Automated Ligand Design in Simulated Molecular Docking

Rob Maccallum and Geoff Nitschke

Computer Science Department
University of Cape Town, South Africa
mccrob015@myuct.ac.za, gnitschke@cs.uct.ac.za

Abstract

The drug discovery process broadly follows the sequence of high-throughput screening, optimisation, synthesis, testing, and finally, clinical trials. We investigate methods for accelerating this process with machine learning algorithms that can automatically design novel ligands for biological targets. Recent work has demonstrated the viability of deep reinforcement learning, generative adversarial networks and auto-encoders. Here, we extend state-of-the-art deep reinforcement learning molecular modification algorithms and, through the integration of molecular docking simulations, apply them to automatically design novel antagonists for the adenosine triphosphate binding site of Plasmodium falciparum phosphatidylinositol 4-kinase, an enzyme essential to the malaria parasite's development within an infected host. We demonstrated that such an algorithm was capable of designing novel molecular graphs with better DSs than the best DSs in a set of reference molecules. There reference set here was a set of 1,011 structural analogues of napthyridine, imidazopyridazine, and aminopyradine.

Introduction

Drug discovery and design comprises three primary categories. First, *hit screening*, which is where classes of drugs or chemotypes which share a similar molecular scaffold are identified as having a notable binding affinity (BA) for a target receptor. Second, *hit-to-lead optimisation*, which is where hits obtained from the previous phase have their BA for the receptor increased through modification by experts. Third, *lead optimisation*, where leads optimised to have high BA for the target have their other physico-chemical properties such as solubility, selectivity, molecular polarizability, charge distribution, molecular weight and others customised for their intended application. The entire drug discovery and design process (including synthesis, testing and clinical trials), takes on average 10 years and costs an average of 2.6 billion US dollars (DiMasi et al., 2016). Also, drug discovery research productivity is on the decline, with average failure rates for clinical trials approaching 90% in all disease categories (Kadurin et al., 2017).

Related Work

In recent years there has been significant progress towards automating the hit-screening and lead optimisation phases through the use of *Evolutionary Algorithms* (EAs) (Eiben and Smith, 2015), *Generative Adversarial Networks* (GANs) (Guimaraes et al., 2017; You et al., 2018) and auto-encoders (Gómez-Bombarelli et al., 2018). For example, Supady et al. (2015) demonstrated that a population of random conformers of a given SMILES sequence could evolve into one containing only low energy conformers. Initially, a string was generated for each individual which represented a vector of torsion angles. The fitness function calculated each individual's conformational energy, estimated with Density Functional Theory (Atkins and de Paula (2010)), and modulated relative to the other individuals in the population. An EA was then used to generate low energy conformers of the initial random population.

Harel and Radinsky (2018) showed that VAEs could be applied to the generation of novel SMILES with optimised properties. Here one-dimensional recurrent convolutional layers were used to map from SMILES strings to a continuous vector encoding. Latent codes were decoded back to SMILES strings by mapping from the continuous vector to a probability distribution over the available characters. This was done iteratively for each character in the sequence until the terminal token was chosen. This method was expanded upon when Gómez-Bombarelli et al. (2018) used Bayesian optimisation with an auxiliary ANN to find latent codes which decoded to molecules with optimal properties. Also, Guimaraes et al. (2017) integrated GANs with RL to generate generate SMILES strings which, in terms of their constitution, resembled those of a set of reference molecules but also had improved solubility ($\log P$), SA and QED. Sanchez-Lengeling et al. (2017) extended this method to optimise the set of novel molecules for melting point, and photovoltaic conversion efficiency.

However, a key drug design goal is achieving suitable binding affinity (BA) for the target macromolecule, and automation of this phase of the drug design process (hit-to-lead optimisation) has received little research attention relative to the hit screening and lead optimisation phases.

In summary, the motivation for our method stems from the need to focus ML algorithms such as these on the hit-to-lead phase of drug design. In this context the goal is to generate novel compounds for which there are limited available reference sets. Supervised learning algorithms such as GANs and VAEs are consequently inappropriate as they require a large supply of empirically labelled training samples. In contrast, purely RL algorithms are capable of learning entirely unsupervised, improving their performance using only the feedback received from the environment through exploration and thus requiring no positive examples in order to learn to generate ligands of the required class. Therefore, in this work we explored the integration of RL algorithms with a docking simulator within which an autonomous agent could learn to design novel ligand candidates using only *a priori* laws of binding energy as a reward signal.

