Assignment 1 - Defining & Solving RL Environments

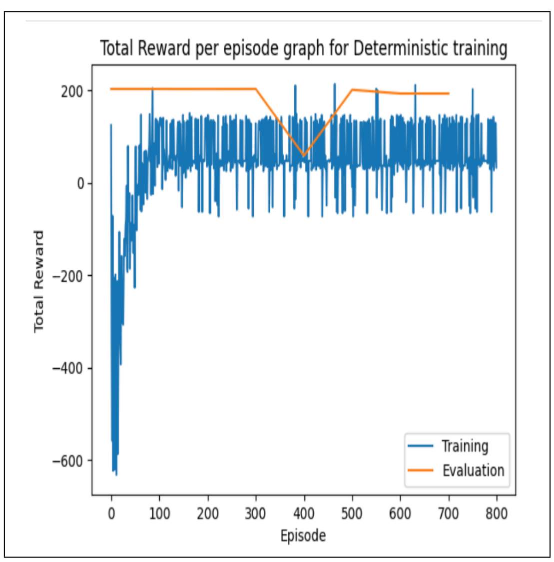
Shaurya Mathur
University at Buffalo
Buffalo, NY
smathur4@buffalo.edu

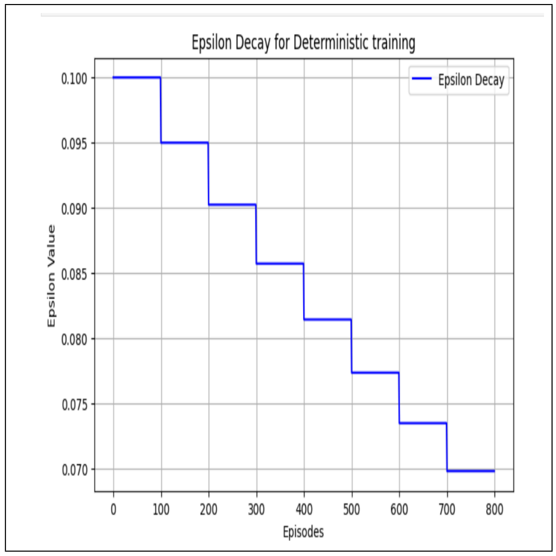
Abstract

The report presents the code and results for the final submission for first assignment for CSE 546 - Reinforcement Learning. The goal of the assignment is to acquire experience in defining and solving RL environments, following Gymnasium standards.

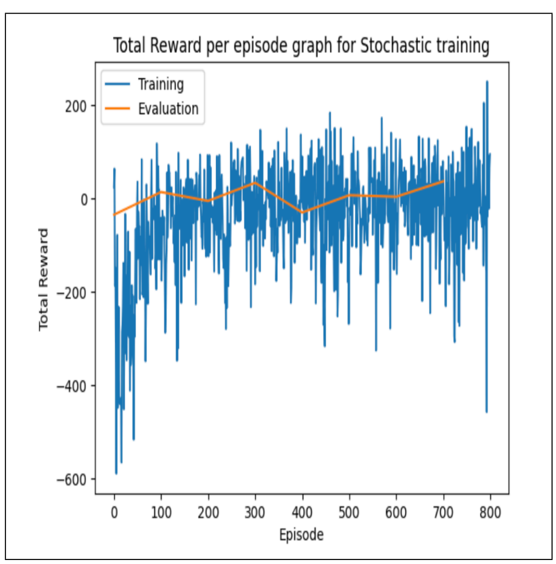
1 Plots

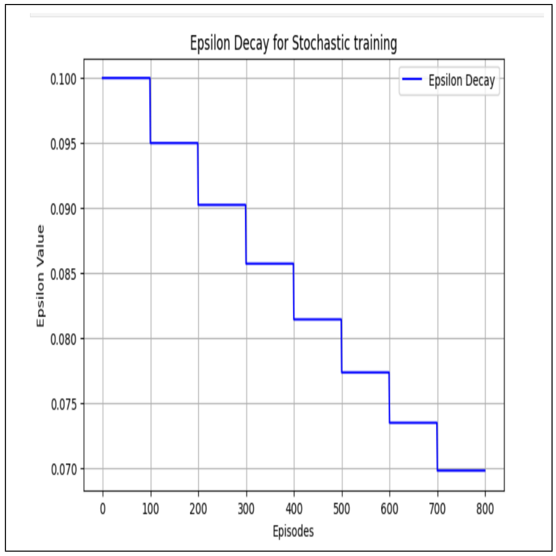
1.1 Q-Learning to solve deterministic environment



