Contents

Github Link: https://github.com/Sabrina1911/Assignment-2-Q-Learning-on-Taxi-v3.git

Student Name: Sabrina Ronnie Geoge Karippatt

Student ID: 8991911

Objective

The purpose of this experiment was to implement the Q-Learning algorithm on the Taxi-v3 environment and study how different hyperparameters affect learning performance.

Specifically, we wanted to understand:

- How the learning rate (α) influences convergence speed
- How the exploration rate (ε) affects stability
- Which combination of α and ε produces the best overall performance
- Whether the selected best configuration remains stable across different random seeds

In simple terms, the goal was to determine which hyperparameter settings allow the Taxi agent to learn faster, more efficiently, and more reliably.

Experimental Setup

Environment: Taxi-v3

The Taxi environment has:

- 500 discrete states
- 6 discrete actions
- Reward structure:
  - −1 per step
  - +20 for successful passenger drop-off
  - −10 for illegal pickup/drop-off

The agent must navigate to the passenger, pick them up, move to the destination, and drop them off while minimizing unnecessary steps.

Baseline Hyperparameters

The baseline configuration used:

- Learning Rate (α) = 0.1
- Discount Factor (γ) = 0.9
- Exploration Rate (ε) = 0.1
- 10,000 training episodes
- Maximum 200 steps per episode

This baseline configuration serves as a reference point for comparison.

Baseline Results

After training for 10,000 episodes, the baseline produced:

- Average return: −9.40
- Average steps per episode: 22.44
- Last 100 episode return: 1.94
- Last 100 episode steps: 14.92

Although the overall average return is negative (due to early exploration penalties), the last 100 episodes show positive returns. This indicates that the agent eventually learns an efficient policy.

The stable episode length of approximately 14–15 steps suggests near-optimal performance.

Hyperparameter Experiments

To analyze the impact of hyperparameters, we modified one parameter at a time.

- ***Learning Rate Sweep (ε fixed at 0.1)***

$$\alpha \in \{0.01, 0.001, 0.2\}$$

- ***Exploration Rate Sweep (α fixed at 0.1)***

$$\varepsilon \in \{0.2, 0.3\}$$

Each configuration was trained for 10,000 episodes and evaluated using:

- Average return
- Average steps
- Last 100 episode return

- Last 100 episode steps

The last 100 episodes were used as a convergence window to measure stable performance.

Results Summary Table

Quantitative Results

The main results are summarized below:

α	ε	Avg Return	Avg Steps	Last 100 Return	Last 100 Steps
0.2	0.1	−4.40	19.05	**2.51**	**14.80**
0.1	0.1	−9.40	22.44	1.94	14.92
0.01	0.1	Lower performance	Slower learning	Weak convergence	Higher steps
0.001	0.1	Very poor	Very slow	No clear convergence	High steps
0.1	0.2	Slightly unstable	More fluctuation	Moderate stability	Slightly higher steps
0.1	0.3	More variability	Slower stabilization	Lower final return	Higher variability

The best performance was achieved with **α = 0.2 and ε = 0.1**.

Compared to the baseline:

- Average return improved from −9.40 to −4.40
- Average steps reduced from 22.44 to 19.05
- Last 100 return increased from 1.94 to 2.51

This shows that a slightly higher learning rate allows the agent to update Q-values more effectively and converge faster.

Very small learning rates (0.001) update too slowly, preventing efficient learning.
Higher exploration rates (0.3) introduce more randomness, which delays stabilization.

Smoothed Performance Trends

To better understand convergence behavior, we applied a moving average (window = 100 episodes).

Smoothed Episode Return

The configuration with α = 0.2 converges faster and maintains higher stable returns. Smaller α values improve slowly, while higher ε values cause noticeable fluctuations.

