MOLPPO - Molecular Optimization through Deep Reinforcement Learning

By

Ahmad Navid

00000329479

Supervisor

Dr. Muhammad Tariq Saeed

School of Interdisciplinary Engineering and Sciences (SINES)

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

August 2022

MOLPPO - Molecular Optimization through Deep Reinforcement Learning

By

Ahmad Navid

00000329479

Supervisor

Dr. Muhammad Tariq Saeed

Co-supervisor

Dr. Ishrat Jabeen

A thesis submitted in conformity with the requirements for the degree of *Master of Science* in Bioinformatics

School of Interdisciplinary Engineering and Sciences (SINES)

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

August 2022

Declaration

I, *Ahmad Navid* declare that this thesis titled "MOLPPO - Molecular Optimization through Deep Reinforcement Learning" and the work presented in it are my own and has been generated by me as a result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a Master of Science degree at NUST
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at NUST or any other institution, this has been clearly stated
3. Where I have consulted the published work of others, this is always clearly attributed
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work
5. I have acknowledged all main sources of help
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself

---

Ahmad Navid,

Reg no. 339479

Copyright Notice

- Copyright in text of this thesis rests with the student author. Copies (by any process) either in full, or of extracts, may be made only in accordance with instructions given by the author and lodged in the Library of SINES, NUST. Details may be obtained by the Librarian. This page must form part of any such copies made. Further copies (by any process) may not be made without the permission (in writing) of the author.

- The ownership of any intellectual property rights which may be described in this thesis is vested in SINES, NUST, subject to any prior agreement to the contrary, and may not be made available for use by third parties without the written permission of SINES, which will prescribe the terms and conditions of any such agreement.

- Further information on the conditions under which disclosures and exploitation may take place is available from the Library of SINES, NUST, Islamabad.

This thesis is dedicated to *my beloved parents*

Abstract

With the rise of computational methods in medicine, there has been several important breakthroughs. Virtual screening, molecular docking and molecular dynamics simulations have revolutionized the field of drug design over the decades. More recently, artificial intelligence has also had major contributions to drug design. However, the problem of computational chemistry is a combinatorial one: molecular function is non-linear and combinatorial in nature. Finding a relationship between chemical space and functional space has been quite challenging. Fortunately, deep reinforcement learning provides some hope in approaching this problem. Earlier work named MOLDQN has used a discrete deterministic approach to modeling molecules from scratch, this thesis aims to use a more generalized probabilistic approach. This thesis uses the Actor-Critic formulation in reinforcement learning to explore the chemical space in terms of the quantitative estimate of drug-likeness (QED), Tanimoto index, and a newly designed diversity score which penalizes highly similar molecules. Results from the algorithm show that the system can learn to model chemical bonds better than earlier work, however the system cannot model aromatic rings accurately. This may perhaps be because of the three-dimensional nature of resonance structures not captured with the Morgan fingerprint which the algorithm uses.

Keywords: *Reinforcement learning, Actor Critic, Q-function approximation, molecular optimization, Gleevec, Imatinib, tyrosine kinases*

Acknowledgments

I want to thank Dr. Muhammad Tariq Saeed for his supervision. He has been foundational not only in academia but also in industry. I would also like to thank Dr. Ishrat Jabeen for being a guide at every step.

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

Π	Policy
ϵ	A constant used in exploration of actions
DQN	Deep Q-Network
PPO	Proximal Policy Optimization
RL	Reinforcement Learning
Q	Q function
V	Value function

Chapter 1

Introduction

1.1 Advent of Artificial Intelligence

After the revolution in image processing with ImageNet [8], deep learning has revolutionized many industries including advertisement [9, 10], speech recognition [11], search engine [12], and self-driving cars [13]. Deep learning has become relevant in every major field of study. This does not preclude drug discovery [14, 15].

Computers have historically played a very important role in drug discovery [16] [17]. Storing and maintaining drug banks and virtual libraries have been part of industry since the 1980s [18]. Even more, with the advent of virtual screening, computers played an important role in the making of drugs. The first major breakthrough was Gleevec [18]. Gleevec was designed to target tyrosine kinases against chronic myeloid leukemia. It was one of the first drugs to have come from a purely computational pipeline: a library of molecules designed and stored are docked against a given target protein. The best candidates are then taken to experimentation. The drug was introduced into the market in 2001 [18].

1.1.1 Artificial Intelligence and sub-branches

There are three main branches of artificial intelligence which are as follows:

Supervised learning

Supervised learning considers the problem of handling labelled data. Both classification and regression problems belong to this class of machine learning [19]. Classification tries to approximate whether a given set of features belong to a class, while regression tries to approximate the range of values where a given set of feature can belong to.

The goal of a supervised learning problem is to find a relationship between an outcome and the attributes selected. The simplest case is that of a linear relationship between the predictors and the outcome. That linear relationship can be mapped with a mathematical equation like logistic or linear regression for classification and regression problem, respectively. Linear regression tries to approximate the best weights for each attribute to outcome while logistic regression fits on the log likelihood of a given class. The equations are as follows:

For the linear regression case [20] where one outcome is dependent on one attribute:

$$y = mx + b \tag{1.1.1}$$

This is the case where y, a dependent variable depends on an independent attribute called x with a factor of m called the slope, with an intercept of b. Both b and m are constants that can be thought of baises, and can be mapped by fitting the slope to a given training data set. An intuitive example of this relationship can be that of the amount of sunlight directly hitting the earth and the temperature of the earth. More sunlight hits the surface of the earth in a concentrated manner, the higher the temperature. The factor b in this case would be a combination of the atmospheric pressure, carbon concentration and the rotation of the earth. Hence, the temperature of the earth can be fitted to a line given this equation. Indeed this has been shown to a good precision as can be seen in the graph below.

Coming to logistic regression case [21], the formula now turns into a measure of probabilities:

$$P = \frac{e^{mx+b}}{1 + e^{mx+b}} \tag{1.1.2}$$

A simple explanation of this equation is the exponent of the constant e to be the linear regression equation both in the numerator and denominator. This equation calculates

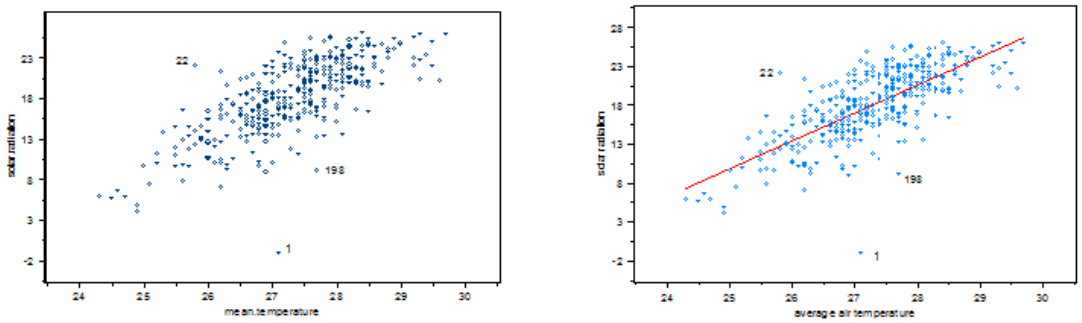

Figure 1.1: Sunlight versus surface temperature of the earth plotted as a linear relationship.

whether a given independent variable is more likely to belong to one class over the other, the numerator divided by the denominator. More often, this equaiton can have more than one attributes to dictate a relationship. And thus each attribute would come with a slope and an intercept as constants also called their bias terms.

To map more complex relationships, there has been tremendous success. Initially, algorithms were made specifically to map gep,etroc functions, but with the help of tree-based and subsequently artificial neural networks, these methods promised mapping any sort of non-linearity to find relationships between the attributes and their outcomes.

Artificial intelligence has had tremendous success in almost every branch of industry. How people have been able to achieve this is beyond the scope of this thesis.

Unsupervised learning

Unsupervised learning or clustering are a set of methods that cluster unlabelled data into maximal hierarchy. The aim of unsupervised learning algorithms is to - without human intervention - find structure in a data set by modeling relationships within the data. Unlike supervised learning approaches, there is no dependent or independent variables. In fact, each variable is somehow interconnected to each other to produce a structure overall [22]. These include K-Means clustering, PCA [23] and more recently, autoencoders [24]. Many of the natural language processing algorithms belong to unsupervised learning [25].

More advanced clustering methods have also been re-purposed to extract important features from noisy data. These methods can be combined with supervised methods for for enhanced data efficiency and better convergence. The advent of these methods are

also beyond the scope of this thesis.

Reinforcement learning

Reinforcement learning is the domain of trial and error. Also called semi-supervised learning, this mode of learning is at the heart of this thesis. Inspired from learning in the real world, reinforcement learning is framed in terms of an agent and its environment. An agent acts in an environment to get a reward and as a result its state changes. This change of state can be evaluated against the end goal. As can be seen, no data is fed to the agent, in fact the agent has to produce data at each time step. Actions taken are stored as data. The agent learns from its actions by maximizing the reward gained from its actions. Hence, the aim of reinforcement learning is to achieve a goal in a complex dynamic environment; complex in that any number of actions can be taken, and dynamic in that the environment changes as a result of the actions taken.

The goal of reinforcement learning is to learn to reach a goal through reward [26]. 1.2 shows a taxonomy of RL algorithms with specific end goals for each branch. RL methods are mainly divided into two classes: ones that require a model of the world, and those that do not. Those that do require either learn the model or produce the model. These methods include the likes of AlphaZero [27] that has successfully produced a generalized model for a wide range of board games like chess and Go [27]. On the other hand, model-free systems focus on learning optimal actions or how good those actions were, policy-optimization and Q-learning respectively. Deep Q-Networks were one of the first to be implemented with success [28] in board games. Following sections go into the details of these two approaches. But before we go into the details of these two branches, the main ideas in reinforcement learning need to be visited to gain an understanding of the thesis.

Main components of RL

Regardless of which algorithm used, reinforcement learning has a few key parts that are common to all. This means that all these algorithms need to be framed in these same terms.

These parts summarized in 1.1 come together into a training loop for an agent to learn the optimal policy or the value functions thereof. The policy is the set of actions taken to

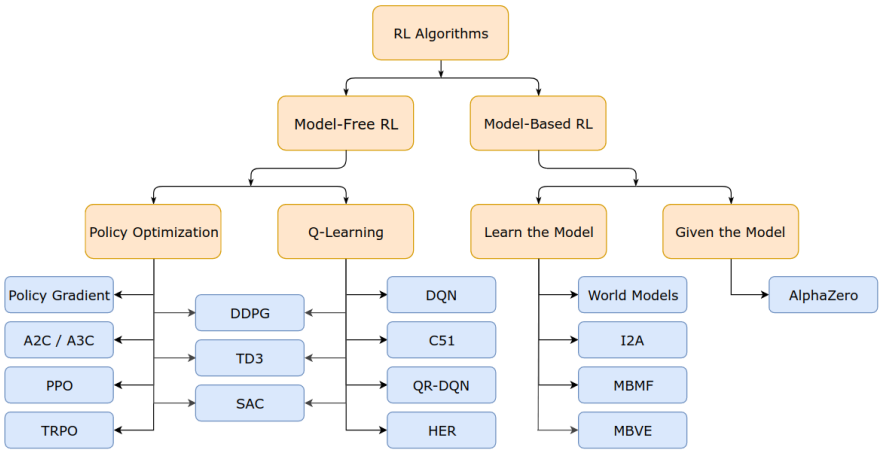

Figure 1.2: Family tree of RL algorithms. Model-based algorithms require specifications on the rules of the environment. Each action has rules associated with it, whereas model-free algorithms do not require rule specifications. Model-based algorithm endeavor to learn the model or produce the model. On the other hand, the counterpart tries to learn the value functions of optimal actions (Q-learning) or optimal actions (policy optimization). This thesis takes the DQN approach and translates it to PPO, a policy gradient method.

