Assignment 1 - Defining & Solving RL Environments

Arjun Ramesh Kaushik
UB ID - 50413327
Department of Computer Science & Engineering
kaushik3@buffalo.edu
https://github.com/arjunrkaushik/Reinforcement_Learning_Fall22

Abstract

The goal of the assignment is to acquire experience in defining and solving reinforcement learning environments, following OpenAI Gym standards. The assignment consists of two parts. The first focuses on defining deterministic and stochastic environments that are based on Markov decision process. In the second part we will apply two tabular methods to solve environments that were previously defined.

1 Defining RL Environments

1.1 Describe the deterministic and stochastic environments, which were defined (set of actions/states/rewards,main objective, etc).

The objective of our experiment is to teach Lionel Messi(**Agent**) the optimal path to win the Qatar World Cup(**Terminal position/End goal**). We have used the same set of States, Actions and Rewards for both Deterministic and Stochastic Environments.

States(S) = { (0,0), (0,1), (0,2), (0,3), (1,0), (1,1), (1,2), (1,3), (2,0), (2,1), (2,2), (2,3), (3,0), (3,1), (3,2), (3,3) }

Actions(A) = { Up, Down, Left, Right }

Rewards(R) = { -10, -5, -3, 0, +5, +10, +20 }

1.2 Provide visualizations of your environments

Figure 1: Agent takes "right" and "up" action in sequence

1.3 How did you define the stochastic environment?

At each timestep, a random action is picked. We have another variable - **temp** - defined inside the environment that picks a random value between 0 and 1. If **temp** is greater than **0.3**, then the agent sticks with the random action that was chosen above. Else, the agent performs any action that is picked randomly from the **Actions Set excluding the previously picked action**. This stochasticity is only present in 2 of 16 possible states, at positions of the **Missed Shot**.

1.4 What is the difference between the deterministic and stochastic environments?

Deterministic Environment	**Stochastic Environment**
The next state of the agent, given a state and action, is always certain at any point in time A real life example would be - traffic signal. There is a certainty with respect to the action taken by the agent. Red signals stop. Green signals go. $P(s', r \| s, a) = \{0, 1\}$	For a given action and state, there's no certainty regarding the agent's next state A real life example would be - playing soccer.The next action and thereby the next state of a soccer player is unpredictable. $\sum_{s', r} P(s', r \| s, a) = 1$

1.5 Safety in AI: Write a brief review (5 sentences) explaining how you ensure the safety of your environments. E.g. how do you ensure that agent choose only actions that are allowed, that agent is navigating within defined state-space, etc

We ensure that the agent is navigating within the defined state-space by using the **np.clip()** function. In our environment, its used as - **np.clip(agent_pos,0,3)**. This keeps the row and column values of the agent between 0 and 3.

Our Action set consists of only 4 actions - {Left, Right, Up, Down} - which are denoted by integers between 0 and 3. The agent is made to pick a random action(integer) by using the **np.random.choice()** function. As an example, **np.random.choice(5,3)** picks 3 integers between 0 and 5.