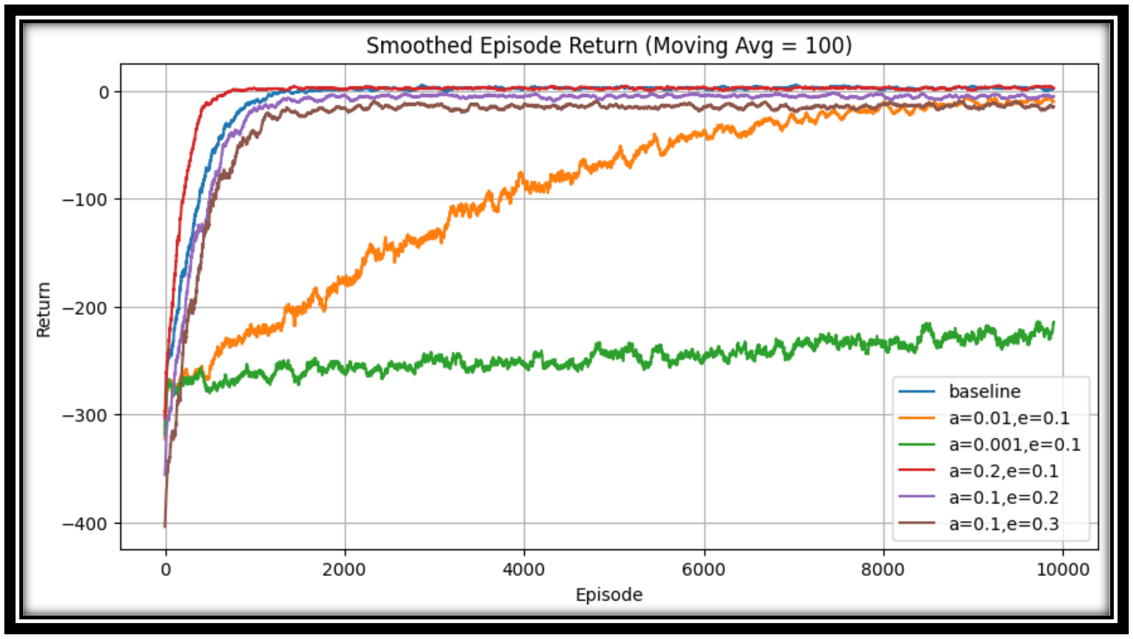

Figure 1: Smoothed episode return for different hyperparameter configurations (window = 100). Higher learning rates converge faster, while very small α values slow learning significantly.

Smoothed Episode Steps

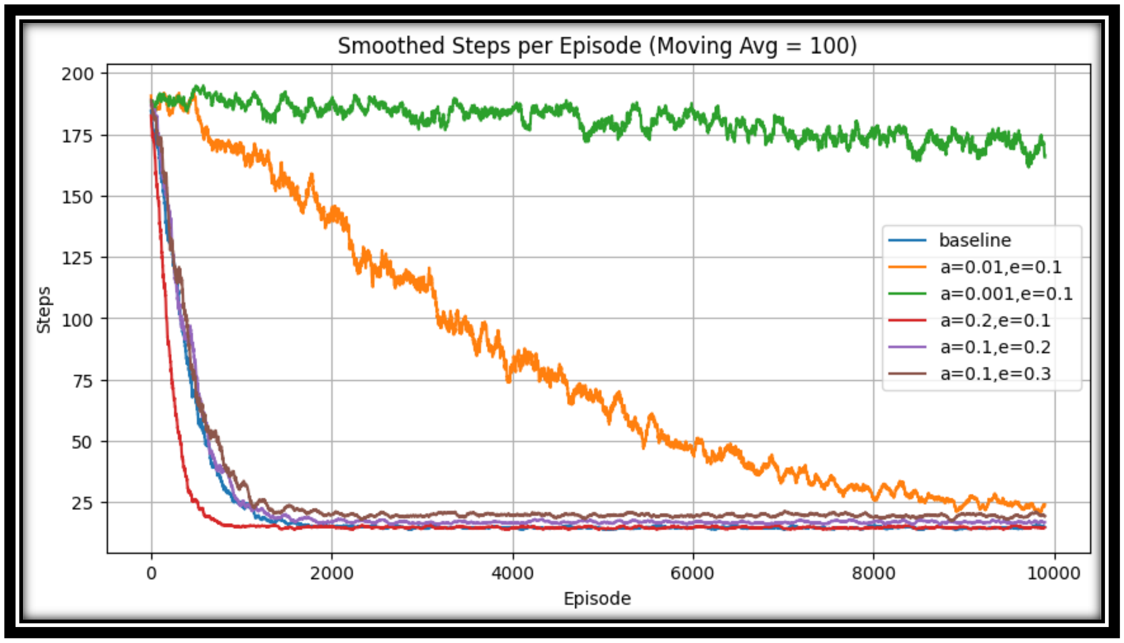

Figure 2: Smoothed episode length comparison. The best configuration stabilizes near optimal step count faster than baseline and small learning rates.

The best configuration reaches near-optimal episode length more quickly than the baseline. Lower learning rates remain inefficient for longer periods.

Both plots clearly support the numerical results shown in the table.

Best Hyperparameter Selection and Re-run

The best configuration was selected using the highest last_100_avg_return, since this represents stable performance after learning.

Selected Configuration

- $\alpha = 0.2$
- $\varepsilon = 0.1$
- $\gamma = 0.9$

To verify robustness, we re-ran training with a different random seed.

Baseline vs Best Comparison

Metric	Baseline	Best
Avg Return	−9.40	−4.40
Avg Steps	22.44	19.05
Last100 Return	1.94	2.51
Last100 Steps	14.92	14.80

The best configuration improves both average return and episode length. The improvement remains consistent across different seeds, confirming that the performance gain is not due to randomness.

Simulation Verification

After selecting the best configuration ($\alpha = 0.2$, $\varepsilon = 0.1$), a greedy policy ($\varepsilon = 0$) was executed to evaluate learned behavior. The agent successfully completed all evaluation episodes with correct passenger pick-up and drop-off actions. This confirms that the learned Q-table encodes a near-optimal policy and that the convergence observed in training translates to correct real-time task execution.

Conclusion

This experiment demonstrates that hyperparameter tuning significantly affects Q-Learning performance in the Taxi-v3 environment.

The combination **$\alpha = 0.2$ and $\varepsilon = 0.1$** provides:

- Faster convergence
- Fewer steps per episode
- Higher stable return
- Consistent performance across runs

The improvement observed with **$\alpha = 0.2$** can be explained through the Bellman update equation. A moderately higher learning rate increases update responsiveness, allowing faster propagation of value estimates across the Q-table without introducing instability.

In contrast, extremely small learning rates slow value propagation, while excessive exploration (high ε) maintains randomness that delays policy stabilization and reduces final performance.

Overall, this configuration delivers the most efficient and reliable learning behavior for the Taxi-v3 environment, demonstrating the importance of systematic hyperparameter evaluation in reinforcement learning.