Robust Reinforcement Learning Through Adversarial Training

Varun Tandon
Department of Computer Science
Stanford University
varunt@stanford.edu

Avi Goyal
Department of Computer Science
Stanford University
sagoyal@stanford.edu

Abstract

In this work, we investigate the impact of adversarial training on the robustness of reinforcement learning (RL) agents subjected to environmental perturbations, with a primary focus on Q-learning within the Cartpole-v1 context. Our results indicate that adversarial training requires a greater number of episodes for convergence compared to traditional tabular Q-learning. Additionally, we observe significant variance in the robustness of models trained via standard tabular Q-learning. While we did not observe notable robustness gains in adversarial Q-learning within the tabular setting for Cartpole, our findings suggest that adversarial training contributes to a noticeable improvement in the robustness of Deep Q Networks (DQN). To explain why adversarial training can result in a more robust agent, comparative analysis of Q-tables between standard and adversarially trained agents was performed. Our study asserts that while adversarial training can potentially improve robustness, these gains are not guaranteed and are accompanied by increased training duration and worsened performance within the default environment.

1 Introduction

In the last decade, reinforcement learning has seen substantial progress, most notably, with the development of the DQN algorithm by OpenAI in 2013 [14], and subsequently the improvement of DQN via Double DQN by DeepMind in 2015 [22]. These advancements in the field led to breakthrough results, in Go [19], protein folding [9], matrix multiplication [6], robotics [15], self-driving [17], and most recently in RLHF [4].

In many real-world situations involving reinforcement learning, robustness to environmental changes is critically important. To illustrate the importance of reinforcement learning in the real-world, consider a reinforcement learning based decision system in a self-driving vehicle. The vehicle needs to be resilient to changes in the external environment, such as heavy rains, winds, or altitude changes, as well as sudden disruptions it may not have encountered in training, commonly known as tail-end events.

More broadly in the realm of distributional robustness, adversarial training has been proposed as a mechanism by which certifiable robustness can be achieved. In particular, we consider the paper *Certifying Some Distributional Robustness with Principled Adversarial Training* [20], where the authors adversarially perturb inputs during training. We discuss these results in further detail in Section 2, but notably the authors provide an example of how adversarial perturbations during Q-learning can result in more efficient training and a more robust RL agent.

In this paper, we investigate the claims of the authors, and attempt to further explore why adversarial robustness can lead to distributional robustness. All experiments are conducted in the cartpole environment. In particular, our contributions are the following:

1. **Convergence:** We present empirical evidence that contradicts the authors' claims that adversarial training is more efficient than standard Q-learning.
2. **Variance of Robustness in Standard Q-Learning Training:** We investigate how standard Q-learning can exhibit high variance in the robustness of the final model, and in particular, we demonstrate that the stopping condition plays a role in final model robustness.
3. **Adversarial Training in Tabular Q-Learning:** We compare adversarial Q-learning to standard Q-learning and demonstrate that in the simple setting standard Q-learning is as robust as adversarial Q-learning. Furthermore, we manually examine the learned Q tables to provide a partial explanation for the differences in the final learned models.
4. **Adversarial Training in Double DQN:** While Sinha et al.[20] only provide results for tabular Q-learning, we also provide results of adversarial training in the Double DQN setting, which represents the modern paradigm for RL models. We demonstrate that adversarial training does improve in robustness over standard training, but at the cost of performance in the default environment.

2 Related Work

In the literature of distributional shifts, there are many ways of addressing the issue of adapting your model to a new test time environment. In this paper we focus on the specific case of being distributional robust in an adversarial setting. That is, given a learning model, how can make sure the model is robust to small adversarial perturbations in the input so that output remains consistent with the results that would have been produced under no perturbations [7]. Usually in this setting the perturbations are small and imperceptible to the human eye [16]. However for each new adversarial method, there have been proposed mechanisms that defend against them, and vice versa [8], [21], [3].

In order to avoid this cycle, researcher have turned towards certified robustness, which hopes to find the perturbation regime under which a model is guaranteed to be robust [5], [20].

Particularly the Sinha et. al. paper [20] we draw inspiration provides a principled adversarial training procedure with a robustness certificate on upper bound on the worst-case loss $\sup_{P:W_c(P,P_0)\leq\rho}\mathbb{E}_P[\ell(\theta;Z)]$ where P defines a ρ-neighborhood of the distribution P_0 under the Wasserstein metric $W_c(\cdot,\cdot)$.

However practically computing this certificate as well as the appropriate algorithm variables are cumbersome for a large neural network with many parameters. So instead we use an unprincipled approach, matching section 5.2 of the Sinha et. al, of simply trying the adversarial training method with variables that are large enough to satisfy smoothness of the Wasserstein cost function. We describe their method as it relates to reinforcement learning in our section 4.

