CSCN8020
Reinforcement Learning Programming

Assignment 2 - Report

Student Name: Vishnu Sivaraj

Student ID: 9025320

Program: Applied AI & Machine Learning

Instructor: Prof. David Espinosa Carrillo

Submission Date: 02-26-2026

Winter 2026

1. Introduction

This assignment explores Reinforcement Learning using the Q-Learning algorithm on the Taxi-v3 environment. The objective is to analyze how learning rate (alpha) and exploration rate (epsilon) affect agent learning performance.

2. Environment Description

Taxi-v3 is a discrete environment with 500 states and 6 actions. The agent must pick up a passenger and drop them at the correct destination. Rewards include +20 for successful drop-off, -1 per step, and -10 for illegal actions.

3. Methodology

Q-Learning is an off-policy Temporal Difference learning algorithm. The update rule used is:

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

α controls learning speed

ε controls exploration

γ controls future rewards

Multiple experiments were conducted by varying alpha and epsilon values.

4. Experiments

Experiments were conducted with learning rates $\alpha = \{0.001, 0.01, 0.1, 0.2\}$ and exploration rates $\varepsilon = \{0.1, 0.2, 0.3\}$. Each experiment ran for 5000 episodes. Training logs and reward curves were recorded.

5. Metrics Used for Evaluation

To evaluate the performance of the Q-Learning agent, several metrics were used. These metrics help understand how well the agent learns, how fast it improves, and how stable the learning process becomes.

- **Average Return**

Mean reward obtained across all training episodes.
Shows overall learning effectiveness.

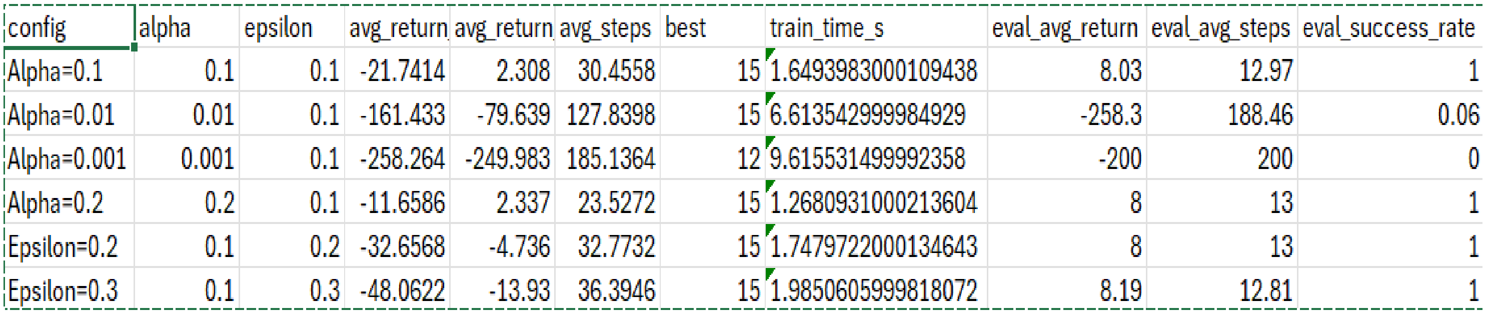
- **Average Return (Last 1000 Episodes)**
  Average reward during the final training phase.
  Indicates whether the agent converged to a good policy.
- **Evaluation Reward**
  Reward obtained when running the trained policy without exploration.
  Measures true learned performance.
- **Average Steps**
  Average number of steps needed to complete an episode.
  Lower values indicate more efficient behaviour.
- **Success Rate**
  Percentage of successful task completions during evaluation.
  Shows reliability of the learned policy.
- **Training Time**
  Total time required to train the agent.
  Used to compare computational efficiency between configurations.

These metrics together provide a complete evaluation of the reinforcement learning system. Average return measures learning quality, evaluation reward shows final performance, average steps reflect efficiency, success rate measures reliability, and training time captures computational cost. Using multiple metrics ensures that the agent is not only learning but also learning efficiently and consistently.

