CSE 574 D: INTRODUCTION TO MACHINE LEARNING

ASSIGNMENT-3

Part-I

1. **RL Environment:**

   Theme: Lawnmower Grid World

   States: There are 16 states (4*4 grid) - {S1 = (0,0), S2 = (0,1), S3 = (0,2), S4 = (0,3), S5 = (1,0), S6 = (1,1), S7 = (1,2), S8 = (1,3), S9 = (2,0), S10 = (2,1), S11 = (2,2), S12 = (2,3), S13 = (3,0), S14 = (3,1), S15 = (3,2), S16 = (3,3)}

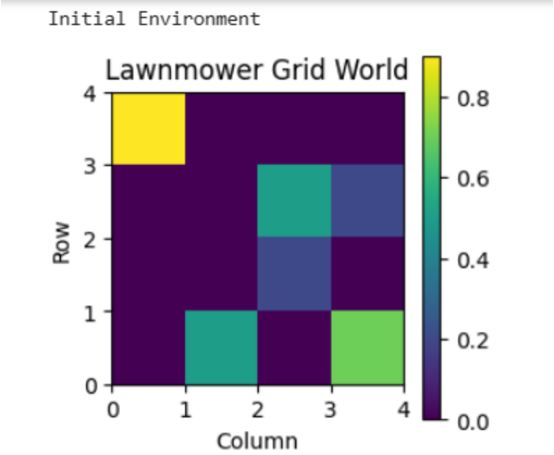
   Actions: Four actions – {Up, Down, Right, Left}

   Rewards: Goal state = 10, two positive rewards = 5 each, two negative rewards = -6 each
   {5, 5, -6, -6, 10}

   Objective: The main objective is for the agent to move in the grid world reaching the goal state to receive a maximum positive reward by avoiding the negative rewards.

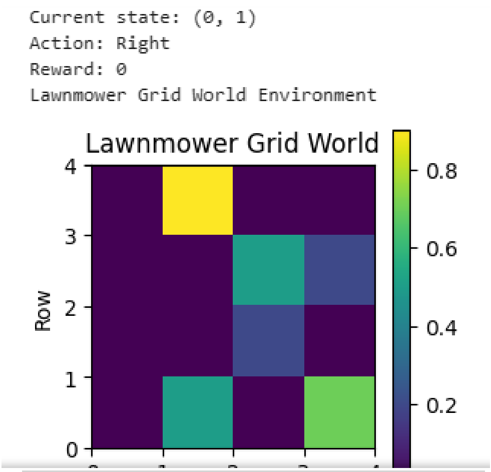
   - Also, the termination conditions are as follows-
     - If the goal state is reached
     - If the max steps given is exceeded
   - The rewards will not be null once after collected. They will remain the same.

2. In the below visualized environment, the goal state is marked as green, positive rewards with teal, negative rewards with blue, and the agent's current position is denoted by yellow.

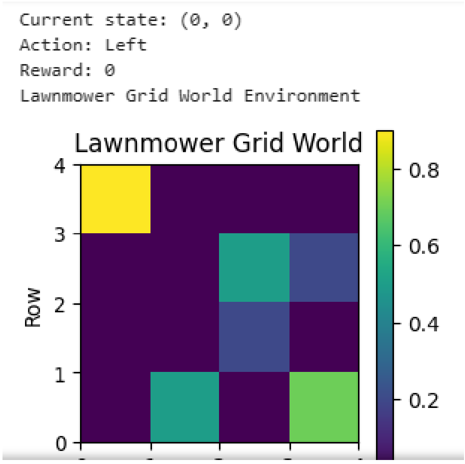


```
Initial Environment
```

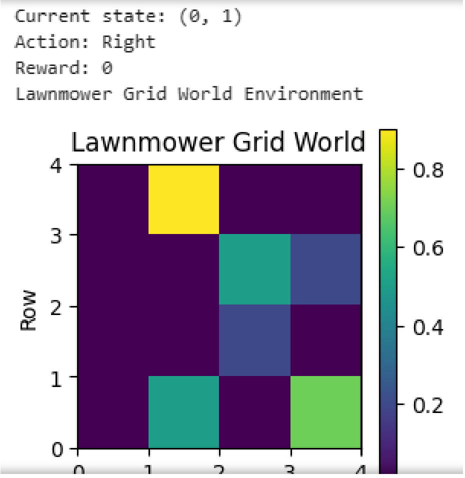
Ten random steps visualization:



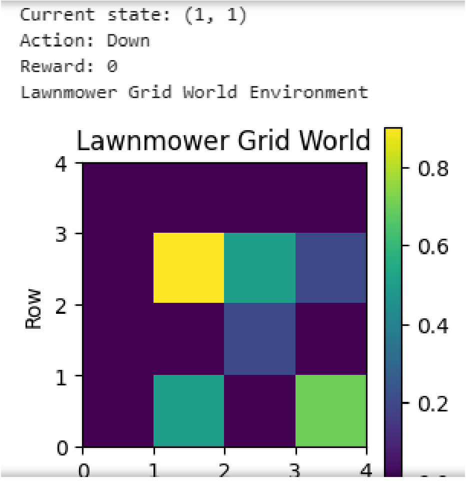
```
Current state: (0, 1)
Action: Right
Reward: 0
Lawnmower Grid World Environment
```