For the specific instance of adversarial robustness for reinforcement learning problems, the main concern is around safety - for example adversarial states can cause autonomous vehicles to swerve into oncoming traffic [11]. Some examples of proposed methods for robustness try to minimize the worst adversarial perturbation range as it's fed through a Deep Neural Network [13], understand natural and sensitive directions of adversarial state adjustment [10], or train GAN like methods that learn an optimal destablization policy [18]. There are also some works that try to instead bound the feasibility of environmental variation, realizing that some worse-case environments may prove too challenge and unrealistic in deployed settings [12].

3 Problem Statement

Consider an agent $\mathcal{A}$ that acts in an environment $\mathcal{E}$ over N timesteps. At each timestep $t \in [0, N]$, the environment has some state $s_t \in \mathcal{R}^d$ where d is the dimensionality of the environment state, and the agent takes some action $a_t \in \mathcal{R}$, where the action space can be either discrete or continuous depending on the environment. After taking action a_t, the environment moves to a new state s_{t+1} according to a dynamics model $\mathcal{P} \sim p(s'|s, a)$. The environment also gives the agent some reward r_t after it takes its action, which can be represented as either as a function of the new state $r(s_{t+1})$ or of the action taken in a given state $r(s_t, a)$. This distinction is environment specific.

In the given environment $\mathcal{E}$ there are certain governing properties that remain consistent

throughout the iterations of the environment. We discuss these governing properties, g, in more detail in the discussion of each individual environment, but an example property might be acceleration of gravity equal to $9.8m/s^2$. Consider $\mathcal{E}$ to be our default environment, with the default governing properties, and let $\mathcal{E}'$ be a modification of the default environment with modified g. Our aim is to train $\mathcal{A}$ which performs well in both $\mathcal{E}$ and $\mathcal{E}'$.

4 Method

4.1 Cartpole Environment

We conduct all experiments in the Cartpole-v1 environment [1]. The task of Cartpole involves balancing a pole on a cart for a maximum of 500 time steps. The environment terminates when the pole has been balanced for 500 time steps, the cart is moved to a position beyond ± 2.4 from the center, or the pole has angle beyond $\pm 12^{\circ}$. At each time step, the agent is given a reward of +1. The agent can take an action 0 or 1, which represent moving the cart to the left or right respectively. Moreover, at each time step, the agent receives an updated state, which includes the cart position, x, cart velocity, $\dot{x}$, angle of the pole, β, and angular velocity of the pole, $\dot{\beta}$.

The above describes our $\mathcal{E}$, and in order to construct $\mathcal{E}'$, our environment with different conditions from the training environment, we modified the pole length, pole mass, and gravity of the environment. These modifications were done in alignment with the results in Sinha et al [20]. The default pole length is 0.5, the default pole mass is 0.1, and the default gravity is 9.8. For our $\mathcal{E}'$, we additionally considered pole lengths of 0.25 and 1, pole masses of 0.05 and 0.2, and gravity of 1.96 and 49.0. We also considered all possible combinations of these environmental modifications.

4.2 Adversarial Perturbation

For both of our Q-learning methods, we follow the following adversarial perturbation as defined in Sinha et al [20]. Consider the standard Q-learning update:

$$Q\left(s^t, a^t\right) \leftarrow Q\left(s^t, a^t\right) + \alpha_t \left(r\left(s^t\right) + \lambda \max_a Q\left(s^{t+1}, a\right) - Q\left(s^t, a^t\right)\right) \tag{1}$$

When we take some action a^t at state s^t, we get back $\widehat{s}^{t+1} \sim p_{sa}\left(s^t, a^t\right)$. We can now adversarially perturb this state via the following equation:

$$s^{t+1} \leftarrow \arg\min_s \left\{ r(s) + \lambda \max_a Q(s, a) + \gamma c\left(s, \hat{s}^{t+1}\right) \right\} \tag{2}$$

We compute the argmin via gradient-based minimization, and we use the L2 norm as our cost function c. Furthermore, since our reward function in cartpole is not differentiable, we use the recommendation of Sinha et al. [20], and compute $r(s) = e^{-|\beta|}$.

4.3 Tabular Q Learning

For our tabular Q-learning results, we use the standard tabular Q-learning algorithm, which can be found in Algorithm 1.

Algorithm 1 Tabular Q learning algorithm.

