Lab 4: Reinforcement Learning

ARI 510

Yola Charara

This assignment solved two reinforcement learning environments using a Deep Q-Network: CartPole-v1 and LunarLander-v3. Training such agents was really challenging, with the main difficulties arising in reaching stable performance. In the case of CartPole-v1, the agent was not able to keep the pole balanced during the first steps because the transitions between exploration and exploitation were unstable. In the case of LunarLander-v3, the dynamics are even more complex, with sparser rewards, making careful tuning necessary for successful landings. By improving the process of training, including main hyperparameter tuning and important code improvements, better results were achieved. In this regard, for CartPole-v1, it was possible to increase the average reward in the last 50 episodes from 20 to 200, resulting in a stable performance captured in a 4-second video. For LunarLander-v3, the average reward in the last 50 episodes increased from -200 to 200, with the spacecraft landing in a 7-second video (Figures 1 and 2 below provide a graph visualizing the performance comparison of the two environments before and after code improvements) . It can be observed that iterative tuning with a careful eye on performance provided improvements.

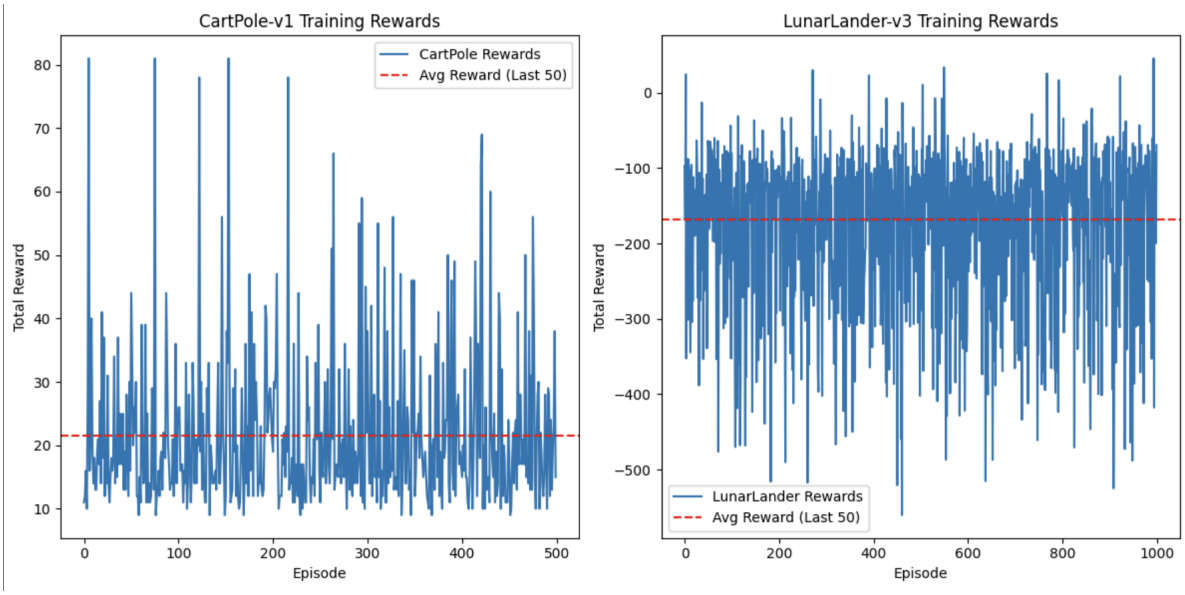

Figure 1: CartPole-v1 and LunarLander-v3 training performance before code improvements.

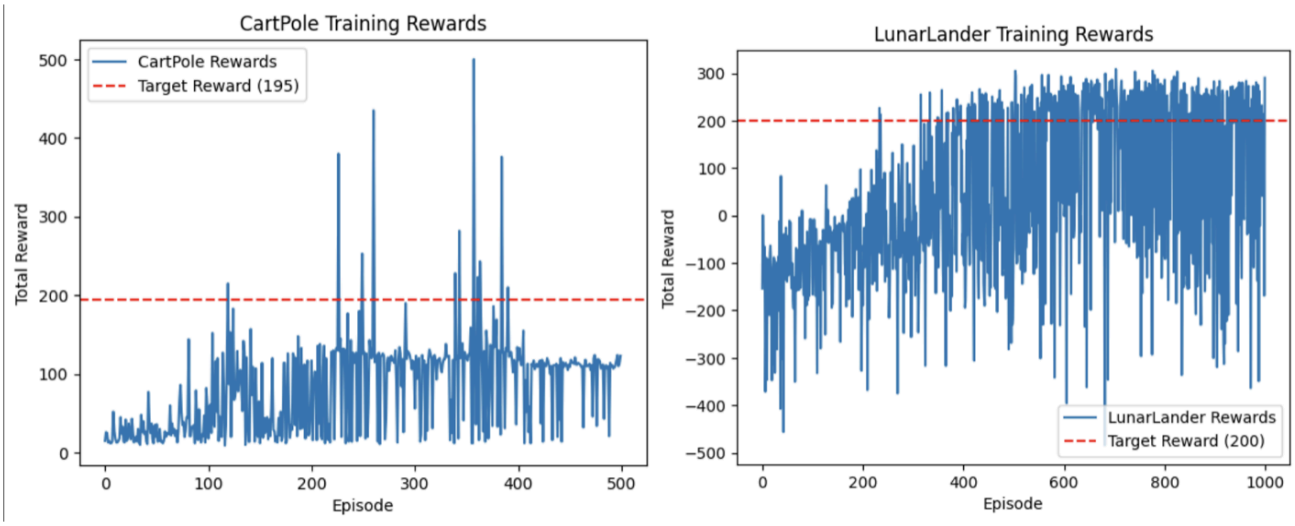

Figure 2: CartPole-v1 and LunarLander-v3 training performance after code improvements.

Key hyperparameters were tuned to obtain optimized agent learning. Both the environment settings implemented use a discount factor γ of 0.99 for prioritizing long-term rewards and a learning rate of 0.001 to get stable convergence. The epsilon-greedy strategy was used by taking $\varepsilon = 1.0$ and decaying by a factor of 0.995 down to 0.01, providing a good balance the agent needed in exploration and exploitation. The size of the replay buffer and the batch size were tuned appropriately in order to provide sufficient diversity for training. Updating the target network at each 10 episodes supplied the necessary stability for learning. These adjustments were very necessary in improving performance, especially in LunarLander-v3, where the ability of the agent to learn precise control and navigate through complex environments depended on these refinements.

The difference between the two environments is evident from the level of complexity each presents and how DQNs scale. CartPole-v1 is relatively simple and requires the agent to balance a pole by applying forces, hence faster convergence and more stable rewards. The agent reached a reward average of 200 and sustained it for the last episodes. However, LunarLander-v3 was considerably a harder task since the agent needed to fly and land a spacecraft under different conditions. The agent initially received unstable rewards, but after several adjustments, it managed to stabilize and exceed the threshold of 200 rewards in the final episodes. While CartPole had given a pretty good baseline understanding of DQN's behavior, LunarLander showed how the algorithm is capable of handling more complex scenes. Improved performance in both these environments indicates that iteration of adjustments is important in practice; even minor changes in code and parameter tuning can yield drastically better performance.