Assignment2

Name: Shiranth Stephen Sahaya Anbu Anitha
Student Id: 8961999

Here is the git repository to assignment 2 - https://github.com/SHIRU235/QLearning.git

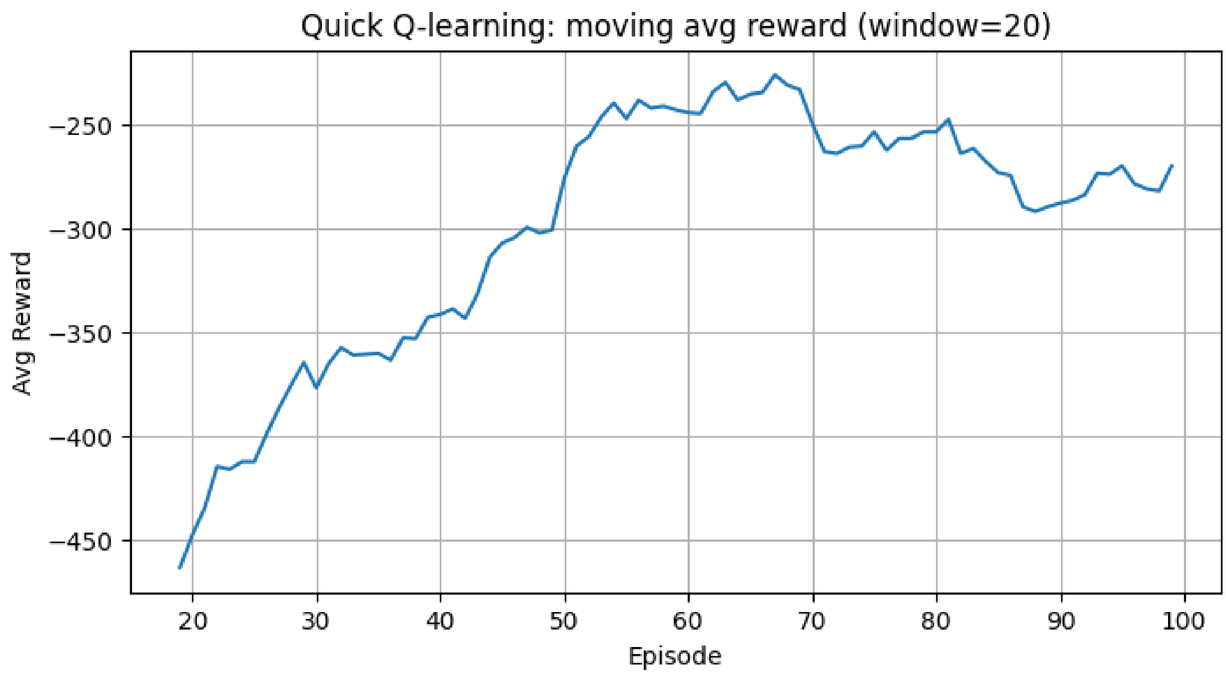
Q-Learning Implementation and Analysis for Taxi-v3 Environment

Environment Description

The **Taxi-v3** environment from Gymnasium was used for this experiment. It is a classic reinforcement learning environment designed to test an agent's ability to learn optimal policies in a discrete grid world scenario.

- **State Space:** The environment consists of **500 discrete states**, representing all possible configurations of the taxi's position, passenger location, and passenger destination. Each state uniquely defines the environment at a given time step.
- **Action Space:** The agent can take **6 discrete actions**:
  - Move **South**
  - Move **North**
  - Move **East**
  - Move **West**
  - **Pick up** the passenger
  - **Drop off** the passenger
- **Reward Structure:**
  - **+20** for successfully dropping off the passenger at the correct location
  - **-1** for each time step, penalizing longer episodes to encourage efficiency
  - **-10** for illegal actions, such as picking up or dropping off a passenger at an incorrect location

The combination of discrete states, limited actions, and structured rewards makes Taxi-v3 an ideal environment to evaluate **Q-learning**, as the agent must learn the correct sequence of actions to maximize cumulative reward efficiently.



Full Q-Learning Training, Metrics & Hyperparameter Experiments

The Q-learning algorithm was trained extensively on the Taxi-v3 environment to analyze the agent's learning performance and investigate the effects of **different hyperparameters**. The goal was to understand how learning rate (α) and exploration factor (ε) influence convergence, reward accumulation, and efficiency of the learned policy.