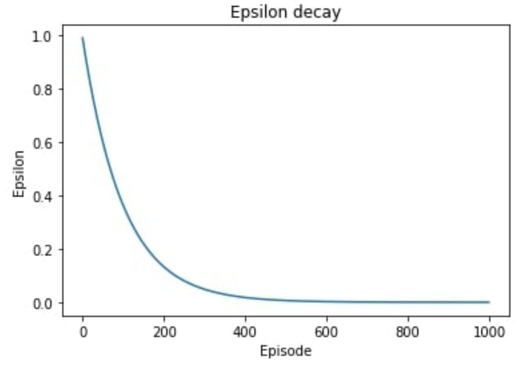
2 Applying Tabular Methods

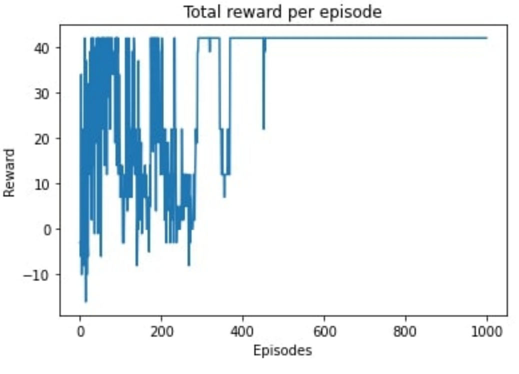
2.1 Training Results

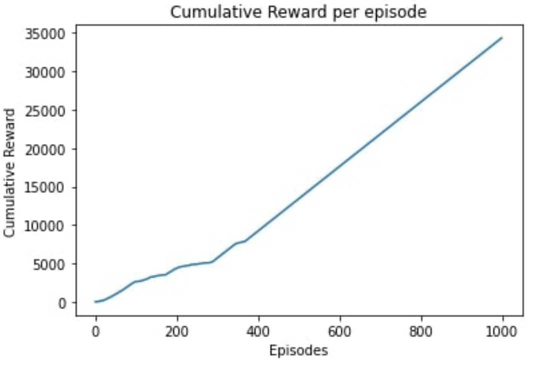
In each of the 4 cases - Q-learning in Deterministic Environment, Q-learning in Stochastic Environment, SARSA in Deterministic Environment and SARSA in Stochastic Environment - the results are as expected. The last row of the Q-tables is filled with 0's since it corresponds to the terminal state. The epsilon decay graph in each case follows a exponential decay trend. The total reward per episode graph increases and stabilizes towards the end, this shows that the agent has learnt to optimize its path to collect maximum reward. A linearly increasing trend on cumulative reward per episode graph is a healthy indication of an agent picking positive rewards and learning.

2.1.1 Q-learning in Deterministic Environment

```
Training Agent using Q-learning in Deterministic Environment

Number of times agent REACHED goal: 799
Number of times agent DID NOT REACH goal: 201
Total reward at the end of training: 34345

Q-table

[[ 4.04567611e+01  1.57753830e+01  1.69588673e+01  1.63941861e+01]
 [ 1.33308815e+01  1.61139595e+01  7.74333511e+00  2.24922494e+01]
 [-7.67010682e-01  3.39438381e+00  3.18924566e-01  1.99557261e+01]
 [ 7.94203255e-02  9.43195534e-02  2.15574887e-01  1.09585253e+00]
 [ 3.58149102e+01  1.55684888e+01  1.59877513e+01  1.83532151e+01]
 [ 1.31091531e+01  1.21314550e+01 -4.06235162e-01  1.91335157e+01]
 [-2.60085808e+00  1.74511396e+01  4.63246300e-02  8.58831649e-01]
 [-9.16837255e-02  1.16446818e-02  6.68858705e-02  4.84939316e-01]
 [ 3.92069800e+01  1.90510763e+01  1.79674766e+01  1.90080605e+01]
 [ 1.94864368e+01  1.62304658e+01  1.67548113e+01  1.73169474e+01]
 [ 2.01787068e+01  6.42438375e-01  2.60594818e+00  1.44847460e+01]
 [ 9.37118000e+00  2.10696778e-03  7.14798441e-01  6.02660683e+00]
 [ 1.78639071e+01  2.15490794e+01  2.95020000e+01  1.75573333e+01]
 [ 1.77717664e+01  1.67325028e+01  2.98000000e+01  1.80882133e+01]
 [ 1.98011463e+01  1.90472382e+01  2.00000000e+01  1.99215835e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

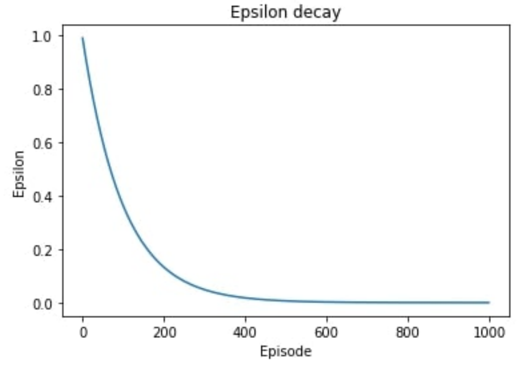


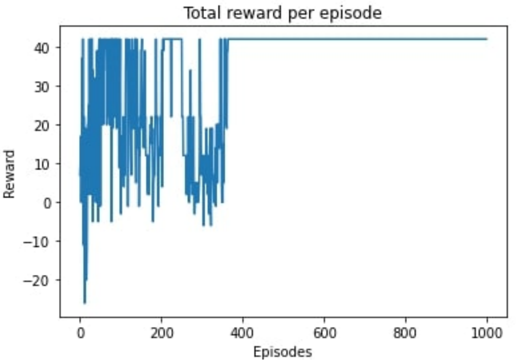


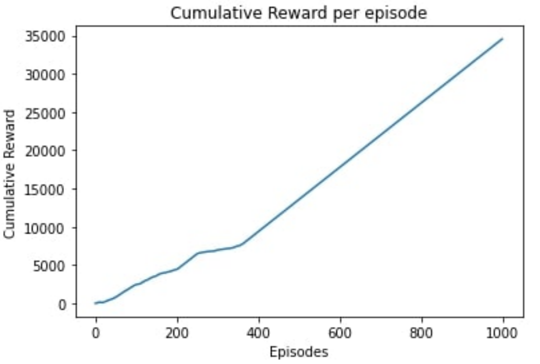


2.1.2 Q-learning in Stochastic Environment