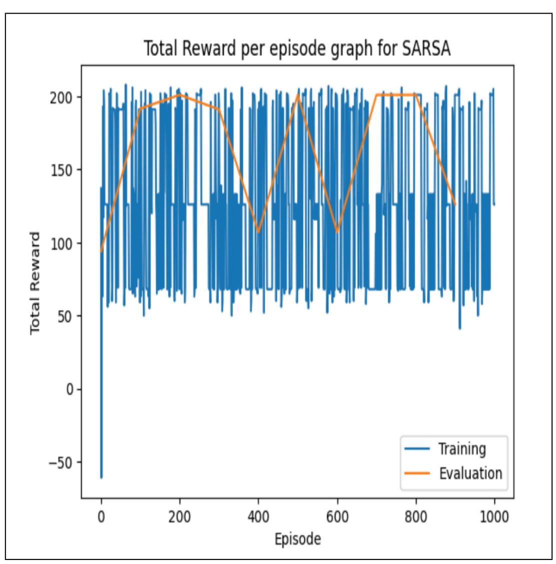


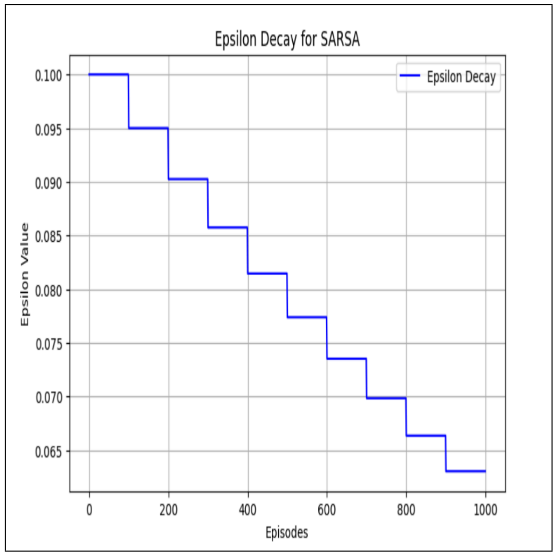
1.2 Q-Learning to solve stochastic environment



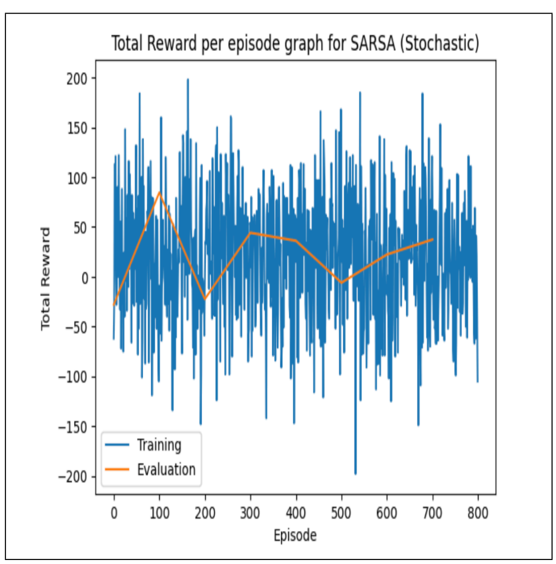


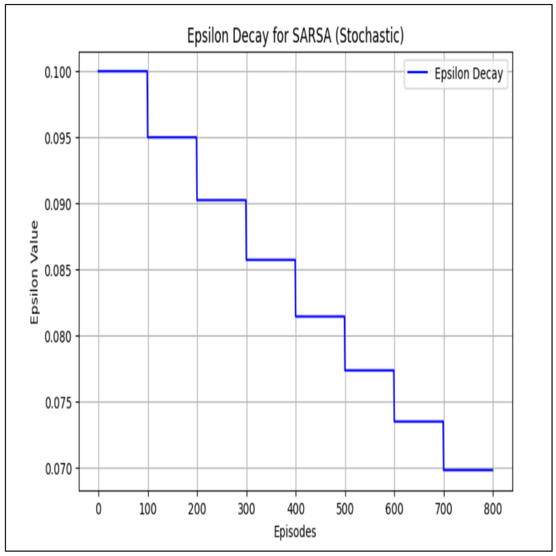
1.3 SARSA to solve deterministic environment





