Evaluation of Value-based and Policy-based methods in Dynamic Multi Drug Therapies for HIV Treatment

Gandharv Patil Raihan Seraj

Abstract

Evaluation of Reinforcement learning(RL) based algorithms on game based environments can sometimes undermine the applicability of these methods to solve real world tasks. In this project we demonstrate the ability of Reinforcement Learning based methods to solve a challenging problem of finding the optimal drug scheduling strategy for treatment of Human Immunodeficiency Virus (HIV) infection. Further we incorporate the environment used for this problem in the popular OpenAI-Gym format with the hope that this problem provides a more practical testbed for benchmarking RL algorithms.

1. Introduction

Over the past few years, evaluation of Reinforcement Learning(RL) based algorithms has been carried out extensively on either game based [6, 5] or robotic environments [17]. However, as mentioned above RL techniques can also be used to solve a wide variety of control problems. Hence we also implement a OpenAI-Gym wrapper for the dynamical model proposed by Adams *et al*. Finally, our contributions in this project can be summarized as follows:

1. We incorporate a new a environment in the OpenAI-Gym frame work that accentuates the applicability of Reinforcement Learning in the health care domain
2. We evaluate the performance of some of the a basic Policy based and Value based methods in the proposed environment.

In the recent years, scientists have made significant progress towards improving quality of life and fostering higher longevity in HIV infected patients thanks to the advances in the treatment of human immunodeficiency virus (HIV). Patients are now able to maintain a low viral load due to the availability of a combination of drug treatments known as ”drug cocktails”.
Presently available anti-HIV drugs can be broadly classified into two categories reverse transcriptase inhibitors (RTIs) and protease inhibitors (PIs). Drugs falling in both of these categories prevent the proliferation of the HIV in different ways, during viral life cycle, HIV hijacks the CD4 + target cell and uses it to replicate itself. The RITs operate by stopping the conversion of HIV RNA to DNA while the PIs reduce the number of virus particles released by infected cell by disrupting the viral assembly process which results in the improper structuring of the viral proteins.
Acutely infected HIV patients are treated using a highly active antiretroviral therapy (HAART). This treatment usually consists of a combination of one or more RTI and PI drugs. Although HAART is successfully able to maintain a low viral load in many patients, its long-term usage comes with a host of complications. In some cases, the HIV mutates producing drug-resistant strains which result in the need to change the drug therapy but many times it is difficult to find pharmaceuticals that provide effective treatment. Further, prolonged exposure to these drugs can cause patients to have benign to severe pharmaceutical side effects which affect the continuity of treatment. Various social and economic barriers in conjunction with high drug cost and convoluted pill regime limits the effective use of HAART in some patients. These overarching concerns about the long-term use of antiretroviral therapy strongly motivate the consideration of optimal schemes for its use.
One such strategy is to use repeated cycles in which the therapy is switched on and off. An advantage of using this protocol is that it can result in reactivating an adaptive immune response during the time frame when the treatment is discontinued (Off). This regime is also called as structured treatment interruption (STI). Researchers have shown that using this kind of strategy can sometimes help patients maintain immune control of virus even in the absence of treatment [14]. A large number of studies have been conducted to investigate the benefits of the STI protocols [3, 13, 19] however; there is no consensus on the optimal drug schedule or interruption schemes that result in effective treatment. Adams *et al*. explore the problem of finding the optimal STI strategy by using a system of ordi-

nary differential equations (ODE) to model the HIV infection dynamics and use continuous control theory to derive an optimal drug scheduling scheme. However, fitting the parameters of an ODE to accurately reflect and quantify biological observations is a difficult task. On the other hand, Reinforcement Learning(RL) is a direct optimal adaptive control framework [21, 22] that can help one discern optimal control strategies or policies by using measured trajectories. This obviates the need of identifying apriori a model of the system dynamics.

Thus in this project, we aim at using Reinforcement learning to determine close to optimal STI strategies using a set of trajectories generated during clinical trials of different STI protocols. More specifically we use the system of ODEs modelling the infection dynamics proposed in [1] to generate new trajectories in response to different STI policies suggested by the RL algorithm and in effect bootstrap these responses to improve the predictions made by the RL algorithms.