Initialize $Q(s, a), \forall s \in S, a \in At = 0$, initial state $s_t = s_0$
Set π_b to be ϵ-greedy w.r.t. Q
while convergence condition not met **do**
  Take $a_t \sim \pi_b\left(s_t\right)$ // Sample action from policy
  Observe $\left(r_t, s_{t+1}\right)$
  $Q\left(s_t, a_t\right) \leftarrow Q\left(s_t, a_t\right) + \alpha\left(r_t + \gamma \max_a Q\left(s_{t+1}, a\right) - Q\left(s_t, a_t\right)\right)$
  $\pi\left(s_t\right) = \arg\max_a Q\left(s_t, a\right)$ w.prob $1 - \epsilon$, else random
  $t = t + 1$
end while

We follow the same approach as Sinha et al. [20], and we only consider $\beta, \dot{\beta}$ in our state, with β

being discretized into 30 buckets, and $\dot{\beta}$ being discretized into 15 buckets. Moreover, since we do not have a differentiable Q function, per Sinha et al. [20], we do not include this term in our adversarial state calculation, simplifying the calculation to:

$$s^{t+1} \leftarrow \arg\min_{s} \left\{ r(s) + \gamma c\left(s, \hat{s}^{t+1}\right) \right\} \tag{3}$$

The justification for this can be found in the original paper, but in short, by removing the dependence on the Q term, we are effectively modifying the environment transition dynamics, and thus maintain the theoretical convergence guarantees of Q-learning. In our experiments, we set $\gamma = 1.3$. We chose this value via hyperparameter tuning, and this was the value that caused the perturbed angle to be off by at most one bucket. Intuitively, these perturbations are large enough that they are meaningful, but not so large that our training fails to converge.

4.4 Double DQN

For our DQN results, we use the standard double DQN algorithm. We opt for double DQN rather than DQN due to its training stability and widespread adoption as the default deep Q-learning method. We also opt to use experience replay for increased training stability. For the sake of brevity, we do not include the pseudocode for DQN training, however, the algorithm we followed the pseudocode provided by Stanford's CS234 course [2].

For our policy and target networks, we instantiate a deep neural network with 4 inputs, one hidden layer of size 256, and an output of size 2. Since we have a differentiable Q function in this setting, we use the full adversarial perturbation from Equation 2.

Just as in the tabular setting, we chose $\gamma = 1.3$.

5 Empirical Results

5.1 Evaluating Convergence Times of Standard vs. Adversarial Training

To assess the claim that adversarial training is more efficient than standard Q-learning, we conducted experiments on both the tabular Q-learning method and the deep Q-learning method. In both cases, we conducted extensive hyperparameter tuning over the learning rate (α), learning rate decay, exploration rate (ϵ), and exploration rate decay and selected the model with the quickest mean convergence time. We defined convergence as the agent surviving for the full 500 time steps in 10 consecutive episodes.

Training Strategy	Mean Convergence Time (95% CI)
Standard	307.8421$\pm$ 29.964
Adversarial	745.3333 $\pm$ 75.08

Table 1: Convergence Times For Standard Training vs. Adversarial Training in Tabular Q Learning

Based on our experiments, standard training was demonstrated to be more efficient than adversarial training in terms of convergence time. This also does not consider the significantly greater running time of adversarial training, as the adversarial minimization step is a significant computational cost.

5.2 Variance of Robustness in Standard Q-Learning

We first considered the tabular Q-Learning agent. While training the agent we observed that there were inconsistent episode run lengths within model and between model specifications, even after holding all hyper-parameters constant. To investigate how robustness changes between models, we looked at three different stopping conditions (stopping when $5, 10$ or 20 consecutive runs reached the max episode length of 500). We found that a stopping condition of 10 consecutive runs reaching the

maximum episode length, gave the highest average robustness scores. See appendix for complete comparison of all stopping conditions on full environmental range.

To investigate how robustness changes within the same model we trained 10 identical agents independently and measured their episode run length standard deviation. We saw that the greatest standard deviation was among the hardest environments (strong gravity and short pole length), however in the model specified with the stopping condition of 10 consecutive runs, we noticed that only strong gravity caused deviations in the episode run length.

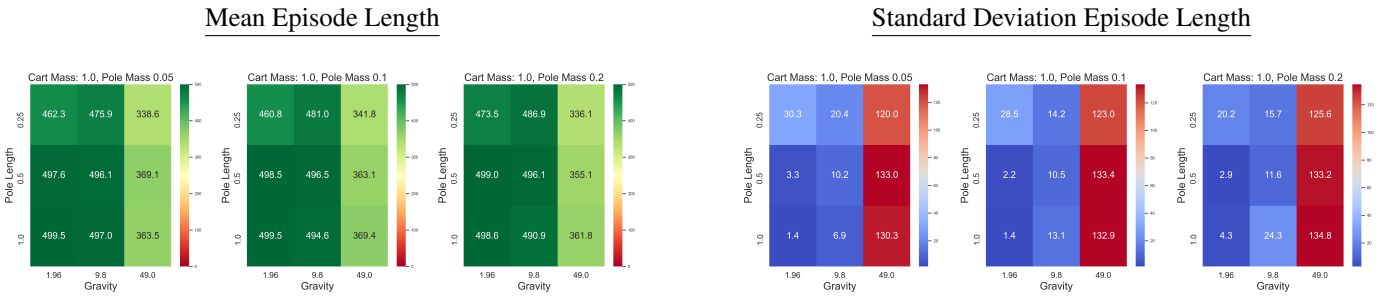


Figure 1: Mean (left) and Standard Deviation (right) episode length in Cartpole under various environment perturbations with a stopping condition of 10 consecutive runs reaching the maximum episode length of 500.

