Replicating and Revising Current Literature on Reinforcement Learning for Strategic Chemotherapy Dosages

CS 221 Final Project Report

Kristina Beck, Daniel Henry, Jimmy Le
kbeck2, dhenry, jimmyle

Abstract

In our project, we built a reinforcement learning (RL) agent to administer chemotherapy to a simulated cancer patient. We modeled the agent's environment with a set of Ordinary Differential Equations, (ODEs), a common practice for training RL agents in healthcare environments. This project served to replicate and reinforce comparable studies in Reinforcement Learning for Healthcare by showing the efficacy of RL methods on this problem. We found that we were able to create agents that effectively learning their respective environments, but also highlighted the need for medically accurate and robust models. Our codebase can be found at `https://github.com/danielhenry1/rl_chemo_simulation`

1 Introduction

1.1 Problem Statement

Cancer refers to a group of diseases involving uncontrolled growth of abnormal tumors in the human body. In 2018, nearly 2 million new cases of cancer were diagnosed in the United States. Fortunately, as of 2016, cancer mortality has fallen nearly 27% from its peak in 1991 as a result of new preventative measures and advances in early detection and treatment.[8]

Yet, treatment scheduling is not perfect. Chemotherapy is a common treatment of cancer involving the administration of drug cocktails that kill tumor cells. Chemotherapy is not a perfect science, and optimal dosages vary tremendously according to the stage of tumor, patient wellness, white blood cell levels, external illnesses, age, and more. With all of these factors, oncologists currently base their treatment off of prior medical knowledge. However, current literature explores using Artificial Intelligence in the hope of assisting the decision making process in chemotherapy scheduling.

In this project, we explored, replicated, and revised two models to generate our own environments, and we trained our own agents to deliver optimal chemotherapy dosages. We showed that a reinforcement learning agent can outperform a naive baseline agent that delivers constant dosages, though we certainly haven't attained the oracle level of a human doctor and their broad medical knowledge. Yet, we believe that the work does further an optimistic outlook that with research and improved models, RL agents could assist doctors in their treatments.

1.2 Related Work/Literature Review

Our models were inspired primarily by two papers. In *Reinforcement learning design for cancer clinical trials*, Zhao et al implement a simple model based on a patient's wellness and tumor size which we call the Tumor Size (TS) model. In *Reinforcement learning-based control of drug dosing for cancer chemotherapy treatment*, Padmanabhan et al implement a more complex model based on the amount of normal cells, tumor cells, and immune cells which we call the Cell Count (CC) model.

Our implementation of both models used the ODEs outlined in the respective paper to model the transitions over time, but differed in a few key components. For one, each of our simulations began with a randomized patient state drawn from a Gaussian distribution. We believed that a Gaussian distribution would be the most realistic for natural phenomenon like medical conditions. The simulation in Zhao et al used a uniform distribution for their start states, and the simulation in Padmanabhan et al used constant start states to train the agent to treat a certain type of patient, such as a "young pregnant woman." In giving our agent a random start state, we simulated the general case where a doctor must be able to provide optimal treatment for any patient that comes into a hospital.

Our reward functions also differed from those in the literature. Padmanabhan et al implemented the

CC model and gave reward to their agent based solely on the proportional drop in tumor cells. For our CC model, we included this component of reward, but also included a hazard function that describes the probability of patient death at a time step, and thus the end of the simulation. Additionally, our reward function gave feedback to the agent if the patient died or was cured.

2 Approach and Infrastructure

We built two models: The Tumor Size Model, and the Cell Count Model.

2.1 Tumor Size Model: Markov Decision Process

We have implemented a simple and straightforward model for our initial implementation.

State Definition: The state models the health and wellness of a patient at a specific time step. The state of a patient is a 3-tuple (W, M, t), where 'W' is a float, representing the patients' negative wellness outcomes in response to chemotherapy, 'M' is a float, representing the size of the patient's tumor size, and 't' is an integer representing the months of treatment that the patient has received. At this point in our model, we assume treatment is given monthly.