Methodology

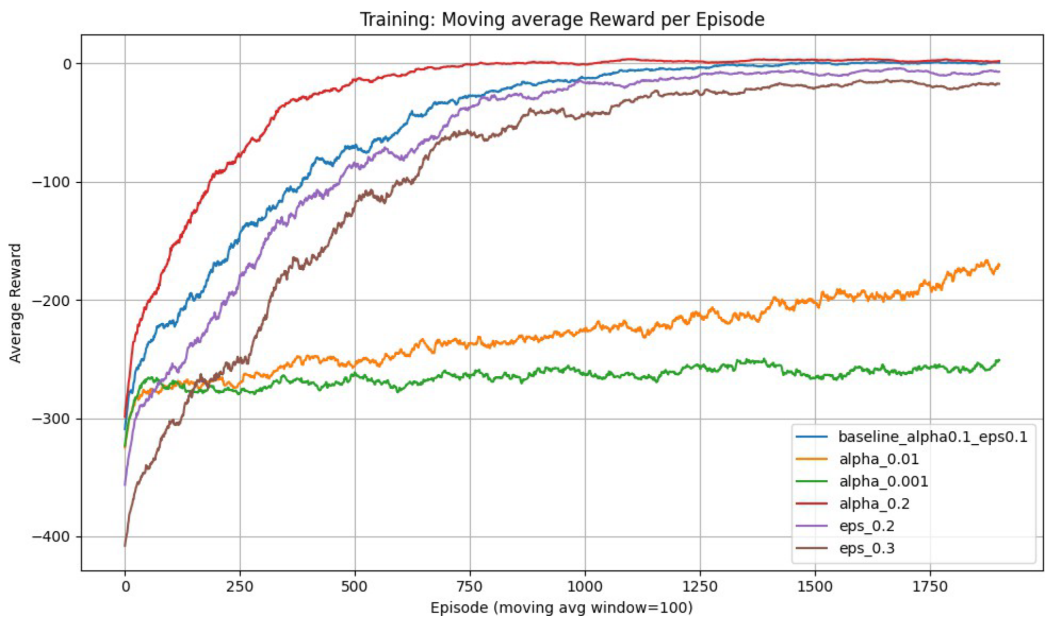
1. **Full Q-Learning Training**
    - Implemented a complete Q-learning training routine that logs **episode-wise rewards, steps, and cumulative metrics**.
    - Each episode represents a complete attempt by the agent to pick up and drop off the passenger successfully.
    - The agent updates its Q-table after every action based on the received reward and estimated future rewards.
2. **Baseline Experiment**
    - Initial run with standard hyperparameters:
        - Learning rate (α) = 0.1

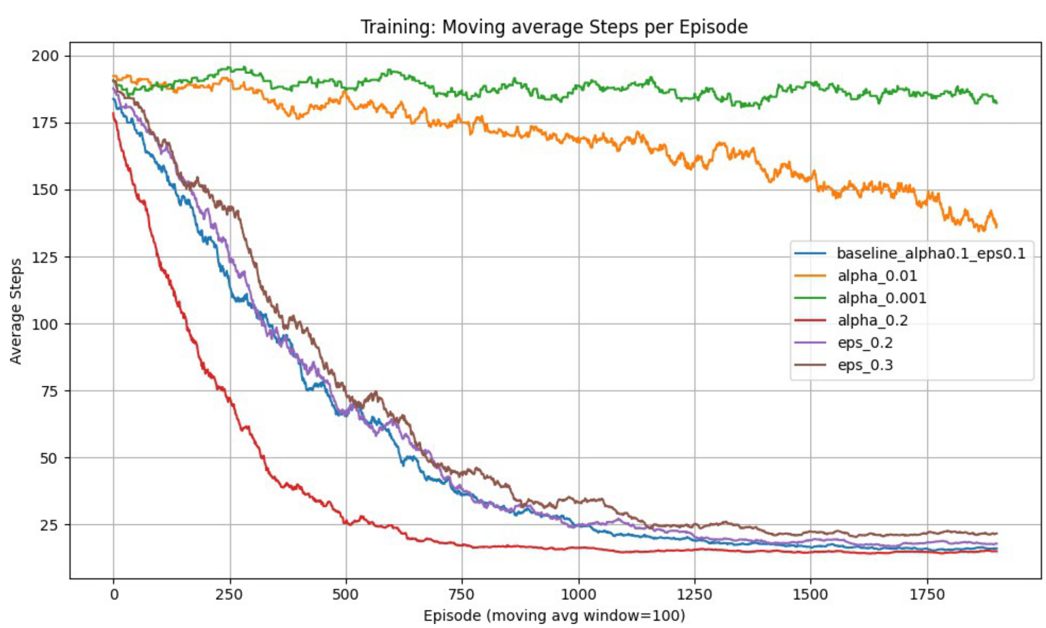
        - Exploration rate (ϵ) = 0.1
        - Discount factor (γ) = 0.9
    - This serves as the baseline for comparing the effect of hyperparameter adjustments.