Agent	An agent that generates actions by acting in the environment
Environment	The playground for the agent to reside in and generate actions
Reward	A scalar value that is observed after an agent acts
Action	What the agent does in the environment
State	The state of the agent before and after acting

Table 1.1: Reinforcement learning basic components

reach an end goal; the value function is the evaluation of too good or bad the policy was at each iteration. The policy and the value function are generated both in the policy-gradient and Q-learning methods. In the former case, policy is calculated directly, whereas in the latter, the policy is inferred from the value function. In general, policy optimization methods are faster in terms of computation than Q-learning approaches [29][30].

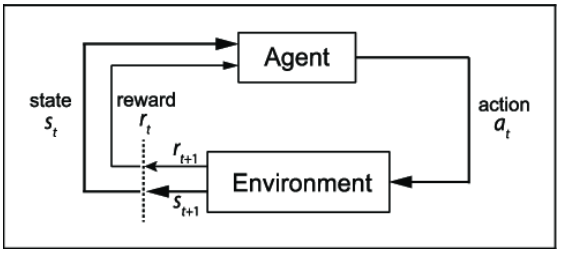

Figure 1.3: An RL setting has an agent that acts, an environment the agent resides in, actions taken produce rewards and move the agent to a new state. The agent learns by storing these states and converging to a set of actions that produce optimal reward.

Q-learning

Q-learning or quality learning was the method of learning on the value function. Q-learning included the likes of DQN [28] which was part of the one first breakthroughs in board games used in AlphaGo [31]. A schematic diagram for AlphaGo is in **??**. An important theme in DQN was that the actions taken are discrete. Monte Carlo Tree Search (MCTS) and DQN had been the underlying algorithms underlying the success of AlphaGo and subsequent system [31]. In the case of Go, each piece in the board can move in four directions corresponding to the grid points in the board. Contrast to the DQN approach, other Q-learning approximate map to continuous action spaces. Application to Q-learning in such algorithms include bi-pedal walking and self-driving cars etc.

DQN was central to this thesis because the work this thesis was based on uses the DQN approach [32] for molecular optimization. Discrete actions taken at each time step were fed into a action-value network to evaluate the value function and handed out a reward. The environment of the molecule was the molecule itself with the reward function corresponding to the quantitative estimate of drug-likeness [33] and Tanimoto index [34] of the molecule to a target, if a target molecule was provided. Actions included addition of bonds, removal of bonds, no modification, inclusion of aromatic rings, and the minimum number of atoms in the ring. These discrete actions were rolled out at each iteration to be passed through QED reward signal. The outputs from each iteration was stored in a replay buffer, or essentially the memory of the system. The instances in the replay buffer were sampled and a gradient ascent was used to maximize the value function. This approach will be discussed in more detail in the Methods section.

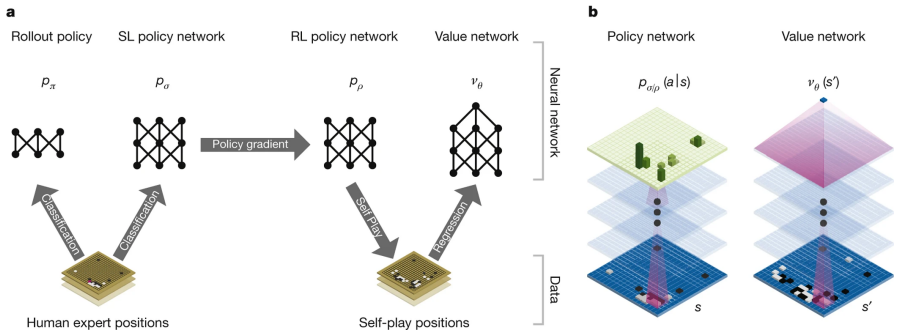

Figure 1.4: Workflow of the Q-learning network in the AlphaGo paper. Steps include, generate action, feed into policy and value networks and evaluate the goodness of the actions until convergence.

Policy optimization

Policy-gradient methods emphasize the actions taken. Even though value functions for each episode are taken into account, it is the actions at each time step that is optimized on. RL provides a wide range of algorithms for this approach. Proximal policy optimization [35], Soft Actor Critic [36] and TD3 [37] are all examples of policy optimization methods. These algorithms have had important contributions, each having its respective strengths and weaknesses.

What is important to note is that there is no clear dichotomy for these algorithms. They can both be policy gradient and Q-based simultaneously as well. In fact, SAC and DDPG [38] are good examples of hybrid methods. What makes them special is that they have been made to tackle the problem of overestimating value functions [30] and smoother learning over the course of simulation [30].

1.2 Molecules

The second pillar of this thesis is biology. Reinforcement learning is a tool used on biological molecules. What is important to realize is that biology has become a computational problem over the years [39]. Finding new therapeutic molecules is at the core of applied biological sciences. It takes years if not decades to find new molecules for therapy. The average lifetime of a molecule to reach the market is 15 years [40]. Just like vaccines, molecules go through four phases of clinical trials. The first phase is about the dose-ranging in lower animals, second phase entails drug efficacy studies on higher

mammals, while the last two phases are effectiveness studies on a few human cases, fourth phase uses a larger group of human trials. These phases exclude grant approvals, scoping and other manufacturing stages. Thus, finding drugs for rare diseases are often challenging because of this overhead [41]. What exacerbates this problem is the monopolistic nature of the pharmaceutical industry today [42]. The top three US-based pharmaceutical companies are worth more than 100 billion USD [42]. The aim of this thesis is to help mitigate with both of these phenomena: firstly, machine learning can contribute to drug design and secondly, reinforcement learning can help produce data.

1.2.1 Computational methods in drug design

The first breakthrough in drug discovery using computational methods was the 2001 introduction of Gleevec into the market [18]. This molecule was computationally predicted using a technique called virtual screening against a library of potential compounds. The methodology will be discussed in the next section. Since then the molecule has been dubbed the "magical bullet" against chronic myeloid leukemia, a type of white blood cell cancer. Patients have been shown to show little signs of tumour progression six years after having had the therapy [18]. Ever since, virtual screening against bigger and bigger libraries has been part of the intiial screening of potential molecules [43]. This thesis uses Gleevec as a case study to see which bonds and atoms are important in the molecule so that similar set of features can be built upon the Gleevec scaffold or to optimize on the molecule further. The latter has already been done with various alternatives to Gleevec already present in the market. These include hydroures [44] and bosutinib [45] both of which are tyrosine kinase inhibitors just like Gleevec itself.

Gleevec: mechanism of action

The mechanism of action for Gleevec is that it competitively binds to the ATP binding site of tyorisne kinases halting function. Tyrosine kinases are a large family of proteins [**?**] out of which Gleevec targets only BCR-ABL [18] and also c-kit and PDGF-R [46]. These proteins are important in cellular signalling driving cellular processes. As these proteins accumulate mutation, they can over-express and lead to tumour progression. Binding of the Gleevec also known as imatinib molecule is given in 1.5. A schematic diagram of the role of BCR-ABL in overall cell processes in provided in 1.5.

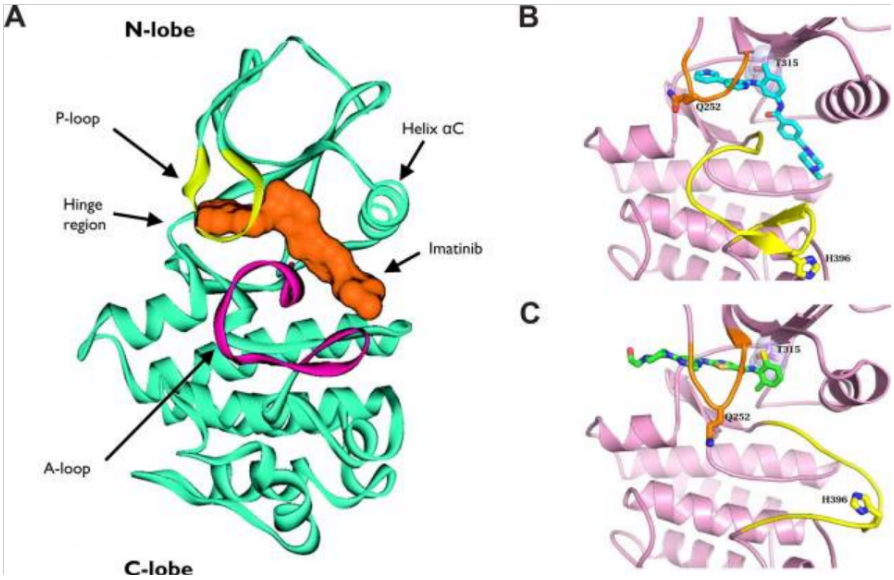

Figure 1.5: Three-dimensional visualization of binding of the Gleevec molecule to BCR-ABL complex in the UCSF Chimera software package [1]. A shows the binding of the protein the hinge region and P-loop of the complex, selectively inhibiting it from binding to ATP molecules driving down abnormal cellular growth in tumours. B and C show different angles of the protein.

Understanding Gleevec's role in inhibiting BCR-ABL complex is key here. Any subsequent molecule proposed must have features that optimize on what the Gleevec scaffold already provides: the molecule must have more or less rigid bonds to be able to dock to the hinge region forming bonds at the P-loop for stability and it must only be selective against the BCR-ABL system and not bind to other proteins in the human body apart from the ADMET properties that are crucial to any potential therapeutic compound . ADMET stands for absorption, distribution, metabolism, excretion and toxicity.

Going forward, the next chapter gives an overview of this history of reinforcement learning and also drug design before coming to applications of machine learning in drug design. The 2 chapter will also highlight the need for quality data for any major breakthroughs to happen in the area. The 3 section will highlight steps taken in this thesis to arrive at one formulation of molecular optimization. The 4 section will demonstrate results from the approach with concluding remarks in 5.

