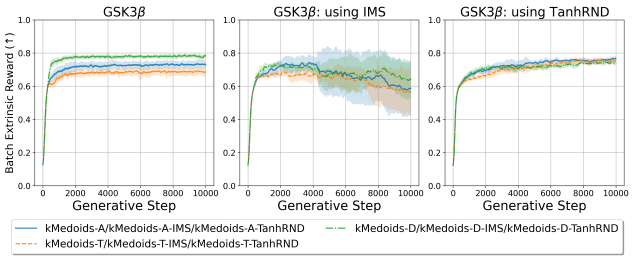


(c) JNK3

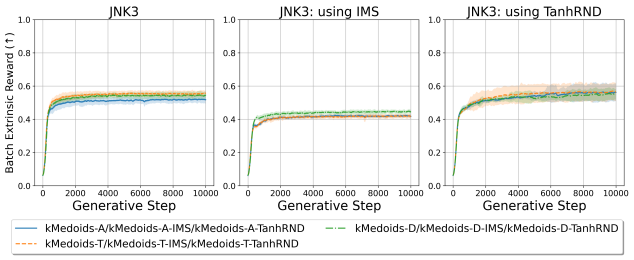
Figure 10: Total number of molecular scaffolds after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model.



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 11: Average extrinsic rewards per generative step across the mini-batch of SMILES evaluated on the DRD2-, GSK3β-, or JNK3-based reward functions. For clarity of presentation, we display the moving averages with a window size of 101.

by

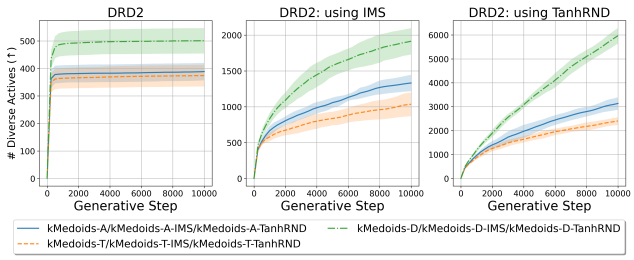
$$\log \pi_{\theta_{\text{aug}}}(A) \coloneqq \sum_{t=1}^{T-2} \log \pi_{\theta_{\text{prior}}}(a_t|s_t) + \sigma R(A), \qquad (12)$$

where $A = a_{1:T-2}$ is a generated molecule, σ is a scalar value, $\pi_{\theta_{\text{prior}}}$ is the (fixed) pre-trained policy. The policy π_θ is optimized by minimizing the squared difference between the augmented log-likelihood and policy likelihood given a mini-batch Y of k SMILES

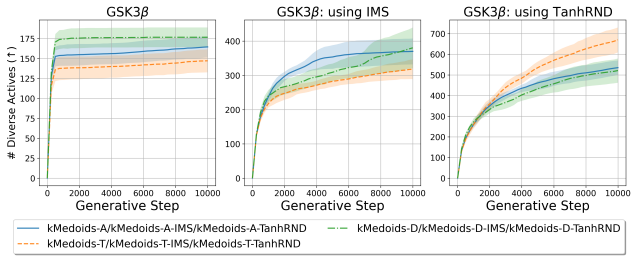
$$\mathcal{L}(\theta) = \frac{1}{k} \sum_{a_{1:T-2} \in Y} \Big(\log \pi_{\theta_{\text{aug}}}(a_{1:T-2}) - \sum_{t=1}^{T-2} \log \pi_\theta(a_t|s_t) \Big)^2 . \qquad (13)$$

Previous work has shown that minimizing this loss function is equivalent to maximizing the expected return, as for policy gradient algorithms (Guo and Schwaller 2024).

