.

Automating reward function configuration for drug design

Marius Urbonas[1] **Temitope Ajileye**[2] **Paul Gainer**[2] **Douglas Pires**[2]
[1]University of Oxford [2]Exscientia
marius.urbonas@stx.ox.ac.uk
{tajileye,pgainer,dpires}@exscientia.co.uk

Abstract

Designing reward functions that can guide generative molecular design (GMD) algorithms to desirable areas of chemical space is of critical importance in AI-driven drug discovery. Traditionally, this has been a manual and error-prone task; the selection of appropriate computational methods to approximate biological assays is challenging and the normalisation and aggregation of computed values into a single score even more so, leading to potential reliance on trial-and-error approaches [16]. We propose a novel approach for automated reward configuration that relies solely on experimental data, mitigating the challenges of manual reward adjustment on drug discovery projects. Our method achieves this by constructing a ranking over experimental data based on Pareto dominance over the multi-objective space, then training a neural network to approximate the reward function such that rankings determined by the predicted reward correlate with those determined by the Pareto dominance relation. We validate our method using two case studies. In the first study we simulate Design-Make-Test-Analyse (DMTA) cycles by alternating reward function updates and generative runs guided by that function. We show that the learned function adapts over time to yield compounds that score highly with respect to evaluation functions taken from the literature [4]. In the second study we apply our algorithm to historical data from four real drug discovery projects. We show that our algorithm yields reward functions that outperform the predictive accuracy of human-defined functions, achieving an improvement of up to 0.4 in Spearman's correlation against a ground truth evaluation function that encodes the target drug profile for that project. Our method provides an efficient data-driven way to configure reward functions for GMD, and serves as a strong baseline for future research into transformative approaches for the automation of drug discovery.

1 Introduction

Drug discovery is intrinsically a multi-objective optimisation problem [6]. Generative molecular design (GMD) algorithms explore the extensive space of drug-like molecules searching for compounds with desirable property profiles, simultaneously maximising or minimising multiple objectives [13]. Many GMD algorithms are designed to generate molecules that maximise a given reward function [2, 9, 17]. Reward functions are a way for designers to communicate their goals to their GMD tools, and directly impact how those tools explore chemical space. Therefore, defining reward functions that can lead to the generation of compounds that meet the goals of the project is of critical importance in drug discovery. However, translating project goals into appropriate reward functions is a challenging task. Decisions must be made on which combinations of computable *in silico* methods are appropriate to

The NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development (AI4D3 2023), New Orleans, LA, USA, 2023.

function as good proxies for the results of comparatively slow and expensive biological assays, and on how to normalise these computed values and combine them into a single score.

When reward functions consist of many components it becomes difficult to quantify the relationships between parameters. Humans in particular struggle at this task, just as they would struggle to quantify the relationships between the features in a high dimensional regression task. Sundin *et al.* [16] observe that the result of this can often be that drug designers may resort to trial and error to explore the parameter space. To alleviate this they introduce a method to integrate human feedback into the optimisation of reward functions employed to guide the REINVENT tool [2] using active learning. The main challenge addressed is the creation of a reward function that aligns with a human chemist's optimisation goals, and the proposed method allows the reward function to be learned directly from user feedback, eliminating the need for manual tuning through trial and error. While this method is shown to be effective for molecular generation, it relies on human expertise to assign binary preferences to generated compounds, and it has been demonstrated that there is low consensus between chemists when prioritising molecules [8, 12].