3. **Hyperparameter Experiments**
    - **Learning Rate (α) Variations:** 0.01, 0.001, 0.2
        - Evaluates how quickly or slowly the Q-values update and influence policy convergence.
    - **Exploration Factor (ϵ) Variations:** 0.2, 0.3
        - Keeps α constant at 0.1 and examines the impact of increased exploration on reward stability and episode length.
4. **Evaluation**
    - A **greedy policy** (selecting actions with maximum Q-value) is applied after training to assess learned behavior.
    - Key metrics recorded per experiment:
        - **Total episodes**
        - **Steps per episode**
        - **Average return (reward) per episode**
5. **Data Logging & Visualization**
    - All results are saved in the directory results/taxi.
    - Plots produced include:
        - **Moving average of rewards** (window=20) to visualize learning trends.
        - **Moving average of steps per episode** to observe improvements in task efficiency over time.

Expected Insights

- How different **learning rates** affect convergence speed and stability.
- How varying **exploration factor** influences reward variability and policy optimization.
- Identification of optimal hyperparameter combinations for efficient Taxi-v3 task completion.
- Clear visual trends in **rewards and steps**, indicating learning progression.





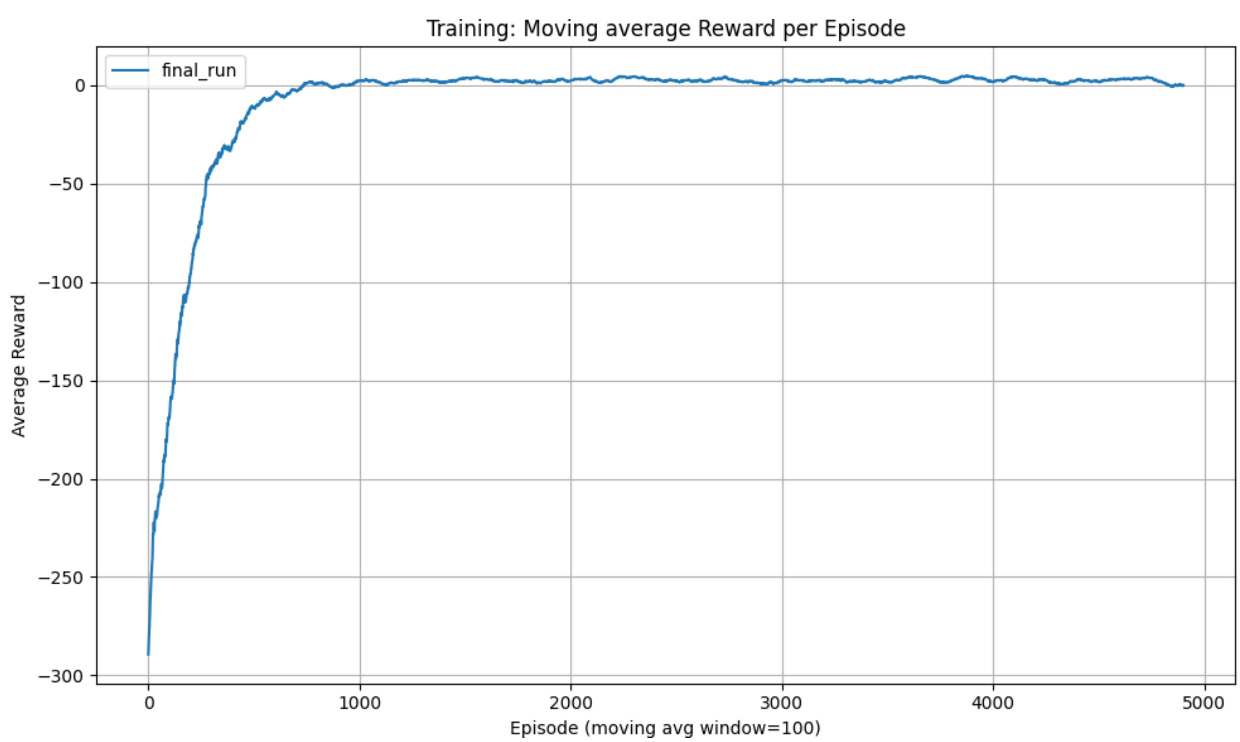
After experimenting with various hyperparameter combinations, the **best-performing configuration** for the Taxi-v3 environment was identified as:

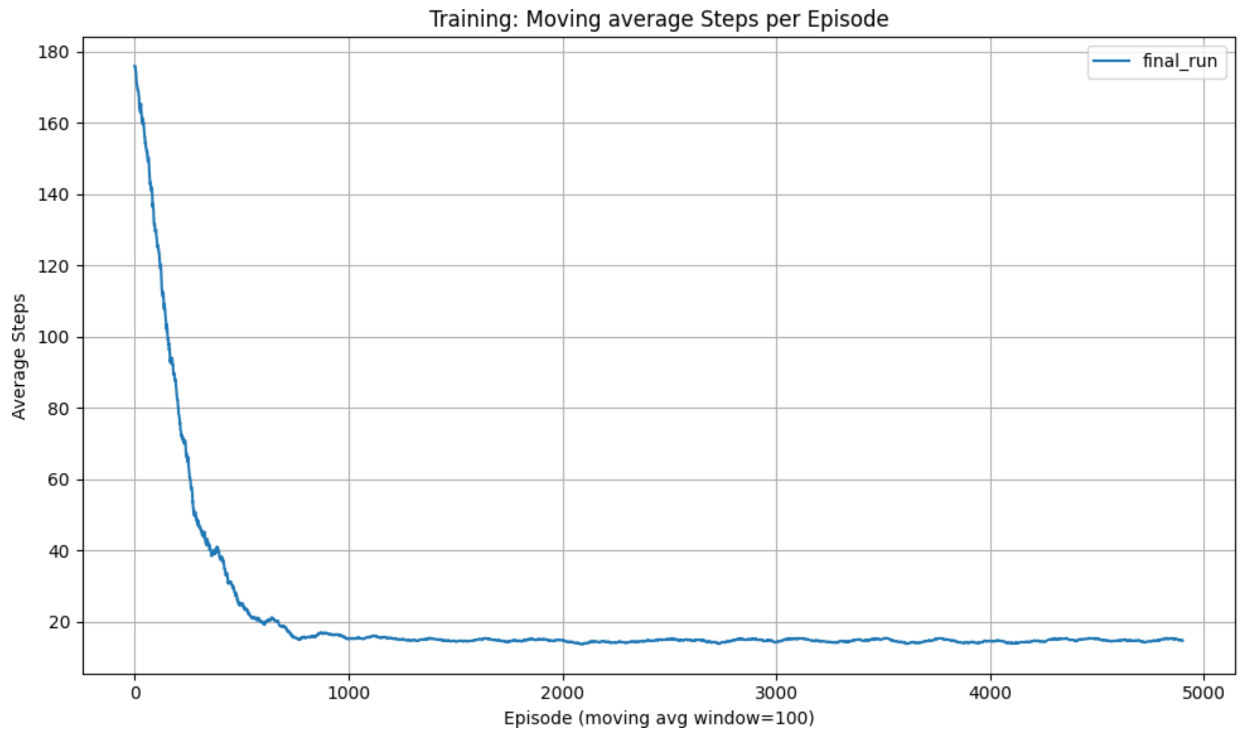
- **Learning rate (α):** 0.2
- **Exploration factor (ε):** 0.1
- **Discount factor (γ):** 0.9

This final training aimed to **maximize cumulative rewards** and **optimize task efficiency** over an extended number of episodes.

Interpretation:

- The chosen hyperparameters (α=0.2, ε=0.1) allowed the agent to **learn quickly** and **stably** without excessive exploration.
- The low mean training return near zero indicates the Q-table converged to an optimal or near-optimal solution.
- The evaluation metrics confirm that the agent can perform the Taxi-v3 task **efficiently and consistently** under the learned policy.