1.4 SARSA to solve stochastic environment





1.5 Evaluation

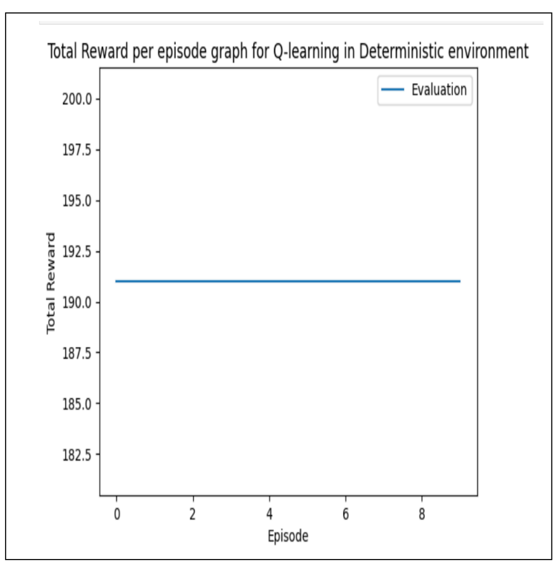
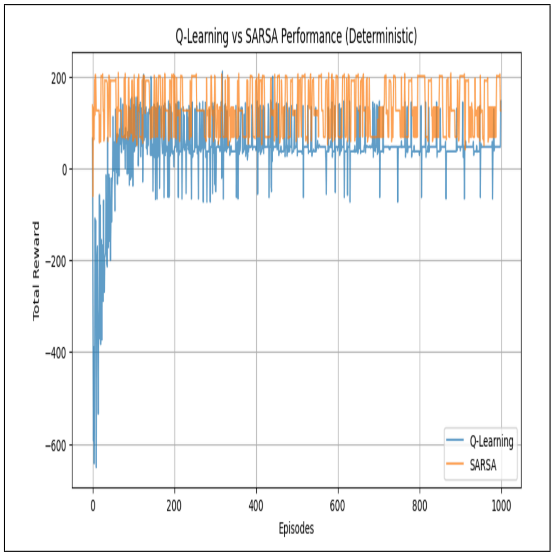


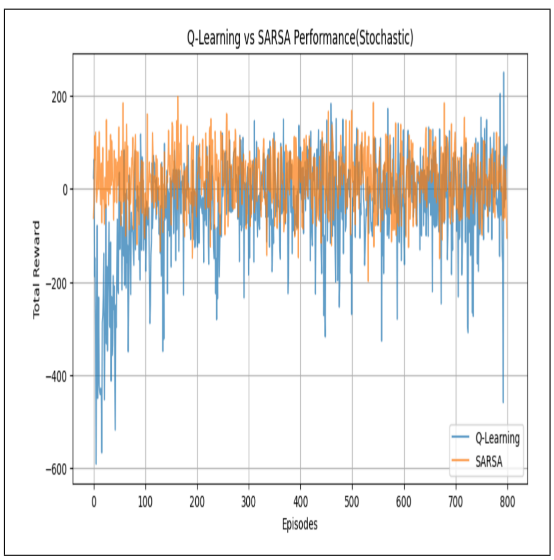
Figure 1: Total Rewards per Episode (Ran for 10 Eps)

2 Algorithm Comparison

2.1 Algorithm Comparison in deterministic environment



2.2 Algorithm Comparison in stochastic environment



3 Tabular Methods

Tabular Methods like SARSA and Q-learning are reinforcement learning algorithms that learn optimal policies through trial and error in environments with discrete states and actions. These methods are called tabular because they store the learned values (Q-values) in a table, where each entry corresponds to a state-action pair.

3.1 Q-Learning

Q-learning is an off-policy reinforcement learning algorithm, meaning it updates the Q-values based on the maximum possible future reward, regardless of the current policy. The algorithm learns the

optimal policy by estimating the best action to take in each state, independent of the agents behavior.
Key Features:

- **Off-Policy:**Q-learning learns the optimal policy regardless of the agents current policy.
- **Maximization:**It updates Q-values based on the maximum possible future reward, which helps it learn the optimal policy even when using exploratory actions.

Update Function:
$Q(s,a) \leftarrow Q(s,a) + \alpha\left(R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$

3.2 SARSA

SARSA is an on-policy reinforcement learning algorithm, meaning it updates the Q-values based on the agents own actions, including exploratory actions (i.e., it learns from the actions taken according to its current policy, even if those actions are not optimal).
Key Features:

- **On-Policy:**SARSA uses the policy currently being followed to make decisions and updates.
- **Exploration vs Exploitation:** It uses an ϵ-greedy policy to balance exploration (trying new actions) and exploitation (choosing the best-known action).