```
Training Agent using Q-learning in Stochastic Environment

Number of times agent REACHED goal: 823
Number of times agent DID NOT REACH goal: 177
Total reward at the end of training: 34522

Q-table

[[ 4.04567611e+01  1.68973603e+01  1.54974689e+01  1.55144109e+01]
 [ 1.29042002e+01  1.48167398e+01  1.18896687e+01  1.53765419e+01]
 [-6.09231994e-01  2.75734086e+00  8.58974778e-01  1.92226544e+01]
 [ 4.16689672e-02  1.43813704e-01  2.24408725e-01  5.44756344e+00]
 [ 3.58149102e+01  1.48147782e+01  1.55711490e+01  1.72897541e+01]
 [ 1.92646930e+01  8.87666528e+00 -1.31366305e+00  8.61274218e+00]
 [-2.17285541e-01  1.62980352e+01  1.68697044e-01  5.76976363e+00]
 [ 1.87034580e-02  2.88388342e+00  1.29914439e-01  9.05102927e-01]
 [ 3.92069800e+01  1.69429651e+01  1.69017655e+01  1.70935515e+01]
 [ 1.78976325e+01  1.74716172e+01  1.83378269e+01  1.73865338e+01]
 [ 2.23840325e+01 -3.01991719e-01  4.68962444e+00  1.45834383e+01]
 [ 1.25305885e+01  1.50960872e-02  7.14780000e-01 -2.24570529e+00]
 [ 1.70275320e+01  1.79743640e+01  2.95020000e+01  1.69496271e+01]
 [ 2.00864259e+01  1.90066924e+01  2.98000000e+01  1.82681222e+01]
 [ 1.88475370e+01  1.96396911e+01  2.00000000e+01  1.99519641e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