Research Objectives

Within the context of automating hit-to-lead optimisation, the focus of this study was the automatic design of novel ligands which are expected to yield high BA for a given target macromolecule. We coupled RL with an environment in which an autonomous agent could design molecular graphs and receive feedback about their docking scores (DSs). The result was a generative algorithm capable of correlating features of molecular structure with DS for a specified binding site of a target molecule. Such an algorithm must generate ligands with maximal DS for the receptor site and, when given a molecular scaffold, must return a ligand with greater DS for the target. Thus, our research question is:

> *How effective is RL as a method for automatically designing ligands possessing a high BA for specific macro-molecular targets?*

We address this question by evaluating a range of DSs that our algorithm attains versus DSs of ligands already available for given targets and the number of training episodes required to generate graphs with a DS above given task-performance thresholds.

Contributions

We contribute a simulation framework for training molecular docking agents to generate novel ligands with high BA, for target macromolecules, where BA is estimated using docking score (DS) as a proxy, for the receptor site of the macromolecule. The framework applies *double-deep Q-learning* (Mnih et al., 2013, 2015), in an environment comprised of a ligand to be modified and a target macromolecule. The agent designs molecular graphs and employs the Morgan algorithm (Rogers and Hahn, 2010) to convert graphs to molecular fingerprints. The environment state is the current molecular graph state and possible actions are graphs accessible from the current state, where the algorithm learns from environments with inconsistent action spaces. Docking between the ligand designed by the agent and the target macromolecule is simulated using the Autodock-GPU (Santos-Martins et al., 2021) package, and the DS calculated by Autodock-GPU defines the reward received by the agent at each step of a design episode.

Our framework potentially accelerates the *hit-screening* and *hit-to-lead* optimisation phases of drug design, since novel lead candidates may result from unintuitive graph arrangements identified by agent correlations between structural features and BA. Such a framework can thus guide and inform chemists during their selection of candidates for screening or when modifying hits to increase candidate BA. Also, the framework is extensible to *lead optimisation* by incorporating any physico-chemical property in rewards.

Methods

This study builds on previous work using *Q-learning* (Zhou et al., 2019) for automated drug design. Previous work explored the generation of molecules with optimised physico-chemical properties such as $\log P$, QED and SA; which were measured with the RDKit chemoinformatics package. However, to develop a ligand for a particular target receptor one needs to measure a given molecule's BA and specificity for that receptor. This can only be accurately measured experimentally, but estimated via simulations. Therefore, our methods incorporate simulations of molecular docking for a receptor identified as a target for a specific pathology into the reward function. Specifically, we use *deep Q-learning* to train agents within docking simulations to optimise a ligand's BA for a protein target, using DS as an estimate.

Agent

The agent was implemented as a pair of fully connected deep ANNs, and a memory buffer, where the input to the network was the concatenation of two vectors. The first was a 2048 bit ECFP_4 molecular fingerprint of each state (graph), accessible from the current state, where each bit in the fingerprint corresponded to a certain functional group. Fingerprint vectors were concatenated with the number of remaining steps in the design episode, so each accessible state was considered relative to number of steps the agent had left to modify the graph. Fingerprints of each state accessible from the current state, concatenated with episode steps remaining, were then presented to the *Q-network* in sequence and for each, the network calculated a *Q-value*.

This approach was taken at each step of the graph design process as the number and type of accessible states changes with the current state of the graph. The action-value distribution over the states accessible from the current state was thus constructed by evaluating each accessible state in sequence. An input layer of 2048 fingerprint neurons and 1 steps-remaining neuron was connected to 3 fully-connected hidden-layers with *Rectified Linear Unit* (ReLU) (Nair and Hinton, 2010) activation functions, comprising 1024, 512, and 128 neurons respectively. The final layer contained only a single output neuron with no activation function. This corresponded to the Q-value of each accessible state, given the steps remaining. This structure was the same for both the behavioural policy network (Q) and the target network (Q^{-}).

Environment

The training environment (simulated with *Autodock-GPU docking* (Santos-Martins et al., 2021)), combined the *ligand* (designed by the agent during its exploration phase over a maximum number of steps), and the target *protein*. Docking simulations then determined if the given ligand could form a stable complex with the target protein and if so what was the change in the free energy of the ligand-protein complex as a result of binding. This change in binding free-energy determined the reward an agent received at the end of each exploration episode. Agent designed molecules were two-dimensional molecular graphs (using the `.mol` format), where these graphs were presented to the agent as molecular fingerprints. Each episode step the agent was presented with the set of possible modifications it could make to the current graph (molecular fingerprint), choosing modifications according to its policy of finding a molecule with the highest DS for the receptor site of the target. After the modification episode, the agent designed ligand was presented to the target for docking.

Simulating Molecular Docking

Molecular docking was simulated with the Autodock-GPU package. Given a molecular graph and the crystal structure of a target protein, docking proceeded as follows.

