ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Artificial Neural Networks/Reinforcement Learning (CS-456)

Deep Deterministic Policy Gradient

Daniil Bobrovskii - 346134
Nicolaj Schmid - 269552

Spring 2023

Contents

1 Introduction

Reinforcement learning (RL) is a computational framework that enables an agent to learn optimal actions through iterative interactions with an environment. Traditionally, RL has primarily focused on problems with finite state and action spaces, whereas deep RL overcomes these limitations by leveraging the capabilities of deep neural networks. While the Deep Q-Network (DQN) agent utilizes deep neural networks to approximate the Q-function in continuous state spaces [1], the Deep Deterministic Policy Gradient (DDPG) algorithm further extends RL to continuous action spaces [2]. DDPG adopts an actor-critic architecture, with the critic network estimating Q values and the actor network selecting actions accordingly. This project aims to develop a DDPG implementation from scratch and employ it to stabilize an inverted pendulum. The implementation is based on the *Gym Pendulum-v1* environment provided by OpenAI [3].

2 Random Policy

First, we have implemented an agent which chooses a random torque between -2 and 2 at each step. The average cumulative reward over 100 episodes is -1227.5 with a standard deviation of 290.8 (average cumulative rewards of this and all further developed agents are reported in Table 1).

3 Heuristic Policy

The heuristic policy is implemented with a hand-crafted actor which applies a constant torque in one of the two possible directions depending on the position of the pendulum. The average cumulative reward over 1000 episodes is shown in figure 1 for different constant torques. For low torques, the performance is comparable to the random actor. When applying higher torques, the results are getting better and reaching the cumulative rewards above -450, and a slight non-monotonicity can be attributed to random discrepancies between simulations. Therefore, we consider $|\text{torque}| = 2.0$ to be the best constant torque for the heuristic policy. This may be seen as a baseline and any RL algorithm should surpass this performance.

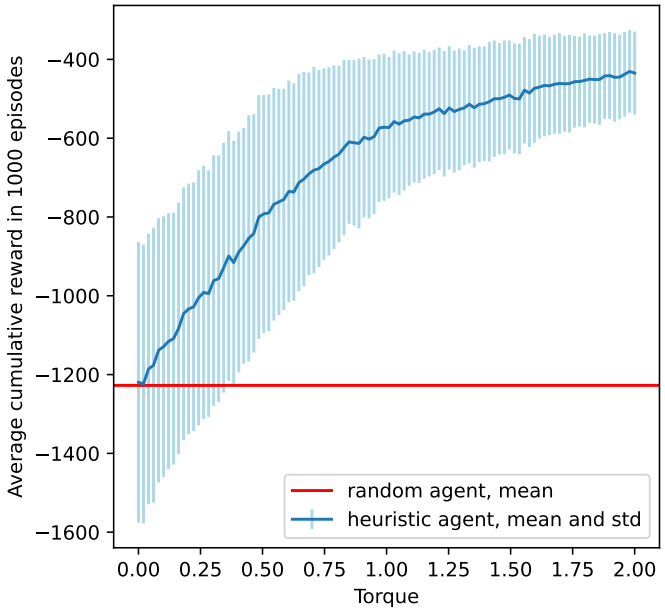


Figure 1. Average cumulative reward for random and heuristic agents.

4 Q Values

The optimal heuristic policy, as discussed in Chapter 3, was employed, and the Q values were learned using a critic neural network. During the training process, the training loss initially exhibits a significant increase because the critic network must adapt its values to match the expected reward (see Figure 2). Subsequently, the model converges rapidly after approximately 200 episodes.

The resulting Q values are illustrated in Figure 3 for various torques and angular velocities (by convention, positive angular velocity corresponds to counter-clockwise motion; on the contrary, positive torque is considered to be clockwise in our report). Figure 3a presents the Q values before training, displaying a completely random distribution. Conversely, Figure 3b demonstrates the Q values after training. Notably, positions around 0° are deemed advantageous when the velocity is zero, aligning with our expectations. The influence of torque