2. Environment Description:

In order to help the reader understand the actual problem being solved we; describe the characteristics of the model proposed by Adams *et al*. [1]. The Dynamic model is characterised by six state variables as described in table 1.

Table 1. State Space

Variable	Description
T_1	Number of healthy CD4 + T-lymphocytes
T_2	Number of healthy macrophages
T_1^*	Number of infected CD4 + T-lymphocytes
T_2^*	Number of infected macrophages
V	Number of free virus particles
E	Number of cytotoxic T-cells

Since our objective is to find the optimal STI strategy; our action space is defined in terms of efficacy of respective type of drug indicated by the variable ϵ_{i_t} where $i \in \{1, 2\}$. Table 2 hence describes the four actions each representing a discrete step in the STI cycle. More formally the action space is described as $A = \{(\epsilon_1, \epsilon_2) | \epsilon_i$ is measurable, $a_i \leq \epsilon_i \leq b_i, t \in [t_1, t_2] \, \forall i = 1, 2\}$.

RTI (ϵ_{1_t})	PI (ϵ_{2_t})	Status
0.7	0	RTI ON, PI OFF
0	0.3	RTI OFF, PI ON
0.7	0.3	RTI ON , PI ON
0	0	RTI OFF, PI OFF

Table 2. Action Space

The reward is expressed as:

$$r(s, a) = -QV_t - R_1\epsilon_{1_t}^2 - R_2\epsilon_{2_t}^2 + SE_t \qquad (1)$$

where: Q = 0.1 , R_1 = 20000, R_2 = 2000, S= 1000, ϵ_{1_t} = $\{0, 0.7\}$, ϵ_{2_t} = $\{0, 0.3\}$. Q, R_1, R_2 & S are weight constants for the virus, control inputs and immune effectors. The second and the third terms represent the systemic costs of drug treatment (severity of unintended side effects as well as treatment cost). The case when $\epsilon_1(t) = b_1$ represents maximal use of RT inhibitors and $\epsilon_2(t) = b_2$ maximal use of protease inhibitors. To summarize the equation (1) represents the goal to minimize both the HIV population and the systemic costs to body while maximizing immune response. Additionally, Adams *et al*. also show that in the absence of treatment (i.e.$\epsilon_{1_t}, \epsilon_{2_t}$ = 0) the system of ODE exhibits three physical equilibrium points within the state space. As shown in table 3, $s_{\text{uninfected}}$ represents a uninfected unstable equilibrium point and $s_{\text{unhealthy}}$ represents a healthy locally stable equilibrium point with a low virus load and High T-cell count & s_{healthy} represents a unhealthy locally stable equilibrium point indicating high viral load and low T-cell count.

State Variables	$s_{\text{uninfected}}$	$s_{\text{unhealthy}}$	s_{healthy}
T_1	10^6	967839	16353
T_2	3198	621	5
T_1^*	0	76	11945
T_2^*	0	6	46
V	0	415	63919
E	10	353108	24

Table 3. Equilibrium points

The simulator built for generating the trajectories starts with the state s_0 = $s_{\text{unhealthy}}$ and monitors it as we change the STI strategies. More concretely, the medication protocol is revised every five days during which patients data is monitored for every state variable. The change in the state variables from one point to another is seen as the state transition as response to the change in the control strategy and the corresponding reward is calculated as per equation (1). As with any RL algorithm our objective is to find a stationary policy π such that it maximises the expected return given by $V^\pi = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r(s, a) | s = s_0\right]$.

3. Methods

Before we begin describing our approach, we would like to highlight that the problem of predicting the optimal STI strategy has been previously studied by Ernst *et al*. [7] wherein they have reported the results of Fitted Q-iteration [2]. Our approach differs from theirs since we use on-line learning setting to test our algorithms as opposed to the off line learning scheme used by Ernst *et al*.

Within the purview of value-based methods, we implement variants of Temporal Difference (TD) Learning methods. TD methods are popular since they combine the sampling of Monte-Carlo with the Bootstrapping of Dynamic Programming. Additionally, these methods differ from Monte Carlo methods as they try to minimise the difference between sequential (temporal) predictions instead of an overall prediction error. Since these methods rewrite the value function update in the form of a Bellman equation, the resulting bootstrapping helps reduce the variance of prediction in every step. In the next section, we define the mathematical notation for a general case of TD methods

3.1. Temporal Difference Learning (TD)

Consider an irreducible periodic Markov chain with a finite or countably infinite state space. We can view the state space to be indexed as a set of positive integers given by $S = \{1, 2, ..., n\}$.