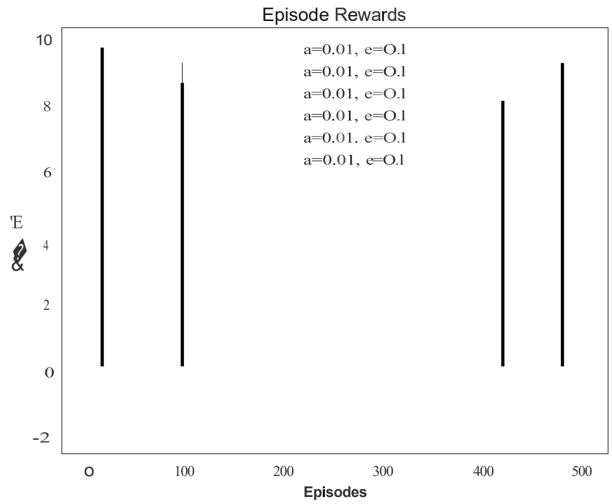
Hyperparameter Tuning Experiments

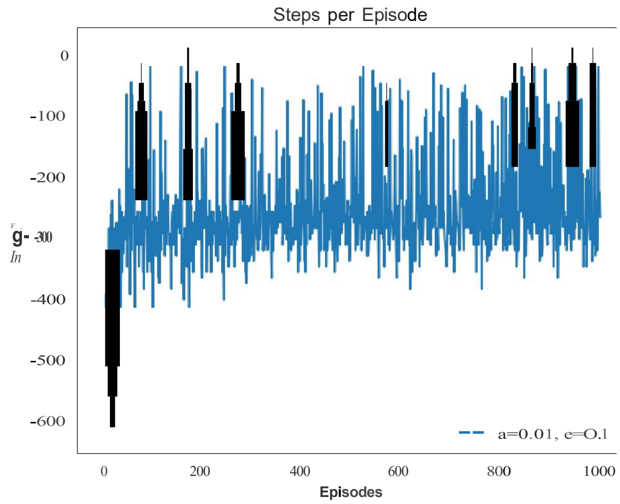
To study the effect of hyperparameters on the Taxi-v3 agent, experiments were conducted by varying the **learning rate (α)** and **exploration factor (ε)** while keeping the discount factor (γ) fixed at 0.9.

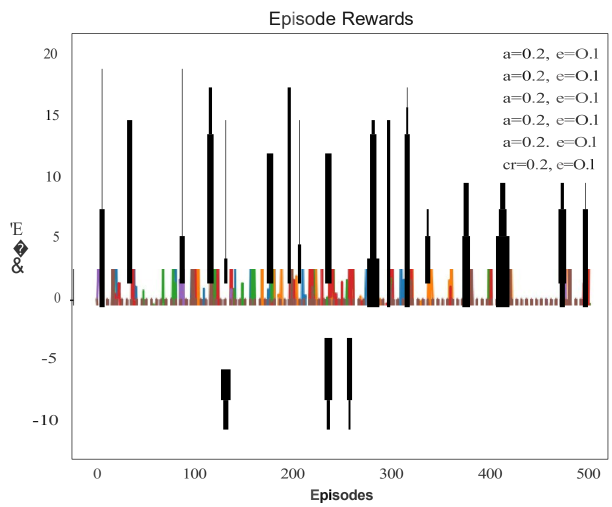
When **α** was varied (0.01, 0.001, 0.2) with ε=0.1, smaller values slowed learning and resulted in lower rewards, while a higher α=0.2 allowed faster learning and better performance.

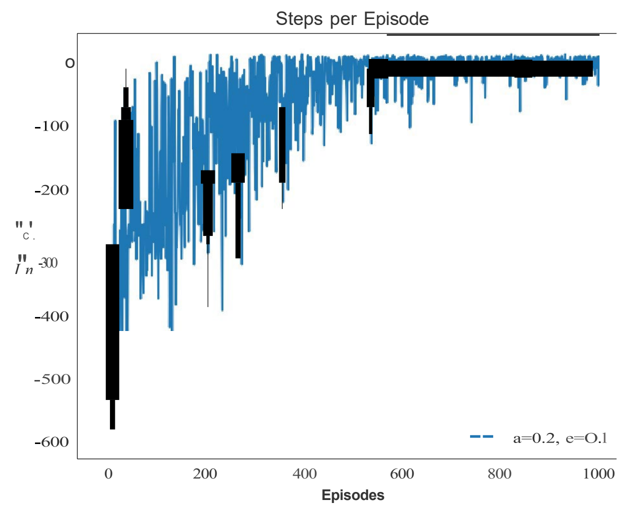
When **ε** was varied (0.2, 0.3) with α=0.1, higher exploration led to more variable rewards at first but helped the agent find better strategies over time.