Action Space: At each state, our agent selects an action corresponding to a dosage of chemotherapy drug. The agent can administer a dose d where $0 \leq d \leq 1$. For this model, we have discretized the action space into k equivalently spaced dosages. As k increases, so to does the action space and the precision at which the agent can administer a dose.

State Succession: The succession between states is defined as follows. Given a state $S_t = (W_t, M_t, t)$ and an action (dose) d

$$S_{t+1} = (W_t + \dot{W}_t, M_t + \dot{M}_t, t+1)$$

$$\dot{W}_t = \alpha M_t + \beta(d - \lambda)$$

$$\dot{M}_t = \chi W_t - \gamma(d - \lambda)$$

For our final implementation, $\alpha = 0.1, \beta = 1.2, \chi = 0.15, \gamma = 1.2, \lambda = 0.5$, which we have tuned to improve realism in our model. To gain some intuition around these hyperparameters, we can see that, for example, the value of .5 for λ is the middle of the dosage range, so half of the action space will have a negative effect of the other half of the action space. γ and β are equal, showing that the tradeoff of a dose's impact on wellness and tumor size are equal. Finally, χ and α are both relatively low, denoting tumors that are less aggressive. We can foresee future simulations where these parameters adjust from patient to patient or even over the course of a simulation.

Note that with this model, a low W value is "healthier." Intuitively, the patiently becomes less well (high value of W) with a larger tumor and a high dosage, but high dosages lead to reductions in tumor size, given the patient is well enough. We can thus see that the agent will need to find a balanced regimen to keep the tumor size down while ensuring the patient's wellness doesn't become too damaged.

While transitioning between any two states, there is a non-zero probability that the patient will die. This probability is calculated via a "hazard function" defined as

$$h = 1 - (e^{-(W+M)} + \psi)$$

where $\psi = .15$. We then assign a random variable V that has a Bernoulli Distribution with probability h.

In the case that $V = 1$, which occurs with probability h, the result of the treatment is death and the simulation ends. Alternatively, if the tumor size M reaches 0 or goes negative, we denoted the patient as cured and ended the simulation. Additionally, if the patient reached a state where $t = 6$, we considered the treatment period over and ended the simulation.

2.2 Tumor Size Model: Algorithms and Reward

Algorithm: At this stage, we have implemented a Q-learning model with function approximation and an epsilon greedy action policy. The algorithm for updating the feature weights is as follows

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\big(\hat{Q}_{opt}(s, a; \mathbf{w}) - (r + \gamma \hat{V}_{opt}(s'))\big)\phi(s, a)$$

Our feature extractor $\phi(s, a)$ returns three features, one for the patient's wellness, one for the patient's tumor size, and one with both combined. Each of these values is discretized into one of k buckets, to allow the algorithm to generalize in its learning.

Reward Function: The reward functions calculated after each transition are as follows.

$$R(s,a) = \begin{cases} 5 & \text{M}_t \leq 0 \\ -5 & \text{V} = 1 \\ R_W(s,a) + R_M(s,a) & t < 6 \\ 2M_0 & t = 6 \end{cases}$$

$$R_W(s,a) = \begin{cases} -\dot{W}_t & \dot{W}_t < -0.5 \\ \dot{W}_t & \dot{W}_t > 0.5 \\ 0 & otherwise \end{cases}$$

$$R_M(s,a) = \begin{cases} -\dot{M}_t & \dot{M}_t < -0.5 \\ \dot{M}_t & \dot{M}_t > 0.5 \\ 0 & otherwise \end{cases}$$

where M_0 is the initial tumor size before the treatment starts.

Intuitively, we see that this reward scheme incentivizes keeping the patient alive and curing them. Additionally, with each dosage, the agent is rewarded for improvements in health and reductions in tumor size. The agent is also rewarded if the patient survives to the end of the treatment cycle, though the reward is less than a complete cure. Our rewards between curing, death, and finishing the procedure are currently inspired by literature, particularly Zhao et al, and an assumption of how certain results are valued. Future developments of this reward function demand better connections between RL, the medical field, and further understanding of what doctors and patients prioritize in a treatment.