In a general sense, the state space is a countable subset of a Euclidean space. Particularly, each state can correspond to a vector of real numbers that describe the state of a physical system which in our case are markers of HIV.

The sequence of states visited by the Markov chain is denoted as $\{s_t | t = 0, 1, 2, , ..., n\}$. A finite or infinite transition probability matrix describes the dynamics of the Markov chain. It is denoted by $P(s, s')$ where the $(s, s')^{\text{th}}$ entry denoted as $p_{ss'}$ is the probability of transitionning to $(s_{t+1} = s')$ given that $(s_t = s)$. For any pair (s, s') we obtain a scalar $r(s, s')$ which represents the reward for transitioning from s to s'. The discount factor is given by $\gamma \in (0, 1)$. Assuming that the expectation in equation (2) is well defined, the value function $V : S \mapsto \mathbb{R}$ associated with this Markov Chain is defined as:

$$V^*(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, s_{t+1}) | s_0 = s\right] \tag{2}$$

Since we are dealing with continuous state space we require function approximation. We consider approximation of $V^* : S \mapsto \mathbb{R}$ using a function approximator such that $\hat{V} : S \times \mathbb{R}^k \mapsto \mathbb{R}$. Generally, for approximating the value function we select a parameter vector $w \in \mathbb{R}^k$ to minimize some error metric between $V^*(\cdot)$ & $\hat{V}(\cdot, w)$.

Assuming that at time t we have set the parameter w to some value w_t. Upon observing the sequence of states s_t, we define the temporal difference d_t corresponding to the transition from s_t to $s_{t+1} = s'$ by:

$$d_t = r(s, s') + \gamma \hat{V}(s_t, w_t) - \hat{V}(s_t, w_t) \tag{3}$$

Subsequently, for $t = 0, 1, ...$, the TD learning method updates the w_t according to the formula:

$$w_{t+1} = w_t + \alpha_t d_t \sum_{k=0}^{t} (\gamma\lambda)^{(t-k)} \nabla_w \hat{V}(s_k, w_t) \tag{4}$$

where w_0 is initialized to some arbitrary vector, α_t is a sequence of scalar step size, λ is parameter such that $\lambda \in [0,1]$, and the gradient $\nabla_w \hat{V}(s_k, w_t)$ is the partial derivative of w.r.t w. We also consider a special case of linear function approximators using which $\hat{V}$ takes the form:

$$\hat{V}(s, w) = \sum_{k=1}^{K} w(k)\phi_k(s) \tag{5}$$

Here, $w = (w(1), ..., w(K))$ is the parameter vector and ϕ_k is a fixed scalar function defined on the State Space S.

We can also define a vector values function $\phi : S \mapsto \mathbb{R}^k$ by letting $\phi'(i) = (\phi_1(s), \cdots, \phi_k(s)$. With this modification, the approximation can be written as :

$$\hat{V}(w) = \Phi \cdot w \tag{6}$$

Where Φ is a $|S| \times K$ matrix whose k^{th} column is equal to ϕ_k, i.e.

$$\begin{bmatrix} | & | \\ \phi_1 & \phi_k \\ | & | \end{bmatrix}$$

Thus the gradient vector becomes

$$\nabla \hat{V}(w) = \Phi' \tag{7}$$

where $\nabla \hat{V}(w)$ is the Jacobian Matrix whose i^{th} column is equal to the $\nabla \hat{V}(s, w)$.