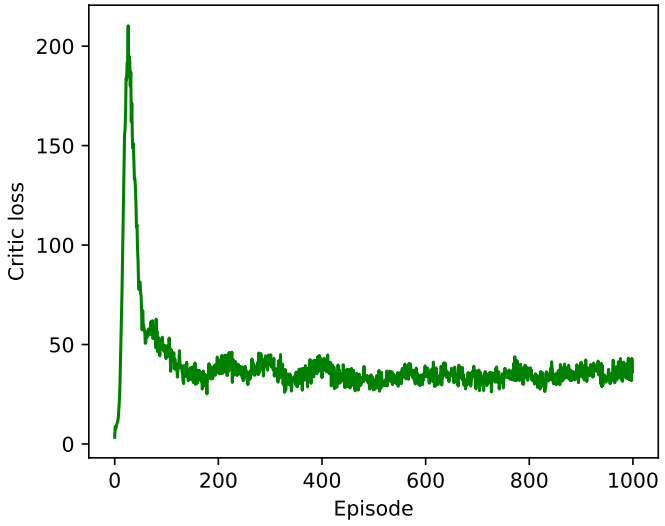
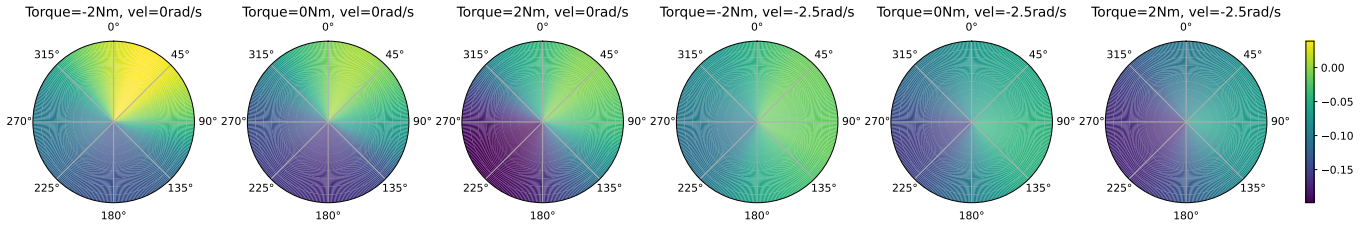
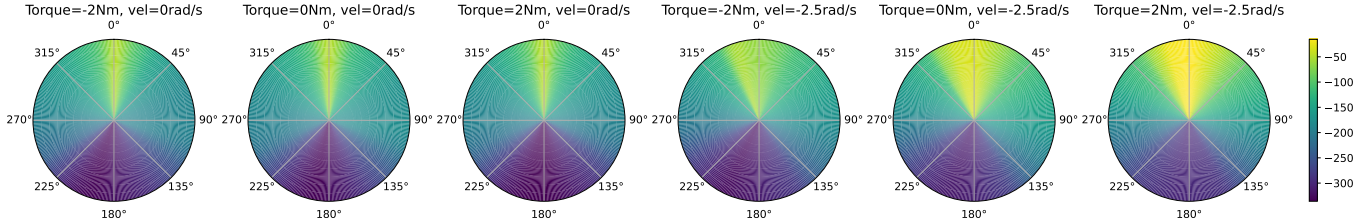


Figure 2. Loss when learning the Q-values



(a) Before training.



(b) After training.

Figure 3. Q values learned by the Q-network with heuristic agent.

on the predicted Q values is minimal, which can be attributed, in part, to the limited options available to our heuristic agent, which did not have an option not to apply any torque or to apply smaller torques. Consequently, the Q-network was unable to learn complex dependencies under these conditions.

When the velocity is non-zero, higher Q values are predicted for angles shifted in the opposite direction of the velocity. In this case, the torque has some effect, with the highest Q values observed when torque is applied in the direction of the velocity. However, interpreting this result proves challenging, and a more detailed analysis of the heatmaps has been left for the further developed models, which could explore different torques as opposed to the current heuristic agent.

5 Minimal DDPG

Next step in building the DDPG model is adding an actor network, which learns an optimal policy. For this model, we observed that training was quite unstable and the resulting performance varied a lot across multiple runs with different random seeds. Two of such runs are shown in Figure 4. Run 1 has converged and reached the average cumulative reward of -154.4 across 100 test episodes, which is reported in Table 1. However, some instability is still observed during the training, with actor losses becoming negative, which means that the critic started predicting positive rewards, which in fact is not possible. On the other hand, run 0 has not converged and the average cumulative reward on test was equal -1057.9.

Q values learned by the critic in run 1 are shown in Figure 5. Similarly to the predictions of the Q-network from the previous chapter, the highest Q values are centered around 0° for zero velocities and shifted in the expected direction for non-zero velocities. What is more, even after replacing the heuristic policy with the actor network, the Q values depend much more on the current velocity than on the applied torque.