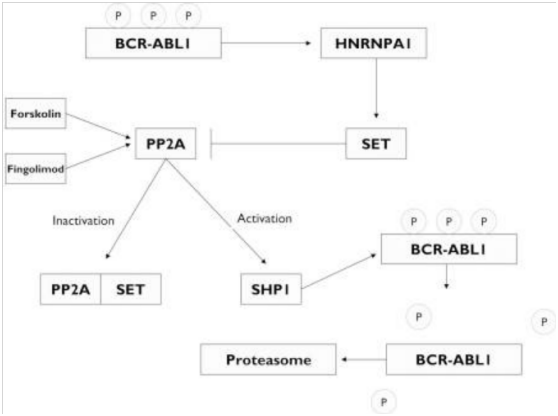

Figure 1.6: Schematic diagram shows the role of BCR-ABL complex in regulating downstream proteins. It is a prime activator of SHP-1, a tyrosine phosphatase protein involved in cell signalling cascade processing. Over-expression of the BCR-ABL complex leads to further activaiotn of SHP-1 which subsequently over-stimulates other proteins in the cell signalling cycle to drive tumourogenesis.

1.3 Research Gap

Perusing literature, what could be surmised was that there is still work needed to be done in the area of drug design with deep reinforcement learning. While both GCPN [47] and blood brain barrier optimization [5] used policy gradient approaches in combination with other deep learning methods to drive molecular optimization, there still remained a gap in basic science research in algorithms. On the other hand, MOLDQN [32] utilized a double Q-learning approach. Finding optimal policy from Q-learning is not a direct step but takes a few additional steps to know which actions are the most important. Additionally, MOLDQN [32] was deterministic in its exploration of the chemical space. This left the question of local minima unanswered.: some chemical combinations may be functionally relevant but could be sub=optimal. Assigning a constant decay term in the chemical space was problematic as the authors have mentioned in their work. The combination of complexity in the applied domain and a rather simplistic algorithmic contribution in the basic science domains, left the gap for proximal policy gradient methods in optimization.

1.4 Problem Statement

Summarizing the problem at hand here, drug discovery is a very cost-intensive process both in terms of time and money. An average molecule takes 15 years to come to the market. And yet, the pharmaceutical industry is worth billions of dollars [48]. Additionally, the industry is monopolistic which means quality data is not freely available [48]. Hence, there is a need for automation and spread of open-source methods in producing molecules [49]. This thesis aims to optimize molecules based on their chemical properties.

1.5 Research objectives

The aim of this thesis is to establish two key points:

- Establish an ablation study to compare which molecular properties are most important in defining a molecule. These include the quantitative estimate of drug-likeness, the hydrophobicity and finally Tanimoto similarity.
- Implement the proximal policy optimization algorithm to compare with the DQN approach

Chapter 2

Literature Review

This section surveys literature on the progress made in applications of reinforcement learning in drug design. But first, we need to establish the importance of reinforcement learning and how it can theoretically be applicable to drug discovery itself. For that advances in reinforcement learning is to explored first and the ideas that inspire the endeavor to apply the technique in drug discovery. The first section discusses reinforcement learning in chess games. The second section discusses the lessons learned from the breakthroughs and lastly, reinforcement learning in drug discovery is discussed.

2.1 Reinforcement Learning

Reinforcement learning is learning through the help of rewards and penalties. A reinforcement learning problem in essence, is an agent that interacts with its environment to generate observations and get rewards. These rewards drive actions of the agent towards a particular goal. The agent can get step-by-step rewards called a rich environment or it can be sparse. An example of the former case is a self-driving vehicle while chess is an example of the latter case. Additionally, the actions taken can be discrete or continuous. A bipedal agent would be taking discrete steps in walking while a self-driving vehicle would be taking continuous values as actions.

AlphaGo

One of the first breakthroughs in reinforcement learning came in 2016 with the AlphaGo agent beating world-class Go player Lee Sedol. AlphaGo won all the five games but the

fourth one [31]. all games were won by resignation by Lee Sedol. AlphaGo had learned from playing with expert Go players. What is important to note is that the system was dynamic in the way it competed against the human player. 1.4 illustrates the workflow for training in the AlphaGo agent.

AlphaGo was a DQN agent. This entailed a few things:

- **Each step was evaluated via a value function.** Compared to policy-gradient methods which arrive at the actions taken. Value functions dictate the goodness of each action, while policy encompass the paths taken.
- The action space was discrete. DQN is a discrete action space algorithm, each step is taken in a step-wise manner.
- **No replay buffer was used making the algorithm less sample efficient.** Sample efficiency is a theme used in the AlphaZero system to narrow down the search space better [27].
- **AlphaGo was trained on playing with human players.** The more advanced AlphaZero system was built entirely on self-play i.e., the agent played against itself to generate actions and learn from those actions [27].

AlphaZero

The next iteration of the AlphaGo system came two years later in 2018 [27]. This stem was a natural abstraction over many board games including Go and chess. What was different was that the agent had learn to play with super-human performance with no prior experience. The system learned to play by itself called self-play. Schematic diagram in 1.3 shows the algorithm at work. Importantly the algorithm used tree-based searching while playing. This is important in board game scenarios and boosted the performance of the system [3].

There are a few changes to the overall algorithm:

- **Policy is decided using a probability distribution.** Adopting such a method means that the agent can explore the action space better.

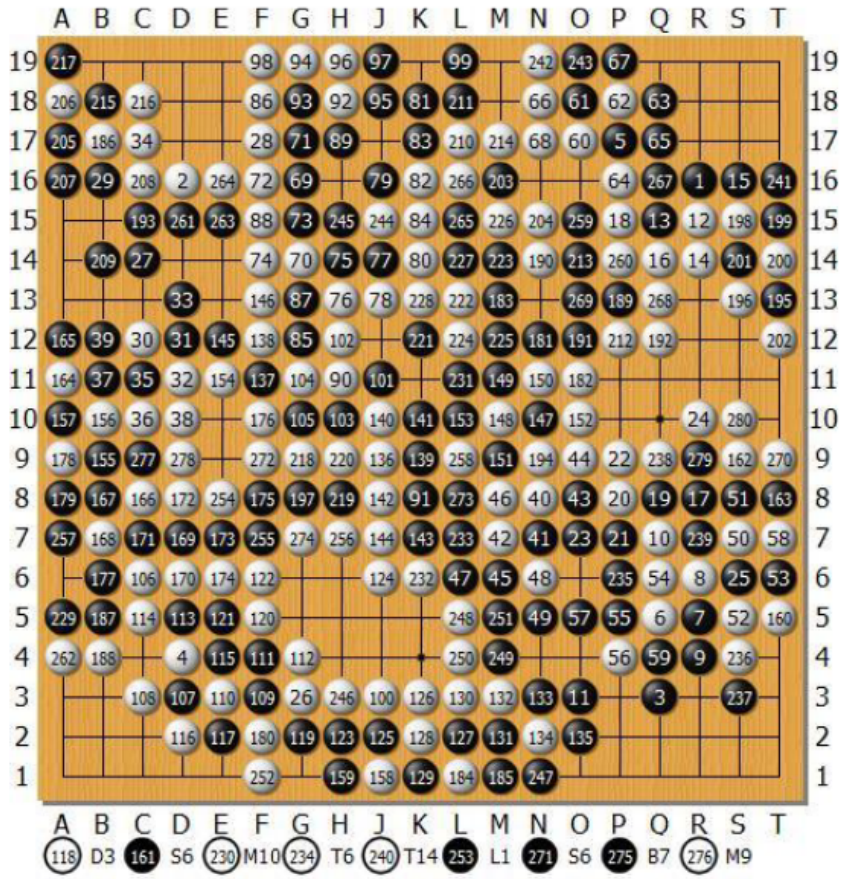

Figure 2.1: Fifth game between Lee Sedol and AlphaGo. White is AlphaGo, black is Lee. White won by resignation [2].

- **The policy and value network are combined.** Unlike the DQN approach which only had separate outputs from the policy and value networks, the AlphaZero system used an output that combined both the policy and value networks [3].
- **As a result of the probabilistic nature of the policy network, the action space becomes continuous.** This approach helps with exploration as it would not be set to some constant value throughout self-play but would assume every-changing probabilities based on the state space in the game board [50].

These two are only just a few instances of applications of reinforcement learning in the real world. There is a large corpus of work being done in the self-driving car domain [13][51][52] and robotic [53] [54]

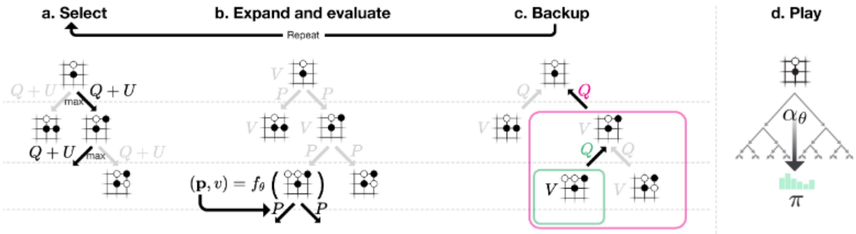

Figure 2.2: Workflow for the AlphaZero system. [3]

2.2 A brief touch to computational drug design

On the other hand, the field of computational chemistry and drug design has had major success over the last few decades in bringing novel drugs, both natural and synthetic, to the market. As of December 2019, there has been more tha 1,500 drugs approved by the Food and Drug Administration [55] which amounts to more than 1.2 trillion USD market share as of 2020 [56]. This speaks to the success of the pharmaceutical industry in reducing human disease.

Relevant to this thesis is one of the first computationally predicted drugs, Gleevec [18]. It was introduced in the market in 2001 used against chronic myeloid leukemia. The drug was discovered to a method that has since become a tried and tested method protocol [43]. This method is called virtual screening. The method includes building a library of different molecules and docking them with the target protein of interest. In the case of Gleevec, it was discovered that the molecule was highly selective to a sub-class of tyrosine kinases that include BCR-Abl, PBGA and others [18]. The ligand interaction map of Gleevec and Abl is shown in the figure below.

This molecule has already been discussed in the previous section. This section was meant to connect the molecule with the continuing line of work in reinforcement learning.

