Comparative Analysis of Q-Learning and SARSA-Learning on Taxi problem

Najeeb Jilani
Computer Science
Habib University
Karachi, Pakistan

Muhammad Ahmed Atif
Computer Science
Habib University
Karachi, Pakistan

***Abstract*—This paper presents a comparative analysis of Q-learning and SARSA learning algorithms applied to the Taxi problem. The performance of both algorithms was evaluated using standardized environments and metrics such as convergence rate, cumulative reward, and exploration-exploitation trade-off. Results show that Q-learning achieved faster convergence and higher rewards, while SARSA demonstrated better adaptability to changing environments. These findings provide insights for selecting the appropriate algorithm based on problem requirements.**

***Index Terms*—Reinforcement learning, Q-learning, SARSA learning, Taxi problem, Comparative analysis, Convergence, Exploration-exploitation trade-off.**

I. Introduction

Reinforcement learning has developed as an effective method for autonomous agents to learn optimum decision-making rules in complex and unpredictable settings. Q-learning and SARSA learning have received substantial attention among other reinforcement learning algorithms due to their efficiency in tackling a wide range of issues. In this study, we compare the Q-learning and SARSA learning algorithms as they are applied to the well-known Taxi issue.

In the subject of reinforcement learning, the Taxi problem is a prominent benchmark test. It entails a taxi agent travelling in a grid-world environment, with the goal of picking up and dropping off customers at predetermined places while optimising specific performance parameters. The agent's requirement to develop an ideal strategy that balances exploration and exploitation while taking into account the changing nature of the environment adds to the task's complexity.

Q-learning is a model-free reinforcement learning system that estimates the anticipated rewards for alternative actions in each state using the idea of Q-values. It employs an iterative strategy to update the Q-values depending on the experiences of the agent, eventually improving the policy towards optimality. Q-learning has been widely used in a variety of fields and has proven to be effective in learning optimum policies.

Another prominent reinforcement learning technique, SARSA (State-Action-Reward-State-Action) learning, is model-free and comparable to Q-learning. SARSA, on the other hand, considers both the present and subsequent state-action pairs in its update process, making it an on-policy algorithm. SARSA provides a more balanced way to adjust to changes in the environment by explicitly including the exploration-exploitation trade-off during learning.

The purpose of this research is to compare the Taxi problem performance of Q-learning and SARSA learning algorithms. We intend to investigate their convergence rates, cumulative rewards, and trade-off features between exploration and exploitation. By comparing and analysing the performance of different algorithms, we hope to get insight into their advantages and disadvantages in addressing the Taxi problem.

The results of this comparison study can help academics and practitioners choose the best method for handling comparable reinforcement learning situations. Understanding the relative strengths and drawbacks of the Q-learning and SARSA learning algorithms in the context of the Taxi problem might yield useful insights for future research and practical implementations.

The rest of this paper is structured as follows: Section II summarises relevant work on Q-learning and SARSA learning algorithms. Section III covers our study's methodology and experimental design. Section IV includes the comparative evaluation findings and analyses. Section V finishes the report with a summary of the findings and future research prospects.

II. Algorithmic Background

A. Q-Learning

Q-Learning is a popular model-free off-policy reinforcement learning algorithm that aims to learn an optimal policy without requiring a model of the environment. It utilizes a value function, called the Q-value, to estimate the expected cumulative rewards for taking specific actions in each state. The Q-value of a state-action pair (s, a), denoted as $Q(s, a)$, represents the expected cumulative reward the agent will receive by starting from state s, taking action a, and following an optimal policy thereafter.

The Q-Learning algorithm employs an iterative approach to update the Q-values based on the agent's experiences. At each time step, the agent selects an action based on an exploration-exploitation strategy, such as ϵ-greedy, which balances between taking actions with the highest Q-value (exploitation) and exploring other actions (exploration). This ϵ-

greedy is our behaviour policy that the agent uses to determine its action(behaviour) in a given state.

The Q-value update equation for Q-Learning is defined as follows:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \quad (1)$$

where s is the current state, a is the chosen action, r is the immediate reward received after taking action a in state s, s' is the next state after taking action a, α is the learning rate $(0 < \alpha < 1)$ that determines the weight given to the new information, and γ is the discount factor $(0 < \gamma < 1)$ that discounts the future rewards. The Target policy and behaviour policies are different as our Q-learning algorithm is an off-policy algorithm

The update equation utilizes the difference between the current Q-value estimate and the updated Q-value estimate based on the observed reward and the maximum Q-value of the next state-action pair. This is our Target policy that the agent uses to learn from the rewards received for its next state actions and this policy is greedy in this case as taking the maximum Q-value of the next state-action pair. The learning rate controls the step size of the update, while the discount factor balances the importance of immediate rewards versus future rewards.