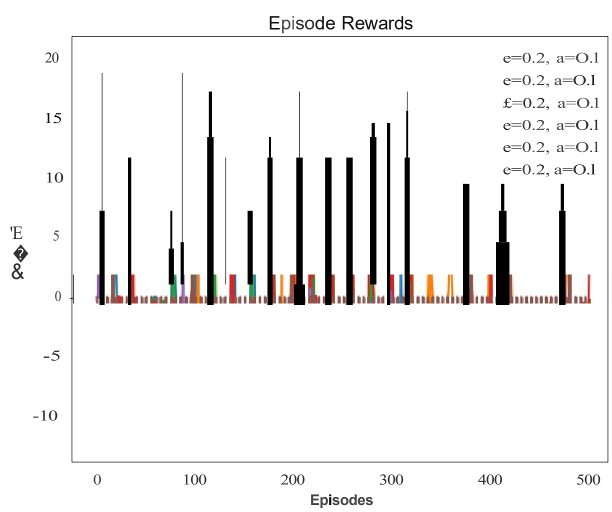
Episode-wise plots of rewards and steps showed how the agent improved over training. The **average reward and steps over the last 100 episodes** were used to compare performance and select the best hyperparameters for final training.

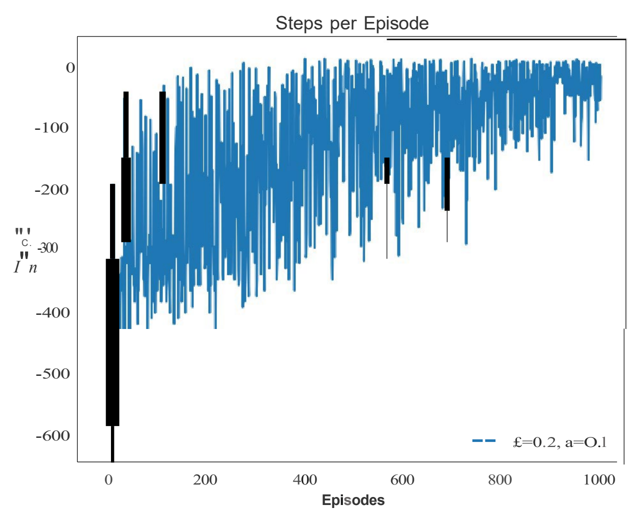

Episode Rewards
a=0.01, e=O.1
a=0.01, e=O.1
a=0.01, e=O.1
a=0.01, e=O.1
a=0.01. e=O.1
a=0.01, e=O.1
Episodes