In order for our algorithm to gather data and learn, we also include an exploration policy that is epsilon greedy and will choose an action uniformly at random with the exploration probability ϵ and choose the optimal action the rest of the time.

We also ran a number of experiments to show how differing methods for adjusting ϵ through the training process may impact hte rate at which the model converges to the optimal policy. See results section.

2.3 Cell Count Model

State Definition: We modeled the normal, tumor, and immune cell count and drug concentration over time. The state is a 5-tuple (N, T, I, C, t). 'N' is a float representing the patients' number of normal cells. 'T' is a float representing the patients' number of tumor cells. 'I' is a float representing the patients' number of immune cells. 'C' is a float representing the drug concentration in the body. Lastly, 't' is an integer representing the number of times treatment has been given to the patient.

Action Space: In the same way as the TS model, at each state, our agent selects an action corresponding to a dosage of chemotherapy drug. The agent can administer a dose d where $d \in (0, 0.01, 0.02, 0.03, 0.04, 0.06, 0.08, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1)$.

State Succession: Given a state $S_t = (N_t, T_t, I_t, C_t, t)$ and an action (dose) d

$$S_{t+1} = (\dot{N}_t, \dot{T}_t, \dot{I}_t, \dot{C}_t, t+1)$$

$$\dot{N}_t = r_2 N_t(1 - b_2 N_t) - c_4 N_t T_t - a_3 N_t C_t$$

$$\dot{T}_t = r_1 T_t(1 - b_1 T_t) - c_2 I_t T_t - c_3 T_t N_t - a_2 T_t C_t$$

$$\dot{I}_t = s + \frac{\rho I_t T_t}{\alpha + T_t} - c_1 I_t T_t - d_1 I_t - a_1 I_t C_t$$

$$\dot{C}_t = d - d_2 C_t$$

Each hyperparameter here represents relationships between cells, between cells and the drug dosage, and between cells and the natural body. The ODE system and ranges of the hyperparameters were drawn from Padmanabhan et al.

a_1, a_2, a_3 represents the fraction kill rate of immune cells, tumor cells, and normal cells, respectively. The ranges are $0 < a_i < 0.5, a_3 \leq a_1 \leq a_2$. In our final model we used $a_1 = 0.32, a_2 = 0.38, a_3 = 0.10$

b_1, b_2 represents the reciprocal carrying capacity of tumor and normal cells, respectively. The ranges are $b_1^{-1} \leq b_2^{-1} = 1$ In our final model we used $b_1 = 0.2, b_2 = 0.2$.

c_1 represents immune cells' competition with tumor cells (intuitively, this is how the two cells interact with each other and the number of immune cells that dies as a result). The ranges for all competition hyperparameters are $0.3 \leq c_i \leq 1$

c_2 represents tumor cells' competition with immune cells.

c_3 represents tumor cells' competition with normal cells.

c_4 represents normal cells' competition with tumor cells.

In our final model, we used $c_1 = 0.5, c_2 = 0.4, c_3 = 0.585, c_4 = 0.6$

d_1 represents the immune cells' natural death rate. The range is $0.15 \leq d_1 \leq 0.3$

d_2 represents the decay rate of the injected drug. The range is $0 \leq d_2 \leq 1$. We used $d_1 = 0.25, d_2 = 1$

r_1 and r_2 represents the per unit growth rate of tumor and normal cells, respectively. The range is $1.2 \leq r_1 \leq 1.6$ and $r_2 = 1$ We used $r_1 = 1.4$

s represents the natural immune cell influx rate. The range is $0.3 \leq s \leq 0.5$. We usued 0.37

α represents the immune cell threshold rate (intuitively, this is the rate at which the number of immune cells can stick around after each state). The range is $0.3 \leq \alpha \leq 0.5$ We used 0.35.

ρ represents the immune cell response rate (which intuitively represents how many of the immune cells will actually stick around). The range is $0.01 \leq \rho \leq 0.05$. We used 0.01.