2.3 What can RL do for us?

In an infinite action space, like chess where the number of moves is larger than the number of atoms in the universe, reinforcement learning can provide a path for efficiently exploring the space [57]. Additionally, emergent properties that have been been explored is also a phenomenon desired oftentimes. So the combination of these two factors are

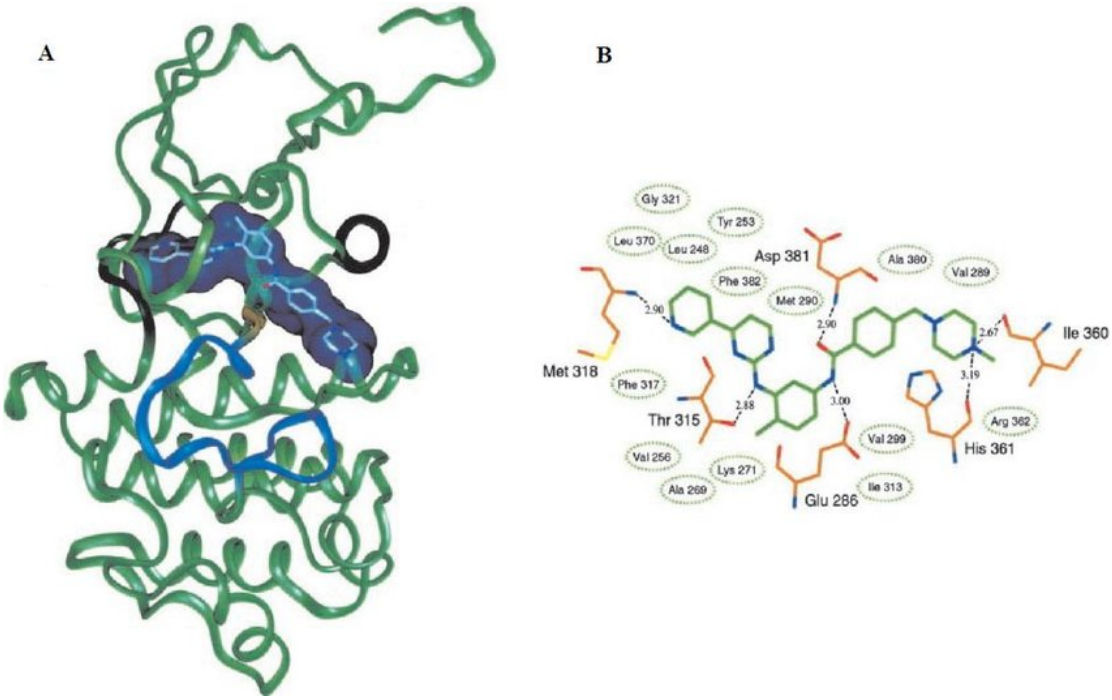

Figure 2.3: Ligand interaction map of Gleevec with Abl tyrosine kinase. Interaction map shows the hydrophobic structure of the Gleevec molecule residing in an iverse V-shaped conformation with the hydrophilic groups interacting with the histidine component of the Abl protein, while the hydrophobic chains interacting with isoleucine and pehnylalanine. [4]

the central themes of this thesis: to explore the molecular space more efficiently, and to find emergent properties within molecules that can be generalized to other compounds. What is important to note is that in the chemical space, the chemical properties are based on the combinatorial nature of the system: a functional group can contribute to the compound's toxicity in a very dramatic manner, for instance. More importantly, a compound's efficacy is essentially a property of the target protein it is designed to bind. Furthermore, the protein structure where the proposed compound needs to bind has to have the right three-dimensional structure and also the right chemical properties to form bonds. The former is a three-dimensional property while the latter is a chemical one. Both of these features are important for any new proposed compound. Thus every compound is therefore, very dependent on the protein itself. However, in this project, the QED and Tanimoto similarity of the Gleevec molecule was considered.

2.4 RL in drug design

Reinforcement learning has already been applied in one form or another in drug design. Graph convolution coupled with policy gradient [47], de-novo policy gradient [58], multi-objective optimization [59] are all instances of RL in drug discovery. To the scope of this thesis, the most important paper was by Google [32].

The graph convolution policy network work used a combination of graph convolutions to characterize a molecule as a network of graphs [47]. Each node represented an atom with specific weights. Each node had an edge which was the molecular property of the atom in relaiton to another atom. For instance, a carbon node oculd be attached to an oxygen atom. Hydrogens were ignored in this study. The edge between carbon and xoygen represented the atomic interactions between carbon and oxygen, for instance their dipole, inter-atomic bond length, and other chemical properties. Trhough algorithms developed for graph convolution for instance neighbor-walking and color mapping [47] coupled with policy gradient optimization, molecules were generated that had similar or even better properties than the starting molecule. Although promising graph convolutions came with computaitonal overhead. Graphs convolutoin have also been known to be very specific for a given class of molecules and cannot be generalized to other molecular classes which need their own training and optimization [60].

On the other hand, molecules that can traverse the blood brain barrier has been a major challenge in drug design [61]. The brain is not directly linked iwth blood, instead the brain interfaces with the bloodstream through cerebrospinal fluid [62]. Targeted therapy for Alzheimer's disease and other mental disorderes need molecules that can pass through the blood brain barrier has been a challenge [63]. Because of the more complex nature of the problem, attempts have been made to optimize available molecules using reinforcement learning which include [5]. The work used a combination of pre-training with generative-adversorial networks [64] and optimized through policy gradient optimization. It seems that policy optimizaiton has been popular in optimizing molecules in academia. A schematic of the neural architecture is given below. Just like the previous work, using GANs coupled with a relatively deep neural arthictecture for policy gradient came with computational overhead [5]. Reinforcement learning has been known to suffer from loss of learning with deeper neural networks [6]. Additionally, GANs itself generated many samples that did not have biological meaning. However, the work was

geared toward molecules that could traverse the blood brain barrier so the problem was specific enough for the algorithm to be implemented in this manner.

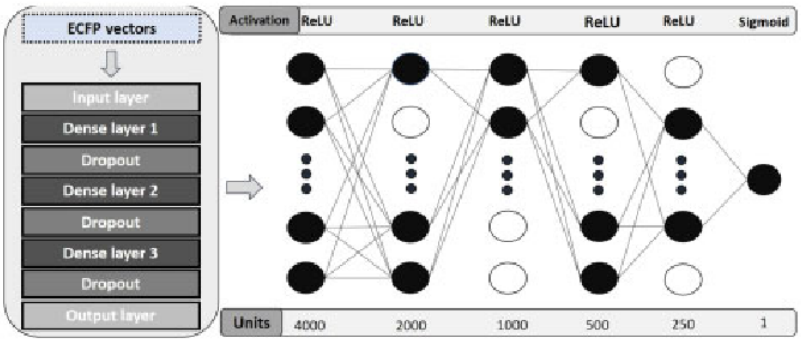

Figure 2.4: Neural architecture for optimization of molecules using deep reinforcement learning targeted for crossing the blood brain barrier. Importantly, the architecture is deep with many layers unusual for deep reinforcement learning tasks [5]. Deep reinforcement learning nueornal layers tend to be shallow for reasons for backpropagation. The gardients may not be able to reach the initial layers costing learning problems [6].

Chapter 3

Materials & Methods

3.1 Introduction

Python 3.8 was used in the project with PyTorch 1.9 as the main framework for the RL environment. The RL environment was built using the Rdkit 2021.03.1b1 library.

PyTorch was utilized for the reinforcement learning framework while RDkit featurized the input molecules. The terminal state of the algorithm was defined as molecule reaching a QED value that cannot be further improved upon further iteration. The other terminal state of the algorithm was when a chemically wrong molecule was produced as a result of wrong actions taken by the policy network. The QED, Tanimoto similarity, and a diversity score were incorporated into the reward signal. The diversity score was unique in this thesis and had not been used in the original work in MOLDQN [32]. The reward signal was termed to be the reward the molecule received as a result generating actions such as bond creation, bond breakage or no modificaiton. The return was defined as the value received at each step from the agent.

An overview of the workflow can be found in the following figure.

Python	3.8
PyTorch	1.9
Rdkit	2021.03.1b1

Table 3.1: Libraries used in MOLPPO

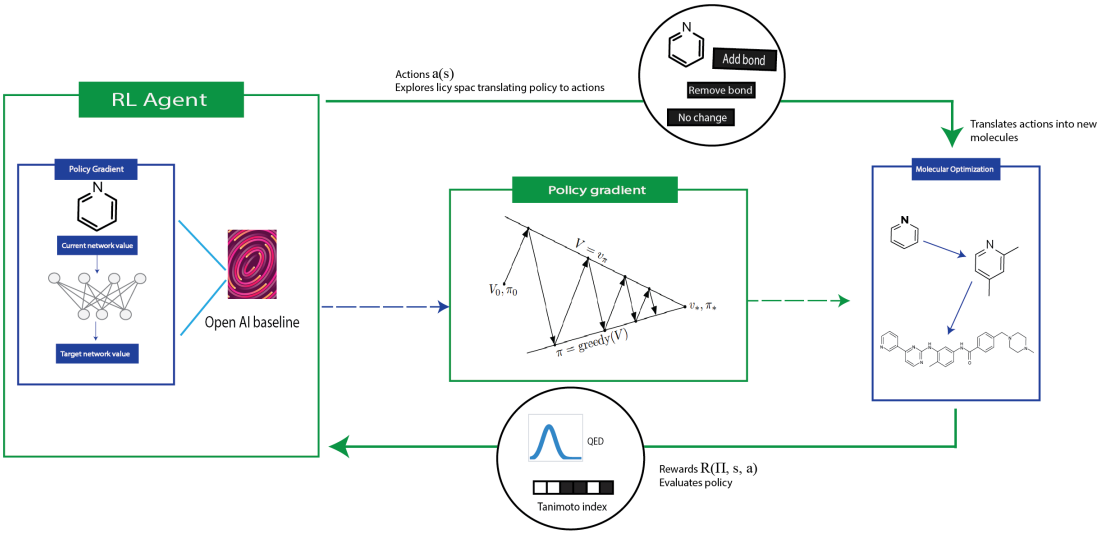

Figure 3.1: Generalized workflow of MOLPPO. A molecule was featurized in the form of a tensor in an Open AI baselines [7] platform and passed through a neural network called the policy network. The molecule in question was then modified by either bond addition or removal and was passed through a second neural network called Critic. The Critic evaluates actions made by the policy network. And a reward was calculated as a result of the modification in the molecular structure. This generation of new bonds and evaluation continued until convergence.

3.2 Formulae in RL

RL provides a set of formulae to deal with a dynamic environment. This section explains what the core formulae of RL can offer:

- Belmman Equations ties the reward received from the environment to the value functions by the agent combined iwth a discount factor. This discount factor emphasizes that the reward received at a given time step decays with the future rewards the agent may take in the future. The discount factor proposition makes sense in an environment where the agent has to explore a maze or learn to walk for instance but not in this thesis. This is because the molecule can optimize at any point in time. Although the factor can be ignored, it has not been done in this thesis. This equation is given as follow:

$$\mathrm{V}^{\pi^*} = max\left(R(s,a) + \gamma \sum P(s',a) V^{\pi^*}(s')\right) \quad (3.2.1)$$

In other words, the value function for the optimal policy is the maximum over the rewards at a given state and action plus the sum over the policy and value functions given in the next step's optimal value function. The discount factor decays as a result of exploration.

- Temporal Difference (TD) is one of the fundamental equations in RL. It provides a way of learning to predict the future before attempting, act on the action, receive a reward, and correct its previous prediction.TD provides a mathematical basis on the way to predict what the agent will achieve. Every step has a TD error associated with it which gauges what the agent predicted and how it performed. The formula for the one-step TD is given below.

$$V(S_t) = V(St) + \alpha R_(t+1) + \gamma V(S_(t+1) - V(S_t) \quad (3.2.2)$$

- Exploration vs. Exploitation. Exploration refers to exploring the environment while exploitation refers to using prior information from the environment. Any RL setting must be able to balance these two succinctly in order to behave optimally. The later two sections provide different formulae on how these algorithms deal with exploration and exploitation.