Model	Reward
Random agent	-1227.5
Heuristic agent, $\|\text{torque}\| = 2.0$	-434.7
Minimal DDPG	-154.4
DDPG, $\tau = 1.0$	-208.5
DDPG, $\tau = 0.5$	-957.6
DDPG, $\tau = 0.1$	-152.0
DDPG, $\tau = 0.05$	-154.9
DDPG, $\tau = 0.01$	**-149.0**
OU noise DDPG, $\tau = 0.01$, $\theta = 1.0$	**-145.3**
OU noise DDPG, $\tau = 0.01$, $\theta = 0.75$	-163.3
OU noise DDPG, $\tau = 0.01$, $\theta = 0.5$	-225.1
OU noise DDPG, $\tau = 0.01$, $\theta = 0.25$	-173.2
OU noise DDPG, $\tau = 0.01$, $\theta = 0.0$	-206.7

Table 1. Average cumulative reward in 100 episodes obtained by trained agents (lowest reward values in bold); note that DDPG with $\tau = 1.0$ and minimal DDPG, as well as DDPG with $\tau = 0.01$ and OU noise DDPG with $\tau = 0.01$, $\theta = 1.0$, correspond to identical models.

To investigate this in more detail, we plotted the difference between the Q-values for torques of -2.0 or 2.0 and the Q-values for a torque of 0.0 (Figure 6). For zero velocities, the predicted Q-values are higher for clockwise angles when a negative torque is applied, as compared to zero torque. Conversely, they are lower for the same angle when a positive torque is applied. While this behavior is logically consistent, it is notable that a symmetrical pattern is not observed for counter-clockwise shifts (compare to Figure 9, which will be discussed later). The same observation can be made for non-zero velocities: the sign of the observed changes is logically consistent, but the magnitude of the differences for various angles is not as symmetrical as one could expect.

Figure 4. Learning curves of the minimal DDPG model over 1000 episodes for two runs with different random seeds; for the cumulative reward, a moving average over 30 episodes is plotted.

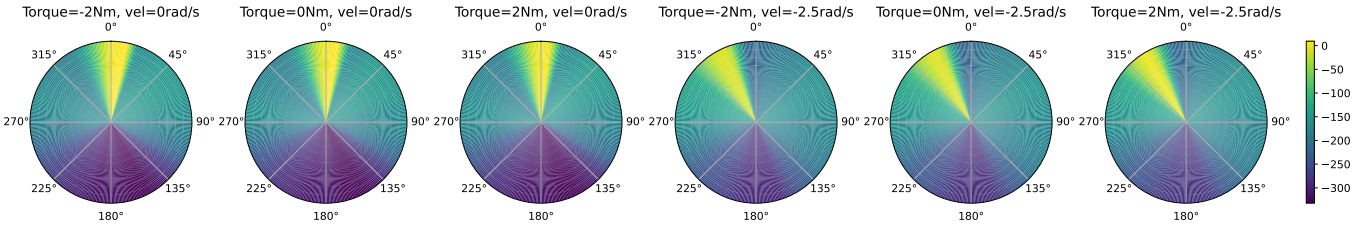


Figure 5. Q values learned by the minimal DDPG agent.