The ligand's molecular graph was converted from 2D graph `.mol` format to 3D representation, including hydrogens, partial charges, and atomic coordinates using the `.pdbqt` format. Target protein crystal structure was obtained from the protein data in `.pdb` format (also converted to `.pdbqt` format). A 3D docking grid was then prepared which encapsulated the receptor site of the target protein. Preparation of the target protein and docking grid was done once prior to training whereas conversion of 2D ligand graphs to their 3D representations was performed after each molecular modification episode. Given the `.pdbqt` ligand and protein files and docking grid, Autodock used its search algorithm to explore conformational states of the given ligand, evaluating the ligand-protein interaction for each conformation. This conformational search was done by a *Lamarckian Evolutionary Algorithm* (EA) (Morris et al., 1999), which searched for the global minimum of equation 1. The conformation with the greatest corresponding release of binding-free energy was saved to a coordinate file.

Autodock implements a semi-empirical scoring function (differentiated from knowledge based and physics based), as a weighted sum of atomic interactions, tuned to structural data. Steric, hydrophobic, and hydrogen bonding interactions between atoms in the ligand, and atoms of the receptor within the docking grid are calculated. The weights of these terms were computed from a non-linear fit of the scoring function to structural data (Trott and Olson, 2010).

Free energy ΔG of a binding pose is given by equation 1.

$$\Delta G = \Delta H_{vdW} + \Delta H_{hbond} + \Delta H_{elec} + \Delta G_{desolv} + \Delta S_{tor} \tag{1}$$

Where, ΔH_{vdW}, ΔH_{hbond}, ΔH_{elec} are the enthalpy changes due to *Van Der Walls* interactions, hydrogen bonding and electrostatic interactions respectively, ΔG_{desolv} is the Gibbs free energy change due to desolvation and ΔS_{tor} is the change in ligand entropy due to the loss of rotatable degrees of freedom in the ligand as a result of binding. Energetic terms are approximated semi-empirically as follows:

$$\begin{aligned}
\Delta H_{vdW} &= W_{vdW} \sum_{i,j} \left(\frac{A_{ij}}{s(r_{ij})^{12}} - \frac{B_{ij}}{s(r_{ij})^{6}} \right) \\
\Delta H_{hbond} &= W_{hbond} \sum_{i,j} E(t) \left(\frac{C_{ij}}{s(r_{ij})^{12}} - \frac{D_{ij}}{s(r_{ij})^{10}} \right) \\
\Delta H_{elec} &= W_{elec} \sum_{i,j} \left(\frac{q_i q_j}{\epsilon(r_{ij}) r_{ij}} \right) \\
\Delta G_{desolv} &= W_{desolv} \sum_{i,j} (S_i V_j + S_j V_i) e^{-r_{ij}^2 / 2\sigma^2} \\
\Delta S_{tor} &= W_{tor} N_{tor}
\end{aligned} \tag{2}$$

Where, sums $\sum_{i,j}$ are over all inter-molecular pairs of ligand-receptor atoms within the docking box. A_{ij}, B_{ij} are constants which depend on the modified Lennard-Jones potentials (Atkins and de Paula, 2010) between atoms i and j, and C_{ij}, D_{ij} are constants which depend on the hydrogen bonding potentials between i and j. S and V are salvation parameters and atom volume respectively and σ is set to 3.5. r_{ij} is the inter-atomic distance between atoms i and j and $s(r_{ij})$ is a smoothing function. $E(t)$ is a function which provides directionality for the hydrogen bond term based on the angle t. $\epsilon(r_{ij})$ is a dielectric function of r_{ij}. N_{rot} is the number of torsions in the receptor in its bound state. Weights W_{vdW}, W_{hbond}, W_{elec}, W_{desolv} and W_{rot} were

empirically set using linear regression on ligand-receptor complexes with known binding constants. Release of binding free-energy was then returned as the ligand's DS, which was an estimate of its BA. This DS then determined the agent's reward.

A single docking calculation was the evaluation of 10^6 to 10^8 scoring function evaluations during the EA run. Autodock-GPU (Santos-Martins et al., 2021) dramatically accelerates the run-time by exploiting the parallel nature of the docking algorithm, decomposing the population of candidate solutions into w work-groups, where work groups ran in parallel on a GPU compute unit.

Rewards and Shaping

The state of the environment (defined by the current molecular graph) is evaluated by the reward function on every step to determine if the goal state has been reached. The goal is that the agent discovers a novel molecular graph with high DS for the given target receptor, thus we cannot specify the goal molecular graph, rather we specify a property that the goal state should possess and the reward function is defined accordingly. Here, the desired property was high DS for the specified receptor site of a given target molecule. Therefore, the reward function returns the result of conformational search and the DS calculation performed by the docking package with its sign inverted, that is:

$$r(s_t) = -1 \times \Delta \text{ Binding Free Energy} \tag{3}$$

The reason for the inversion is that a greater reduction in binding free energy is of higher value. As a result, the Q-value (state-action value), of any molecular graph returned by the agent's behavioural network, equates to the expected cumulative DS expected in the remaining steps (after choosing the given action).