Elaborating more on the need of function approximation; a large state space presents two major challenges. The most obvious one is the storage problem, as it becomes impractical to store the value function (or optimal action) explicitly for each state. The other is the generalization problem, assuming that limited experience does not provide sufficient data for each and every state. Both these issues are addressed by the Function Approximation approach [21], which involves approximating the value function by functional approximators with given architectures and a manageable number of adjustable parameters. Obviously, the success of this approach rests on some regularity properties of the state space, possibly induced by appropriate feature selection, and on the proper choice of an approximation architecture and size. In a linear approximation architecture, the value of a state is computed by first mapping it to a low dimensional feature vector, and then linearly weighting these features using adjustable weights. The functions used to compute each entry in the feature vector are called the

basis functions. A notable class of linear function approximators is that of Radial Basis Function (RBF) networks [8, 12, 18, 16]. RBFs can be thought of as neural networks with a single hidden layer where each neuron in the hidden layer is a RBF kernel function as shown in figure 1. Mathematically RBF kernel can be written as follows:

$$\phi(x) = \frac{1}{\sigma\sqrt[2]{2\pi}} exp\left(\frac{-|x-\mu|^2}{2\sigma^2}\right) \tag{8}$$

The update rule then follows as:

$$f(x) = \sum_i w_i \phi(x, x^i) \tag{9}$$

RBF kernels also have a particular advantage that although the kernel function $f(x)$ in non-linear w.r.t x, it can be expressed as a linear combination of $\phi(x)$

In the RL context, linear architectures uniquely enjoy convergence results and performance guarantees, particularly for the problem of approximating the value of a fixed stationary policy [24]. Thus we use RBF functions for feature extraction and then approximate the value function using semi-gradient methods.

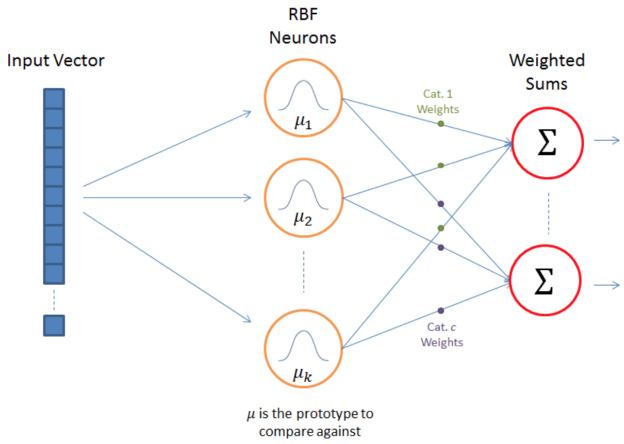


Figure 1. High level schematic of a RBF network from [15]

3.2. Policy Based Methods

The value based methods described in the section 3.1 parameterise the value function and derive the near optimal policy from it. In contrast, Policy Based methods are class of methods which aim at learning the policy directly by parameterizing it. Actor Critic algorithm [11] can be viewed as a TD variant of Policy based methods. From algorithm 1 we can see that in this framework, the Actor updates the parameters of the policy using Policy Gradient theorem [23] whereas the Critic updates the parameters of the approximate action value function $\hat{Q}(s,a)$ using the TD method described in section 3.1. Owing to these unique properties,

Algorithm 1: Actor Critic Algorithm based on Action value critic

Initialise s,w, θ;
Sample action , $a \sim \pi_\theta$;
for *every step* **do**
  Sample reward $r = r(s,a)$;
  Sample transitions $s \sim P_a^{ss'}$;
  Sample next action $a' \sim \pi_\theta(s', a')$;
  $d = r(s,s,a) + \gamma\hat{Q}_w(s', a') - \hat{Q}_w(s,a)$;
  $\theta = \theta + \beta\nabla_\theta log\pi_\theta(s,a)\hat{Q}_w(s,a)$;
  $w \leftarrow w + \alpha d_\phi(s,a)$;
  $a \leftarrow a'$, $s \leftarrow s'$;
end

we implement the Actor-Critic algorithm using non-linear function approximation.

In conjunction with the Actor-Critic algorithm we also implement the recently introduced approach called Trust Region Policy Optimization (TRPO) [20]. These methods demonstrate robust performance on a wide variety of tasks and tend to give monotonic improvements with "little hyper-parameter tuning", this property warrants their use in our task.

3.3. Implementation

As discussed in section 3 we implement the three most basic variants of TD Algorithms; Q-learning ,SARSA & Expected SARSA. For the function approximation part we implement, four RBF kernels with exemplar values of 0.05, 1, 0.5, 0.1 respectively. Note that these values have been empirically chosen and we don't dwell on the rational of choosing them for the sake brevity. We use a constant step size α = 0.1 and set the discount rate γ = 0.99. We repeat 5 cycles of 10000 steps each to study the variance of these methods for a comparative analysis.