3.3 MOLDQN

MOLDQN from Google [32] used a discrete value learning reinforcement learning algorithm. Molecular optimization could be done in a host of modes, including bootstrapping, multi-objective, or just over one molecular property. The state and action were the same, i.e., every turn in the iteration, a new molecule would be built from scratch. In other words, it was not a step-by-step building of a molecule but rather generation of a new molecule in each episode. The actions included, removal, addition or no modification bonds, the number of aromatic rings that could be added. The reward signal was a function of the molecular property optimized on at hand which included the hydrophobicity [33], QED [33], surface accessibility [65] of the molecule. Each cycle, the molecule would be factorized into a Morgan fingerprint tensor [66]

3.4 MOLPPO

This thesis uses the proximal gradient method to optimize molecules. Although the action space is defined the same was as in the DQN method, what is different is that each cycle the overall scaffold of the molecule remained the same with modifications each turn. The policy was rolled out and evaluated through the on-policy actor and critic networks. The actor network is defined as the component that generated actions while the critic evaluated those actions through a value function. Similar to the DQN case, the molecule was turned into a tensor by calculating its tensor using Morgan fingerprint [66].

The exploration vs. exploitation trade-off in MOLDQN was done through the classic ϵ method. If the constant was lower than than the threshold, it would explore otherwise it would explore as the agent interacted in the environment. This exploraiton can be observed to be deterministic and not a funciton of the dynamic environment of the system. At the start of training, more and more exploration was done and as time passed, 4ϵ shrunk and the system slowly transitioned to exploitation, or trying out bonds that it had previously learned before.

Figure 3.2 gives an overall overview of th the process in MOLPPO.

The program got a molecule as input in the form of SMILES format. The molecule was featurized via Morgan fingerprint and fed into a tensor. A few molecules were generated based on the fingerprint as a result of interacting in the environment. These molecules were stored in a replay buffer until the maximum number of iterations. After this, the replay buffer was sampled, the the actions taken were evaluated through a gradient ascent. In PyTorch the replay buffer was fed to a stochastic gradient ascent algorithm and back-propagated subsequently. Best actions from the replay buffer were then used in the nest cycle of iterations and optimization with the updated weights in the neural networks.

The algorithm for MOLPPO is given below. What was notable in the process is that the exploration and exploitation was done through a probability distribution in contrast to a constant value seen in DQN methods. What is also important to note is that this algorithm also utilized a replay buffer just like the previous approach. Replay buffers have been shown to be better with sample efficiency. This was aimed to reduce the

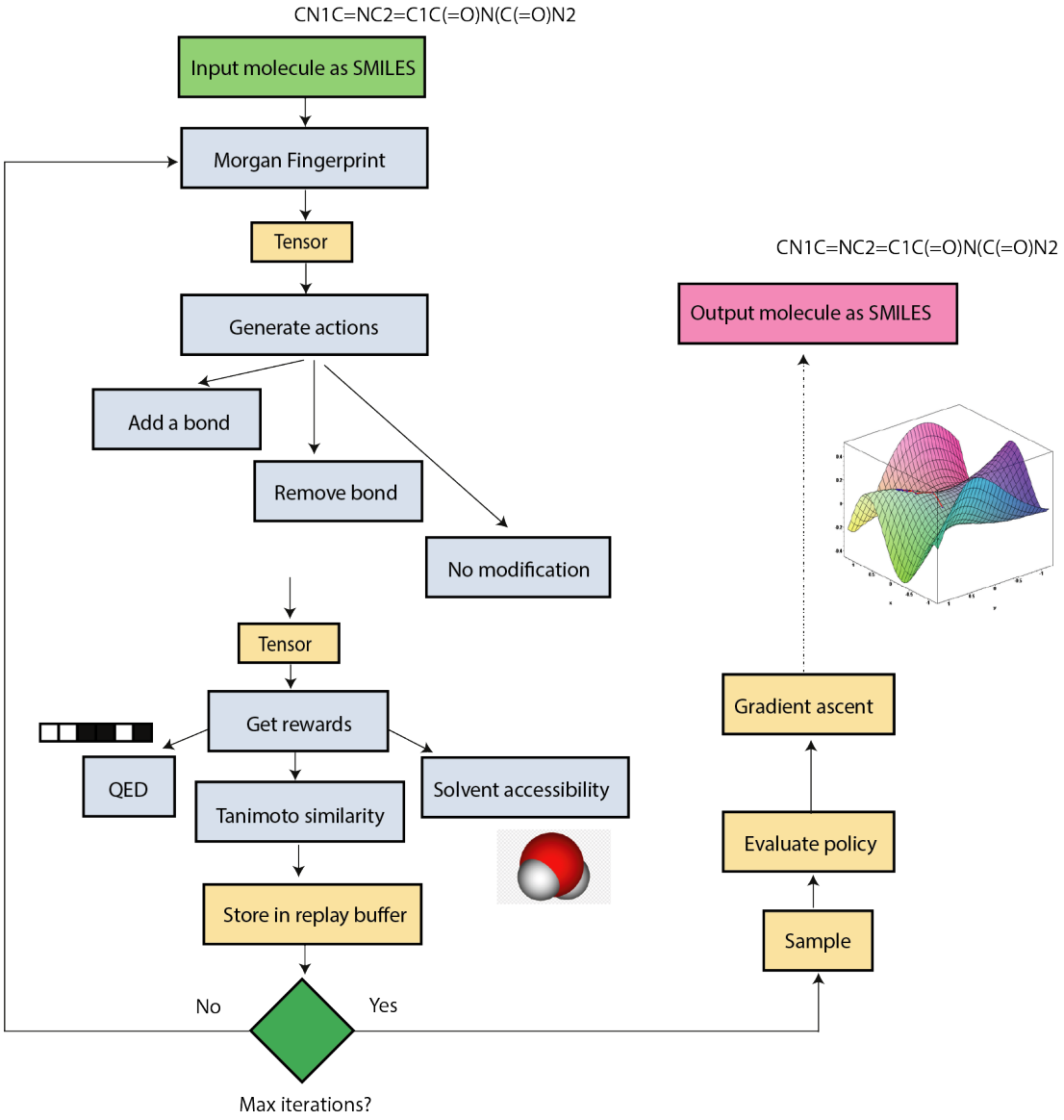

Figure 3.2: Workflow for MOLPPO

complexity of sample efficiency [67]. In other words, by having a memory of sorts to optimize upon, the algorithm was hoped to arrive at better molecules quicker because it could sample and optimize online instead of having one single molecule to optimize upon.

3.4.1 Things to note in PPO

Looking at the algorithm above, a few things are worth noting.

- The advantage function had not appeared till now. It was defined as the ratio over

the policy gained in the policy given the best one. The function was parameterized by a reward clipping function that limited the exploration of the algorithm. The reward function was calculated by the equation below. Though the effect of this equation remained to be seen in the molecular optimization case.

$$L(s, \theta_k, \theta = min\left(\frac{\pi_\theta(a_t, s_t)}{\pi_{\theta_k}(a_t, s_t)} A^{\theta_k}(s, a), clip\left(\frac{\pi_\theta(a_t, s_t)}{\pi_{\theta_k}(a_t, s_t)} A^{\theta_k}(s, a), e - \epsilon, 1 + \epsilon\right) A^{\theta_k}(s, a)\right) \tag{3.4.3}$$

- The algorithm is on-policy meaning that the value function is estimated online as the molecule is built in the environment. This reduces the computational complexity of the system in exchange for molecular unreliability. No replay buffer in the PPO case. This means that the algorithm can potentially re-visit molecules it has constructed. This was a choice made in this thesis but could be implemented in a future exercise.

- The reward signal in the PPO case has a few additional components. Firstly a diversity function was introduced which penalized the algorithm above 95% similarity. Secondly, the surface accessibility score was also incorporated into the reward signal. These were attempts to find a molecule with overall better binding properties not just optimizing on a given set of properties.

3.4.2 Neural Architectures in MOLPPO

Given that MOLPPO used an on-policy Actor-Critic approach, there were two main shallow neural architectures. One that dealt with the policy called the Actor and the other that dealt with the value function, the Critic. Unlike off-policy approaches where there were two policy methods: the target and the behavior, there was only one policy method here, however. Off-policy algorithms uses two networks while on-policy uses one.

- **Actor.** The actor method took an observation state's dimensions and gave out another observation state-like tensor. The Actor in MOLPPO expanded to a 256-dimensional tensor sand witched between TanhH [68] activation function and then moved back to the Morgan fingerprint one dimensional tensor using a softmax function [68].

- **Critic.** The Critic method evaulated the policy of the Actor by assigning it a q-value. Simply put, the Critic modeleled the actions of the policy network on to the environment and scored it. The architecture in MOLPPO had a Critic that took the Morgan fingerprint expanded to a tensor with 256 dimensions then came back to a scalar value. In between the layers, Critic also used TanhH activation function [68]. However the output was a scalar quantity, not another molecule like the Actor.

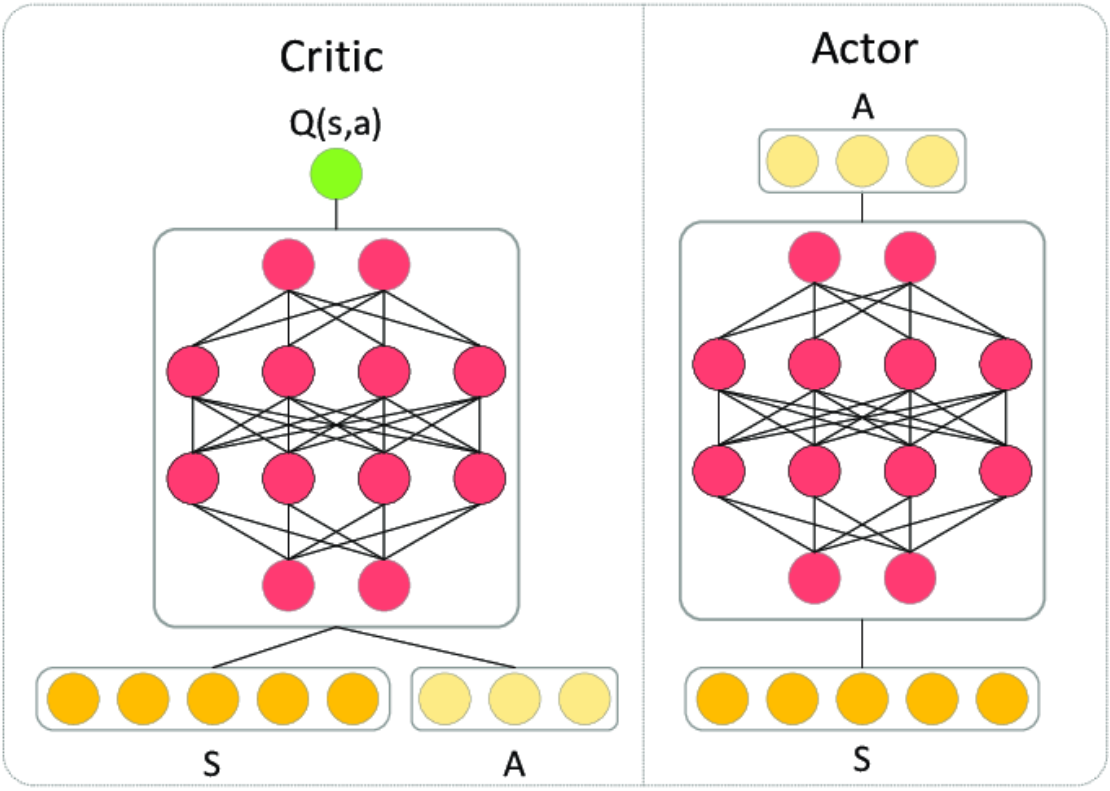

Figure 3.3: Neural architecture e for MOLPPO.