The Q-Learning algorithm continues to interact with the environment, updating the Q-values based on observed rewards and transitioning to new states until it converges to an optimal policy. Convergence is achieved when the Q-values stabilize and no further significant updates occur.

Q-Learning has been widely applied in various domains, demonstrating its effectiveness in learning optimal policies in complex environments. Its model-free nature and ability to handle large state spaces make it a valuable tool for reinforcement learning tasks.

B. *SARSA-Learning*

SARSA (State-Action-Reward-State-Action) is a model-free On Policy reinforcement learning algorithm that aims to learn an optimal policy by directly incorporating the exploration-exploitation trade-off during learning. Similar to Q-Learning, SARSA estimates the Q-values, representing the expected cumulative rewards for taking specific actions in each state. However, SARSA is an on-policy algorithm, meaning it learns and updates its Q-values based on the current state-action pair and the subsequent state-action pair.

The SARSA update equation is defined as follows:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma Q(s',a') - Q(s,a) \right] \quad (2)$$

where s is the current state, a is the chosen action, r is the immediate reward received after taking action a in state s, s' is the next state after taking action a, a' is the action chosen in the next state s', α is the learning rate $(0 < \alpha < 1)$ that determines the weight given to the new information, and γ is the discount factor $(0 < \gamma < 1)$ that discounts the future rewards.

The update equation calculates the difference between the current Q-value estimate and the updated Q-value estimate based on the observed reward, the Q-value of the next state-action pair. SARSA takes the action a' in the next state s' based on the same strategy used in selecting the current action a. As the Sarsa agent is an On Policy learner so the behaviour policy is the same as the Target policy. So, for both the times for the current action and the next state action, the Q-value estimate is based on the Max Q-value of the state-action pair.

The SARSA algorithm proceeds by iteratively updating the Q-values based on observed rewards, transitioning to new states, and selecting actions according to the exploration-exploitation strategy. The learning process continues until convergence, where the Q-values stabilize and no significant updates occur.

SARSA has been widely applied in various reinforcement learning tasks, showcasing its ability to learn optimal policies while adapting to changes in the environment. Its on-policy nature makes it suitable for environments with stochastic dynamics or when the agent needs to consider the immediate consequences of its actions.

III. OpenAI Gym Taxi Problem

We conducted our experiments on OpenAI's Gym Taxi Problem (version-3). In the Taxi grid world, there are four specified sites denoted by R(ed), G(reen), Y(ellow), and B(lue). The episode begins with the taxi arriving in a random square and the customer arriving at a random place. The taxi arrives at the passenger's location, picks up the passenger, travels to the passenger's destination (another of the four stated places), and then drops the customer off. The episode concludes after the passenger is dropped off.

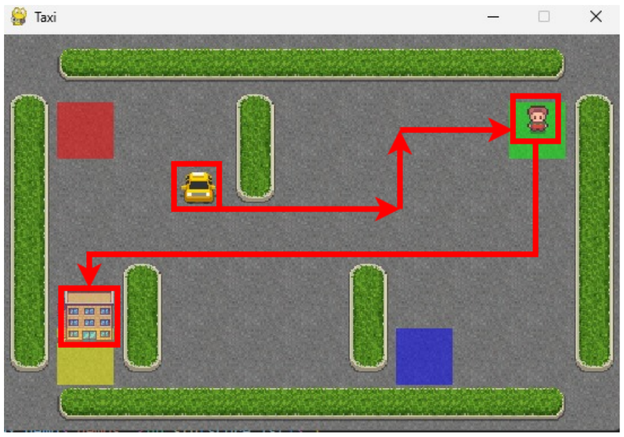

Fig. 1. Screenshot of the Taxi Problem Render

There are 500 discrete states since are 25 taxi positions, 5 potential passenger locations (including the scenario when the passenger is in the cab), and 4 destination locations, there are 500 distinct states.