For Actor Critic implementation we use a simple 2 layer feed-forward MLP with $tanh$ activation applied between successive linear layers for approximating $\hat{Q}_w(s,a)$ as well as policy π_θ. We use the Adam optimizer [10] with the learning rate of 10^{-1}, due to its demonstrated advantages of improving convergence by approximating second-order methods For TRPO, we follow the same structure as for the actor critic implementation. Additionally, we use a softmax layer at the end of our policy network which returns probability values corresponding to each value. Here we use the learning rate of 10^{-4}.

4. Results & Discussion

We now discuss results of our implemented algorithms and discuss their implications.

4.1. Results - Value based Methods:

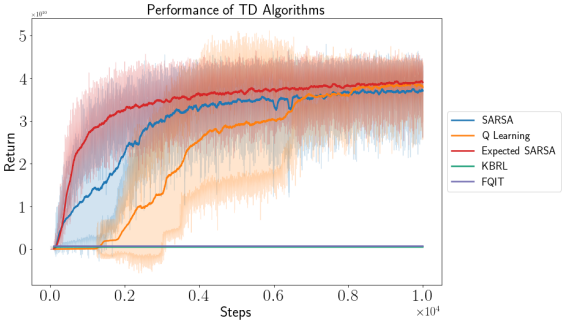


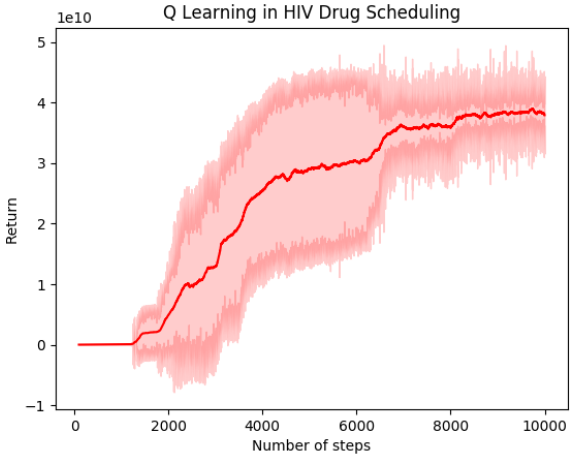
Figure 2. Performance of TD-algorithms vs reported baselines by Ernst *et al*. [7] & Barreto *et al*. [4]

Figures 2 ,3 show the results of TD algorithms, we find that our implementation of TD algorithms outperform the results reported in Barreto *et al*. by a significant measure. We can also see that Expected-SARSA yields better performance as compared to Q learning and SARSA. We handle the explanation of this phenomenon on a case by case basis.

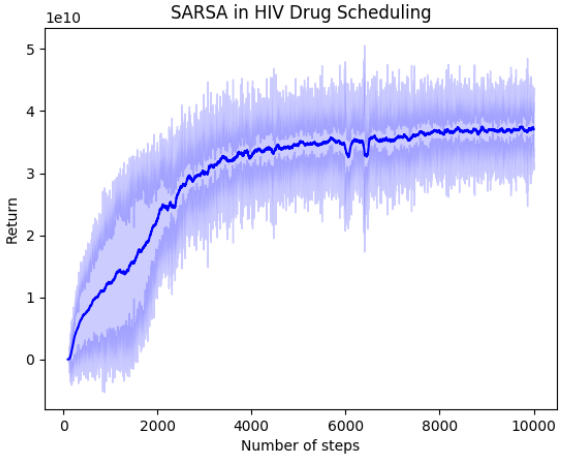
Expected SARSA vs Q-learning: The main difference between Expected SARSA and Q-learning algorithm is that the former is a On-policy method while the latter is a Off-policy method. In Q learning we choose the action maximizes the state-action value function $Q(s, a)$, in a way this operation prohibits exploration and makes the policy greedy with respect to the values of Q. This can some times hurt the agents performance as it never learns to avoid certain bad actions. On the other hand on-policy methods try to estimate the optimal way to behave given the exploration that is occurring. Hence we can conclude that Expected SARSA will perform better than Q-learning in cases where the ϵ-soft policy is better than the ϵ-soft policy based on $Q^*(s, a)$ [25]

Expected-SARSA vs SARSA: From figures 3(b) & 3(c) we can that the variance of Expected SARSA is lower than SARSA. The high variance is the result of the substantial difference in the state-action($Q(s, \cdot)$) values for every action a given state s. This variance is eliminated in Expected SARSA due to the expectation operation performed in the target update.