To eliminate the dependence on biased human feedback we propose a novel approach that relies solely on the results of biological assays to guide the learning of a suitable reward function. To achieve this we build on advancements in the field of Inverse Reinforcement Learning (IRL), that we use as the backbone of our method. IRL is the problem of learning a reward function by observing a set of expert demonstrations, where the optimality of any learned function depends on the quality of the demonstrations [14]. Agyemang *et al*. [1] use entropy maximisation to learn a reward model to guide a Reinforcement Learning agent to optimise single-objective tasks for molecular generation, including the octanol-water partition coefficient (logP) and binding to the DRD2 protein. They show that their approach is effective when high-quality molecular data exhibiting the objective is readily available. However, when optimising for multiple objectives it is difficult to obtain a set of high-quality demonstrations. For real drug discovery projects, available molecular data is usually scarce or nonexistent, and expensive to augment as the project progresses due to the high cost of conducting biological assays. This is particularly challenging during the early stages of a project, where only a handful of noisy observations are available.

To address this problem, we adopt the approach of Brown *et al.* [3] where a reward function that yields better-than-demonstrator performance is obtained by learning to justify preferential rankings over demonstrations, rather than learning to imitate them. There are two main approaches to construct preference rankings over molecules for multi-parameter optimisation tasks [13]: aggregation methods and Pareto-based methods. The first approach combines multiple objective functions into a single score that can be used to rank the molecules. However, the manual specification of the scalarisation function faces the same challenges as that of manually specifying a reward function, and encodes the bias of human chemists when exploring trade-offs between objectives.

The main contribution of this paper is a principled Pareto-based method to learn reward functions that relies solely on experimental assay data. We achieve this by defining partial rankings using pairwise Pareto dominance relationships, where one molecule is preferred to (*dominates*) another if it as least as good for every objective, and better for at least one objective. We then use these partial rankings to train the reward function as a preference predictor over molecules. We demonstrate empirically that our method can learn reward functions that yield rankings that outperform human-defined functions over data from real drug discovery projects. Furthermore, we apply our method to synthetic goals taken from a well established benchmark for GMD, and show that we can learn reward functions that can guide a method taken from that benchmark to generate compounds that score as highly as those generated using the target reward function.

In Section 2 we describe our proposed method, which we then evaluate using data from public benchmarks and real drug discovery projects in Section 3. We conclude the paper with a discussion of our results and an outline of future directions for work in Section 4.

2 Methods

Let $\mathcal{M} = \{m_1, m_2, \cdots, \}$ be a set of molecules, and let $\alpha_1, \cdots, \alpha_K$ be real valued evaluation functions over the molecular space: these functions map each molecule into $\mathbb{R}^K$. We will often refer to these functions as assays and think of molecules as points in that multi-dimensional space. Experimentally characterising each molecule designed by a discovery team is infeasible; the decision

of which subset to evaluate is guided by a reward function r, ideally inexpensive to compute, that gives higher scores to molecules that are closer to $\mathfrak{T}$ in evaluation space.

Our method consists of two main steps. In the first step we construct rankings in $\mathbb{R}^K$ based on Pareto dominance: this encodes the notion that, lacking any overriding directives, m_1 can only be said to be preferred over m_2 if it is closer to $\mathfrak{T}$ on all assays. In the second step we train a neural network to generate a reward function such that the rankings determined by the function correlate with those constructed in the previous step. The neural network is given access to the components that humans would usually include in their reward functions.

2.1 Pareto Dominance Ranking

Traditionally, in order to establish a preference relationship between two molecules, designers encode an opinion on the scaling functions and relative weights of each assay [13], so that an average of the results can be used to determine preferences. Our method only requires the selection of relevant evaluation functions, rather than the specification of those parameters, and as a consequence is less biased.

The evaluation-based ranking over $\mathcal{M}$ is constructed by determining Pareto dominance relationships over the multi-objective space defined by the results of K biological assays with respect to some point $\mathfrak{T}=(t_1,t_2,\ldots,t_k)$ that corresponds to a drug candidate criteria, where each $t_i \in \mathbb{R}$ is the target value for assay α_i, as illustrated in Figure 1.