In this study, it was not possible to calculate the DS of each transition along a roll-out as this would have made the run-time of a complete training session infeasible. Hence, we calculated the DS of the final state along a roll-out. Given that a roll-out trajectory was composed of 40 transitions, only one of which receives an extrinsic reward, the problem was one of sparse rewards, and a shaping method was implemented to encourage learning. Given the reward for the terminal state of a roll-out trajectory, rewards for the preceding transitions were calculated using a method similar to that of potential-based reward shaping (Ng et al., 1999). Here the potential of the terminal state was taken as the DS received from the environment, and the potentials of the preceding states were estimated by linearly interpolating from the full DS in the final state to zero in the initial state (equation 4).

$$\phi(s_t) = r_f - \frac{r_f}{t_{max}} \times (t_i - 1) \tag{4}$$

Where, r_f is the reward of the terminal state, t_{max} is the number of steps in a trajectory and t_i is the steps remaining between state i and the terminal state.

Learning Algorithm

The RL algorithm (Algorithm 1) used to train the agent was double deep Q-networks (DDQN) (Mnih et al., 2013, 2015). First the agent was initialised with an empty molecular graph (line 4, Algorithm 1). The agent was given a maximum of 40 steps (chosen based on previous work (Zhou et al., 2019; You et al., 2018)) in which to construct a ligand. Each step corresponded to the potential addition or removal of an atom or bond, given a limit of 40 atoms as the maximum size of the ligands, excluding hydrogen, with a mass in the range of 500 to 1000 Daltons, common for small-molecule drugs (Chhabra, 2021). At each step of the modification phase, the environment generates all possible and valid molecular graphs that are accessible from the current state and returns these to the agent in a tensor.

The next states s_{t+1} are the actions a_t (Algorithm 1), so at each step of an episode the action space was defined by accessible states. The agent then uses its behavioural network to select a state (action) to move into (lines 5, 6, and 7, Algorithm 1). At each step, docking between the ligand and target receptor was simulated. The DS was calculated and returned to the agent as a reward (r_t in line 7, Algorithm 1), and the transition $(s_t, s_{t+1}, r_t, term)$ was stored in the agent's memory (line 9, Algorithm 1), where $term$ is a flag indicating if s_{t+1} was a terminal state. A trajectory of at most 40 such transitions (t_{max} in algorithm 1) constituted a single roll-out or exploration episode.

After T_{bp} roll-out episodes (backpropagation period), where data was sampled from the policy π_θ, the agent randomly sampled a mini-batch of transitions from the replay memory (line 12, Algorithm 1) and optimised its ANN approximation of the optimal action-value function Q_θ using stochastic gradient descent in backpropagation (lines 13, 14 and 15, Algorithm 1). The parameters of the target network θ^- were then updated with a fraction τ of the updates made to their counterparts in the behavioural network (line 16, Algorithm 1). Thus training alternated between T_{bp} sampling episodes e and mini-batch gradient descent in backpropagation. This loop continued for e_{max} episodes or until convergence in the policy network's loss function.

Experiments and Results

The learning framework was evaluated with the following three experiments, defined such that their results would answer our research questions (posed in the introduction).

1. Generation of ligands with maximal DS for the PfPI4k target receptor when the agent begins each training episode from an empty starting molecule.

2. Generation of ligands with maximal DS for the PfPI4k target receptor when the agent begins from three reference scaffolds known to be structural analogues of ligands with high BA, namely napthyridine, imidazopyridazine, and aminopyradine.

3. Docking of 1,011 structural analogues of napthyridine, imidazopyridazine, and aminopyradine with the PfPI4k target receptor.

In the first two experiments the agent's goal was to construct ligands through the addition or removal of atoms or bonds, such that the terminal state after 40 transitions received a high DS. When the environment was reset at the beginning of each new episode the state of the ligand was returned to its initial state. These experiments explored the impact of the initial state on the final agent DS. In the first experiment, the agent begins from a single carbon atom, whereas in the second experiment, the agent begins from one of three reference scaffolds. The purpose of the third experiment is to define a reference for comparing the scores of the ligands generated by the agent.

Generation from an Empty Molecule

The first part of experiment 1 (figure 1, blue curve) was to evaluate whether it was possible for the agent to achieve some degree of success with such sparse feedback from the environment. Figure 1 (left) shows the sparse and shaped reward curves over the course of training. Both curves correspond to training session where the agent begins from an empty molecule on each episode, where in the sparse reward training session (blue curve) only the terminal state received a DS reward, and in the shaped reward training session (red curve), reward shaping was used to estimate a reward for all transitions in the roll-out episode.

The impact of reward shaping was investigated since when rewarding the agent with a ligand's DS, it was not possible to return a reward at every step of a roll-out episode. This was because the time taken to calculate a single ligand's DS made the run-time for a full training session of 5000 episodes untenable. Thus, a DS was returned only for the terminal state of the roll-out episode, meaning preceding transitions were added to the replay memory with no associated reward.