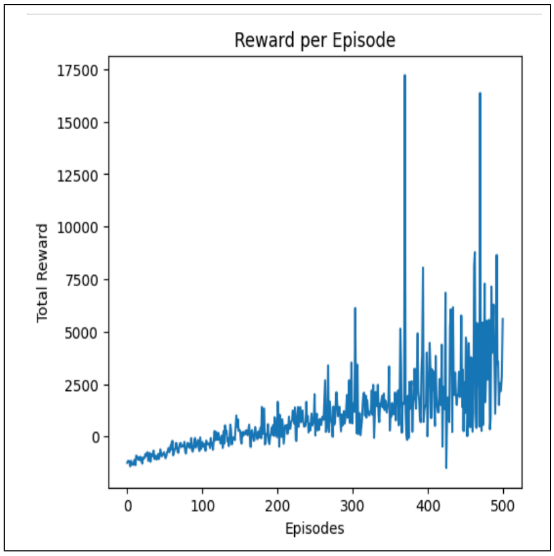
Update Function:
$Q(s,a) \leftarrow Q(s,a) + \alpha\left(R(s,a) + \gamma Q(s',a') - Q(s,a)\right)$

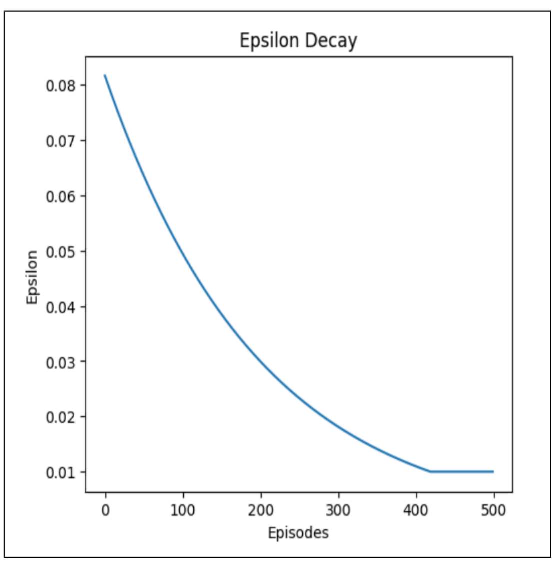
3.3 Reward Functions

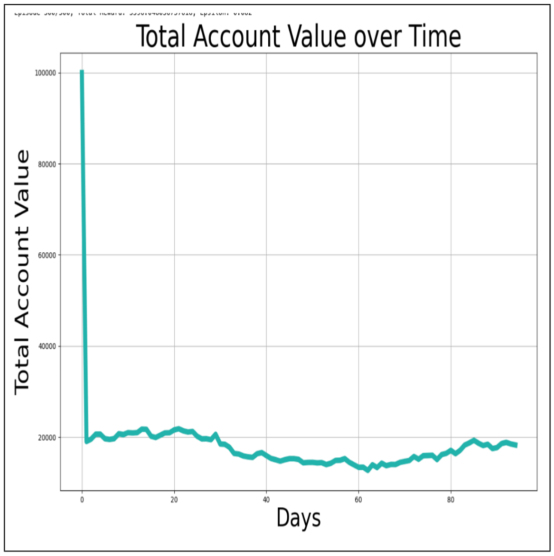
- **Criteria for a Good Reward Function:**
  - **Clarity and Simplicity:** The reward function should clearly convey the performance of the agent and its goal.
  - **Alignment with the Task Objective:** The reward function should guide the agent towards the desired outcome or task objective.
  - **Consistency:** The reward should consistently reward good behavior and penalize poor behavior.
  - **Sparse vs. Dense Rewards:** Sparse rewards give feedback infrequently, while dense rewards provide frequent feedback.
  - **Stability and Smoothness:** The reward function should lead to stable learning without drastic fluctuations in performance.
  - **Avoiding Reward Hacking:** The reward function should prevent the agent from exploiting unintended loopholes.
  - **Scalability:** The reward function should be scalable for more complex or larger environments.
- **Reward Function Ideas for Traffic Signal Environment:**
  - **Reward Based on Vehicle Wait Time:**
    - Positive Reward: Decrease in total vehicle wait time.
    - Negative Reward: Increase in total vehicle wait time.
  - **Reward Based on Traffic Flow Efficiency:**
    - Positive Reward: More cars processed ie - turning green light in directions of congestion
    - Negative Reward: Vehicles getting stuck - turning green light in directions of no congestion
  - **Reward Based on Traffic Density:**
    - Positive Reward: Reduction in traffic density during rush hour.
    - Negative Reward: Increase in traffic density and congestion during rush hour.