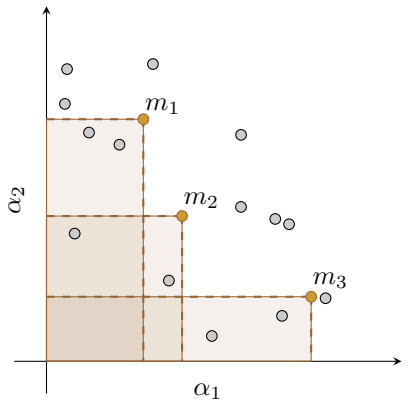

Figure 1: Example of Pareto dominance relationships: for each molecule m_i the corresponding shaded region shows the set $\mathrm{dom}(m_i)$ of molecules dominated by m_i in the multi-objective space defined by the results of two biological assays α_1 and α_2.

Definition 1. *Given two molecules m_1 and m_2, and a drug candidate criteria $\mathfrak{T} = (t_1, t_2, \ldots, t_K)$, we say that* m_1 dominates m_2 *if $|\alpha_i(m_1)-t_i| < |\alpha_i(m_2)-t_i|$ for all $1 \leq i \leq K$. We denote this relationship by $m_1 \succ m_2$.*

The set $\mathcal{D}_{\hat{t}}(\mathcal{M})$ of all domination pairs of molecules can then be constructed as:

$$\mathcal{D}_{\hat{t}}(\mathcal{M}) = \bigcup_{m_1 \in \mathcal{M}} \{(m_1, m_2) \mid m_2 \in \mathrm{dom}(m_1)\}, \tag{1}$$

where $\mathrm{dom}(m)$ denotes the set of molecules in $\mathcal{M}$ that are dominated by m.

2.2 The Reward Network

The learnable reward function is a composition of *components* $\{\mathrm{c}_i\}_{i=1,\ldots,N}$, where each component is a physics-based, machine-learned, or otherwise-computable function mapping compounds to scores in $\mathbb{R}$. Components are unbounded, therefore for each c_i there is a corresponding normalising function $\phi_i : \mathbb{R} \to [0, 1]$. To ensure interpretability and reliability of any learned function, we fix the shape of $\hat{r}_\theta$ to be a linear combination over components:

$$\hat{r}_\theta(m) = \mathbf{w} \cdot \begin{bmatrix} \phi_1(\mathrm{c}_1(m)) \\ \phi_2(\mathrm{c}_2(m)) \\ \ldots \\ \phi_N(\mathrm{c}_N(m)) \end{bmatrix} \tag{2}$$

We follow [16] and use two different normalising functions: a sigmoid function to bound a maximisation property and a Gaussian activation to bound a property that is expected to have an optimal range. In both cases the learning framework estimates the parameters of these functions; slope and midpoint for the sigmoid, and mean and variance for the Gaussian activation.

To train $\hat{r}_\theta$ on preference pairs we follow the approach adopted in [3], and model the probability of a molecule being preferred as depending exponentially on the value of the reward using a softmax-normalised distribution:

$$P(m_1 \succ m_2) \approx \frac{\exp \hat{r}_\theta(m_1)}{\exp \hat{r}_\theta(m_1) + \exp \hat{r}_\theta(m_2)}. \tag{3}$$

The cross-entropy between predictions from $\hat{r}_\theta$ and the Pareto dominance relation pairs $\mathcal{D}_{\hat{t}}(\mathcal{M})$ is then minimised using the following loss function that has been demonstrated to be effective for training models from preferences [3, 5, 10]:

$$\mathcal{L}(\theta) = -\sum_{m_1 \succ m_2} \log P(m_1 \succ m_2) \tag{4}$$

$$= -\sum_{m_1 \succ m_2} \log \frac{\exp \hat{r}_\theta(m_1)}{\exp \hat{r}_\theta(m_1) + \exp \hat{r}_\theta(m_2)}. \tag{5}$$

3 Experiments and Results

We conduct two experiments to evaluate our method. In the first we investigate the effectiveness of our learned reward functions for guiding a generative method to desirable regions of molecular space. To achieve this we simulate DMTA cycles by defining synthetic proxies for project goals using reward functions taken from a publicly available benchmark for GMD. For the second experiment we apply our method to data taken from four real drug discovery projects, and evaluate its performance when compared to human-defined reward functions taken from those projects. For our method to be successful in this experiment, we expect rankings obtained using our method to have a higher correlation with project goals than rankings obtained using human defined reward functions.