The second part of experiment 1 (figure 1, red curve) created dense reward signals from the sparse extrinsic reward returned by the environment at the end of the agent's roll-out trajectory (equation 4). Instead of pushing the states received from the environment along with a zero reward to the agent's replay memory (as they were received), they were instead buffered until the episode's end.

Equation 4 was then used to calculate a non-zero reward for every preceding transition using the terminal state reward. The initial generation experiment where the agent received a reward only for the terminal state demonstrated that this is insufficient information from which to extract a useful policy as the DS of generated ligands failed to increase in 5,000 episodes. The terminal state DS after 4,500 training episodes was lower than the DSs of terminal states after random exploration in the first 500 episodes.

However, simple reward shaping that linearly extrapolates backwards from the final state, to calculate a reward for preceding states (equation 4), immediately enabled the agent to learn an effective design policy. The "shaped" plot (figure 1, left), shows that after starting at an initial DS of greater than -6 kcal/mol (from random exploration), the agent converged on a minimum of approximately -13 kcal/mol with the best candidates exceeding -15 kcal/mol.

The effectiveness of this shaping methods is likely due to the fact that the value of a given state is not determined by the state in isolation, but the terminal state of the path in which the state occurred. Thus, states are valuable if they are close to terminal states with high DSs. For example, if a given roll-out episode resulted in a terminal state which received a high DS, then the penultimate state along that path would also be valuable as it permits access to the high-scoring terminal state. The further back along the path one is from the terminal state, the less valuable the states become. The shaping function (equation 4) was designed to implement this logic, thus conditioning the agent to select features in the extended-connectivity fingerprints leading to high-scoring terminal states within the 40-step limit.

Given that the use of reward shaping enabled the agent to learn where it had previously failed in the sparse reward environment, we can assume that a more sophisticated sparse-reward amelioration strategy such as hindsight experience replay, which has been shown experimentally to outperform reward-shaping (Andrychowicz et al., 2017), should improve the algorithm's performance.

Algorithm 1: Double Deep Q-Learning (DDQN) (Mnih et al., 2013, 2015)

Randomly initialise action-value function Q_θ and target network Q_{θ^-} with parameters $\vec{\theta}$ and $\vec{\theta^-}$ respectively.
Initialise replay memory $\mathcal{D}$ to capacity $\mathcal{N}$
repeat
  Reset environment
  repeat
    With probability ϵ sample random action (next state) s_{t+1}
    Otherwise select $s_{t+1} = \underset{s_{t+1}}{\max} Q^*_\theta(s_t, s_{t+1})$
    Move into next state s_{t+1} and observe reward r_t
    Store transition $(s_t, s_{t+1}, r_t, term)$ in $\mathcal{D}$
  until t_{max}
  if $e \mod T_{bp} == 0$ **then** optimise behavioural and target networks
    Sample random minibatch of transitions $(s_j, s_{j+1}, r_j, term)$ from $\mathcal{D}$
    Set$y_j = \begin{cases} r_j & \text{terminal } s_{j+1} \\ r_j + \gamma \underset{a'}{\max} Q(s_{t+1}, a') & \text{non-terminal } s_{j+1} \end{cases}$
    Update Q_θ via one step of gradient descent by backpropagation
    $\Delta\vec{\theta} = \nabla_\theta J(\theta) = \mathbb{E}\left[\left(r + \gamma \underset{a'}{\max} Q_{\theta^-}(s', a') - Q_\theta(s, a)\right)\nabla_\theta Q_\theta(s, a)\right]$
    $\theta^- \leftarrow \tau\theta$
  end
until e_{max}

Generation from Reference Series

This experiment explored the effect of focusing the search on regions of chemical space near to the structural features of chemical series known to have an affinity for the receptor. Here the variant of the algorithm incorporating reward shaping was applied to three runs, each composed of 5000 episodes starting from the *aminopyridine*, *napthyridine* and *imidazopyridazine* molecular backbones. hese chemical series are being investigated as derivatives of these three molecular backbones with varying substituents have demonstrated high inhibition potency against the PfPI4K target enzyme in phenotypic whole-cell screenings.

Plots of the terminal state DS over the course of training for these three runs are shown in figure 1 (right). The curves are colour-coded to indicate which scaffold the agent was modifying in each training session.

Whereas the previous experiment evaluated agent ability to find a path from an arbitrary point in chemical space (a single carbon atom) to a region of high DS for the receptor, this experiment evaluated the performance impact of simplifying the problem by beginning training from a region of chemical space known to contain high-BA structures.

Figure 1 (right) shows that when starting a 40-step rollout from each of these scaffolds the initial ligands generated by the agent (from random exploration), obtained a better terminal state DS than those obtained when beginning from only a single carbon atom, as the terminal state DSs from random roll-outs starting from napthyridine, aminopyradine and imidazopyridazine were approximately -9 kcal/mol.