There are two architectures at play in MOLPPO. Unlike DQN methods, the policy and value functions are calculated by separate networks called the Actor and the Critic. The Actor method evaluates the policy while the Critic method outputs the q-value or the policy at each step. Both neurons take the Morgan fingerprints of a molecule as input, which is also the observed state and action, expands the number of neurons to 256 in the hidden layer with a TanH activation function. While the Actor squeezes the dimensions back to the same Morgan fingerprint-like tensor, the Critic network outputs a scalar quantity.

Next chapter documents the results for running this algorithm for about 10,0000 iter-

ations, the optimized molecule it produced and the time it took to arrive at an end result.

Algorithm 1 Molecular Actor Critic

Require: initial policy and value function parameters θ

1: **for** $k \leftarrow 1$ to N **do**

2: Roll out policy $_k$ for given molecule

3: Compute reward R on the go

4: Compute the advantage function A_t based on current value function V_t

5: Update policy by maximizing the clip function

$$\theta_{k+1} = \frac{1}{D_k} \sum_{t=D_k} \sum_{t=0}^{T} min \left(\frac{\pi_\theta(a_t, s_t)}{\pi_{\theta_k}(a_t, s_t)} A^{\theta_k}(s, a), g(\epsilon, A^{\theta_k}(s, a)) \right) \tag{3.4.1}$$

6: through stochastic gradient descent with Adam

7: Fit function with mean-squared error

$$\psi_{k+1} = \frac{1}{D_k} \sum_{t=D_k} \sum_{t=0} (V_\psi(s_t) - R_t) \tag{3.4.2}$$

8: with gradient descent

9: **end for**

Chapter 4

Results

This section documents the results from running MOLPPO on Gleevec. The importance of the Gleevec molecule has already been mentioned in 2. This section is divided into two main parts. Firstly, the tool can take a molecule as a starting point and optimize on its QED [33], Tanimoto similarity [34] and a new diversity function that penalizes the Tanimoto similarity to be more than 95%. The latter is relevant so that the tool does not converge to the same molecule. Secondly, the tool can also use a molecule as a target and construct moieties and functional groups to build the target molecule. This allows us to get a better understanding of the entities important to a molecule.

MOLPPO has been run using NVIDIA 1020 graphics on an Intel Cor i5 processor. Many experiments have been run ranging from 6000 iterations to 6 million. The q-function versus returns graphs (4.6) have been generated on ten thousand iterations while graphs showing reward and loss 4.1 amount to 4 million cycles.

4.1 Optimizing a molecule

Starting from a molecule given as input, the tool can optimize based on multiple objectives. Multi-objective optimization is based on multiple criteria, in this case the quantitative estimate of drug-likeness [33], Tanimoto similarity [34] and a newly introduced function called the diversity score. This score penalizes the system if the molecule becomes more than 98% similar to the starting molecule. This is because we do not want the algorithm to converge to the starting molecule and allow the esystem to explore other functional groups. Metrics for any RL system includes the loss and the

reward functions logged across iterations. The reward signal has just been described to be multi-objective while the loss is the TD error used in RL.

Of note is the fact that each cycle, different molecules are generated and are independent of the iteration before and the state and action spaces are the same. Given a roll-out instance, a molecule already in the state space is taken and a new molecule is generated and stored in the action tensor and evaluated whether they are chemically sound - meaning that the molecular bonds follow the laws of chemistry - and then subsequently passed through the reward signal. The snapshots show not only that the algorithm optimizes the molecule but also learns to make chemically correct structures.

Figure 3.2 gives an overall overview of th the process in MOLPPO.

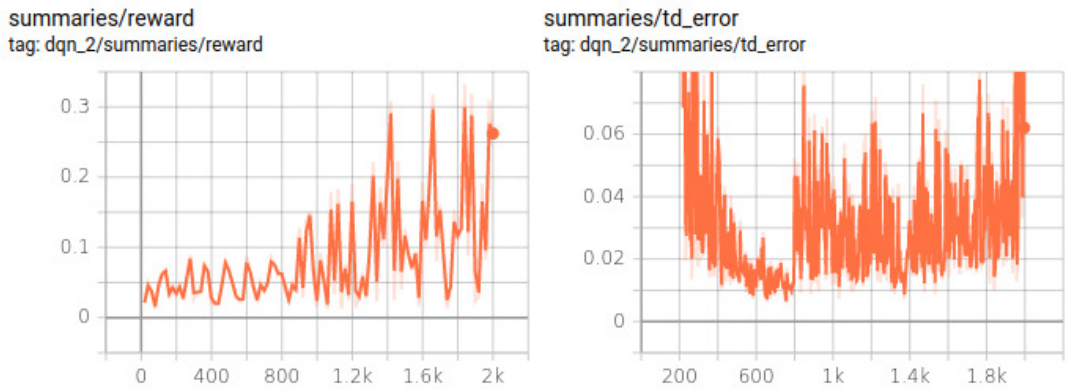

Figure 4.1: Loss and reward function logged across 6 million iterations. The highly jagged nature of graphs is normal when it comes to algorithms like Actor-Critic and DQN systems. This is because here, the algorithm also explores the molecular space as time goes on despite the total reward increasing over time.

The roll-out feature is itself one of the strategies in RL systems [69] to sample the environment where an action is taken based on an exploration/exploitation channel and stored in a tensor. There are other methods used for instance the Monte Carlo tree search method used in [2].

For a total of 6 million runs, the molecule with the highest reward is also showcased in figure. Interestingly, the highest reward is only in decimals. This may considered a limitation of how the state is represented.

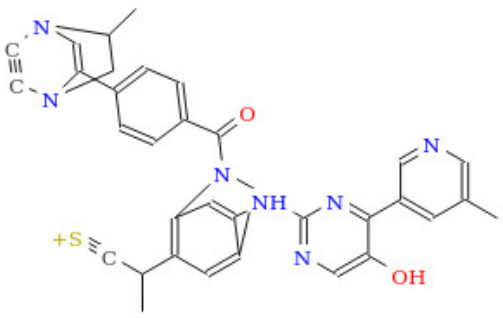

Figure 4.2: One of the initial molecules produced by the optimizer. The strange bonding can be noticed in the upper right. Through each cycle the RL agent learns to build molecules that law the laws of chemistry. The nitrogenin the center of the ring is making bonds with resonance carbon ring while a "shadow" aliphatic ring sticks out of the carbon and nitrogen on the upper right.

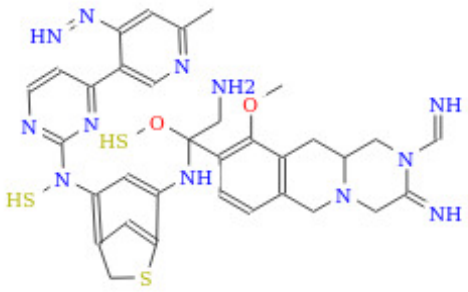

Figure 4.3: Best molecular as predicted by the RL optimizer algorithm based on Gleevec.

4.2 Understanding functional gorups within a molecule

Perhaps the more blackbox-like part of the tool is building the molecules by fragments and understanding the important moieties. While we understand that molecular functional groups are a function of the scaffold [70]. But the tool here reaches the target molecular structure as the tool converges. Convergence is logged in terms of the loss and reward. The reward function has already been elaborated at the start of this section while the loss is the TD error explained in 3. These two metrics are the same as optimizing molecules because their underlying evaluation is the same.

4.3 Evaluating the q-value versus returns

What can be noted from the figure below is that with no molecule given for optimization, the MOLPPO system can randomly generate molecules that have a direct relationship between its q-values and returns. This is not the case when the system gets molecules to optimize on. We have seen so far that modeling aromatic rings is a challenge for both DQN and MOLPPO. What can be concluded from this set of experiments is that how directly related are the q-values with returns. If the q-values align with the returns, it means that the RL system "gets" the environment and can accurately generate actions to get higher reward. This cannot be seen in the case for complex molecular structures like pencillin G [71] and gleevec [18] which both have aromatic sets of rings.

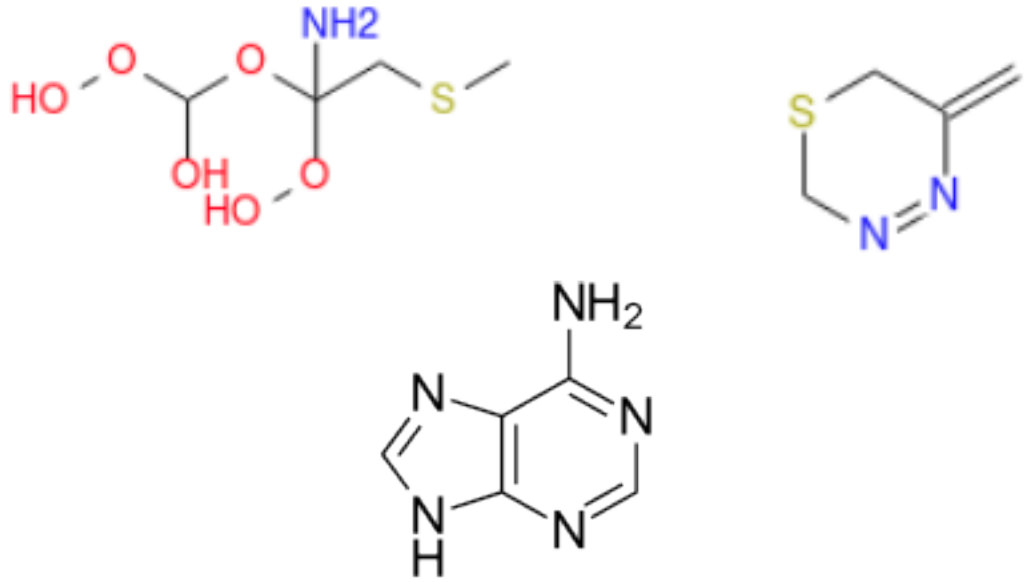

Figure 4.4: Functional groups indicated by the RL system to be important. While building scaffolds of the Gleevec molecule, enlisted are some of the functional groups deemed highest by the reward function.The two on the top had the highest values. But as can be noted both are structurally unstable entities. Oxygen having an alcohol group is not prevalent in nature. Sulfur in a ring is a ring with ntirogens is also rare. More importantly, the adenine ring was also foud to be important. Of note is the fact that functional groups are a function of the 3-dimensional structure of the proteins these molecules bind and not necessarily an innate property of the molecules themselves. Nevertheless, a pattern can be drawn for scaffolds on the kinds of functional groups that increase their QED scores.