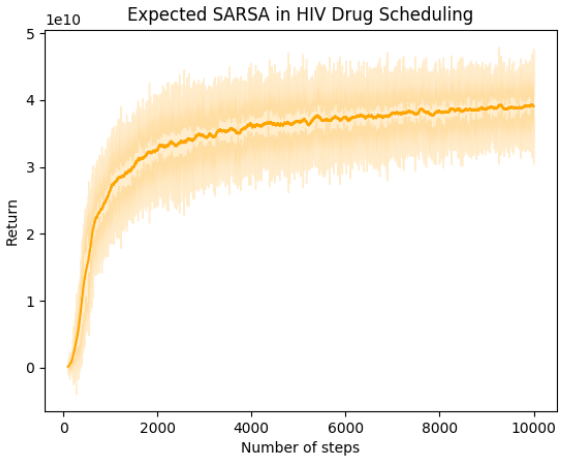
In figure 5 we plot the evolution of state variables $T_1, T_1^*, T_2, T_2^*, V, E$ as the learning progresses to verify the efficacy of the STI policy learnt by our RL agent. For all the three algorithms we notice that the learnt STI strategy successfully helps lower the viral load V and higher T-cell count and is able to bring the patient within the domain of attraction of the healthy steady state.



(a)



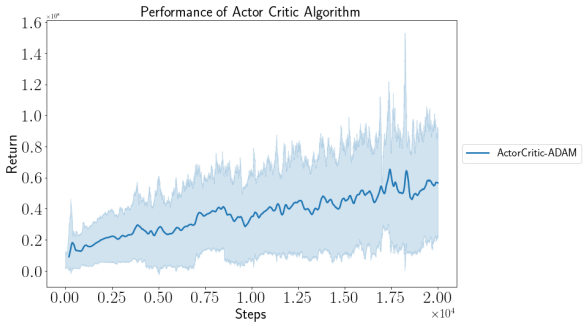
(b)



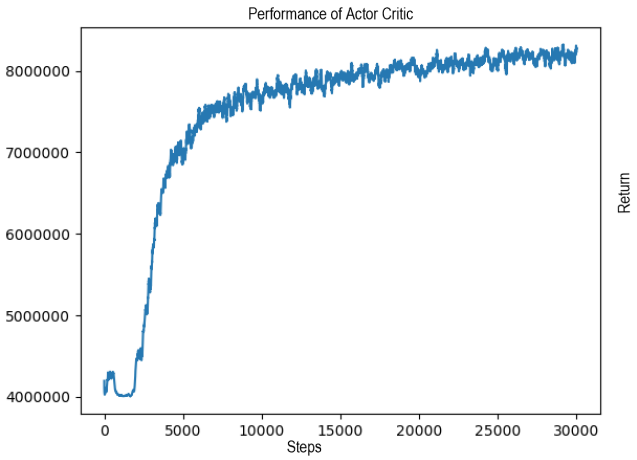
(c)

Figure 3. Performance of TD-algorithms averaged across five runs

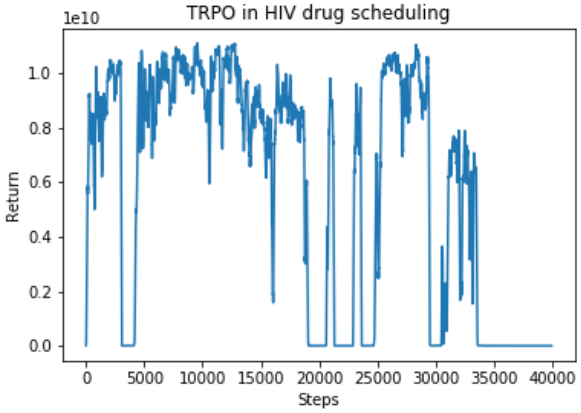
4.2. Results - Policy Based Methods



(a)