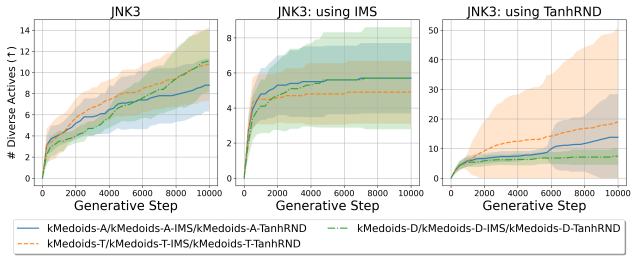
In practice, at each step g of the generative process, B full trajectories/episodes (until reaching a terminal state) are rolled out, to obtain a batch $\mathcal{B}$ of generated SMILES. Each token in the SMILES is sampled from the multinomial distribution induced by the policy's action probabilities. Subsequently, k-DPP, the MaxMin algorithm or k-medoids clustering is used to select a mini-batch of k trajectories (SMILES) from $\mathcal{B}$. The modified reward $\hat{R}(A)$ for each molecule $A \in Y$ is observed by the agent and subsequently used for fine-tuning. The modified reward $\hat{R}(A)$ is computed using the penalty function $f(A)$ and/or intrinsic reward R_I (depending on which reward function is used). The penalty functions and intrinsic rewards use a bucket size of M to determine the desired number of generated molecules with the same scaffold (we refer to (Blaschke et al. 2020b; Gummesson Svensson et al. 2025) for more details). The modified reward is used to compute the loss $\mathcal{L}(\theta)$ in equation (13), i.e., we let $R(A) = \hat{R}(A)$ if the extrinsic reward is modified, e.g., via intrinsic reward or reward penalty. The policy parameters θ are updated with respect to the $\mathcal{L}(\theta)$ using a learning rate α. Algorithm 2 illustrates the specific procedure utilized for *de novo* drug design. The source code is



(a) DRD2

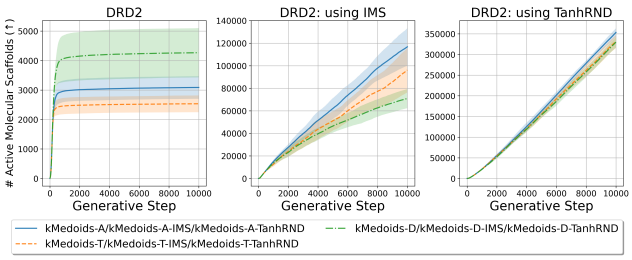


(b) GSK3β

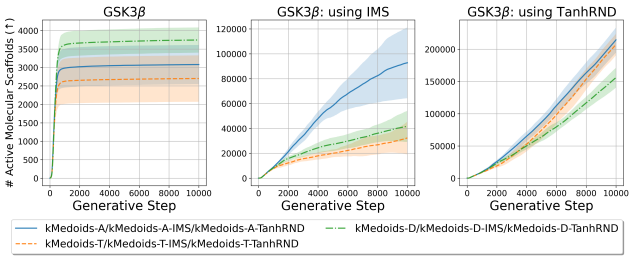


(c) JNK3

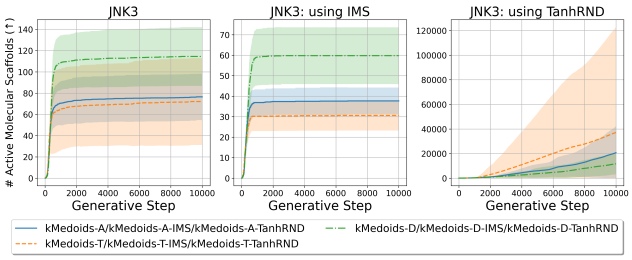
Figure 12: Total number of diverse activities after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model. The total number of diverse actives is plotted for every 250th generative step.



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 13: Total number of molecular scaffolds after g generative steps evaluated on reward functions based on the DRD2, GSK3β, or JNK3 predictive model.

available in a GitHub repository.[2] Fine-tuning of the policy network is done on a single NVIDIA A40 GPU with 48GB RAM or NVIDIA T4 GPU with 16GB RAM using PyTorch 1.12.1 and CUDA 11.3 on a Linux-based system. We use the DPPy package (Gautier et al. 2019) with version 0.3.3 to perform exact sampling from k-DPP, using the default random seed. For k-medoids clustering, we use the FasterPAM algorithm (Schubert and Rousseeuw 2021) from the kmedoids package (Schubert and Lenssen 2022) with version 0.5.3.1. We use random initialization of medoids and at most 100 iterations. We use the MaxMin algorithm implemented by RDKit (Landrum 2006) with version 2023.9.6.