This results from Figure 1 indicate that robustness to adversarial setting can be achieved by finding the appropriate stopping condition or by instantiating multiple models with the sample specifications simply taking an ensemble approach. Furthermore, the results sets a baseline for robustness for the Q-learning agent.

5.3 Adversarial Training in Tabular Q-Learning

To evaluate the role of adversarial training in tabular Q-learning, we trained Q-learning agents in the standard tabular setting and the adversarial setting, in accordance with the experimental setup in Section 4.3. Per our results in Section 5.2, we found the optimal stopping condition to be 10 consecutive iterations at 500, and thus, we trained both the standard and adversarial agents with this convergence condition. We then evaluated the agents on out of distribution environments, $\mathcal{E}'$, as described in Section 4.1. The mean episode length across 30 episodes for our standard agent and adverasrial agent are presented in Figure 2

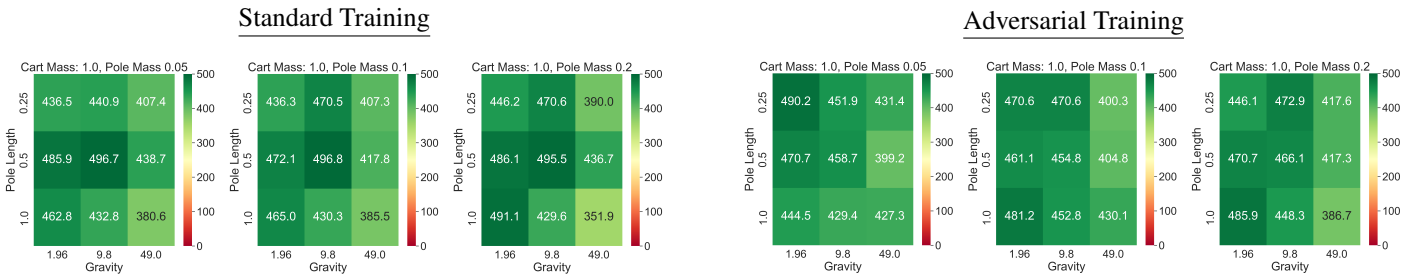


Figure 2: Mean episode length in Cartpole under various environment perturbations. On the left is a model obtained from standard tabular Q learning, and on the right is a model obtained from adversarial tabular Q learning.

From these results, we do not notice a significant improvement in robustness from adversarial training in the tabular setting. This deviates from the result presented in Sinha et al. [20], notably since our model that was trained without adversarial perturbations is fairly robust. We believe this is due to the variance in robustness discussed in Section 5.2. Perhaps Sinha et al. [20] obtained a less robust

model from their standard training, just due to random chance. We present the learned Q-tables and a discussion of these results in Section 6.1.

5.4 Adversarial Training in DQN

Finally, we conducted standard and adversarial training with DQN agents. These results evaluate the role of adversarial perturbations in the neural network setting, and thus represent a more realistic setting given modern reinforcement learning. Notably, the neural network setting is not a setting that was previously evaluated by Sinha et al [20]. We used the same convergence condition as before: 10 successive episodes where the agent survives for 500 time steps. Our results can be seen in Figure 3.

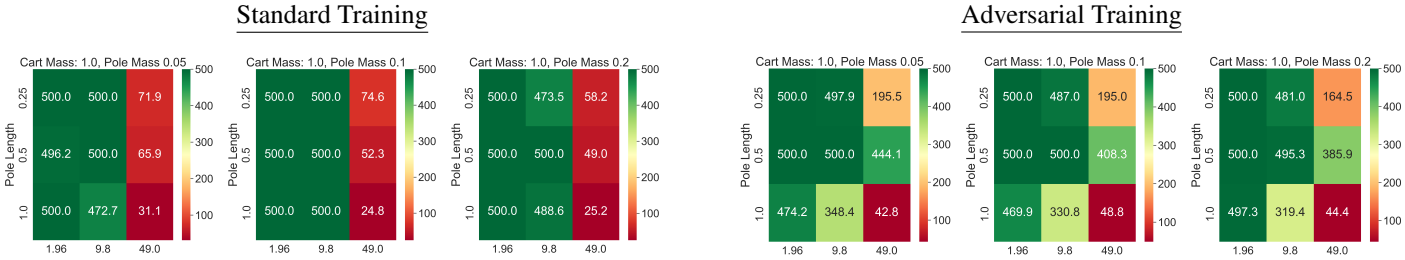


Figure 3: Mean episode length in Cartpole under various environment perturbations. On the left is a model obtained from standard DQN training, and on the right is a model obtained from adversarial DQN training.

Based on these results, the adversarially trained agent is more robust to environmental changes. Interestingly, while the adversarially trained agent is more robust to environmental changes, it does have a decreased performance in the default environment. Moreover, the agents trained via DQN seem to be more sensitive to enviornmental changes. We hypothesize that this is due to high number of parameters in the neural network, which in some sense "overfits" to the standard setting.