3.1 Experiments on Simulated Design Cycles

We first evaluate the effectiveness of our method when used to guide a GMD algorithm towards desirable regions of molecular space. To simulate the cycles of a drug discovery project we extract a set of reward functions from the GuacaMol [4] benchmark that serve as proxies for drug discovery project goals. We use these functions as evaluation functions for determining of true rankings. We identify 6 reward functions with which to evaluate our method from the goal-directed section of that benchmark. Of the 20 available functions, 9 are single parameter objectives, and hence out of scope for what we are aiming to show, 2 have primarily binary scoring components, and hence are not representative of the continuous scores that result from biological assays, and 2 consist of simple maximisation objectives. Finally, we omit the remaining *Ranolazine MPO* function as the components for that function reported in the paper differ from those available in the corresponding data repository.

Algorithm 1 Experiment procedure for simulated DMTA cycles.

Require: E an evaluation function
Require: $\mathfrak{T}$ the drug candidate criteria derived from E
Require: $\mathcal{G}$ a GMD tool
Require: T number of repetitions
  for $t = 1 \dots T$ **do**
    $\mathcal{M} \leftarrow$ a random sample of 20 molecules from ChEMBL
    for $i = 1 \dots 20$ **do**
      compute $\mathcal{D}_{\hat{t}}(\mathcal{M})$ and use to train $\hat{r}_\theta$
      $\mathcal{M}_i \leftarrow$ molecules generated by $\mathcal{G}$ using $\hat{r}_\theta$ and seed molecules $\mathcal{M}$
      $\mathcal{M}_i^+ \leftarrow$ top 10 molecules in $\mathcal{M}_i$ or a diverse set of 10
      $\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{M}_i^+$
      compute the mean and max values obtained by evaluating E over $\mathcal{M}_i$
    end for
  end for

For each reward function from the benchmark we construct a drug candidate criteria $\mathfrak{T}$ by reading the values from the reward function components. We set the components of the learnable reward function to be those of the target function. However, we initialise the parameters for aggregation and scaling for the learnable function by sampling from a unit Gaussian. Algorithm 1 shows the procedure used to simulate DMTA cycles and evaluate our approach. To simulate the sub-optimal molecules available at the start of a drug discovery project we initially sample 20 molecules uniformly at random from the subset of ChEMBL [7] molecules used for that task in [4]. We then simulate 20 cycles by repeatedly constructing preferences over the existing set of evaluated molecules. For each cycle we fit a reward function using Pareto-dominance pairs, use a GMD method to optimise that function and generate new molecules, and evaluate the top 10 scoring molecules. We repeat the whole experiment three times, and aggregate the results.

We use the highest performing GMD method (*Graph GA*) from [4] to evaluate our approach. Graph GA is a genetic algorithm that follows the implementation of Jensen [11]. We configure it with a population size of 200, an offspring size of 200, and mutation rate of 0.01, and run for 100 generations at each iteration. We run two versions of this experiment. In the first, we select the 10 top scoring molecules after each iteration to use for subsequent iterations. In the second, we cluster all generated molecules at each iteration by their ECFP6 fingerprint [15] using the K-means clustering algorithm and selecting the molecules closest to the cluster centres to encourage diversity.