It's worth noting that there are 400 states that may be visited during an episode. The missing states relate to occasions in which the passenger is in the same place as their destination, which usually indicates the conclusion of an episode. Four further stages can be seen immediately following a successful episode, when both the passenger and the cab arrive at their destination. This results in a total of 404 distinct states that can be reached.

There are a total of 6 actions, which were:

$$Actions = \{1, 2, 3, 4, 5, 6\}$$

where,

- 0: move south
- 1: move north
- 2: move east
- 3: move west
- 4: pickup passenger
- 5: drop off passenger

The reward space was as follows:

- -1 per step unless other reward is triggered.
- +20 delivering passenger.
- -10 executing "pickup" and "drop-off" actions illegally.

IV. Methodology and Experimental Design

To set up this study, we used the OpenAI GYM library in Python. Using the GYM, we instantiate an environment of Taxi Problems. The environment contains all the information about the environment such as its Markov Decision Process (MDP), transition probabilities and so on. Since we are doing the analysis between model-free environments, we did not need to extract its MDP information and transition probabilities.

We trained the agent for both Q-Learning and SARSA-Learning using a fixed number of episodes. We obtained an average reward for each experiment defined by a fixed number of episodes. And we ran multiple experiments with different numbers of episodes to train. Lastly, we plotted the average reward against a different number of training episodes.

V. Analysis

Q-Learning

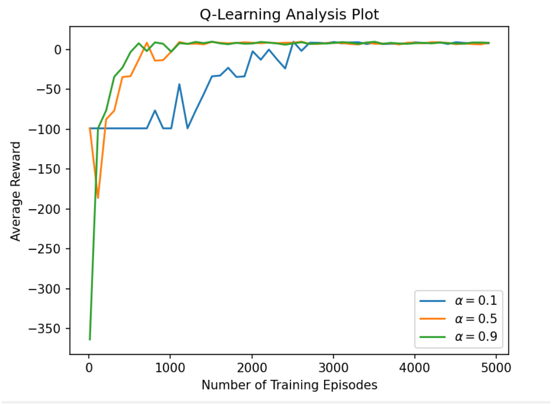

Fig. 2. Plot of Q-Learning Average Rewards vs Number of training Episodes for different values of α (Learning Rate)

Here we can see that $\alpha = 0.9$ converges the fastest, followed by $\alpha = 0.5$ and then $\alpha = 0.1$. This is because when $\alpha = 0.9$, the agent learns much faster as it weights updated target Q-table more than previous Q-table. Once every variant has converged, all of them have same average reward.

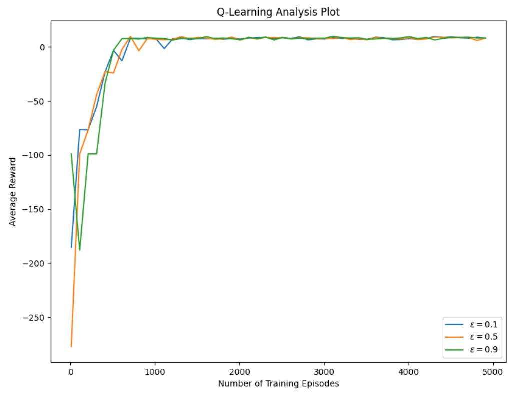

Fig. 3. Plot of Q-Learning Average Rewards vs Number of training Episodes for different values of ϵ (Exploration-Exploitation Rate)

Here we can see that all values of ϵ converges at the same time. This hence show that there is minimal effect of ϵ on the reward and convergence of the algorithm.

SARSA-Learning

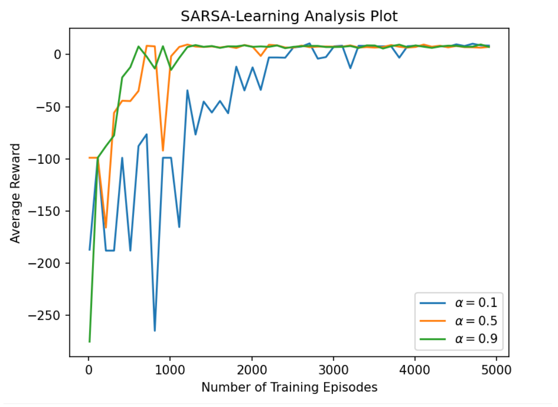

Fig. 4. Plot of SARSA-Learning Average Rewards vs Number of training Episodes for different values of α (Learning Rate)

Here we can see that $\alpha = 0.9$ converges the fastest, followed by $\alpha = 0.5$ and then $\alpha = 0.1$ which converges around 3000 episodes. This is because when $\alpha = 0.9$, the agent learns much faster as it weights updated target Q-table more than previous Q-table. Once every variant has converged, all of them have same average reward.