6 Discussion and Conclusions

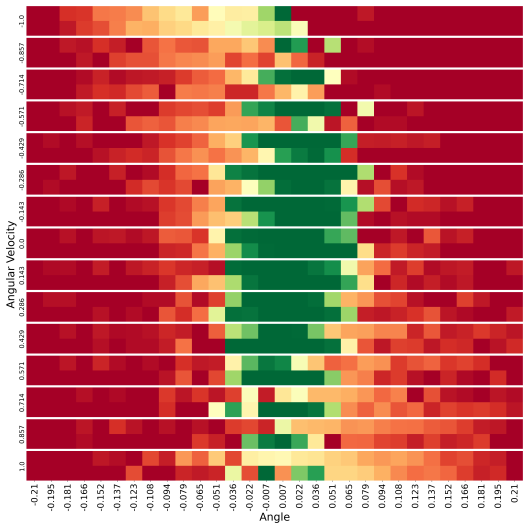
6.1 Examining Standard vs. Adversarially Learned Q-Tables

We expect the standard Q value table to differ from the adversarially perturbed Q value table in a systematic way. Particularly, since a perturbation shift a next-state $\widehat{s}^{t+1}$ to the adversarial next-state s^{t+1}, we expect the Q value to be more cognizant of the future rewards achieved under this adversarial state. To understand these differences we visualize the Q value table for the standard and adversarial setting.

Based on Figure 4 we can see that more state action cells $Q(s, a)$ exist that are non-zero, indicating that adversarial perturbation lead to more exploration during each episode roll out. This behavior likely appears during training since the agent is unsure which action to select and takes a step that places it at a disadvantage and it later has to recover from. This disadvantaged next-state acts as the state-action cell that has now been visited in the adversarial setting but wasn't in the standard case. Importantly it should be noted that the adversarial next-states are never directly added to the reply buffer. This aligns with the hypothesis posed by the original authors that adversarial perturbations encourage better exploration of the environment.

However even though adversarial training offers more exploration, in our experiments we see that it has no benefit in robustness for the tabular setting. This might be explained by the deviation in the Q-tables. Notice that in the standard setting, for regions of small angular velocity and small angle, the agent is ambivalent towards which action to take, however in the adversarial case the model is much more opinionated (significant difference in state-action value for left vs right action). Once the adversarial agent takes these decision in a hard environment (strong gravity, short pole length, heavy pole mass) it immediately ends up in a large angular velocity region (first / last row of table) where the action-values may not be as fully realized.

Q-Table for Standard Training on Standard Environment



Q-Table for Adversarial Training on Standard Environment

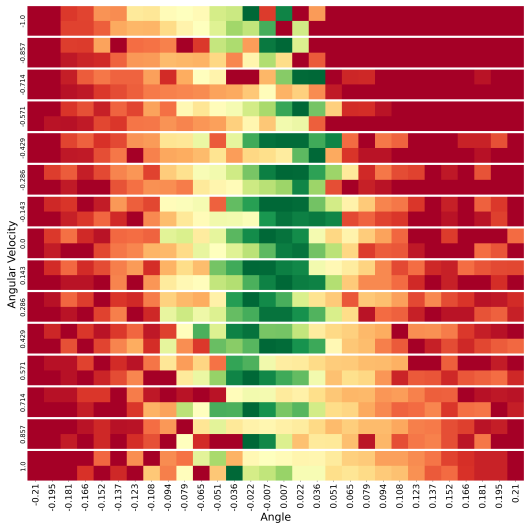


Figure 4: This is the Q value table showing the state action space. Each row represents angular velocity (moving from negative to positive as you move down) and each column represents an Angle (moving from negative to positive as you move right). Each top and bottom sub-rows of each row represents the action to move left and right, respectively. The left is the Q value table for an agent that is trained under the standard setting and on the right is an agent trained in the adversarial setting. Red corresponds to a value of 0 and as you move to Green the value increases.

This above discussion calls into question what are the correct adversarial states to even consider in the first place. The original method [20] was to find adversarial next states that minimizes the reward and are close to the initial next state. However it seems that some state perturbations have more priority and should be learned more robustly than others. To understand this, we trained an agent specifically in a hard environment to evaluate it's Q value table.

Based on the Figure 5 we can see that the agent in the hard environment has a much more narrowed region of ambivalence, and it doesn't ever visit states that have low probability of occurring in the environment. For example the agent on the right has almost no cell values for regions of high clockwise angular velocity and a pole angle in the negative direction (bottom left of plot), while the standard agent in the standard environment (Figure 4 left) does explore these state-action cases.