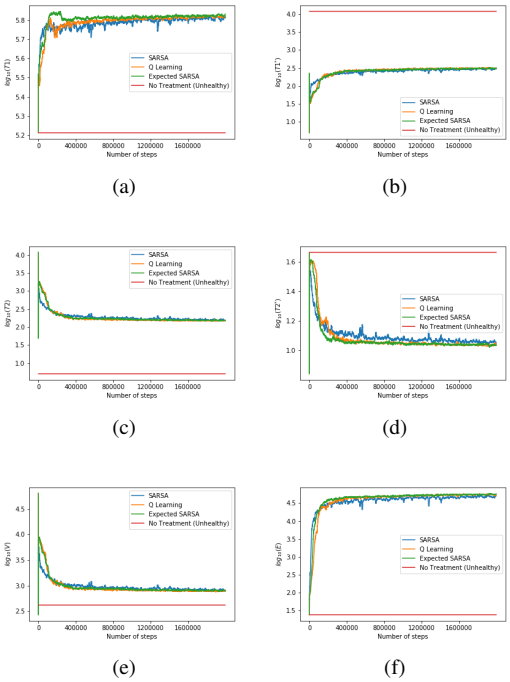
(b)



(c)

Figure 4. Performance of Policy based methods

Based on our implementations, we find the results of policy based methods using non-linear function approximation to be inconclusive. Figure 4(a) shows the results of 10 runs of Actor-Critic algorithm using a vanilla neural network and Figure 4(b) shows the results of the Actor-Critic algorithm for a single run. From figure 4(a) we can see that our implementation diverges in variance with the number of runs in-spite of promising convergence in a single run. Paying heed to the concerns raised by Islam *et al.* [9] we do not fix the seed during multiple runs of our algorithm. This reinforces the susceptibility of divergence in non-linear function approximation methods due to lack of careful tuning of hyper-parameters [24, 9]. This observation is further bolstered by the results of our TRPO implementation in figure 4(c), where algorithm doesn't seem to learn anything in-spite of using smaller learning rates and experimenting with the network architecture. Due to the lack of computing resources at our disposal, we were not able to run a full-fledged grid search for the optimal hyper-parameters, and report the best performance of our implementations.



(a) (b) (c) (d) (e) (f)

Figure 5. The curves in the figure(s) represent the evolution of state variables as the learning progresses as compared to the unhealthy steady state ($s_{\text{unhealthy}}$) as indicated in table 3.

4.3. Discussion

Although our algorithms seem to have outperformed the some of the previously tested approaches in terms of the expected return, we would like to highlight the fact that this does not nearly present a strong case for vouching for the efficiency of the RL in solving for the optimal STI strategy. In table 3 we have defined the notion of a locally stable equilibrium point that denotes the healthy state of the pa-

tient, and the goal of our algorithm should be to reach that state as soon as possible and remain there for the life-time of the patient. Additionally the according to the model proposed by Adams *et al*., the treatment protocol is evaluated and changed every 5 days, thus based on our results in fig of states , our algorithms reach this steady state at around 750 days. In a practical scenario this time frame is too long for a person to reach a healthy state. This calls in for the need to additional tunning of parameters of the learning algorithm using certain heuristics. While using these heuristics might be possible in a simulated environment, real life deployment of such algorithms will require consultation with medical experts to ensure proper definition of the objective.

Moreover, the interaction dynamics of HIV and the immune system can vary from person to person. The model proposed by Adams *et al*. assumes the interaction dynamics to be same for every patient, and is quite likely to be violated in a real life scenario. In order to capture these subtleties the state space needs to incorporate additional variables which act as indicators of these features.

5. Conclusion

In this paper, we have used a real-life problem for benchmarking the performance of some basic RL algorithms. To this end, we have also wrapped an environment simulating a plausible mathematical model of the HIV infection dynamics in the OpenAI-Gym framework, which is extensively used by the RL community to test their algorithms. Additionally, we have tested the performance of value-based and policy-based methods on this new environment.

Based on our results, we can conclude that reinforcement learning techniques could help to design effective real-life STI strategies using trajectories generated from the simulated model. However, we feel that the simulator can be improved further by modelling various difficulties encountered in the real-life deployment of such systems. Further, we also think that this environment provides a platform to test the effectiveness of model-based reinforcement learning methods. Finally, we also acknowledge that in spite of meticulous modelling of the interaction dynamics, the problem of finding the optimal treatment strategies is still a difficult one to solve and, will need multiple rounds of reformulation and testing before it is ready to be deployed in the real world.