C.1 Reward Function

Our experiments utilize scoring components of REINVENT4 (Loeffler et al. 2024) to define the extrinsic reward using a geometric mean. In addition, we implement a scoring component using the predictive oracles of the Dopamine Receptor D2 (DRD2), Glycogen Synthase Kinase 3 Beta (GSK3β) and c-Jun N-terminal Kinases-3 (JNK3) oracles from TD Commons (Huang et al. 2021; Velez-Arce et al. 2024). The weight and parameters for each scoring component are displayed in table 2. Predictive oracle functions, providing the activity values, are provided by PyTDC 1.1.4. Fingerprints and QED are computed using RDKit 2023.9.6. For the custom alerts, we use the following default chemical patterns in the SMARTS language:

- [*;r8]
- [*;r9]
- [*;r10]
- [*;r11]
- [*;r12]
- [*;r13]
- [*;r14]
- [*;r15]
- [*;r16]
- [*;r17]
- [#8][#8]
- [#6;+]
- [#16][#16]

[2]https://github.com/hampusgs/diverse-mini-batch-selection-rl

Component	Weight	Transform type	high	low	c_{div}	c_{si}	c_{se}	k
Molecular weight	1	Double sigmoid	550	200	500	20	20	–
# hydrogen bond doners	1	Reverse sigmoid	6	2	–	–	–	0.5
QED	1	None	–	–	–	–	–	–
Custom Alerts	1	None	–	–	–	–	–	–
Predictive oracle	5	None	–	–	–	–	–	–

Table 2: Parameters for scoring components in the REINVENT4 (Loeffler et al. 2024) framework. A geometric mean is used to combine them into the extrinsic reward observed by the agent.

- [#7;!n][S;!$(S(=O)=O)]
- [#7;!n][#7;!n]
- C#C
- C(=[O,S])[O,S]
- [#7;!n][C;!$(C(=[O,N])[N,O])][#16;!s]
- [#7;!n][C;!$(C(=[O,N])[N,O])][#7;!n]
- [#7;!n][C;!$(C(=[O,N])[N,O])][#8;!o]
- [#8;!o][C;!$(C(=[O,N])[N,O])][#16;!s]
- [#8;!o][C;!$(C(=[O,N])[N,O])][#8;!o]
- [#16;!s][C;!$(C(=[O,N])[N,O])][#16;!s]

C.2 Hyperparameters

Table 3 displays the hyperparameters used in the experimental evaluation. We run for $G = 10000$ generative/training steps to investigate the chemical exploration over a large number of steps. We generate a large set $\mathcal{B}$ of $|\mathcal{B}| = B = 640$ instances/items since we argue that ten times the number of items we want to choose (i.e., k) is sufficient to generate diverse solutions. This is supported by our experiments. We use a distance threshold $D = 0.7$ as suggested by (Renz, Luukkonen, and Klambauer 2024) since there is a significant decrease in the probability of similar bioactives beyond this threshold (Jasial et al. 2016). When computing the diversity in terms of both scaffolds and diverse actives, we only regard active molecules, defined as molecules with both QED and predicted activity larger than $h = 0.5$. The activity models are trained on binary classification tasks, such that a value larger than $h = 0.5$ means that the molecule is most likely to be active. A QED of 0.5 is close to the mean QED of approved drugs (Bickerton et al. 2012). Otherwise, we use the default hyperparameters of REINVENT4 (Loeffler et al. 2024).