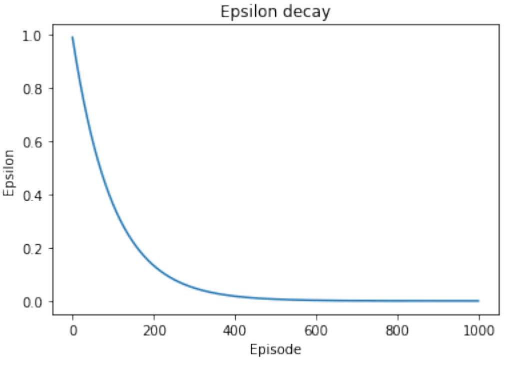


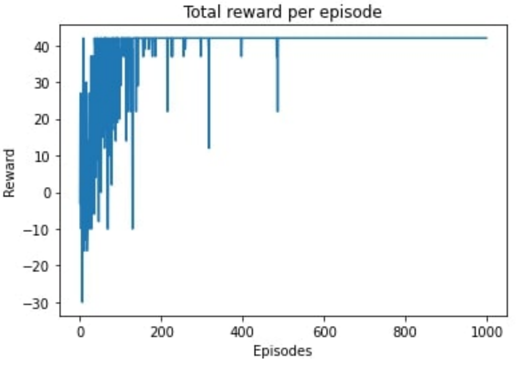


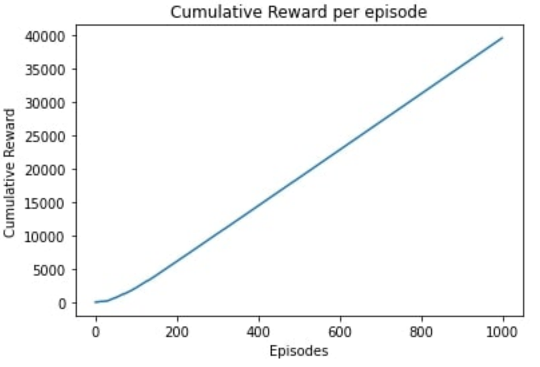


2.1.3 SARSA in Deterministic Environment

```
Training Agent using SARSA in Deterministic Environment

Number of times agent REACHED goal: 983
Number of times agent DID NOT REACH goal: 17
Total reward at the end of training: 39595

Q-table

[[ 4.04567611e+01  1.68973603e+01  1.54974689e+01  1.55144109e+01]
 [ 1.29042002e+01  1.48167398e+01  1.18896687e+01  1.53765419e+01]
 [-6.09231994e-01  2.75734086e+00  8.58974778e-01  1.92226544e+01]
 [ 4.16689672e-02  1.43813704e-01  2.24408725e-01  5.44756344e+00]
 [ 3.58149102e+01  1.48147782e+01  1.55711490e+01  1.72897541e+01]
 [ 1.92646930e+01  8.87666528e+00 -1.31366305e+00  8.61274218e+00]
 [-2.17285541e-01  1.62980352e+01  1.68697044e-01  5.76976363e+00]
 [ 1.87034580e-02  2.88388342e+00  1.29914439e-01  9.05102927e-01]
 [ 3.92069800e+01  1.69429651e+01  1.69017655e+01  1.70935515e+01]
 [ 1.78976325e+01  1.74716172e+01  1.83378269e+01  1.73865338e+01]
 [ 2.23840325e+01 -3.01991719e-01  4.68962444e+00  1.45834383e+01]
 [ 1.25305885e+01  1.50960872e-02  7.14780000e-01 -2.24570529e+00]
 [ 1.70275320e+01  1.79743640e+01  2.95020000e+01  1.69496271e+01]
 [ 2.00864259e+01  1.90066924e+01  2.98000000e+01  1.82681222e+01]
 [ 1.88475370e+01  1.96396911e+01  2.00000000e+01  1.99519641e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

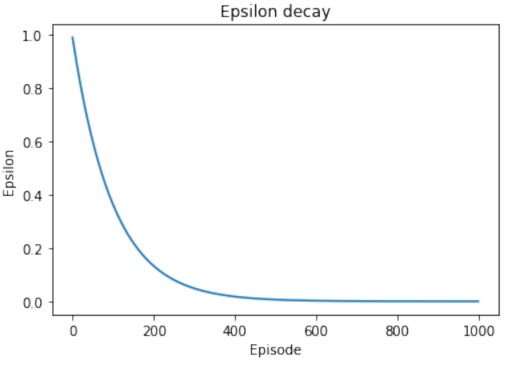


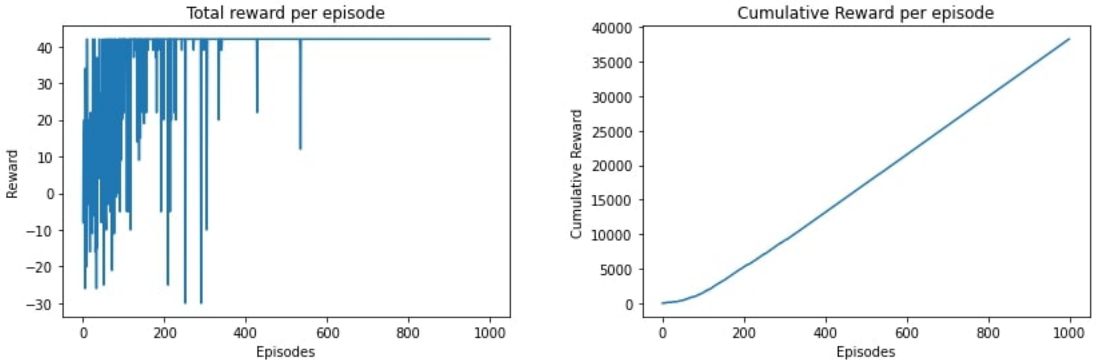




2.1.4 SARSA in Stochastic Environment