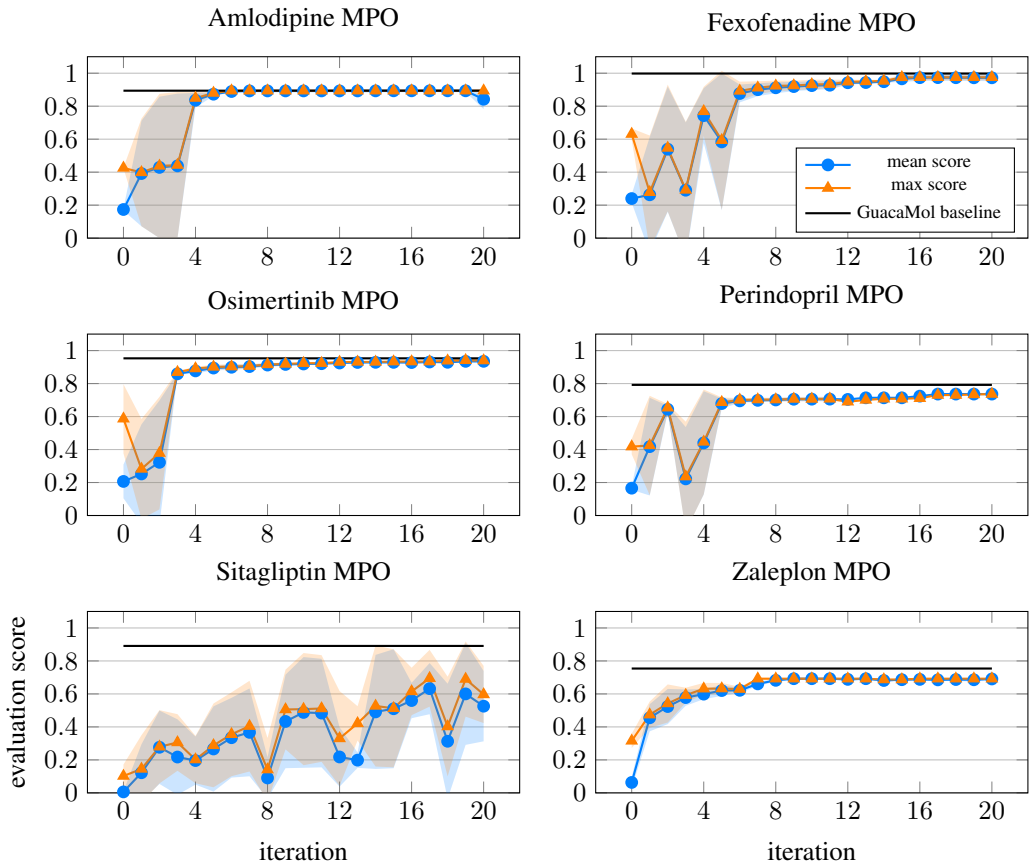

Figure 2: Mean and maximum evaluation scores for molecules generated at each iteration of Graph GA using the top scoring selection method. The shaded regions show variance across repeated runs.

Results The results for this experiment are plotted in Figures 2 and 3. For each reward function that we seek to learn we plot a horizontal line indicating the performance of Graph GA in the benchmark, reported as the average over the mean scores of the top scoring molecule, the top 10 scoring molecules and the top 100 scoring molecules, when seeding generative runs with the *highest scoring molecules* from the corresponding subset of ChEMBL. While that measure is not directly comparable to our mean scores, it represents the performance that we would expect if we used the evaluation function in