After approximately 2,500 training episodes the agent converged on candidate ligands with an average terminal state DS of approximately -14 kcal/mol, with numerous candidates exceeding -15 kcal/mol and the best exceeding -16 kcal/mol. This result is comparable to the performance of MolDQN (Zhou et al., 2019), which converged on a score of 0.8 after 3,000 episodes when training to optimise QED.

In summary, when starting from scaffolds known to be structural analogues of ligands with high DS for the target, the agent begun training by finding (initially via random exploration), states with a better DSs than those discovered when beginning from only a single carbon atom, and converged on states with better DSs than those obtained when starting from only a single carbon atom. The agent likely converges on better molecules when building from known scaffolds since the search is now being constrained to the region of chemical space surrounding the given scaffold. Since the *aminopyradine*, *napthyridine* and *imidazopyridazine* scaffolds are the backbones of three chemical series known to contain high-BA ligands, by constraining the search to this region, the problem of finding high-BA candidates is simplified since the agent is more likely to discover rewarding states through exploration.

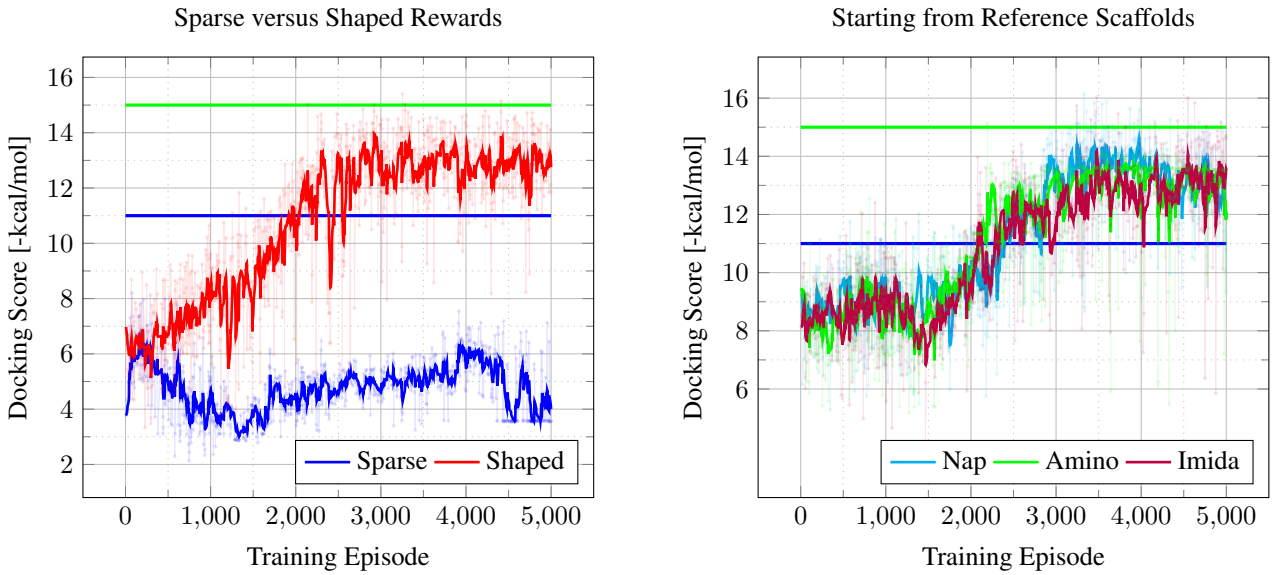

Figure 1: Training curves from the various DS rewards experiments showing the Autodock DS of the terminal state per training episode. The plots show the raw data as well as a that data smoothed with the following first-order IIR low-pass filter $f(x_t) = x_{t-1} * \alpha + (1-\alpha)x_t$ with $\alpha = 0.8$. Data are agent performance when learning from sparse rewards (left), using reward shaping (left), and starting from known reference molecules (right). Legend abbreviations (right) correspond to reference scaffolds *napthyridine*, *aminopyridine* and *imidazopyridazine*. Experiments starting from the reference molecules also incorporated reward shaping. The two horizontal lines show the mean and maximum DS of the ligands in the reference set (figure 2).

This is expected since structural analogues, (molecules sharing similar scaffolds but with different substituents and sufficiently similar molecular backbone), are candidates for functional analogues. These are two molecules with similar pharmacological properties. They exhibit similar biochemical or physiological effects on the human body, but with variations in efficacy and side effects (Bruice, 2011).

This result is also supported by other EAs using specially pre-initialised populations to boost the task performance of evolved solutions by evolving novel functional analogues in initial populations (Rupakheti et al., 2015; Brown et al., 2004; Lameijer et al., 2005). In other RL approaches the network has also been initially pre-trained structural analogues (Sumita et al., 2018) to boost task performance. The notion of structural analogues has also been incorporated in RL algorithms using *Tanimoto similarity* (Zhou et al., 2019), where the agent attempts to maximise similarity to a given scaffold when generating novel structure.