```
Training Agent using SARSA in Stochastic Environment

Number of times agent REACHED goal: 971
Number of times agent DID NOT REACH goal: 29
Total reward at the end of training: 38289

Q-table

[[ 4.04567611e+01  1.68973603e+01  1.54974689e+01  1.55144109e+01]
 [ 1.29042002e+01  1.48167398e+01  1.18896687e+01  1.53765419e+01]
 [-6.09231994e-01  2.75734086e+00  8.58974778e-01  1.92226544e+01]
 [ 4.16689672e-02  1.43813704e-01  2.24408725e-01  5.44756344e+00]
 [ 3.58149102e+01  1.48147782e+01  1.55711490e+01  1.72897541e+01]
 [ 1.92646930e+01  8.87666528e+00 -1.31366305e+00  8.61274218e+00]
 [-2.17285541e-01  1.62980352e+01  1.68697044e-01  5.76976363e+00]
 [ 1.87034580e-02  2.88388342e+00  1.29914439e-01  9.05102927e-01]
 [ 3.92069800e+01  1.69429651e+01  1.69017655e+01  1.70935515e+01]
 [ 1.78976325e+01  1.74716172e+01  1.83378269e+01  1.73865338e+01]
 [ 2.23840325e+01 -3.01991719e-01  4.68962444e+00  1.45834383e+01]
 [ 1.25305885e+01  1.50960872e-02  7.14780000e-01 -2.24570529e+00]
 [ 1.70275320e+01  1.79743640e+01  2.95020000e+01  1.69496271e+01]
 [ 2.00864259e+01  1.90066924e+01  2.98000000e+01  1.82681222e+01]
 [ 1.88475370e+01  1.96396911e+01  2.00000000e+01  1.99519641e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```



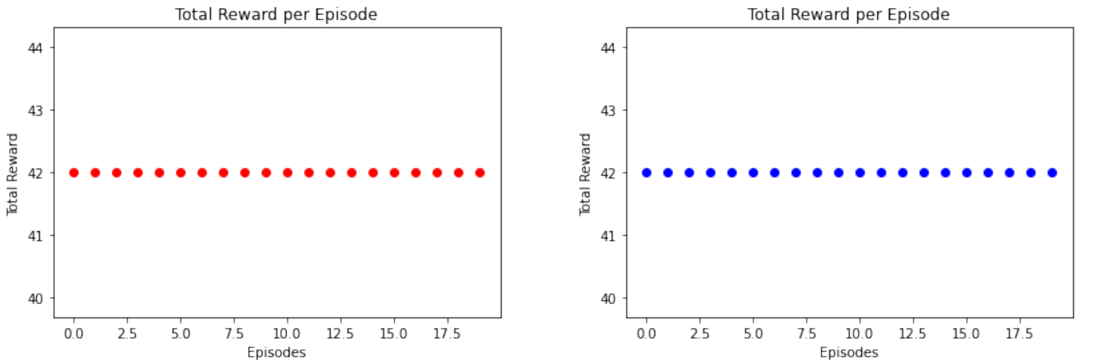


2.2 Evaluation of Agent in Deterministic and Stochastic Environments

For each Reinforcement learning algorithm, we run the agent in each Environment for 20 episodes. We find that, in each of the 4 cases, the agent picks up 42 points as reward consistently. This shows that the agent has learnt well and is picking maximum rewards everytime.