Similar to the TS model, we included a non-zero probability that the patient will die between each time step. This probability, calculated by the "hazard function", in this model is:

$$h = 1 - \frac{I_t}{T_t}$$

We then assign a random variable V that has a Bernoulli Distribution with probability h.

In the case that $V = 1$, which occurs with probability h, the result of the treatment is death and the simulation ends. Alternatively, if the number of tumor cells reaches 0 or goes negative, we mark the patient as cured and the simulation ends. If the patient reaches a state where $t = 7$, we consider the treatment period over and end the simulation.

2.4 Cell Count Model: Algorithms and Reward

Algorithm: We implented a Q-leaning model with function approximation and an epsilon greedy action policy, like the previous model. We update the feature weights as before:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\big(\hat{Q}_{opt}(s,a;\mathbf{w}) - (r + \gamma \hat{V}_{opt}(s'))\big)\phi(s,a)$$

Our feature extractor $\phi(s,a)$ returns four features, one for the patient's normal cell count, one for the patient's tumor cell count, one for the patient's immune cell count, and one with all combined. Each of the values are discretized with k.

Reward Function: The reward functions calculated after each transition are as follows.

$$R(s,a) = \begin{cases} 5 & \mathrm{T}_t \leq 0 \\ -5 & \mathrm{V} = 1 \\ \frac{(T_t - (T_t + \dot{T}_t))}{T_t} & t < 6 \\ T_0 & t = 6 \end{cases}$$

In order for our algorithm to gather data and learn, we also include an exploration policy that is epsilon greedy and will choose an action uniformly at random with the exploration probability ϵ and choose the optimal action the rest of the time.

3 Results and Error Analysis

3.1 Tumor Size Model: Results and Experimentation

Our initial results are encouraging with the Tumor Size Model. The agent learns how to cure the patient over many trials, although with the simplicity of the model, often times the agent finds itself solely curing the tumor and neglecting the wellness of the patient.

Our agent surpasses the baseline. In our model, a non-trivial *baseline* can be viewed as a chemotherapy schedule that gives out constant dosages over the time span. In *Figure 1*, we find that our agent is able to perform better than constant dosages. Notably, the constant dosage of 1 cures a large number of patients because a strong dosage would indeed eliminate a tumor. This is a flaw in the mathematical models because strong amounts of chemotherapy leads to adverse effects on the patient's wellness. While we would expect a dosage of 1 to increase the wellness term significantly and thus the probability of death in our hazard function, in practice, this dosage also decreases the tumor size, leading to insignificant changes in the probability of death. Thus, improvements in both the hazard function and the relative impact of a dose on tumor size and wellness need to be implemented. Further knowledge of how significantly tumor size and chemotherapy affect the body's ability to function is necessary to improve the model more accurately.

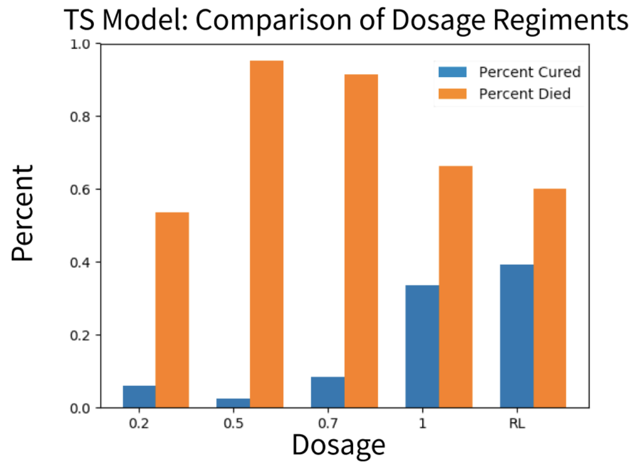

Figure 1: A graph comparing percent of patients cured and died depending on being given constant dosages and using our agent.