Comparative Analysis

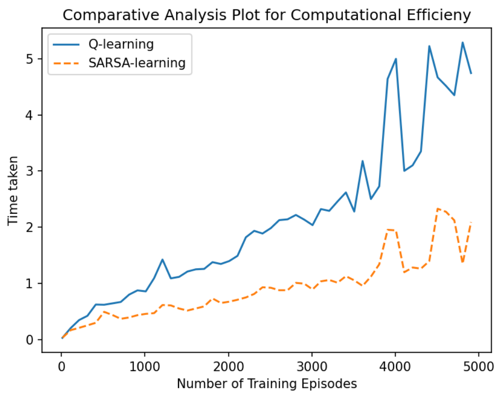

Fig. 5. Plot of Comparative Analysis between SARSA-Learning and Q-Learning in terms of Time taken to run a certain number of training Episodes.

From this plot we can observe that SARSA-Learning outperforms Q-Learning in time complexity by a significant margin.

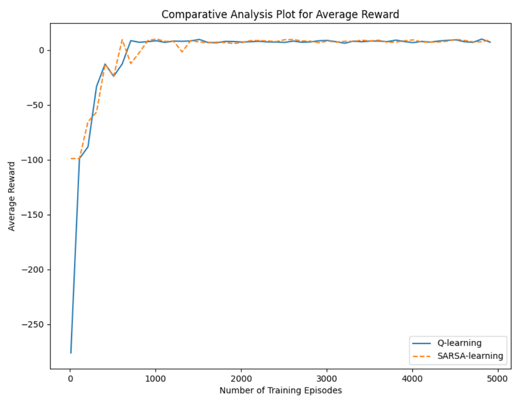

Fig. 6. Plot of Comparative Analysis between SARSA-Learning and Q-Learning in terms of Average Rewards and Number of training Episodes.

From this plot we can observe that both SARSA-Learning and Q-Learning have same convergence and their max rewards are also same. Hence we can see no significant difference between both in terms of Average Rewards and Number of training Episodes.

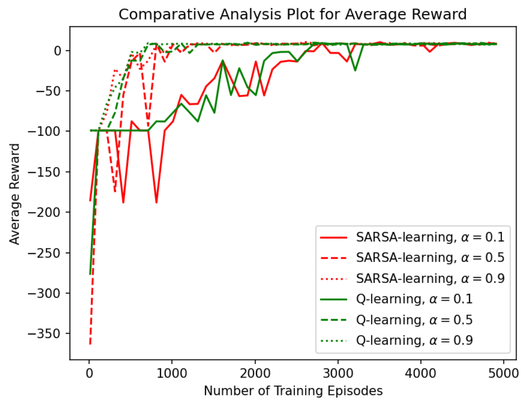

Fig. 7. Plot of Comparative Analysis between SARSA-Learning and Q-Learning in terms of Average Rewards and Number of training Episodes with values of α (Learning Rate)

Here in this plot we can observe that Q-Learning and SARSA-Learning with $\alpha = 0.1$ has the slowest convergence. Q-Learning and SARSA-Learning with $\alpha = 0.5$ has the fastest convergence.

VI. Conclusion:

In conclusion, our project demonstrated the effectiveness of SARSA and Q-learning algorithms for solving the Open AI Gym Taxi Toytext problem. We found that SARSA is faster and more computationally efficient than Q-learning in some scenarios while keeping the learning rate high can lead to faster convergence.

These findings can be applied to other reinforcement learning problems and can help inform the selection of appropriate algorithms for different scenarios. We hope that our project will contribute to the wider community of researchers and practitioners working in the field of reinforcement learning and inspire further exploration and development of these techniques.

Overall, this project provided us with valuable insights into the capabilities and limitations of SARSA and Q-learning algorithms and allowed us to gain a deeper understanding of reinforcement learning in general. We look forward to further exploring this exciting field and building on the knowledge and experience gained through this project.

References

[1] Sutton, R.S. and Barto, A.G. (1998) Reinforcement Learning: An Introduction. Vol. 1, MIT press, Cambridge
[2] https://www.gymlibrary.dev/environments/toy_text/taxi/