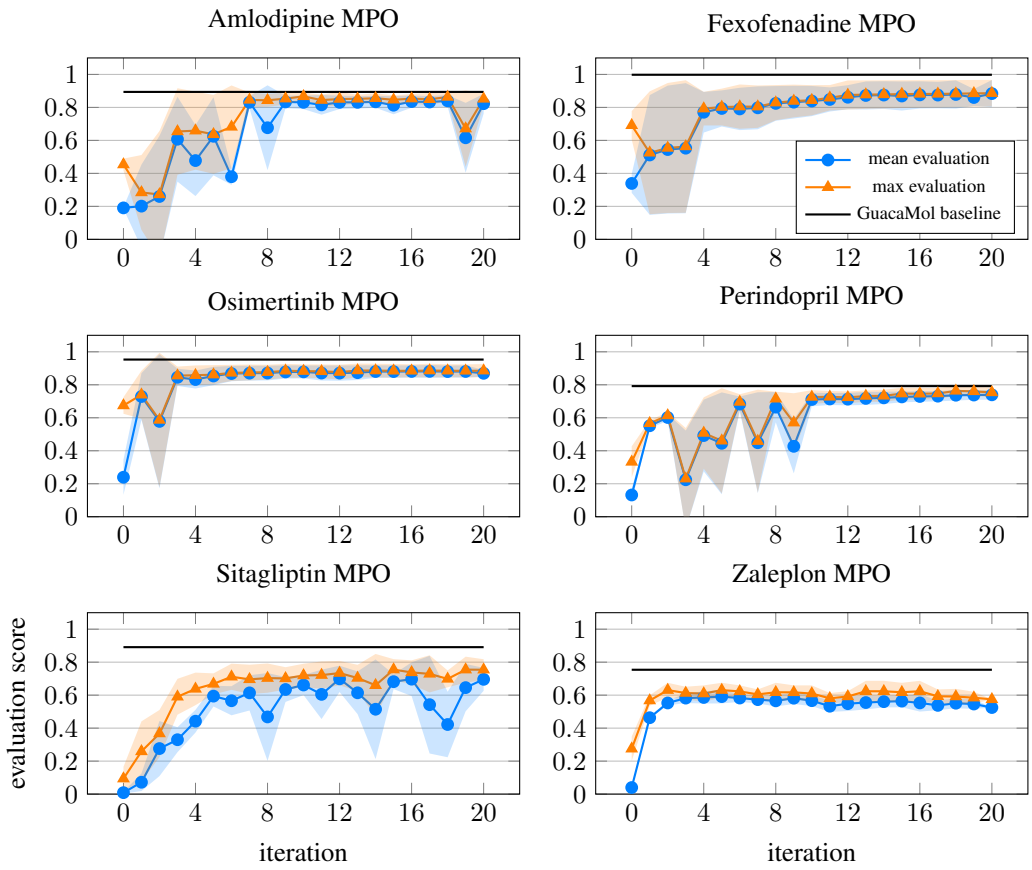

Figure 3: Mean and maximum evaluation scores for molecules generated at each iteration of Graph GA using the K-means selection method. The shaded regions show variance across repeated runs.

place of the learned reward function. For both selection strategies our method learns a reward function that guides Graph GA to molecules with a score comparable to those achieved in the benchmark within 10 iterations, despite seeding our runs with a randomly initialised population from the dataset. When selecting top scoring compounds we achieve within 0.146 of the baseline by iteration 10 on average, and within 0.054 by iteration 20. For cluster selection we achieve within 0.138 by iteration 10, and within 0.108 by cycle 20. Performance was better across all functions when using top scoring compounds, with the exception of Sitagliptin. We postulate that this is due to this function having a larger number of Gaussian components, which are harder to optimise when molecules are selected by the same reward functions used to generate them.

3.2 Experiments on Discovery Project Data

We now evaluate our model on data taken from four active drug discovery projects (undisclosed targets): $\mathsf{Project}_1$, $\mathsf{Project}_2$, $\mathsf{Project}_3$, and $\mathsf{Project}_4$. Each project has an associated target property profile that specifies desirable ranges for assay results. If all assay results for a molecule satisfy this profile then the molecule is a potential drug candidate for that project. Therefore, we use the assays and ranges from that profile to define the drug candidate criteria $\mathfrak{T}$ in the objective space. We treat half-open intervals as minimization or maximization objectives by setting the target value to an arbitrary large or small float. For closed intervals we set the target value to the midpoint of the interval.