The agent has learned actions. Because our TS agent learns the best policy at each state depending on the current tumor size and patient wellness, we are able to visualize what dosage our agent would give patients given their tumor size and wellness. Figure 2 shows what dosage a patient should receive depending on their wellness and tumor size.

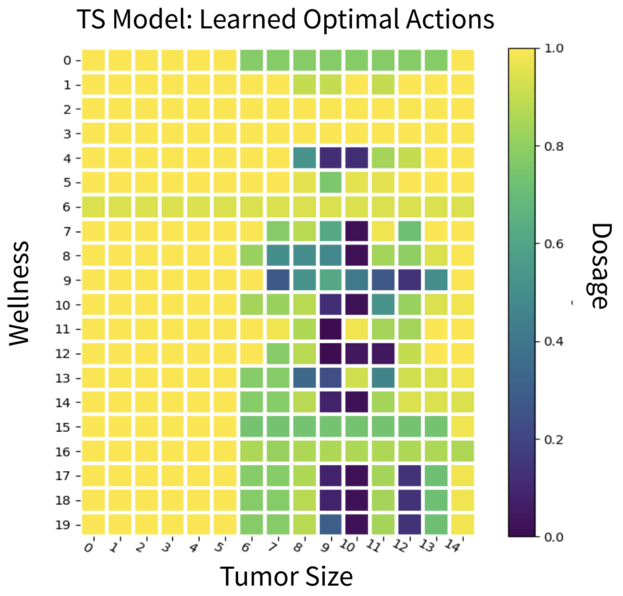

Figure 2: A chart indicating the amount of dosage depending on current tumor size and wellness.

After an analysis of this chart, the policy the agent learns makes intuitive sense. If the tumor size is fairly small (lower than 0.6), the agent will give it a large dosage no matter the wellness in hopes of getting rid of the tumor completely during that month. If the patient has a large tumor but is otherwise healthy, then the agent knows the patient can handle a high dosage and thus eliminate the tumor. The agent treads finer lines of lower dosages if the patient is unwell and the tumor is large. The agent knows that it will need multiple months to cure the patient, and a high dosage will likely lead to patient death. Medically, this makes sense because a doctor wouldn't put a patient through high pain of high dosages of chemotherapy if they know low dosages across time could be better.

One part of this chart that surfaces a point of future work is how the agent will still give full dosages even if the tumor size is small. It would make sense that a small dosage could do the trick. We believe this is because the agent values eliminating tumor size as soon as possible, and in these cases eliminating the tumor outweighs using smaller dosages. We could add another reward based on the end wellness of the patient so that the agent would want to have the minimum final dosage to get rid of a tumor.

The agent still converges with varying adjustment policies of epsilon. In the following figure, we experiment with no exploration, a constant epsilon, a linear decay in epsilon, and an exponential decay in epsilon. The constant epsilon was 0.2 for 350,000 trials and 0 after. The linear decay in epsilon decreases the value by 0.00001 after every trial until it reaches 0.001. The exponential decay in epsilon continues with 99.995% of the previous epsilon after each trial. These experiments showed that the agent was able to converge at different rates depending on the exploration factor.

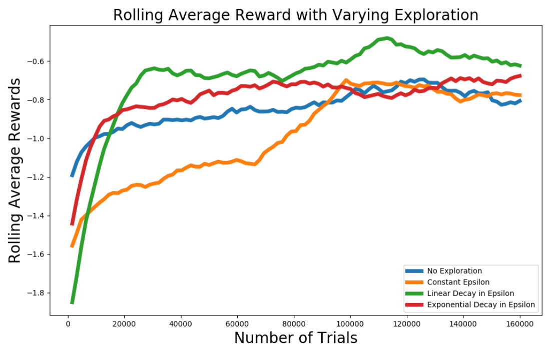

Figure 3: A graph comparing the varying modifications of epsilon.

Experimenting with discretization. We were able to train our agent in the TS model using different values of k. This discretization factor both increased the precision at which the agent could administer dosages, as well as the specificity to which it could measure the patients wellness and tumor size. We noted that higher values of k did indeed lead to better performance, as the agent was able to better understand the current status of the patient. However, this came with a tradeoff of longer learning time for higher k. See results in Figure 4.