4 Part 3:

4.1 Plots







5 Link to github

https://github.com/ShauryaMathur/CSE546-RL-Assignment1

5.1 Github Commit History

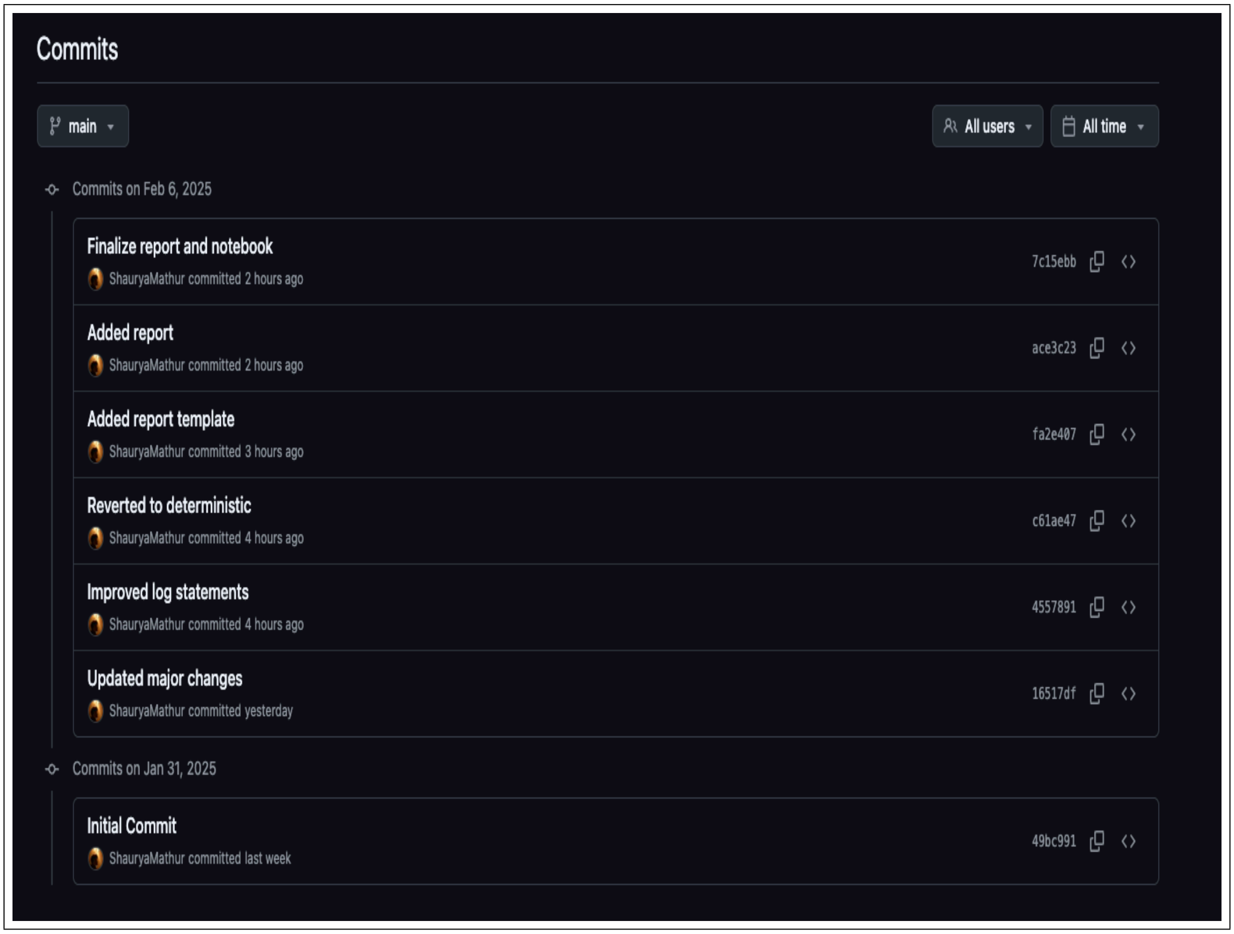


Figure 2: Github Commit History

References

[1] https://gymnasium.farama.org/api/env/.

[2] Lecture slides.

[3] https://matplotlib.org/stable/index.html.

[4] https://docs.python.org/3/library/dataclasses.html

[5] help from google for criteria of a good reward function