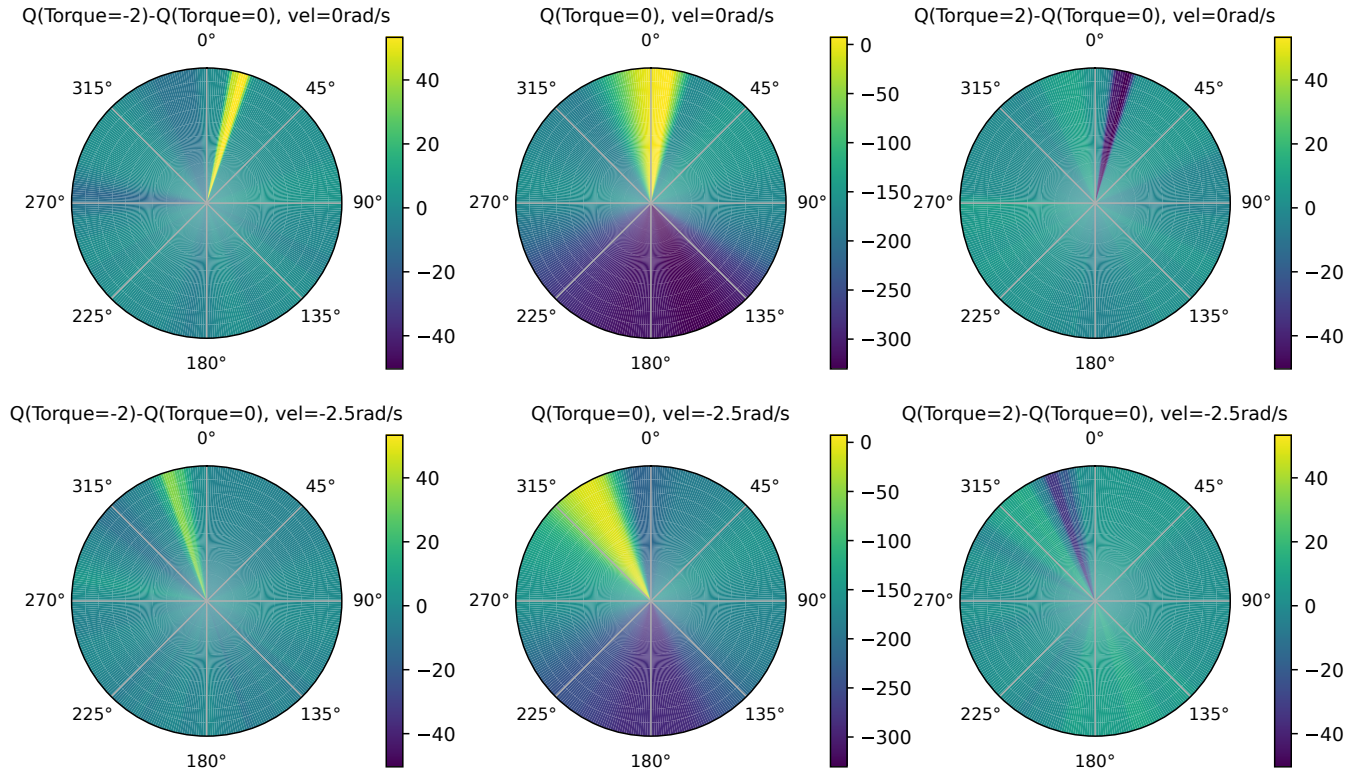


Figure 6. Q values learned by the minimal DDPG agent (left: difference between the Q values for torque $= -2.0$ and torque $= 0.0$; middle: Q values for torque $= 0.0$; right: difference between the Q values for torque $= 2.0$ and torque $= 0.0$).

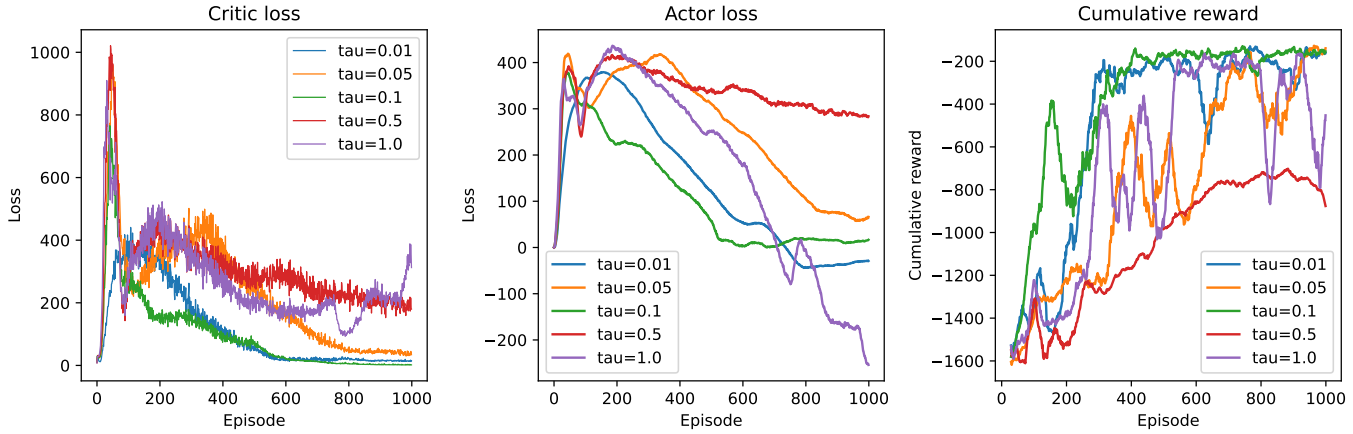


Figure 7. Learning curves of the DDPG with target networks over 1000 episodes for different values of τ; for the cumulative reward, a moving average over 30 episodes is plotted.