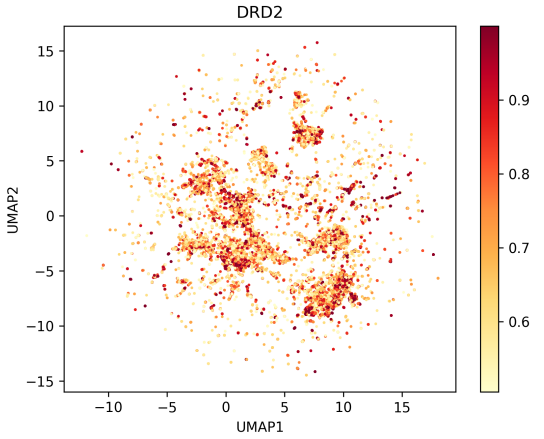
D Analysis of predictive activity models

To better understand the underlying reward space, we visualize the molecules from ChEMBL25 (Gaulton et al. 2017), in total 2474589 molecules, on the three predictive activity models (oracles) investigated in this work. ECFP (Morgan) fingerprints with 2048 bits are reduced to 200 features using principal components analysis (PCA) using scikit-learn (Buitinck et al. 2013). These 200 features are subsequently reduced to 2 dimensions using UMAP (McInnes, Healy, and Melville 2018). For clarity, we only display the active molecules with a predicted activity of more than 0.5. Only fingerprints are used for the different predictive models to match the fingerprints used for the corresponding predictive models in Therapeutics Data Commons (Huang et al. 2021; Velez-Arce et al. 2024).

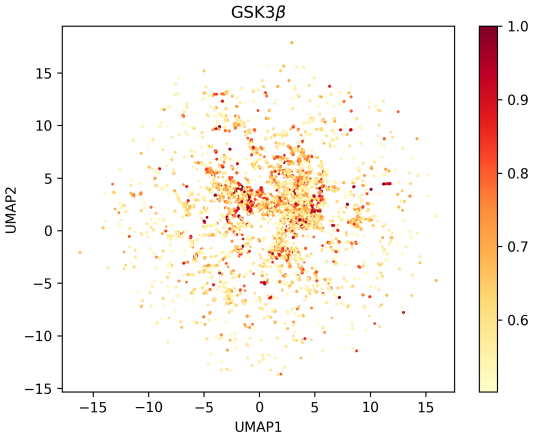
For DRD2 oracle, ECFPC3 fingerprints (using counts and features) are calculated using RDKit (Landrum 2006) and visualized in figure 14a. There are 58843 active molecules in total for the DRD2 oracle. For the GSK3β oracle, ECFP2 fingerprints are calculated using RDKit (Landrum 2006) and visualized in figure 14b. There are 44066 active molecules in total for the GSK3β oracle. For the JNK3 oracle, ECFP2 fingerprints are calculated using RDKit (Landrum 2006) and visualized in figure 14c. There are 7249 active molecules in total for the JNK3 oracle.

Parameter	Value
B	640
G	10 000
h	0.5
k	64
D	0.7
σ	128
α	0.0001
Optimzer	Adam (Kingma and Ba 2017)
M	25
$\lvert\mathcal{A}\rvert$	34

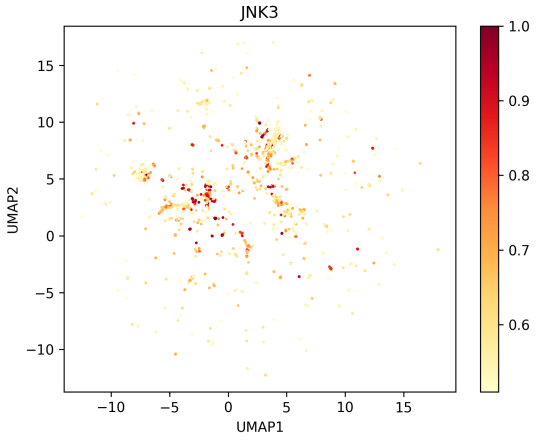
Table 3: Hyperparameters for the experimental evaluation using the REINVENT4 (Loeffler et al. 2024) framework.



(a) DRD2



(b) GSK3β



(c) JNK3

Figure 14: 2-dimensional UMAP projection of 200 PCA features. The PCA features are derived from 2048-bits ECFP (Morgan) fingerprints. We only display the active molecules with a predicted activity of more than 0.5.