HW4: Q-Learning and DQN

Atharv Kulkarni - u1322897

February 25, 2025

---

1 Tabular Q-Learning to Win Big at Blackjack

The results show that the Q-Learning agent's average reward converges to about -0.084, whereas the random policy remains around -0.346. While both are negative, which is typical for Blackjack due to the house advantage, the Q-Learning agent clearly performs significantly better than random play. The learning curve shows some fluctuation due to the noisy environment like Blackjack, but overall there is a clear upward trend, indicating that the agent gradually refines its strategy to reduce losses and occasionally win. A small negative reward in Blackjack is expected, so the near break-even outcome suggests that the learned policy is making good decisions (for example, when to hit or stick) and is able to outperform a purely random strategy by a good margin.

Policy	**Average Reward (1000 episodes)**
Learned Q-Learning Policy	-0.084
Random Policy	-0.346

Table 1: Comparison of Q-Learning and Random Policy in Blackjack

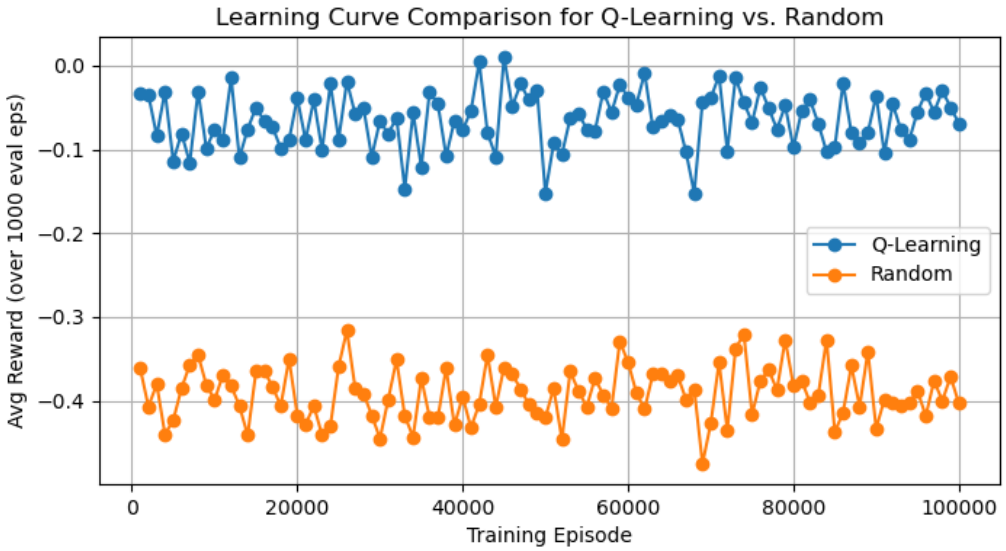

Figure 1: Learning Curve Comparison for Q-Learning vs. Random Policy.

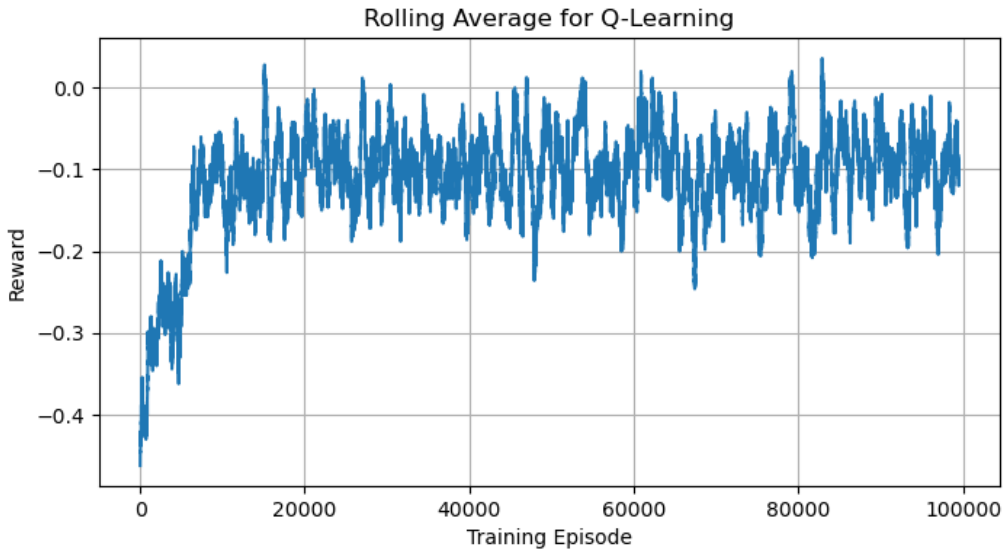

Figure 2: Rolling Average of Training Rewards for Q-Learning.

1.1 Hyperparameters Used in Training Blackjack

Hyperparameter	**Value**
Number of Episodes (n_{episodes})	100,000
Learning Rate	0.01
Initial Epsilon ($\epsilon_{\text{initial}}$)	1.0
Epsilon Decay (ϵ_{decay})	1×10^{-4}
Final Epsilon (ϵ_{final})	0.1
Discount Factor (γ)	0.95

Table 2: Hyperparameters used for training the Blackjack Agent.