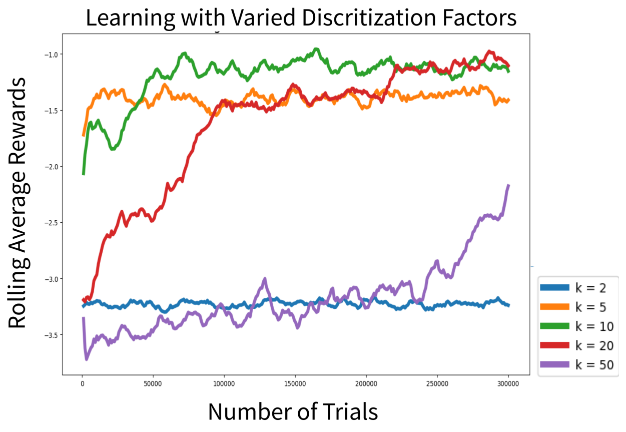

Figure 4: A graph comparing rewards over training trials for different discretization constants k.

3.2 Cell Count Model: Results and Experimentation

The tuning of the cell count model was much more difficult. As can be seen in Figure 5, it took over one million trials for the model to attain significant changes in the reward function.

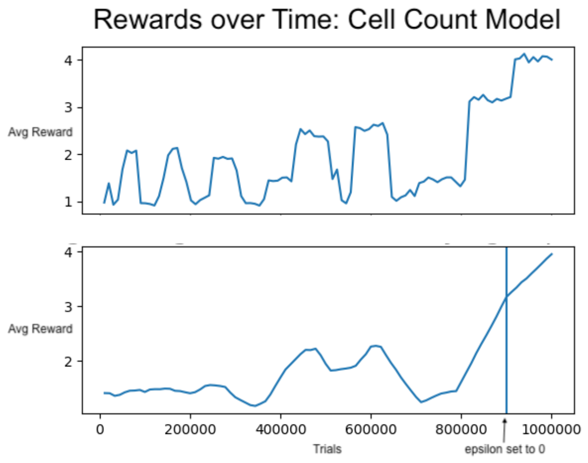

Figure 5: A graph showing the increasing reward function over training time for the CC model.

Additionally, it was not always clear that the agent was performing in the way we expected it to. In an early version of the model, we discovered that the agent was keeping patients alive as long as possible, for over 100 iterations in some cases, because it discovered that it could attain a higher reward that way than by curing patients outright.

Ultimately, the model environment was not realistic. Although our RL model attained higher rewards than the baseline models, the percentage of cures was not greater. This shows that while the Reinforcement Learning algorithm was able to optimize its reward in the simulation, the simulation did not ultimately incentivize or make possible effective treatments. We worked to fine-tune our hyperparameters and found that a small adjustment would drastically change the results, leading to only cures or only deaths. We believe that the model requires precision tuning and the hazard function needs to be improved in order to have more accurate and realistic results. However, in all cases the RL agent was able to learn an optimal policy in order to maximize its own reward. We can see in Figure 6 that the RL agent was able to outperform the naive constant agents on this CC model as well.

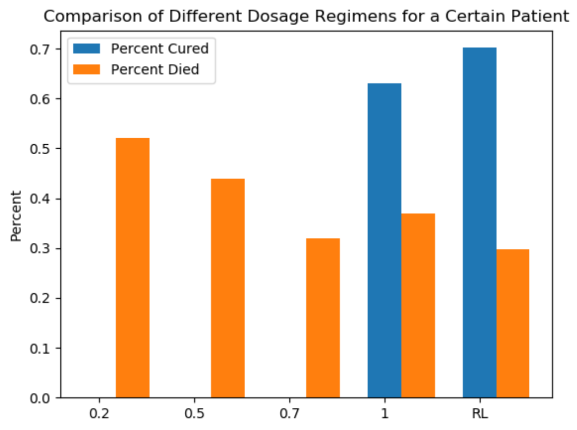

Figure 6: A graph showing the cure and death rates for the CC model.