Each project is associated with a human-defined evaluation function E that aggregates normalised assay scores into a single score in the unit interval using the geometric mean. This ensures that molecules only score highly (close to 1) when all of the assay component scores are close to the drug candidate criteria. Molecules with high scores are potential drug candidates. The evaluation function

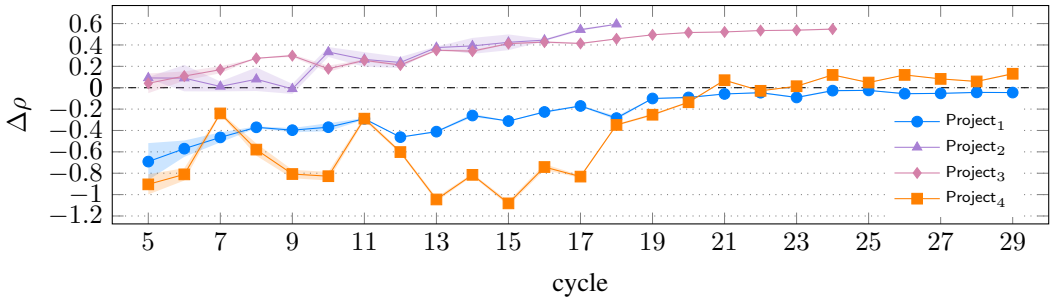

Figure 4: Difference between the Spearman's correlation of E and learned reward functions rankings, and the Spearman's correlation of E and human-defined reward function rankings.

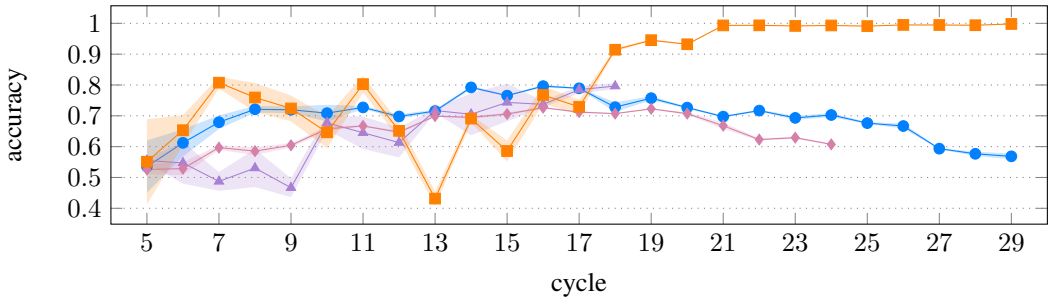

Figure 5: Ranking accuracy of our learned rewards. The dip in accuracy towards the end for some projects is explained by the increase in variance from smaller test datasets in later cycles.

gives us another way to rank molecules based on experimental results. Although we do not train our method to maximise correlation with rankings determined by E, we compute the Spearman's rank correlation coefficient between our learned rewards and E as a way to compare our method against an existing project-based measure of progress.

For each project we obtain the most recent reward function configured by the discovery team on that project, and limit our method to only use components from that reward function. This limitation is imposed to assess the ability of the method to identify suitable weights and scaling parameters, and we defer an analysis of model performance when extending the set of available components to future work. We then take the set of molecules sent for synthesis for that project, along with their associated assay scores, and split it temporally across DMTA cycles. For each cycle i we fit the model using rankings computed over the assay results for molecules up to cycle $i-1$, and perform inference on molecules from cycle i to obtain a predicted ranking. Many of the components of these reward functions are predictive models that are trained on internal data. When a component of the reward function is a predictive model we retrain a restricted version of that model, limiting the training set by excluding data that was not available before the timestamp associated with cycle i.

Due to time limitations we do not use our restricted models for the evaluation of the human-defined reward function, and directly use the latest versions of those models from the projects. In addition, we do not include any of the foundational data from sources such as ChEMBL [7] that are often included in the training of those models. Therefore, in this experiment the human-defined reward function has access to predictive models with better performance than those that we use for training $\hat{r}_\theta$.