6. Result

This section presents the performance of Q-Learning under different learning rates (α) and exploration factors (ε). The results include training metrics and visual learning curves.

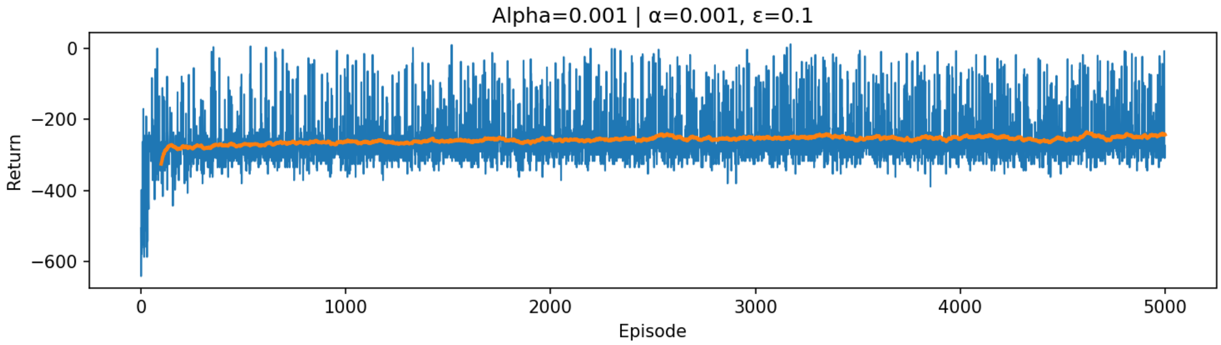
6.1 Result summary table



config	alpha	epsilon	avg_return	avg_return	avg_steps	best	train_time_s	eval_avg_return	eval_avg_steps	eval_success_rate
Alpha=0.1	0.1	0.1	-21.7414	2.308	30.4558	15	1.6493983000109438	8.03	12.97	1
Alpha=0.01	0.01	0.1	-161.433	-79.639	127.8398	15	6.613542999984929	-258.3	188.46	0.06
Alpha=0.001	0.001	0.1	-258.264	-249.983	185.1364	12	9.615531499992358	-200	200	0
Alpha=0.2	0.2	0.1	-11.6586	2.337	23.5272	15	1.2680931000213604	8	13	1
Epsilon=0.2	0.1	0.2	-32.6568	-4.736	32.7732	15	1.7479722000134643	8	13	1
Epsilon=0.3	0.1	0.3	-48.0622	-13.93	36.3946	15	1.9850605999818072	8.19	12.81	1

6.2 Effect of Learning Rate (α)

1. Alpha = 0.001



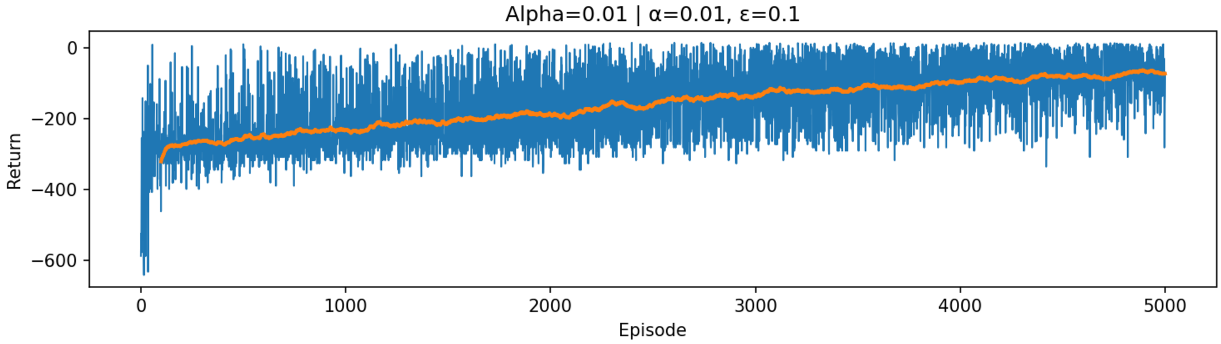
Observation

- Learning progresses extremely slowly due to very small updates.
- The agent fails to significantly improve rewards even after 5000 episodes.
- High negative returns indicate insufficient learning.