6 Target networks

To enhance the stability of the training process, we implemented two target networks, one for the critic and one for the actor, which are used to compute the target Q value in the critic loss. The parameters of these target networks were updated at a slower rate compared to the parameters of the trained networks, utilizing a soft update mechanism parameterized by τ. A value of $\tau = 1.0$ makes the model identical to the one without target networks, while $\tau = 0.0$ would have resulted in the target networks retaining their initial weights throughout the training process.

Figure 7 depicts the learning curves of the DDPG algorithm with target networks, showcasing various values of τ. Additionally, the performance of the models across test episodes is presented in Table 1. Generally, it was observed that lower values of τ contributed to more stable training, as expected. The highest average cumulative reward achieved was -149.0 with $\tau = 0.01$; however, it only exhibited a marginal improvement over the reward of -154.4 attained initially with one of the runs using the minimal DDPG model. Therefore, the model demonstrated the ability to converge and achieve relatively high performance even in the absence of target networks, while the primary advantage of employing target networks was a more assured outcome due to the stabilization of the learning process. However, in some of the runs we performed, even the model with $\tau = 0.05$ failed to converge consistently. It should be noted that to establish a valid comparison, multiple runs of each model should be conducted, and the resulting performance distribution should be thoroughly examined. Unfortunately, due to limited computational resources, conducting such a comprehensive analysis involving numerous model training runs was not feasible in our case.

The Q-values learned by the best model with $\tau = 0.01$ exhibited substantial similarity to those learned by the minimal DDPG agent. Furthermore, the difference between Q-values for different torques displayed similar symmetry issues. For brevity, we do not include the heatmaps depicting these results.

7 Ornstein-Uhlenbeck noise

Finally, we implemented an Ornstein-Uhlenbeck (OU) noise as an alternative to the uncorrelated Gaussian noise used in the training of the previously discussed models. The OU noise introduces correlated perturbations to successive actions, theoretically enabling more efficient exploration by the agent. The noise is parameterized by θ, where $\theta = 1.0$ makes the model indistinguishable from the one employing uncorrelated noise, while $\theta = 0.0$ results in the highest degree of noise correlation.

We trained the DDPG with OU noise and target networks, employing a τ value of 0.01, and investigating different θ values. The corresponding training curves are presented in Figure 8, while the performance across test episodes is reported in Table 1. Overall, we did not observe consistent improvements in terms of training stability or test performance when employing OU noise. While OU noise was initially proposed by the authors of DDPG [2], the lack of significant gains compared to uncorrelated noise aligns with the findings of other studies concerning deep RL for continuous control [4, 5].

However, it is important to note that OU noise did not result in performance degradation. As the training outcomes vary across different runs, the rewards listed in Table 1 represent only one set of possible results, corresponding to the runs for which the training curves are plotted in Figure 8. Notably, the highest test performance across all the models was observed in a different run of the OU noise model with $\theta = 0.0$, yielding an average cumulative reward of -144.7. Furthermore, even in the run with $\theta = 0.0$ that yielded lower performance of -206.7, the heatmap of the difference in Q-values exhibited the symmetry lacking in both the minimal DDPG and the DDPG with target networks (Figure 9).