6.2 The Role of Adversarial Training in RL

Based on our results, we believe that adversarial training can improve the robustness of a reinforcement learning agent; however, it is not a silver bullet. Adversarial training comes with many caveats: it is much more computationally expensive to train, it converges more slowly, and as we observed in the DQN setting, it is possible that gains in robustness come at the expense of performance in the standard setting. Thus, we see adversarial training as an option that should be considered when alternatives have been exhausted.

For example, in section 5.2, we demonstrated how the final model obtained in standard Q-learning can widely vary in its robustness to environmental change, and that the stopping conditions can play a role in the robustness of the final model. Thus, we would recommend first examining the role of these factors, since these experiments tend to be quicker to conduct. However, in the DQN setting our empirical evidence clearly supports the hypothesis of adversarial training leading to greater robustness. Even in the tabular setting, although we did not empirically measure improvements in robustness, we can see from Section 6.1 that the learned Q-tables have learned "softer" Q-values, with more signal in the areas between good and bad states.

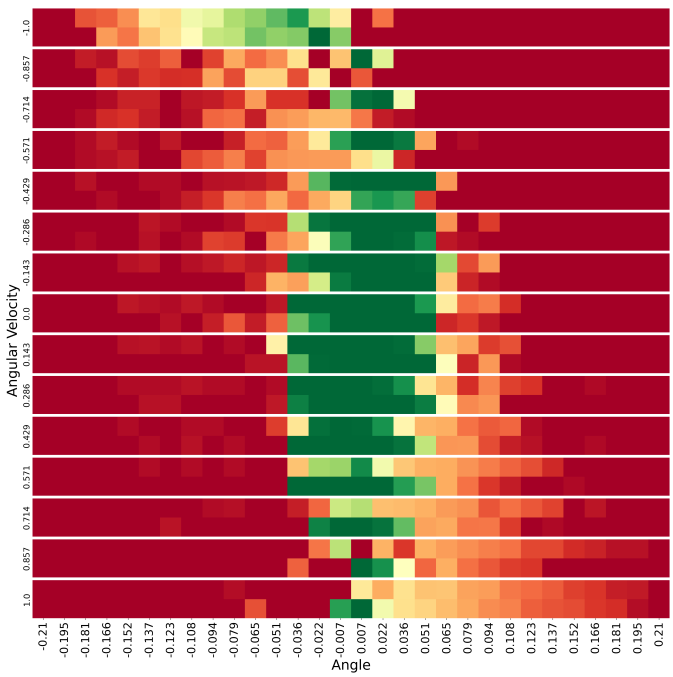


Figure 5: This is the Q value table showing the state action space for an agent that undergoes standard training in a (hard) adversarial environment: strong gravity, short pole length, heavy pole mass.

6.3 Future Work

In this paper we have only attempted to interpret adversarial states by leveraging and visualizing the Q-value tables and hypothesize there maybe a priority adversarial states that need to be hardened to achieve more robustness. This poses an interesting open area of future work that investigates which adversarial states are more important and thus the model needs to be pushed towards, and how the relationship between adversarial states depend on the final downstream new environment. A way of understanding this could be by tracing the trajectory of an episode to see which angle, and angular velocity states the agent reaches in an adversarial setting.

Code

All the code for our work can be found here: https://github.com/varun-tandon/CS329DFinal

References

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[2] Emma Brunskill. Lecture 5: Value function approximation - stanford university.

[3] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017.

[4] Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2023.

[5] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR, 2019.

[6] A Fawzi, M Balog, B Romera-Paredes, D Hassabis, and P Kohli. Discovering novel algorithms with alphatensor, 2022.

[7] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[8] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In *WOOT*, pages 15–15, 2017.

[9] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[10] Ezgi Korkmaz. Adversarial robust deep reinforcement learning requires redefining robustness. *arXiv preprint arXiv:2301.07487*, 2023.

[11] Tencent Keen Security Lab. Experimental security research of tesla autopilot. *Tencent Keen Security Lab*, 2019.

[12] John Banister Lanier, Stephen McAleer, Pierre Baldi, and Roy Fox. Feasible adversarial robust reinforcement learning for underspecified environments. *arXiv preprint arXiv:2207.09597*, 2022.

[13] Björn Lütjens, Michael Everett, and Jonathan P. How. Certified adversarial robustness for deep reinforcement learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1328–1337. PMLR, 30 Oct–01 Nov 2020.

[14] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

[15] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving rubik's cube with a robot hand, 2019.

[16] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against deep learning systems using adversarial examples. *arXiv preprint arXiv:1602.02697*, 1(2):3, 2016.

[17] Darsh Parekh, Nishi Poddar, Aakash Rajpurkar, Manisha Chahal, Neeraj Kumar, Gyanendra Prasad Joshi, and Woong Cho. A review on autonomous vehicles: Progress, methods and challenges. *Electronics*, 11(14):2162, Jul 2022.

[18] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2817–2826. PMLR, 06–11 Aug 2017.

[19] David Silver, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016.

[20] Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training, 2020.