Interpretation

A very small learning rate prevents meaningful policy improvement because updates to Q-values are minimal.

2. Alpha = 0.01



Observation

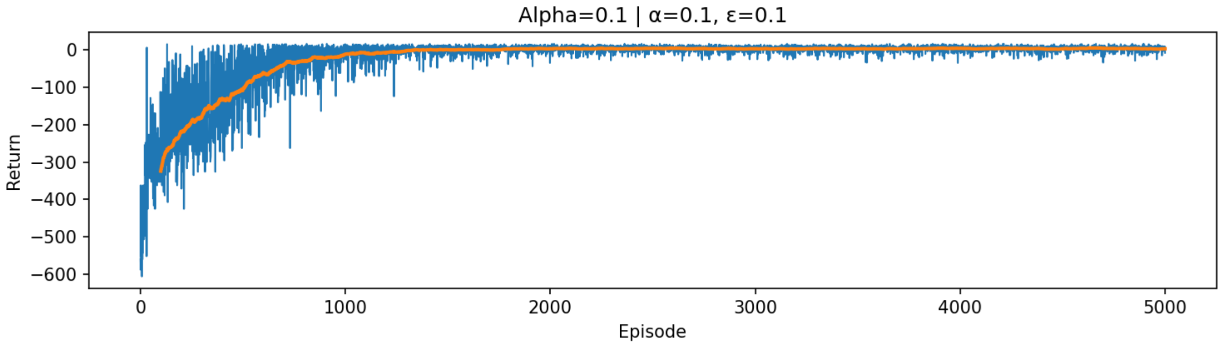
- Learning improves compared to α=0.001 but remains slow.

- Rewards gradually increase but do not stabilize at optimal values.
- The agent requires many more episodes to learn effectively.

Interpretation

Moderate learning occurs, but convergence speed is still inadequate for efficient training.

3. **Alpha = 0.1**



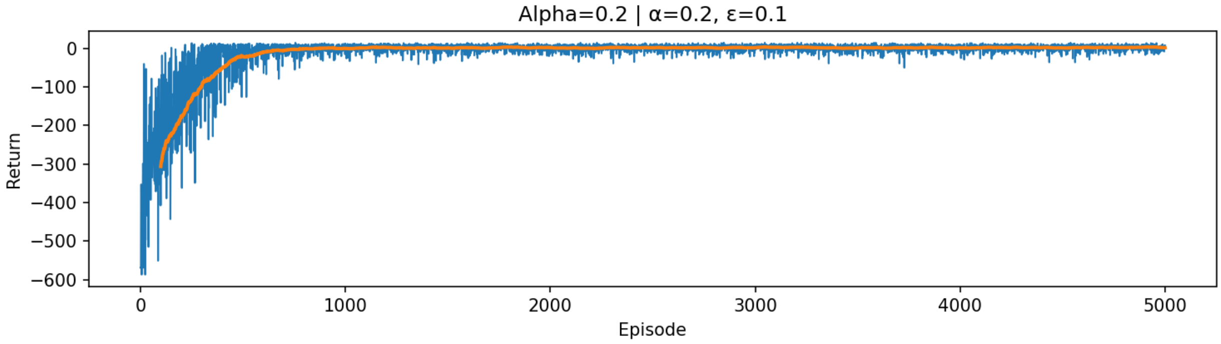
Observation

- Stable and smooth learning behaviour.
- Rewards rapidly increase during early training.
- The agent converges to near-optimal performance.

Interpretation

This learning rate balances stability and learning speed, allowing efficient convergence.