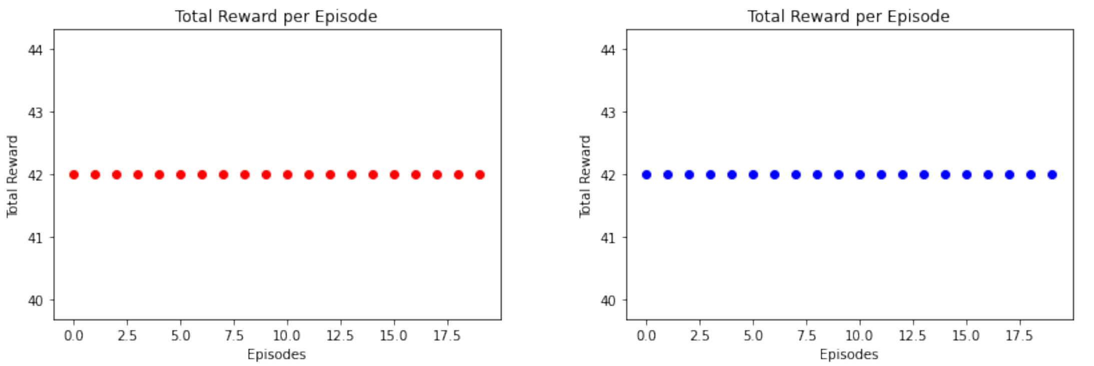
2.2.1 Q-learning

Deterministic Environment Stochastic Environment



2.2.2 SARSA

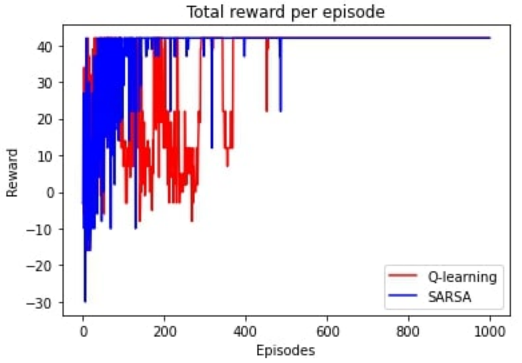
Deterministic Environment Stochastic Environment

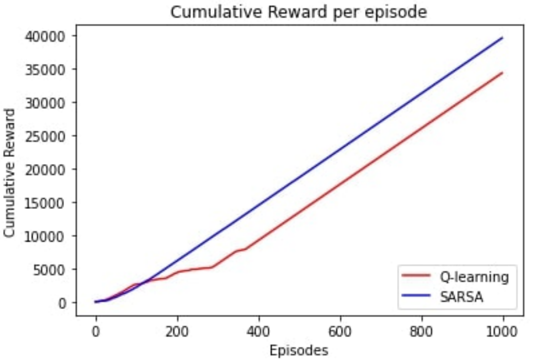


2.3 Performance of algorithms

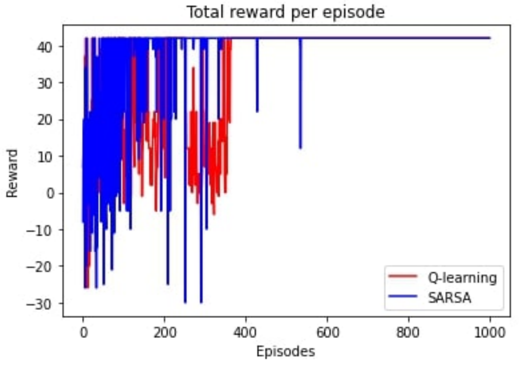
Here we compare the performance of Q-learning and SARSA in both Deterministic and Stochastic environments using their reward dynamics. From the obtained graphs, we can see that the agent learns faster and better while using SARSA.