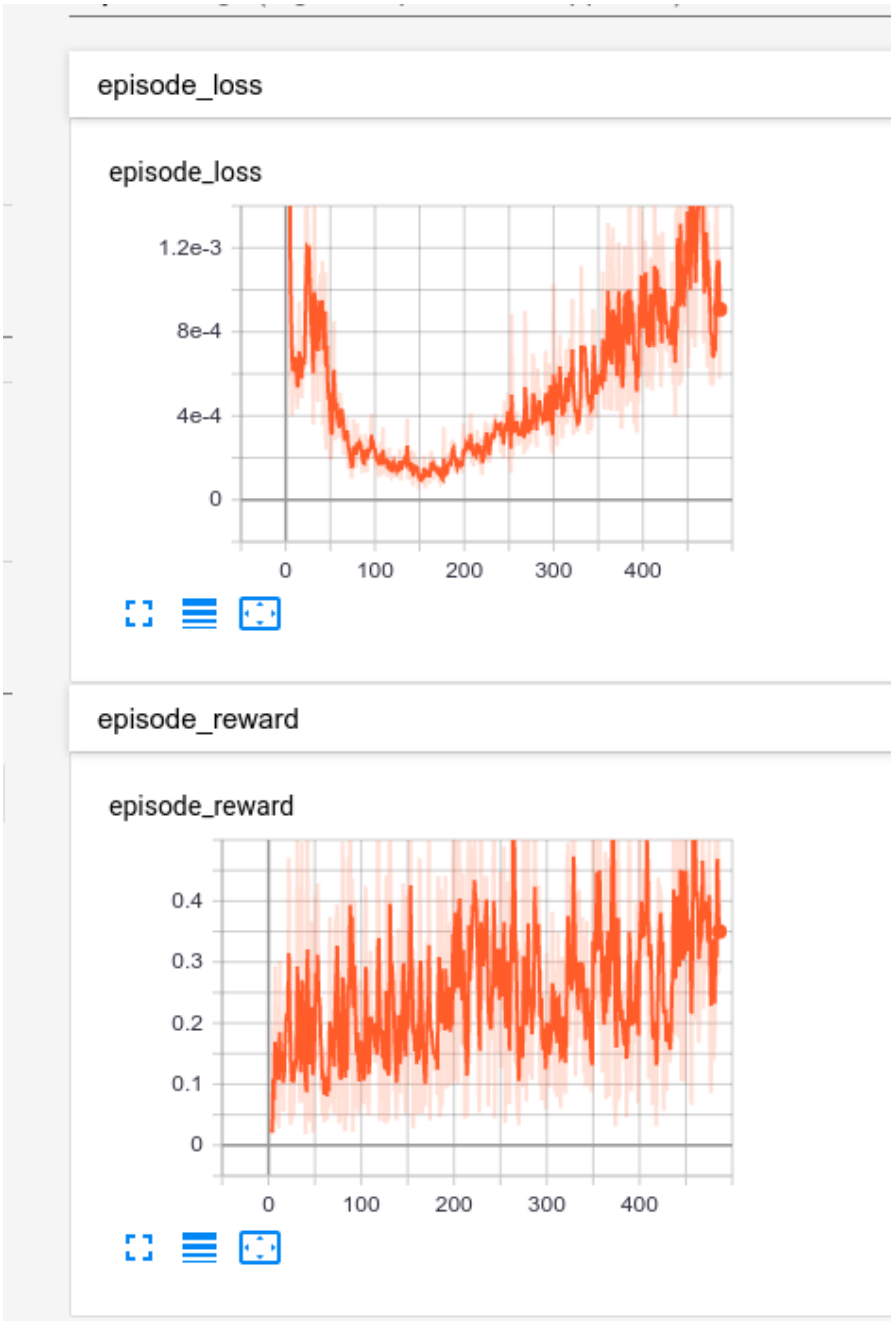

Figure 4.5: Reward and loss graphs for constructing the Gleevec molecule. Taking Gleevec as target, the RL system generates molecules that have similarity to the Gleevec molecule to understand the different moieties imporant to the structure. What can be seen is that despite the haphazard shape of the rewawrd graph, it is still going up with intense exploration. On the other hand, loss is also increasing meaning that the algorithm is far from modeling the environment accurately. Run over 400,000 cycles, the algorithm needs to run more to reduce its loss/TD error.

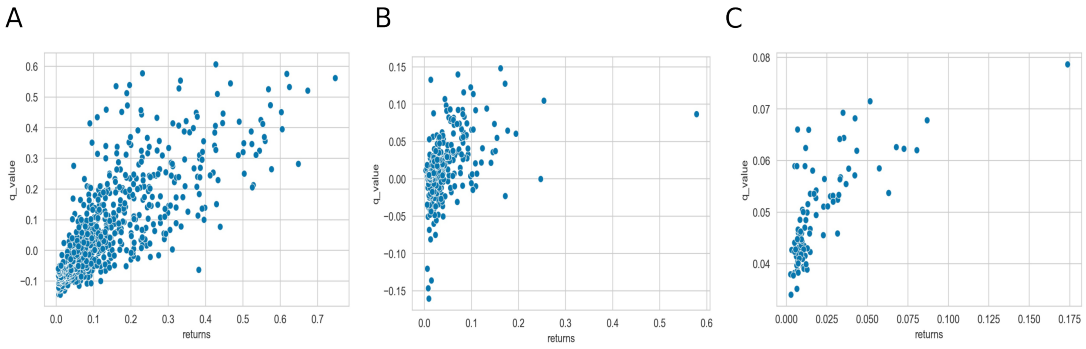

Figure 4.6: Q-value versus reward for molecules. A. No starting molecule is given as input. The q-function (y-axis) is fairly correlated with the returns (x-axis). **B.** Q-function versus returns for the Penicillin G molecule. Even for higher q-values, the returns remain low. This indicates that the system still has to model the chemical space. **C.** Q-values versus returns for the Gleevec molecule, the exploration space becomes sparser with only a handful with higher value and return relationship. This graph can be thought of as the mode's understanding of the chemical space and the return it gets from trying new actions. With more complex molecules, the value-return relationship should also cluster around some mean value.

Chapter 5

Discussion

This work can be considered a novel experimental approach rather than improving a tradition of drug design. Reinforcement learning has had major successes in the last decade [2, 3, 13]. This gives us hope that RL can also be used in other optimization tasks, in this case, bridging the gap between functional and chemical space. Other efforts will follow.

Looking at the results from the previous section, a few things can be noted.

- Aliphatic chains can easily be modeled with the help of Morgan fingerprint. These aliphatic chains can also contain functional groups including carboxyl, amide, and alcohol groups.
- Policy gradient was able to optimize on structurally different scaffold than the starting molecule as compared with the DQN approach. This was because of the probabilistic nature of proximal policy as its action space was wider and could explore a wider functional space.
- Contrary to aliphatic chains, modeling aromatic rings is challenging with Morgan fingerprint. There can be multiple reasons including absence of pi-bond featurization. This pi-bond phenomenon can impart more stability to aromatic rings and make them less likely to form bonds with functional gorups outside the ring. What was interesting to notice was that with Morgan fingerprints, the aromatic rings always generated bonds with reactive atoms like nitrogen and oxygen.
- Hence, aromatic rings need to have a richer representation. Molecular properties are not linear in combination but combinatorial in nature. Aromatic rings attain

a highly stable system due to the linear nature of the ring with pi-bonds sharing their electrons over the whole molecule. It can be deemed of as a micro-level emergent property. Morgan fingerprint was not about to represent this accurately to the reinforcement learning system.

- Finally, three-dimensional featurization of molecules is key here in order for moelcular optimization to successfully be modeled to the functional space.

Functional space is dependent upon the proteins molecules bind to. Designing molecules that bind to its target must keep the target in mind. Knowing the local distribution of the functional space is important, featurizing these properties is tricky. Reinforcement learning itself is computationally expensive. So featurizing the 3-dimensional structure of a molecule, not just the Morgan fingerprint [66], with an RL system would require considerable hardware capabilities. Keeping these two points in mind, RL systems heavily depend on state representation. This work only used Morgan fingerprint [66] which is simply a binary array of features.

However, even though the state representation is simplistic, the RL system could still come to some conclusion about the molecular space. Importantly, the architecture itself has performed well, but what seems to be the challenge is describing molecules in a neuron-friendly manner. Previous work has used graph convolutions [47] but those tend to be very specific to a small molecular space. A possible future direction would be to use a richer 3-dimensional action space with more emphasis on keeping the aromatic rings intact while generating actions.

Chapter 6

Conclusion

This thesis was an attempt to bring reinforcement learning and drug design together. Although there is sizable literature in the field, this work was still a novel attempt at optimizing molecules through an algorithm that has not yet been attempted before.

In summary, there has been many attempts at molecular optimization through direct and mixed reinforcement learning approaches. Direct methods include MOLDQN [32], drug discovery through transcriptomic data [50] while hybrid methods include graph convolution based policy gradient method GCPN [47] and blood brain barrier drug optimization [45]. However, this thesis was motivated by MOLDQN [32] which used a direct RL mode. MOLDQN features included an off-policy discrete action space with two Q value networks, the behavior and the target. Whereas MOLPPO, the current work, uses a probabilistic on-policy approach with two separate networks for the policy and q function. The idea behind separating these two is that learning tends to be faster and more generalizable [54]. The policy network is responsible for producing molecules while the critic network works on evaluating those newly generated molecules. Results show that the Actor-Critic method just like MOLDQN successfully learns to generate aliphatic chains, learning to model the aromatic rings as a chemical entity is a challenge. A possible reason is that the two methods use a binary array for each feature while resonance is a multi-atom property that cannot be mapped on to a binary array of properties like Morgan fingerprint [66]. This can be part of a future work where richer representation of the molecular space could be postulated and pushed to the Actor-Critic networks. Comparatively, the MOLPPO method provided an easier method of optimizing on a given molecule with a better representation of the chemical space.

PPO was designed for discrete action spaces and can be generalized for different optimization tasks. One direction PPO can be implemented is antibody optimization. Antibodies are bio-markers in the immune system that target cells bodily and foreign cells for identification [72]. Currently there is a need for antibody design [72] and has been deemed the "basis for engineering therapeutics". PPO can be used to modularize antibody building and optimization: addition of antibody components can be postulated as actions in the antibody space with affinity score as a reward signal. Although this formulation will be more computationally intensive, but a pipeline can stimulate some interest in the field.