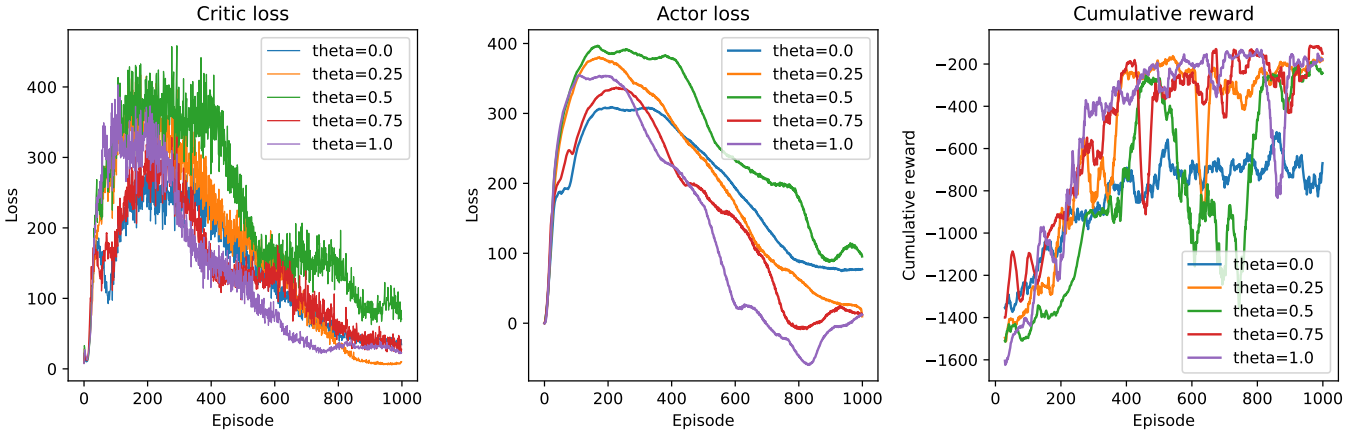


Figure 8. Learning curves of the DDPG with target networks ($\tau = 0.01$) and OU noise over 1000 episodes for different values of θ; for the cumulative reward, a moving average over 30 episodes is plotted.

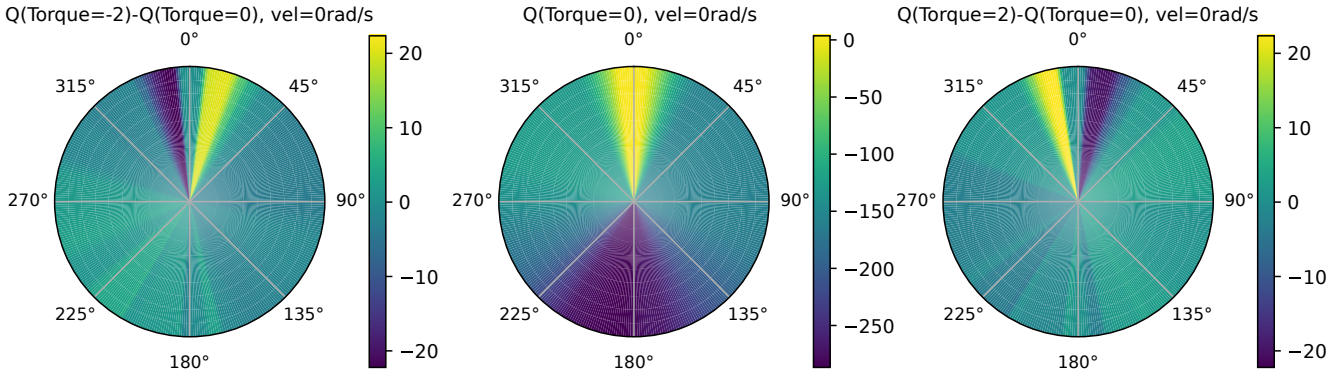


Figure 9. Q values learned by the DDPG with target networks ($\tau = 0.01$) and OU noise ($\theta = 0.0$) (similar to Figure 6).

This observation may still be attributed to the random discrepancies between runs, but an alternative explanation could be that lower θ values, indeed, result in improved exploration of states, leading to more consistent estimation of Q-values even if does not improve the resulting performance of the agent.

All in all, similar to our observations regarding different τ values, a comprehensive investigation of the impact of OU noise on training should encompass multiple runs for each model and an analysis of the performance distribution.

8 Conclusion

In this project we developed a DDPG implementation for stabilizing an inverted pendulum. The heuristic policy provided a baseline, while the DDPG model exhibited unstable training and varying performance across runs. The addition of target networks improved convergence and achieved slightly higher performance. The use of Ornstein-Uhlenbeck noise did not yield consistent improvements compared to uncorrelated noise. Overall, the DDPG model showed promise in stabilizing the pendulum, being able to achieve reasonable performance compared to the heuristic agent. Further analysis with multiple runs is needed to better understand the models' behavior and potential improvements.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[2] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv:1509.02971*, 2019.

[3] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv:1606.01540*, 2016.

[4] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*, pp. 1587–1596, PMLR, 2018.

[5] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. Tb, A. Muldal, N. Heess, and T. Lillicrap, "Distributed distributional deterministic policy gradients," *arXiv:1804.08617*, 2018.