Eris Leksi – 9067882

10/16/2025

CSCN8020 - Reinforcement Learning

Assignment 2

Assignment Report

1. Introduction

Purpose: Implement and evaluate a Q-Learning agent in the Taxi-v3 environment using the Gymnasium framework. Our goal is for the agent to learn an optimal policy that efficiently picks up and drops off passengers in a 5x5 grid world which pops up in random locations, but always over a designated space.

Why Q-Learning? Q-Learning is an off-policy reinforcement learning algorithm. It was chosen because it allows the agent to learn from experience without requiring a model of the environment. This assignment investigates how the learning rate (α) and exploration factor (ε) influence performance and convergence. The results of these experiments help determine the best hyperparameter combination for maximizing long-term rewards and achieving stable policy behavior.

2. Methodology

2.1. Environment Setup

What does the Taxi-v3 environment consist of?

- A 5×5 grid with four key locations: Red, Green, Blue, and Yellow.

- The taxi must pick up a passenger from one of these points and drop them at another.

- A state is represented as a single integer (0–499) encoded by:

((taxi_row×5+taxi_col)×5+passenger_location)×4+destination

2.2. Algorithm

The Q-Learning update rule used was:

$$Q(s,a)\leftarrow Q(s,a)+\alpha[r+\gamma a'\max Q(s',a')-Q(s,a)]$$

where:

- s, s′ are current and next states
- a is the chosen action
- r is the immediate reward

- α is the learning rate
- γ is the discount factor

An ε-greedy policy controlled the balance between exploration and exploitation.

2.3. Hyperparameters

Parameter	Description	Values Tested
α (Learning rate)	Determines how much new information overrides old estimates	0.1 (baseline), 0.01, 0.001, 0.2
ε (Exploration rate)	Probability of random action selection	0.1 (baseline), 0.2, 0.3
γ (Discount factor)	Weight of future rewards	0.9
Episodes	Training duration	5000
Max steps per episode	Safety limit to prevent infinite loops	200

3. Results

3.1. Training Overview

Each variation was trained for 5000 episodes.

Below I am pasting the sample output from the training log:

```
Best found: baseline (alpha=0.1, eps=0.1) — re-running to confirm.

Ep 1/5000 avg_last100=-569.00

Ep 1000/5000 avg_last100=-14.82

Ep 2000/5000 avg_last100=-0.60

Ep 4000/5000 avg_last100=2.72

Ep 5000/5000 avg_last100=2.12

All experiments finished.
```

3.2. Summary Table

Label	α	ε	γ
baseline	0.1	0.1	0.9
alpha_0.01	0.01	0.1	0.9
alpha_0.001	0.001	0.1	0.9
alpha_0.2	0.2	0.1	0.9
epsilon_0.2	0.1	0.2	0.9
epsilon_0.3	0.1	0.3	0.9

3.3. Visual Results

Please check the PDF generated by one of the cells in the Jupyter Notebook which also part of the GitHub Repository: LINK . All the plots are also saved as ".png" files under the "results" folder in the assignment's repository.

4. Discussion

The experiments show that learning rate α=0.1 and exploration rate ε=0.1 yield the most stable and consistent results.
Smaller α values (0.01, 0.001) caused slower learning, while higher α (0.2) introduced instability in the reward trend.
Increasing ε to 0.2 or 0.3 increased exploration but reduced convergence speed.

The baseline run consistently achieved positive average rewards after around 2,500 episodes, indicating that the agent learned an effective policy for navigating to passenger pickup and drop-off locations.

The last-100-episode average reward for the best configuration showed stable convergence, confirming that the Q-table successfully encoded an optimal policy.

5. Conclusion

The baseline configuration *(α=0.1, ε=0.1, γ=0.9)* provided the best overall balance between learning speed and stability.
After retraining under these parameters, the saved Q-table was tested in simulation using the `simulate_episodes()` function, and the taxi successfully completed pickup and drop-off tasks consistently.

This confirms that Q-Learning can effectively solve the Taxi-v3 task through tabular updates and ε-greedy exploration.