4 Conclusion

4.1 Social Impact

As artificial intelligence and reinforcement learning improve, and as research in the medical field expands, this project posits a possible future for oncology. It's indeed possible to train an agent to pick strategic dosages to cure patients. We hope our project conveys that with sophisticated mathematical models, doctors indeed can turn to artificial intelligence as a sidekick in their work. Optimistically, this is a technology that would revolutionize the treatment process of medicine, saving time and money in our current system.

Yet, there are more nuances. The mathematical models used in our simulations are immature and don't adequately capture the complexity of cancer's progression. Additionally, the state space we utilized is quite small. As important as cell count and tumor size are, many other factors influence cancer such as medical history, geographic location, age, other diseases, and diet. There are also factors such as access to resources and cost that influence real world treatments. Cancer is a complex disease, meaning the future of this project must involve sophisticated medical research.

4.2 Future Work

For our project, future work must ensure models are grounded in science and medical research while keeping pace with improved learning models. Specifically:

- Realism is a desired objective. We would like to see better ODEs that more realistically model the human body. Additionally, we would like reward functions to consider more factors such as the cost of a treatment.
- Modeling the differences in patients. Our current models use hyperparameters as a mechanism for distinguishing the natural differences among patients, but more research can advance these hyperparameters.
- Different cancers require different types of chemotherapy. Again, our models have hyperparameters that addresses this issue, but we're hopeful research can provide more insight into the bounds of these hyperparameters.
- Improve the realism of the hazard function to incorporate more factors and provide realistic mortality statistics.

4.3 Conclusion

We were able to successfully train reinforcement learning agents to optimize rewards in the custom environments we built. Our results supported the findings of both Zhao et al and Padmanabhan et al, as the basic ODE structures they presented led to environments with intuitive, learnable, yet non-naive optimal policies. Additionally, our work highlighted the importance of robust, accurate, and coherent environments for reinforcement learning agents. While we wouldn't trust our personal health to the trained agent at this stage, we are optimistic in the belief that Reinforcement Learning and Artificial Intelligence will continue to improve medical practices with further research.

5 References

1. Zhao, Yufan, Michael R. Kosorok, and Donglin Zeng. "Reinforcement learning design for cancer clinical trials." Statistics in medicine 28, no. 26 (2009): 3294-3315.

2. https://arxiv.org/abs/1908.08796

3. Yauney, Gregory, and Pratik Shah. "Reinforcement learning with action-derived rewards for chemotherapy and clinical trial dosing regimen selection." In Machine Learning for Healthcare Conference, pp. 161-226. 2018.

4. A Tumor Growth Inhibition Model for Low-Grade Glioma Treated with Chemotherapy or Radiotherapy. Benjamin Ribba, Gentian Kaloshi, Mathieu Peyre, Damien Ricard, Vincent Calvez, Michel Tod, Branka Čajavec-Bernard, Ahmed Idbaih, Dimitri Psimaras, Linda Dainese, Johan Pallud, Stéphanie Cartalat-Carel, Jean-Yves Delattre, Jérôme Honnorat, Emmanuel Grenier and François Ducray Clin Cancer Res September 15 2012 (18) (18) 5071-5080; DOI: 10.1158/1078-0432.CCR-12-0084

5. Padmanabhan, Regina, Nader Meskin, and Wassim M. Haddad. 2017. \Reinforcement Learning-Based Control of Drug Dosing for Cancer Chemotherapy Treatment." Mathematical Biosciences 293 (November): 11{20. doi:10.1016/j.mbs.2017.08.004.

6. https://www.cancer.gov/about-cancer/understanding/s

7. https://www.cancer.org/latest-news/facts-and-figure

8. Rebecca L. Siegel, Kimberly D. Miller, Ahmedin Jemal. Cancer statistics, 2019. CA: A Cancer Journal for Clinicians, 2019; DOI: 10.3322/caac.21551