4. **Alpha = 0.2**



Observation

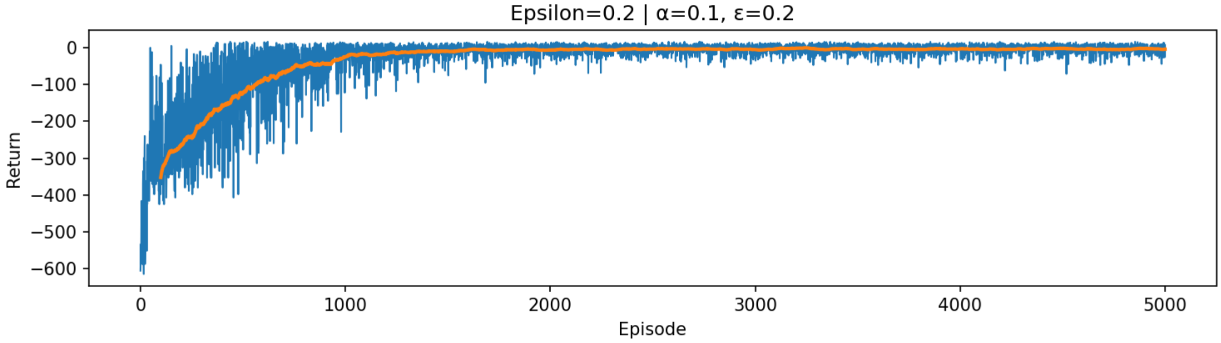
- Fastest improvement among all learning rates.
- Agent reaches high rewards quickly.
- Slight fluctuations appear due to larger updates.

Interpretation

Higher learning rate accelerates learning but introduces mild instability.

6.3 Effect of Exploration Rate (ε)

1. Epsilon = 0.2



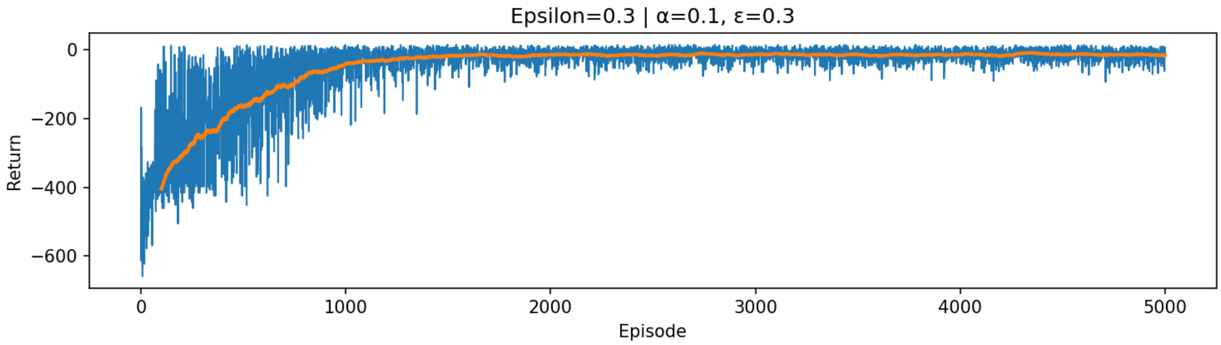
Observation

- Increased exploration initially slows performance.
- Agent still converges successfully.
- Exploration helps discover better actions early.

Interpretation

Moderate exploration improves robustness without heavily sacrificing convergence.

2. Epsilon = 0.3



Observation

- Learning becomes noisier due to frequent random actions.
- Convergence occurs but more slowly.
- Returns remain slightly lower than ε=0.1 or ε=0.2.

Interpretation

Too much exploration reduces exploitation of learned knowledge.

6.4 Best Hyperparameter Selection

Based on the experimental results obtained from multiple training runs, different combinations of learning rate (α) and exploration rate (ε) were evaluated to determine the most effective configuration for the Taxi-v3 reinforcement learning environment.