[21] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

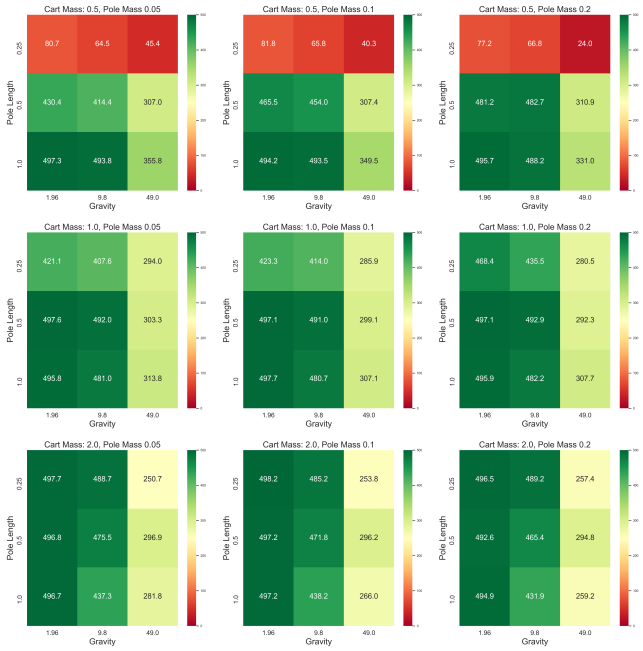
[22] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning, 2015.

7 Appendix

7.1 Full Environmental Range

Here we include results which show how the tabular Q-learning agent performs with different stopping criteria.

Mean Episode Length - Stop 5



Standard Deviation Episode Length - Stop 5

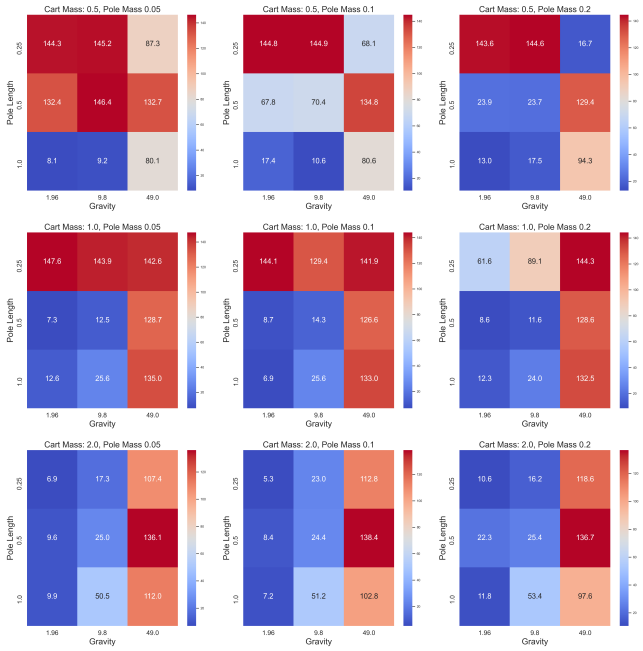


Figure 6: Mean (left) and Standard Deviation (right) episode length in Cartpole under various environment perturbations with a stopping condition of 5 consecutive runs reaching the maximum episode length of 500.