To conclude, this thesis formulated molecular optimization in the form of a proximal policy optimization method. Morgan fingerprint could capture straight chain properties but it could not model conjugated systems well. Hence, a richer representation of molecules is needed, either through graph convolutions or tree-based methods. This thesis established a baseline for future improvements. Additionally, one domain that PPO can be extended is antibody design. The reward signal could be formulated as the binding affinity score while actions could be the additoin of molecular components.

References

[1] Jaime Rodríguez-Guerra Pedregal and Jean-Didier Maréchal. Pychimera: use ucsf chimera modules in any python 2.7 project. *Bioinformatics*, 34(10):1784–1785, 2018.

[2] Juan Martınez-Miranda and Arantza Aldea. Emotions in human and artificial intelligence. *Computers in Human Behavior*, 21(2):323–341, 2005.

[3] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676): 354–359, 2017.

[4] Yen-Lin Lin, Yilin Meng, Wei Jiang, and Benoît Roux. Explaining why gleevec is a specific and potent inhibitor of abl kinase. *Proceedings of the National Academy of Sciences*, 110(5):1664–1669, 2013.

[5] Tiago Pereira, Maryam Abbasi, Jose Luis Oliveira, Bernardete Ribeiro, and Joel Arrais. Optimizing blood–brain barrier permeation through deep reinforcement learning for de novo drug design. *Bioinformatics*, 37(Supplement_1):i84–i92, 2021.

[6] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pages 1407–1416. PMLR, 2018.

[7] Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvari, Satinder Singh, et al. Behaviour suite for reinforcement learning. *arXiv preprint arXiv:1908.03568*, 2019.

[8] Dmytro Mishkin, Nikolay Sergievskiy, and Jiri Matas. Systematic evaluation of convolution neural network advances on the imagenet. *Computer Vision and Image Understanding*, 161:11–19, 2017.

[9] Christopher D Manning. Computational linguistics and deep learning. *Computational Linguistics*, 41(4):701–707, 2015.

[10] Atsuho Nakayama and Daniel Baier. Predicting brand confusion in imagery markets based on deep learning of visual advertisement content. *Advances in Data Analysis and Classification*, 14(4):927–945, 2020.

[11] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. pages 173–182, 2016.

[12] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

[13] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[14] Jessica Vamathevan, Dominic Clark, Paul Czodrowski, Ian Dunham, Edgardo Ferran, George Lee, Bin Li, Anant Madabhushi, Parantu Shah, Michaela Spitzer, et al. Applications of machine learning in drug discovery and development. *Nature Reviews Drug Discovery*, 18(6):463–477, 2019.

[15] Natalie Stephenson, Emily Shane, Jessica Chase, Jason Rowland, David Ries, Nicola Justice, Jie Zhang, Leong Chan, and Renzhi Cao. Survey of machine learning techniques in drug discovery. *Current drug metabolism*, 20(3):185–193, 2019.

[16] David S Wishart, Craig Knox, An Chi Guo, Dean Cheng, Savita Shrivastava, Dan Tzur, Bijaya Gautam, and Murtaza Hassanali. Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic acids research*, 36(suppl_1):D901–D906, 2008.

[17] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem 2019 update: improved access to chemical data. *Nucleic acids research*, 47(D1): D1102–D1109, 2019.

[18] Maria Debiec-Rychter, Bartosz Wasag, Michel Stul, Ivo De Wever, Allan Van Oosterom, Anne Hagemeijer, and Raf Sciot. Gastrointestinal stromal tumours (gists) negative for kit (cd117 antigen) immunoreactivity. *The Journal of Pathology: A Journal of the Pathological Society of Great Britain and Ireland*, 202(4):430–438, 2004.

[19] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[20] Ronald R Hocking. Developments in linear regression methodology: 1959–l982. *Technometrics*, 25(3):219–230, 1983.

[21] Stephan Dreiseitl and Lucila Ohno-Machado. Logistic regression and artificial neural network classification models: a methodology review. *Journal of biomedical informatics*, 35(5-6):352–359, 2002.

[22] Hans-Hermann Bock. Clustering methods: a history of k-means algorithms. *Selected contributions in data analysis and classification*, pages 161–172, 2007.

[23] Ka Yee Yeung and Walter L. Ruzzo. Principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774, 2001.

[24] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[25] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

[26] Paul W Glimcher. Understanding dopamine and reinforcement learning: the dopamine reward prediction error hypothesis. *Proceedings of the National Academy of Sciences*, 108(Supplement 3):15647–15654, 2011.

[27] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[28] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[29] Maozu Guo, Yang Liu, and Jacek Malec. A new q-learning algorithm based on the metropolis criterion. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(5):2140–2143, 2004.

[30] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

[31] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[32] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):1–10, 2019.

[33] G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4 (2):90–98, 2012.

[34] Dávid Bajusz, Anita Rácz, and Károly Héberger. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of cheminformatics*, 7(1):1–13, 2015.

[35] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[36] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[37] Stephen Dankwa and Wenfeng Zheng. Twin-delayed ddpg: A deep reinforcement learning technique to model a continuous movement of an intelligent robot agent.

In *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, pages 1–5, 2019.

[38] Shihui Li, Yi Wu, Xinyue Cui, Honghua Dong, Fei Fang, and Stuart Russell. Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4213–4220, 2019.

[39] Paul Medvedev. Modeling biological problems in computer science: a case study in genome assembly. *Briefings in bioinformatics*, 20(4):1376–1383, 2019.

[40] Jorge Ivan Castillo-Quan, Kerri J Kinghorn, and Ivana Bjedov. Genetics and pharmacology of longevity: the road to therapeutics for healthy aging. *Advances in Genetics*, 90:1–101, 2015.

[41] Craig A Umscheid, David J Margolis, and Craig E Grossman. Key concepts of clinical trials: a narrative review. *Postgraduate medicine*, 123(5):194–204, 2011.

[42] Michele Boldrin, David K Levine, et al. Against intellectual monopoly. 2008.

[43] Nataraj S Pagadala, Khajamohiddin Syed, and Jack Tuszynski. Software for molecular docking: a review. *Biophysical reviews*, 9(2):91–102, 2017.

[44] Jodi B Segal, John J Strouse, Mary Catherine Beach, Carlton Haywood, Catherine Witkop, Haeseong Park, Renee F Wilson, Eric B Bass, and Sophie Lanzkron. Hydroxyurea for the treatment of sickle cell disease. *Database of Abstracts of Reviews of Effects (DARE): Quality-assessed Reviews [Internet]*, 2008.

[45] Alison Steinbach, Stephen M Clark, and Amber B Clemmons. Bosutinib: a novel src/abl kinase inhibitor for chronic myelogenous leukemia. *Journal of the advanced practitioner in oncology*, 4(6):451, 2013.

[46] Thomas Kindler, Frank Breitenbuecher, Andreas Marx, Joachim Beck, Georg Hess, Birgit Weinkauf, Justus Duyster, Christian Peschel, Charles J Kirkpatrick, Matthias Theobald, et al. Efficacy and safety of imatinib in adult patients with c-kit–positive acute myeloid leukemia. *Blood*, 103(10):3644–3654, 2004.

[47] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31, 2018.

[48] Michele Boldrin, David K Levine, et al. *Against intellectual monopoly*, volume 9. Cambridge University Press Cambridge, 2008.

[49] Leonardo LG Ferreira and Adriano D Andricopulo. Admet modeling approaches in drug discovery. *Drug discovery today*, 24(5):1157–1165, 2019.

[50] Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*, 2016.

[51] Jiman Kim and Chanjong Park. End-to-end ego lane estimation based on sequential transfer learning for self-driving cars. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 30–38, 2017.

[52] Sen Wang, Daoyuan Jia, and Xinshuo Weng. Deep reinforcement learning for autonomous driving. *arXiv preprint arXiv:1811.11329*, 2018.

[53] Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Conference on robot learning*, pages 482–495. PMLR, 2017.

[54] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[55] Vincent J Venditto and Francis C Szoka Jr. Cancer nanomedicines: so many papers and so few drugs! *Advanced drug delivery reviews*, 65(1):80–88, 2013.

[56] John Hudson. Generic take-up in the pharmaceutical market following patent expiry: a multi-country study. *International Review of Law and Economics*, 20(2): 205–221, 2000.

[57] Abdul J Jerri. The shannon sampling theorem—its various extensions and applications: A tutorial review. *Proceedings of the IEEE*, 65(11):1565–1596, 1977.

[58] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):1–14, 2017.

[59] Christos A Nicolaou, Nathan Brown, and Constantinos S Pattichis. Molecular optimization using computational multi-objective methods. *Current Opinion in Drug Discovery and Development*, 10(3):316, 2007.

[60] Yotam Hechtlinger, Purvasha Chakravarti, and Jining Qin. A generalization of convolutional neural networks to graph-structured data. *arXiv preprint arXiv:1704.08165*, 2017.

[61] Julia V Georgieva, Dick Hoekstra, and Inge S Zuhorn. Smuggling drugs into the brain: An overview of ligands targeting transcytosis for drug delivery across the blood–brain barrier. *Pharmaceutics*, 6(4):557–583, 2014.

[62] Laurent Sakka, Guillaume Coll, and Jean Chazal. Anatomy and physiology of cerebrospinal fluid. *European annals of otorhinolaryngology, head and neck diseases*, 128(6):309–316, 2011.

[63] Anil Kumar, Arti Singh, et al. A review on alzheimer's disease pathophysiology and its management: an update. *Pharmacological reports*, 67(2):195–203, 2015.

[64] Tiago Pereira, Maryam Abbasi, Jose Luis Oliveira, Bernardete Ribeiro, and Joel Arrais. Optimizing blood–brain barrier permeation through deep reinforcement learning for de novo drug design. *Bioinformatics*, 37(Supplement_1):i84–i92, 2021.

[65] Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1(1):1–11, 2009.

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

[67] Shangtong Zhang and Richard S Sutton. A deeper look at experience replay. *arXiv preprint arXiv:1712.01275*, 2017.

[68] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*, 2018.

[69] Dimitri Bertsekas. Multiagent reinforcement learning: Rollout and policy iteration. *IEEE/CAA Journal of Automatica Sinica*, 8(2):249–272, 2021.

[70] Dilyana Dimova, Dagmar Stumpfe, and Jürgen Bajorath. Computational design of new molecular scaffolds for medicinal chemistry, part ii: generalization of analog series-based scaffolds. *Future Science OA*, 4(2):FSO267, 2017.

[71] Qingfen Liu, Jiang Yu, Wangliang Li, Xuesheng Hu, Hansong Xia, Huizhou Liu, and Ping Yang. Partitioning behavior of penicillin g in aqueous two phase system formed by ionic liquids and phosphate. *Separation science and technology*, 41(12): 2849–2858, 2006.

[72] Mark L Chiu, Dennis R Goulet, Alexey Teplyakov, and Gary L Gilliland. Antibody structure and function: the basis for engineering therapeutics. *Antibodies*, 8(4):55, 2019.