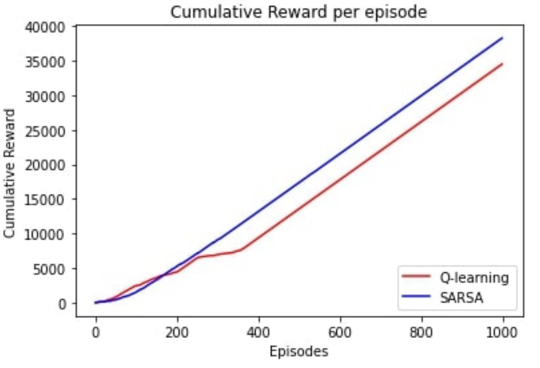
2.3.1 Deterministic Environment





2.3.2 Stochastic Environment





2.4 Reinforcement Learning Algorithms

2.4.1 Q-learning

Q-learning is a model-free and off-policy Reinforcement Learning algorithm. In this algorithm, the agent maintains a Q-table of "S" rows and "A" columns, where "S" corresponds to number of states and "A" corresponds to number of actions. The agent picks maximum expected future reward of the next state, irrespective of the next action, while calculating the Q-value.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma max_{a'} Q(s_{t+1}, a^{'}) - Q(s_t, a_t) \right]$$

2.4.2 SARSA

Q-learning is a model-free and on-policy Reinforcement Learning algorithm. In this algorithm, the agent maintains a Q-table of "S" rows and "A" columns, where "S" corresponds to number of states and "A" corresponds to number of actions. The agent picks the expected future reward of the next state, with respect to the next action, while calculating the Q-value.

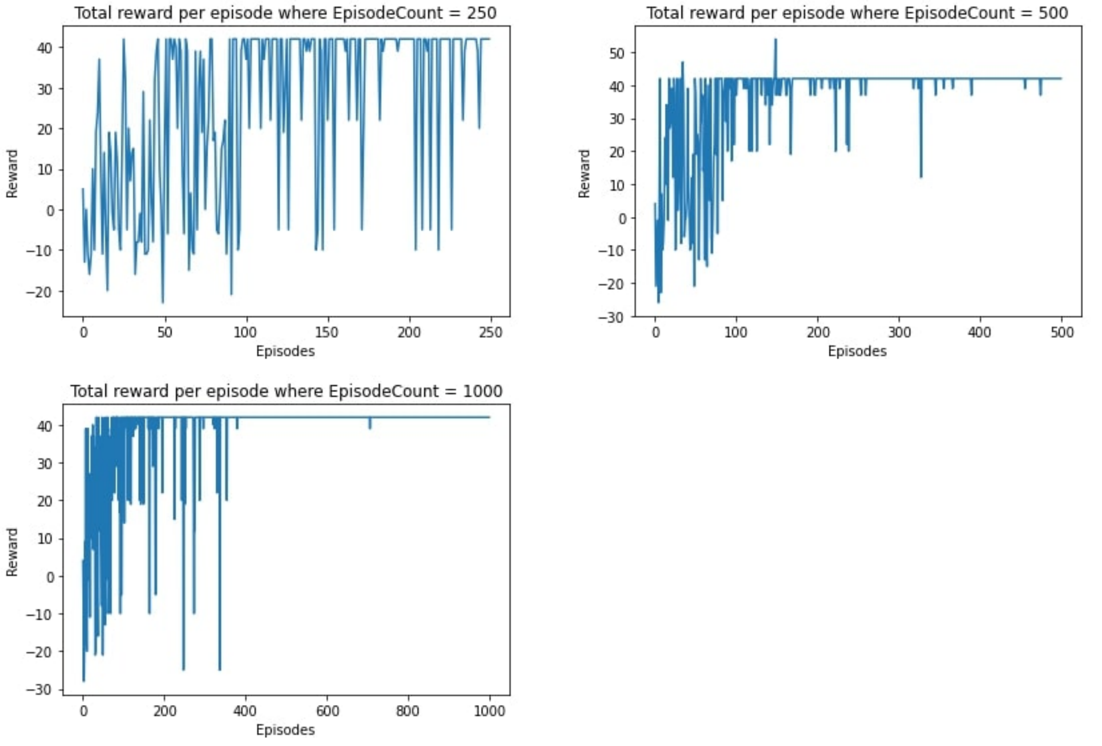
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right]$$

2.5 Hyperparameter Tuning

In this section, we explore the changes in training results when the discount factor and number of episodes are changed for a SARSA agent in the Stochastic Environment.