Mean Episode Length - Stop 10

Standard Deviation Episode Length - Stop 10

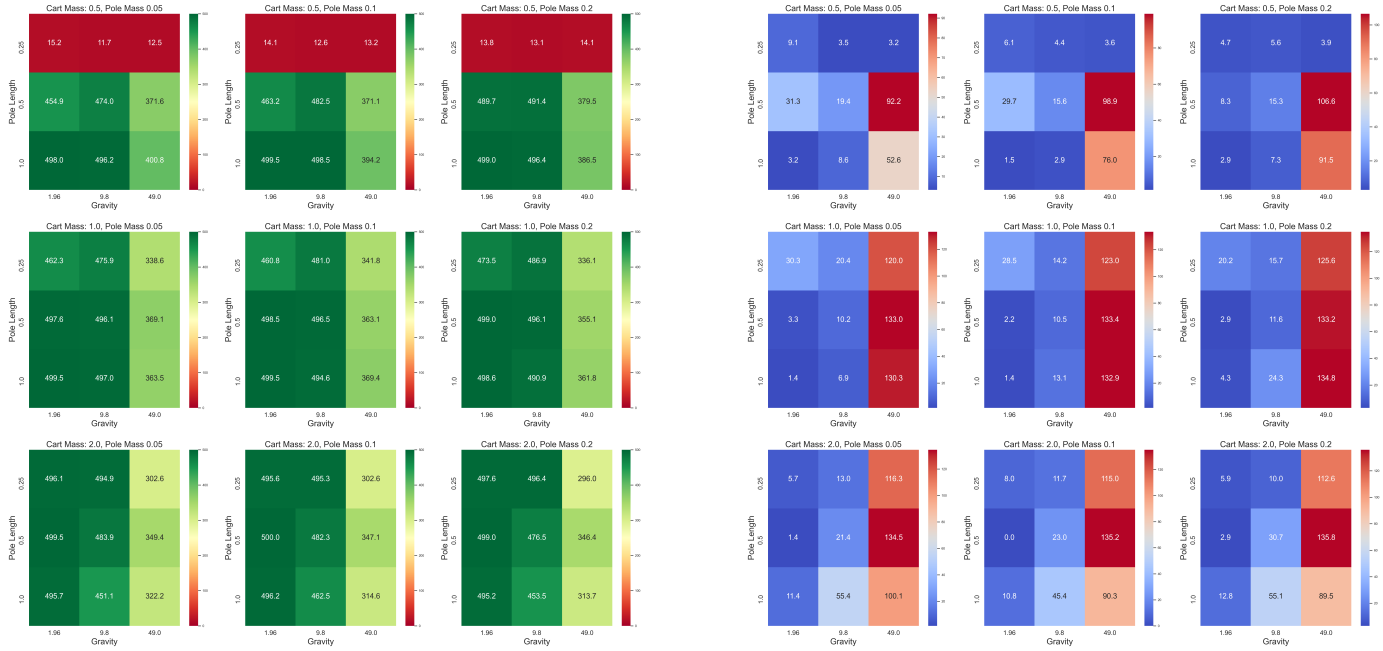


Figure 7: Mean (left) and Standard Deviation (right) episode length in Cartpole under various environment perturbations with a stopping condition of 10 consecutive runs reaching the maximum episode length of 500.

Mean Episode Length - Stop 20

Standard Deviation Episode Length - Stop 20

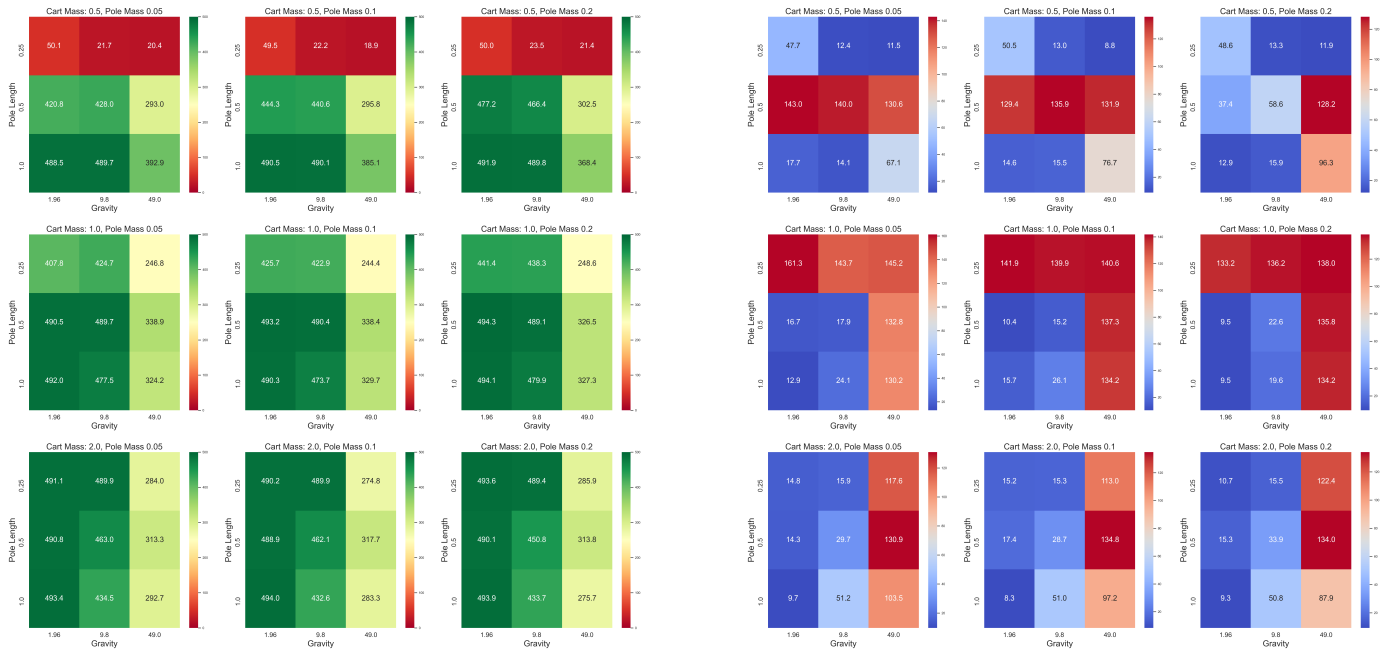


Figure 8: Mean (left) and Standard Deviation (right) episode length in Cartpole under various environment perturbations with a stopping condition of 20 consecutive runs reaching the maximum episode length of 500.

We see that the stopping condition of 10 consecutive runs of that reach the maximum episode length yield the smallest standard deviation (most robust) result.