Results For each project, we compute the Spearman's rank correlation coefficient between our learned reward at each cycle and the evaluation function E. We compute the same coefficient for the reference reward function and plot the difference as $\Delta\rho$ in Figure 4. The results show that our method generates reward functions that meet or exceed the predictive power of the reference reward functions, despite the disadvantages explained above. In general we see a positive trend across all projects. In the case of $\mathsf{Project}_2$ and $\mathsf{Project}_3$ our method performs decisively better than the reference by cycle 10. For $\mathsf{Project}_1$ and $\mathsf{Project}_4$ we meet or surpass the references in later cycles.

4 Discussion

In this paper we have proposed a novel approach for automated reward configuration for drug discovery that relies solely on experimental data. We demonstrated the effectiveness of our approach when applied to historical data from real drug discovery projects, as well as in a simulated environment built on synthetic objectives taken from the GuacaMol benchmark.

The experiments on historical project data show that our method is able to learn functions that match or outperform the predictive power of human-configured functions when given access to the components from those functions, despite having access to inferior predictive models. In future work we will investigate the performance of our method when it is given access to all feasible components. This will allow us to determine if the constraint on available components applied in our experiments hinders or contributes towards the effectiveness of the approach. Applying our method to simulations of DMTA cycles showed that our method can be used to guide a generative molecular design tool towards molecules with scores comparable to those achieved in the GuacaMol benchmark within 10 iterations, despite seeding our runs with a randomly initialised population.

The contribution presented in this paper constitutes a significant step towards the automation of reward function configuration for generative molecular design, serving as a baseline for future research.

References

[1] Brighter Agyemang, Wei-Ping Wu, Daniel Addo, Michael Y Kpiebaareh, Ebenezer Nanor, and Charles Roland Haruna. Deep inverse reinforcement learning for structural evolution of small molecules. *Briefings in Bioinformatics*, 22(4):bbaa364, 2021.

[2] Thomas Blaschke, Josep Arús-Pous, Hongming Chen, Christian Margreitter, Christian Tyrchan, Ola Engkvist, Kostas Papadopoulos, and Atanas Patronov. Reinvent 2.0: an ai tool for de novo drug design. *Journal of chemical information and modeling*, 60(12):5918–5922, 2020.

[3] Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pages 783–792. PMLR, 2019.

[4] Nathan Brown, Marco Fiscato, Marwin HS Segler, and Alain C Vaucher. Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108, 2019.

[5] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[6] Matthew D. Segall. Multi-parameter optimization: identifying high quality compounds with a balance of properties. *Current pharmaceutical design*, 18(9):1292, 2012.

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

[8] Laurent Gomez. Decision making in medicinal chemistry: The power of our intuition, 2018.

[9] Sai Krishna Gottipati, Boris Sattarov, Sufeng Niu, Yashaswi Pathak, Haoran Wei, Shengchao Liu, Simon Blackburn, Karam Thomas, Connor Coley, Jian Tang, et al. Learning to navigate the synthetically accessible chemical space using reinforcement learning. In *International conference on machine learning*, pages 3668–3679. PMLR, 2020.

[10] Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31, 2018.

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

[12] Peter S Kutchukian, Nadya Y Vasilyeva, Jordan Xu, Mika K Lindvall, Michael P Dillon, Meir Glick, John D Coley, and Natasja Brooijmans. Inside the mind of a medicinal chemist: the role of human bias in compound prioritization during drug discovery. *PloS one*, 7(11):e48476, 2012.

[13] Sohvi Luukkonen, Helle W van den Maagdenberg, Michael TM Emmerich, and Gerard JP van Westen. Artificial intelligence in multi-objective drug design. *Current Opinion in Structural Biology*, 79:102537, 2023.

[14] Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *ICML*, volume 1, page 2, 2000.

[15] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

[16] Iiris Sundin, Alexey Voronov, Haoping Xiao, Kostas Papadopoulos, Esben Jannik Bjerrum, Markus Heinonen, Atanas Patronov, Samuel Kaski, and Ola Engkvist. Human-in-the-loop assisted de novo molecular design. *Journal of Cheminformatics*, 14(1):1–16, 2022.

[17] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):10752, 2019.