2 Landing on the Moon using DQN

The training progression of the Deep Q-Network (DQN) agent is evident from the plotted episode duration and reward graphs. Initially, during the first few episodes, the agent performed poorly as it was making random actions, which resulted in frequent crashes and inefficient movements. Since the exploration strategy (ϵ-greedy with $\epsilon_{\text{start}} = 0.9$) encouraged high exploration in early episodes, the agent had no understanding of landing mechanics and was simply trying different actions without any structured goal.

As training progressed, the agent realized that staying in the air longer resulted in higher cumulative rewards due to the small per-step reward. This led to a phase around episode 120 where the agent started hovering for most of the episode instead of attempting to land. This behavior is reflected in the increased episode durations around this period, as the agent was exploiting the reward mechanism rather than focusing on landing.

However, after sufficient experience replay and Q-value updates, the agent discovered that soft landings result in even greater rewards. From episode 150 onwards, we see a gradual reduction in episode duration, while the reward trend continues to increase. This transition signifies that the agent has learned that landing earlier and smoothly yields better rewards than simply hovering for an extended period. The final training phase, around episode 400-500, shows a consistent increase in reward values, with the moving average stabilizing around 185. This suggests that the agent has successfully optimized its landing strategy to achieve high rewards.

The learning curve also gave a good insight that, longer flight durations do not necessarily mean better performance. Early on, the agent mistook duration for a good strategy but later refined its policy to prioritize safe and smooth landings rather than just staying in the air.

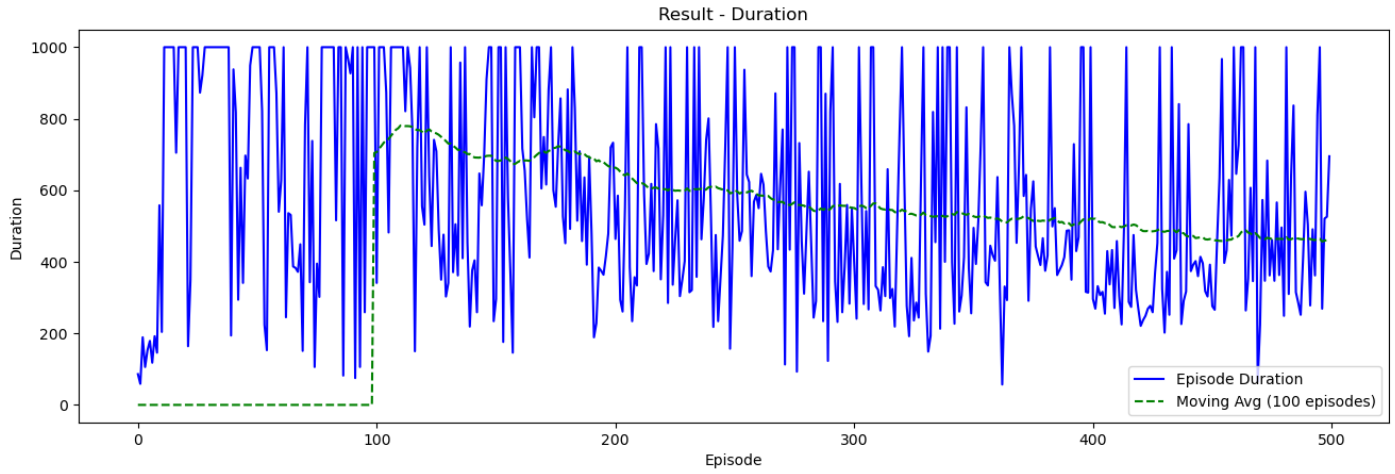

Figure 3: Duration plot of the Lunar Lander agent during training.

Figure 4: Reward plot of the Lunar Lander agent during training.

2.1 Performance Comparison: Learned Policy vs. Human and Random

To further validate the trained model's performance, we compare it against a human-controlled policy and a random policy in terms of average reward and average episode duration.

Policy	Average Reward	Average Duration
Random Policy	-190.28	91.25
Human Policy	-109.32	N/A
Learned Policy	**185.30**	**597.57**

Table 3: Performance comparison between human, learned, and random policies.

The random policy performs the worst, achieving an average reward of -190 and surviving for an average of only 91 steps before either crashing or going out of bounds. Since it takes actions arbitrarily, it neither stays airborne long enough nor learns to land properly.

The human policy, while slightly better, still performs worse than the trained agent. With an average reward of -109, it suggests that landing the lander optimally and precisely is hard. A human player might struggle with controlling thrust precisely or maintaining balance, leading to hard landings and negative rewards.

The DQN-trained agent significantly outperforms both. With an average reward of 185.30 and an average episode duration of 597 steps, it has effectively learned an optimal balance between controlled descent and smooth landings. This confirms that the reinforcement learning framework successfully optimized the lander's behavior beyond both random exploration and human intuition.

2.2 Hyperparameters Used in Training Lunar Lander

Hyperparameter	Value
Batch Size	128
Discount Factor (γ)	0.99
Initial Exploration Rate (ϵ_{start})	0.9
Final Exploration Rate (ϵ_{end})	0.05
Exploration Decay (ϵ_{decay})	1000
Target Network Update Rate (τ)	0.005
Learning Rate (LR)	1×10^{-4}
Experience Replay Memory	10,000

Table 4: DQN hyperparameters used during training.

2.3 Final Trained Model Location

The final trained model is available at:
Trained_Lunar_Lander/lunar_lander_dqn_v3.pth