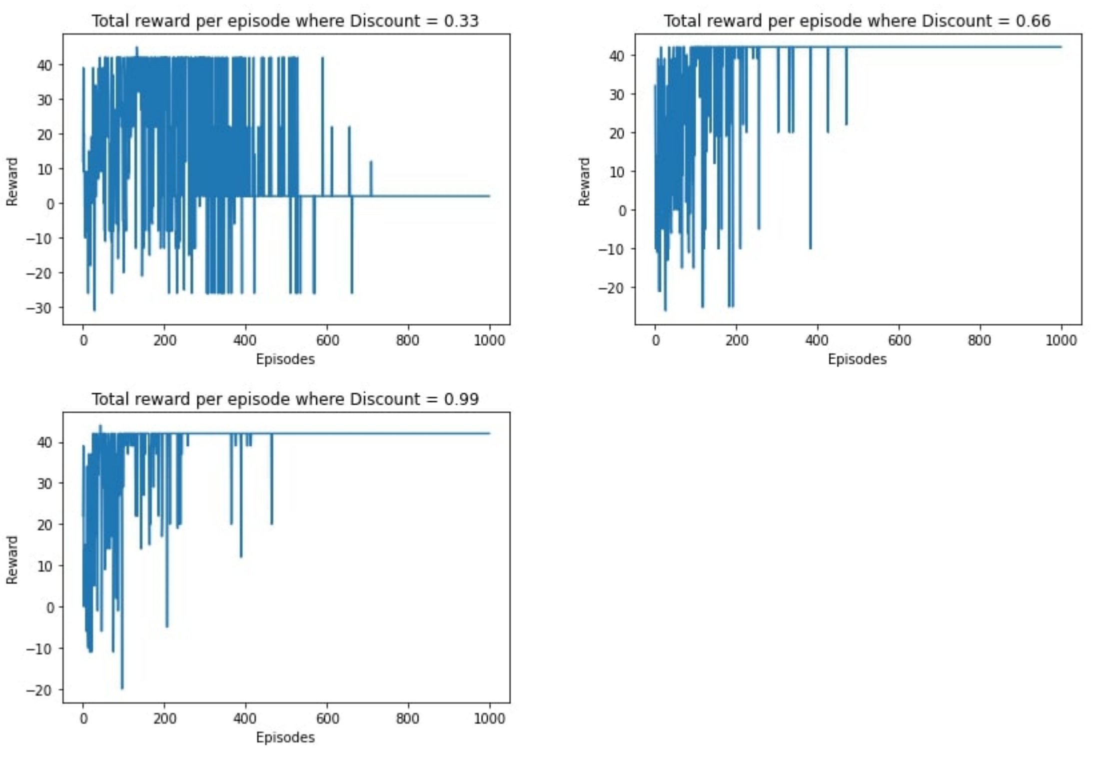
2.5.1 Number of Episodes

From the training results, we see that a higher value in the number of episodes leads to better training of the agent. The agent is able to learn better, and collect the maximum reward consistently when episode count is incremented. **Most efficient value for number of episodes is 1000.**



2.5.2 Discount Factor

We can observe that lower the discount factor, longer the agent takes to learn and stabilize on the maximum reward. **Most efficient value for discount factor is 0.99.**



3 Changes since Checkpoint

1. Added visual effects to environment

2. Added new rewards
3. Changed the number of stochastic states from 3 to 2

4 References

1. Class lectures and notes
2. https://stackoverflow.com/
3. https://matplotlib.org/