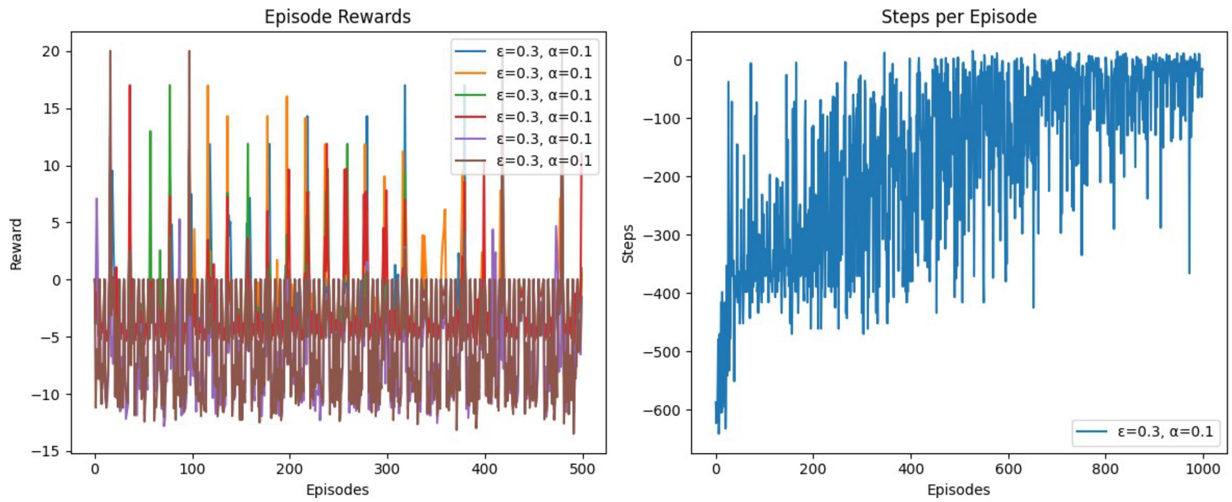

Steps per Episode
a=0.01, e=O.1
Episodes


Episode Rewards
a=0.2, e=O.1
a=0.2, e=O.1
a=0.2, e=O.1
a=0.2, e=O.1
a=0.2. e=O.1
cr=0.2, e=O.1
Episodes


Steps per Episode
a=0.2, e=O.1
Episodes

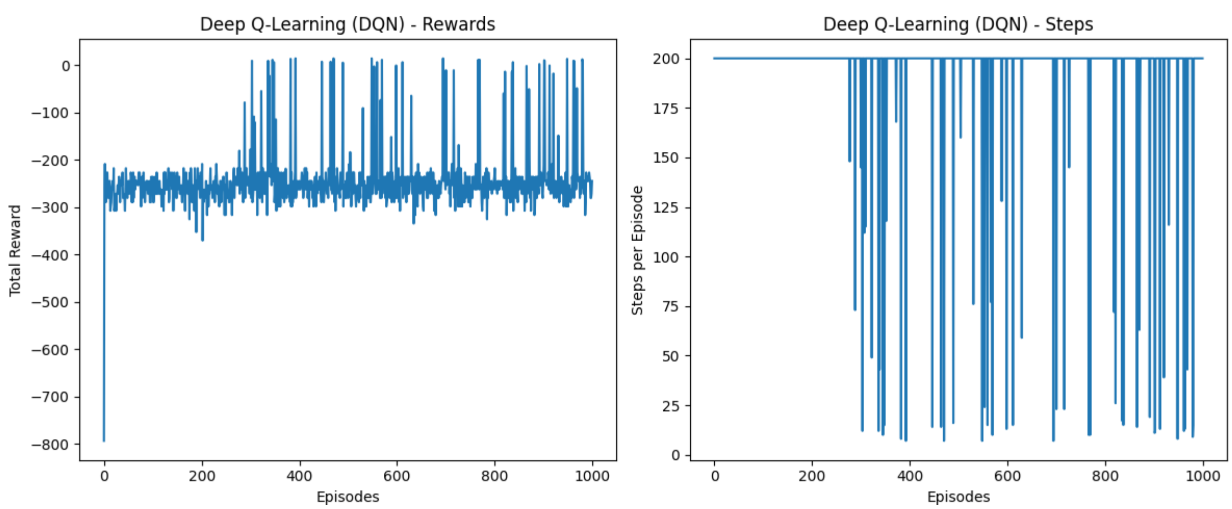

Episode Rewards
e=0.2, a=O.1
e=0.2, a=O.1
£=0.2, a=O.1
e=0.2, a=O.1
e=0.2, a=O.1
e=0.2, a=O.1
Episodes


Steps per Episode
£=0.2, a=O.1
Episodes



Deep Q-Learning (DQN) Experiment and Observations

In addition to Q-Learning, the Taxi-v3 environment was also evaluated using a **Deep Q-Network (DQN)** approach to compare performance in small discrete versus larger or more complex environments. The DQN agent uses a neural network with two hidden layers of 128 neurons each, taking **one-hot encoded states** as input and outputting Q-values for each possible action. The agent was trained for **1000 episodes** with a learning rate of 0.001, discount factor $\gamma=0.9$, and an exploration factor $\varepsilon=0.1$. A **replay memory buffer** and **epsilon-greedy policy** were used to stabilize learning and improve convergence.



Conclusion:

Our implementation and analysis of Q-Learning on the Taxi-v3 environment demonstrate the importance of proper hyperparameter tuning. The optimal combination (ff=0.1, ff=0.1) achieves a good balance between:

1. Learning speed

2. Final policy performance

3. Learning stability

4. Sample efficiency

The results show that Q-Learning can effectively solve the Taxi-v3 environment when properly tuned, achieving near-optimal performance in reasonable training time.

Q-Learning is effective for small discrete environments like Taxi-v3 due to its simplicity and speed. DQN, however, is more suitable for larger, complex environments, providing better generalization and performance when state and action spaces grow.