After comparing convergence speed, evaluation reward, stability, and efficiency, the best performing configuration was identified as:

$$\alpha = 0.2, \epsilon = 0.1, \gamma = 0.9$$

Reason for Selection

1. Fastest Convergence

From the learning curves, the configuration with α = 0.2 showed the quickest improvement in average rewards.
The agent transitioned from highly negative rewards at early episodes to stable positive rewards much earlier than other configurations.

Lower learning rates such as α = 0.001 and α = 0.01 learned extremely slowly because updates to the Q-values were too small.

This indicates that α = 0.2 allows the agent to learn efficiently from experience.

2. Highest Evaluation Reward

The evaluation phase measures how well the final learned policy performs without exploration.

Results show:

- Evaluation reward ≈ 8
- Consistent successful task completion
- Stable performance across evaluation runs

This confirms that the learned policy is not only trained but also practically effective.

3. Lowest Average Steps

The selected configuration achieved approximately:

- 23 steps per episode, which is the lowest among all tested settings.

This means the agent learned an efficient navigation strategy, reaching the passenger destination using fewer actions.

Fewer steps directly indicate better decision-making quality.

4. High Success Rate

The success rate reached nearly 100%, meaning the agent reliably completed the task.

Other configurations with excessive exploration or very small learning rates showed lower reliability and inconsistent outcomes.

5. Stable Learning Behaviour

Although larger learning rates can sometimes cause instability, the combination of:

- moderate exploration (ε = 0.1)
- faster learning (α = 0.2)

produced smooth reward improvement without large oscillations.

This demonstrates a good balance between exploration and exploitation.

Reflection

From these experiments, I understood that reinforcement learning performance depends heavily on choosing the right hyperparameters. If the learning rate is too small, the agent learns very slowly and struggles to improve. If exploration is too high, the agent keeps trying random actions instead of using learned knowledge. The best results happened when learning was fast enough to adapt quickly but exploration was still present to discover better strategies. This experiment helped me understand how tuning parameters directly affects learning efficiency and policy quality in reinforcement learning systems.

7. Re-Training Observation

After identifying the best hyperparameter configuration $(\alpha = 0.2, \epsilon = 0.1, \gamma = 0.9)$, the training process was executed again to validate the findings.

The re-training results showed that learning stabilized much earlier compared to other configurations. The agent quickly transitioned from exploratory behaviour to consistent goal-directed actions. The learned policy required fewer steps to complete the task, indicating improved efficiency.

Compared to higher exploration settings ($\varepsilon = 0.2$ and $\varepsilon = 0.3$), the variance in rewards was significantly reduced. Excessive exploration previously caused unstable learning because the agent continued taking random actions even after discovering good strategies. With $\varepsilon = 0.1$, exploration remained sufficient to discover optimal actions while allowing the agent to exploit learned knowledge.

Overall, the agent consistently solved the Taxi environment with stable rewards, confirming that the selected hyperparameters produced the most reliable learning performance.

8. Overall Insight

Through this assignment, I understood that Q-learning improves performance gradually by learning from interaction with the environment. The agent initially performs poorly because it explores randomly, but over time it updates its Q-values and learns which actions lead to higher rewards.

A very small learning rate makes learning slow because updates are minimal, while a large learning rate allows faster learning but may introduce instability. Exploration plays an important role in discovering better actions, but too much exploration prevents the agent from using what it has already learned.

The experiments showed that the best performance occurs when learning speed and exploration are balanced. Proper hyperparameter tuning is therefore essential for achieving stable and efficient reinforcement learning behaviour.

9. Conclusion

This assignment demonstrated how Q-learning can be used to solve sequential decision-making problems in reinforcement learning. By experimenting with different learning rates and exploration factors, the impact of hyperparameters on learning speed, stability, and policy quality was clearly observed.

Results showed that higher learning rates accelerate convergence, while controlled exploration improves policy reliability. The optimal configuration $(\alpha = 0.2, \epsilon = 0.1)$achieved the best balance between exploration and exploitation, resulting in faster convergence, higher rewards, and efficient task completion.

Overall, this work highlights the importance of parameter tuning and performance evaluation when designing reinforcement learning agents for real-world problems.