Comparison with Existing Ligands

In order to evaluate the algorithm's performance when generating ligands for the PfPI4K receptor, a reference point was needed. Therefore, a set of ligands currently being explored as potential antagonists for the PfPI4K enzyme were docked against the target receptor for comparison. The DSs of the reference ligands are plotted in figure 2.

Figure 2 shows the DSs of 1,011 structural analogues of the napthyridine, aminopyradine and imidazopyridazine scaffolds. These structures were docked with the same parameters used to reward the agent during the generation experiments. From the histogram we see that the mean of the set is approximately -11 kcal/mol with only a very small number of structures receiving scores better than -14 kcal/mol with none being better than -16 kcal/mol. Thus, figure 2 indicates the agent was able to design ligand candidates containing a substantial number of ligands with DSs better than the best DSs in a set of reference ligands.

Conclusion

This study sought to investigate the potential for applying RL to the development of generative algorithms for automated drug design. Our methods comprised a deep RL agent and simulation environment where the agent could construct molecular graphs bond-by-bond and dock the resulting ligands with the crystal structure of a target receptor. The experiments evaluated the agent and environment as the task of maximising DS for a given receptor, specifically the ATP binding site of the PfPI4K enzyme. Experiments investigated the impact of *sparse* versus *shaped* rewards, focusing the search on a particular region of chemical space, and comparing the ligands generated by the agent with those currently being evaluated in chemical laboratories.

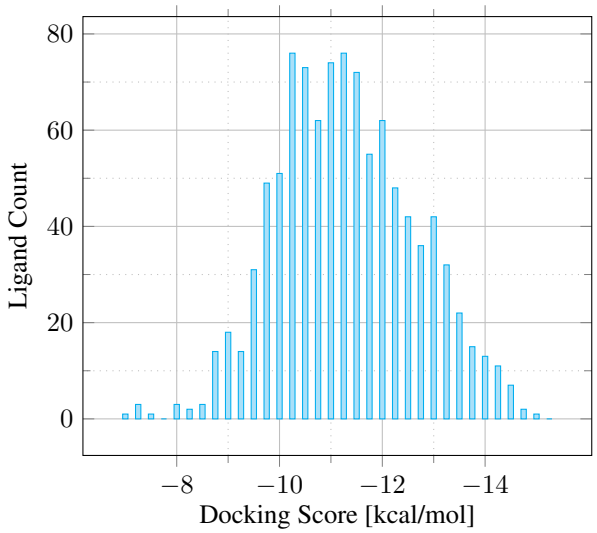

Figure 2: Result of docking 1011 structural analogues derived from *napthyridine*, *aminopyridine* and *imidazopyridazine* scaffolds against the PfPI4K receptor. x-axis: DS intervals of 0.25 kcal/mol. y-axis: Ligands in each interval. Histogram shows range and distribution of Autodock DSs for known ligands: benchmark for agent task-performance.

Rewarding only the terminal state during training significantly reduced training time and using reward shaping facilitated learning. Reward shaping was successful since the shaping function assumed the state value to be proportional to both the DS received by the (episode) terminal state and its distance from the terminal state. However, we are investigating the efficacy of other sparse reward strategies such as hindsight experience (Andrychowicz et al., 2017).

When beginning training episodes from reference scaffolds known to be analogues of ligands with high DSs, the agent was able to discover more molecules with higher DSs. This was due to the search space being focused to the region of chemical space where the *napthyridine*, *aminopyridine* and *imidazopyridazine* series are located, as this appears to simplify the search problem for the agent thus leading to the policy converging on ligands with a higher DSs.

Finally, when comparing the automatically designed ligands with those currently available, we observed that the agent was able to generate a substantial number of ligands with higher DSs than the best ligands in the reference set.

Future Work

The most apparent shortcoming of the current implementation is the number of impossible atomic arrangements which appear in the graphs designed by the agent, rendering the proposals impossible to synthesise even though they possess high DSs. This is a consequence of the reward function considering DS in isolation. The agent therefore searches for graphs which optimally fit the receptor site without any consideration for other properties of those graphs. There are a few methods which could potentially address this. The first would be to hard-code filters, which prevent specific modifications that would lead to these unrealistic structures, into the environment. Alternatively, instead of rules which filter out certain actions that would lead to undesirable features, the agent could instead be presented with modifications which change whole functional groups. For example, instead of choosing only between adding a single carbon, nitrogen or oxygen atom, the agent's choices would also include the options to add carobxylic acid, aldehyde, amine or phenyl functional groups. In combination with these methods, QED and SA could be incorporated into the reward function. The goal would then be to maximise QED and SA simultaneously with DS. In addition to having high DSs, the resulting ligands would then also have high QED and SA scores as well, thus improving their utility.