Contributions: **Gandharv Patil** created the GYM environment, implemented the Q-learning and Actor-Critic methods and authored the report.**Raihan Seraj** implemented the SARSA, Expected-SARSA & TRPO methods, and helped writing the report

References

[1] B. Adams, H. T. Banks, H.-D. Kwon, and H. T. Tran. Dynamic multidrug therapies for hiv: optimal and sti control approaches. *Mathematical biosciences and engineering : MBE*, 1 2:223–41, 2004. 2

[2] A. Antos, C. Szepesvári, and R. Munos. Fitted q-iteration in continuous action-space mdps. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 9–16. Curran Associates, Inc., 2008. 2

[3] S. H. Bajaria, G. Webb, and D. E. Kirschner. Predicting differential responses to structured treatment interruptions during haart. *Bulletin of Mathematical Biology*, 66(5):1093–1118, Sep 2004. 1

[4] A. M. Barreto, D. Precup, and J. Pineau. Practical kernel-based reinforcement learning. *Journal of Machine Learning Research*, 17(67):1–70, 2016. 5

[5] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013. 1

[6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016. 1

[7] D. Ernst, G. B. Stan, J. Goncalves, and L. Wehenkel. Clinical data based optimal sti strategies for hiv: a reinforcement learning approach. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 667–672, Dec 2006. 2, 5

[8] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 1998. 4

[9] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep reinforcement learning that matters. *arXiv preprint arXiv:1709.06560*, 2017. 6

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4

[11] V. R. Konda and J. N. Tsitsiklis. On actor-critic algorithms. *SIAM J. Control Optim.*, 42(4):1143–1166, Apr. 2003. 4

[12] R. P. Lippmann. Pattern classification using neural networks. *IEEE Communications Magazine*, 27(11):47–50, Nov 1989. 4

[13] J. Lisziewicz and F. S. H. Lori. Structured treatment interruptions in hiv/aids therapy. *Microbes and infection*, 4 2:207–14, 2002. 1

[14] J. Lisziewicz, E. Rosenberg, J. A. Lieberman, H. J. Jessen, L. Lopalco, R. F. Siliciano, B. A. Walker, and F. S. H. Lori. Control of hiv despite the discontinuation of antiretroviral therapy. *The New England journal of medicine*, 340 21:1683–4, 1999. 1

[15] C. McCormick. Radial basis function network (rbfn) tutorial, 2013. 4

[16] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991. 4

[17] M. Plappert, M. Andrychowicz, A. Ray, B. McGrew, B. Baker, G. Powell, J. Schneider, J. Tobin, M. Chociej,

P. Welinder, V. Kumar, and W. Zaremba. Multi-goal reinforcement learning: Challenging robotics environments and request for research, 2018. 1

[18] M. J. D. Powell. Algorithms for approximation. chapter Radial Basis Functions for Multivariable Interpolation: A Review, pages 143–167. Clarendon Press, New York, NY, USA, 1987. 4

[19] L. Ruiz, J. Martinez-Picado, J. C. Romeu, R. Paredes, M. Zayat, S. Marfil, E. Negredo, G. Sirera, C. Tural, and B. Clotet. Structured treatment interruption in chronically hiv-1 infected patients after long-term viral suppression. *AIDS*, 14 4:397–403, 2000. 1

[20] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. 4

[21] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. MIT Press, Cambridge, Massachusetts, 1998. 2, 3

[22] R. S. Sutton, A. G. Barto, and R. J. Williams. Reinforcement learning is direct adaptive optimal control. In *1991 American Control Conference*, pages 2143–2146, June 1991. 2

[23] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, pages 1057–1063, Cambridge, MA, USA, 1999. MIT Press. 4

[24] J. N. Tsitsiklis and B. V. Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, May 1997. 4, 6

[25] H. van Seijen, H. van Hasselt, S. Whiteson, and M. Wiering. A theoretical and empirical analysis of expected sarsa. In *2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pages 177–184, March 2009. 5