Finally, there is potential to avoid the inaccuracies and computational demands of simulations entirely by leveraging available IC50 data to develop surrogate models. These could replace the computationally demanding docking simulator to accelerate training. By first training a discriminative network to predict IC50 values from molecular fingerprints, this predictive model could be used as the reward function for a generative agent. In addition to accelerating training, this could potentially be more useful than DSs calculated from simulations, as IC50 measurements are obtained from whole-cell phenotypic screenings, and so they also account for off-target interactions within the cell. The reward returned from this predictive IC50 model could of course also be combined with QED and SA as previously described.

References

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. (2017). Hindsight Experience Replay. *arXiv*.

Atkins, P. and de Paula, J. (2010). *Physical Chemistry*. Oxford University Press, Oxford, 9 edition.

Brown, N., Mckay, B., and Gasteiger, J. (2004). The de novo design of median molecules within a property range of interest. *Journal of Computer-Aided Molecular Design*, 18(12):761–771.

Bruice, P. Y. (2011). *Organic Chemistry*. Pearson, sixth edition.

Chhabra, M. (2021). Biological therapeutic modalities. *Translational Biotechnology*, pages 137–164.

DiMasi, J. A., Grabowski, H. G., and Hansen, R. W. (2016). Innovation in the pharmaceutical industry: New estimates of R&D costs. *Journal of Health Economics*, 47:20–33.

Eiben, A. and Smith, J. (2015). *Introduction to Evolutionary Computing*. Natural Computing Series. Springer Berlin Heidelberg, Berlin, Heidelberg, 2nd edition.

Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. (2018). Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Central Science*, 4(2):268–276.

Guimaraes, G. L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C., and Aspuru-Guzik, A. (2017). Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models. *arXiv preprint arXiv:1705.10843*.

Harel, S. and Radinsky, K. (2018). Prototype-Based Compound Discovery Using Deep Generative Models. *Molecular Pharmaceutics*, 15(10):4406–4416.

Kadurin, A., Aliper, A., Kazennov, A., Mamoshina, P., Vanhaelen, Q., Khrabrov, K., and Zhavoronkov, A. (2017). The cornucopia of meaningful leads: Applying deep adversarial autoencoders for new molecule development in oncology. *Oncotarget*, 8(7):10883–10890.

Lameijer, E. W., Bäck, T., Kok, J. N., and Ijzerman, A. P. (2005). Evolutionary algorithms in drug design. *Natural Computing*, 4(3):177–243.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning. *CoRR*, abs/1312.5.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Morris, G. M., Goodsell, D. S., Halliday, R. S., Huey, R., Hart, W. E., Belew, R. K., and Olson, A. J. (1999). Automated Docking Using a Lamarckian Genetic Algorithm and an Empirical Binding Free Energy Function. *Journal of Computational Chemistry*, 19(14):16391662.

Nair, V. and Hinton, G. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning (ICML)*, ICML 2010, pages 807–814, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Ng, A. Y., Harada, D., and Russell, S. J. (1999). Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 278–287, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Rogers, D. and Hahn, M. (2010). Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754.

Rupakheti, C., Virshup, A., Yang, W., and Beratan, D. N. (2015). Strategy to discover diverse optimal molecules in the small molecule universe. *Journal of Chemical Information and Modeling*, 55(3):529–537.

Sanchez-Lengeling, B., Outeiral, C., Guimaraes, G. L., and Aspuru-Guzik, A. (2017). Optimizing distributions over molecular space. An Objective-Reinforced Generative Adversarial Network for Inverse-design Chemistry (ORGANIC). *ChemRxiv*.

Santos-Martins, D., Solis-Vasquez, L., Tillack, A. F., Sanner, M. F., Koch, A., and Forli, S. (2021). Accelerating AutoDock 4 with GPUs and Gradient-Based Local Search. *Journal of Chemical Theory and Computation*, 17(2):1060–1073.

Sumita, M., Yang, X., Ishihara, S., Tamura, R., and Tsuda, K. (2018). Hunting for Organic Molecules with Artificial Intelligence: Molecules Optimized for Desired Excitation Energies. *ACS Central Science*, 4(9):1126–1133.

Supady, A., Blum, V., and Baldauf, C. (2015). First-Principles Molecular Structure Search with a Genetic Algorithm. *Journal of Chemical Information and Modeling*, 55(11):2338–2348.

Trott, O. and Olson, A. J. (2010). AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of Computational Chemistry*, 31(2):455–461.

You, J., Liu, B., Ying, R., Pande, V., and Leskovec, J. (2018). Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. *Advances in Neural Information Processing Systems*, 2018-Decem(NeurIPS):6410–6421.

Zhou, Z., Kearnes, S., Li, L., Zare, R. N., and Riley, P. (2019). Optimization of Molecules via Deep Reinforcement Learning. *Scientific Reports*, 9(1):10752.