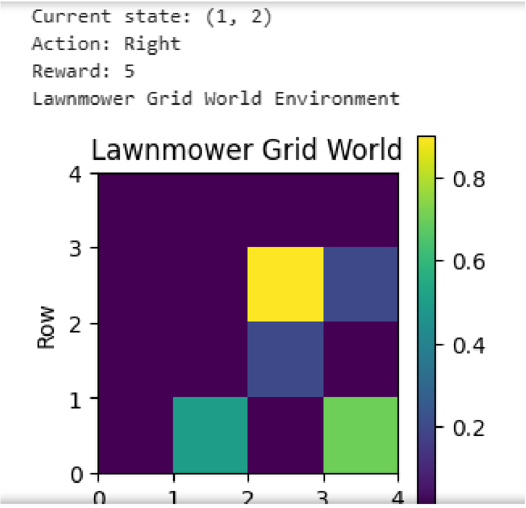
```
Current state: (0, 0)
Action: Left
Reward: 0
Lawnmower Grid World Environment
```



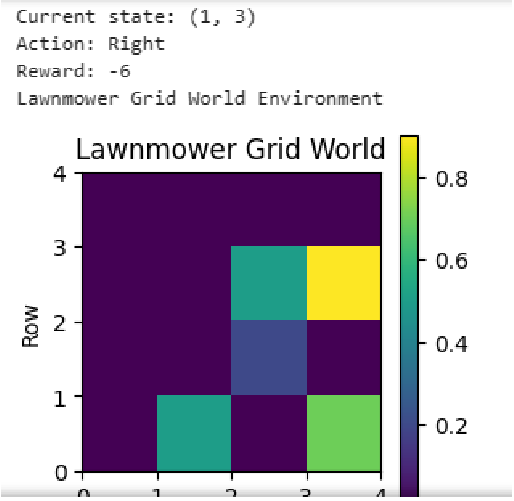
```
Current state: (0, 1)
Action: Right
Reward: 0
Lawnmower Grid World Environment
```



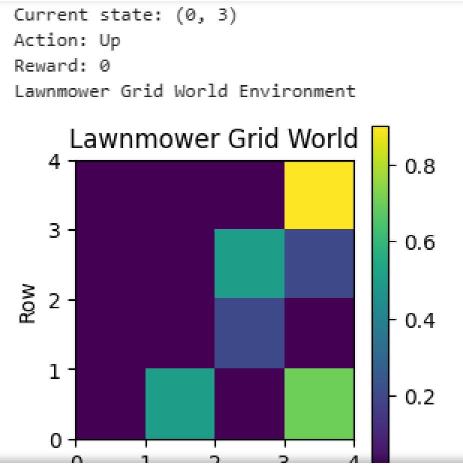
```
Current state: (1, 1)
Action: Down
Reward: 0
Lawnmower Grid World Environment
```



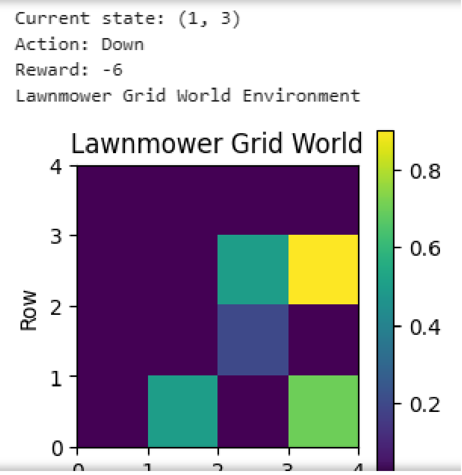
```
Current state: (1, 2)
Action: Right
Reward: 5
Lawnmower Grid World Environment
```



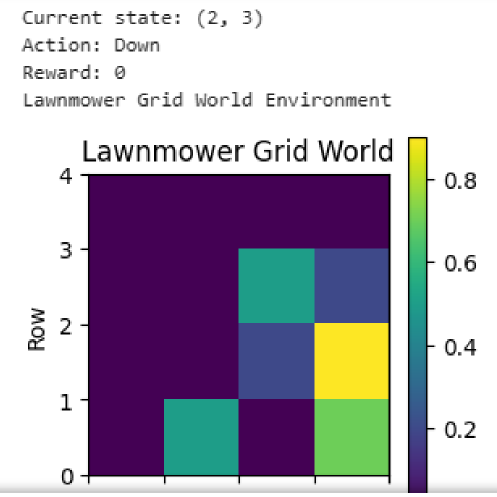
```
Current state: (1, 3)
Action: Right
Reward: -6
Lawnmower Grid World Environment
```



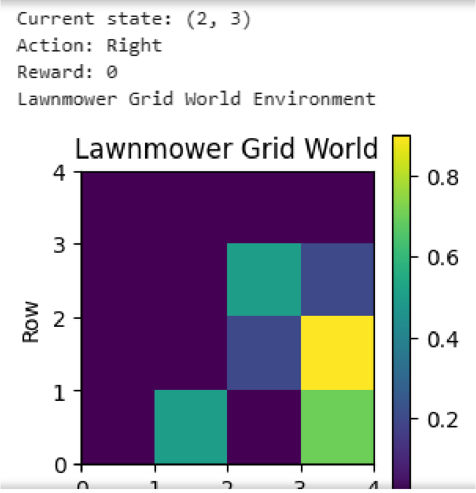
```
Current state: (0, 3)
Action: Up
Reward: 0
Lawnmower Grid World Environment
```



```
Current state: (1, 3)
Action: Down
Reward: -6
Lawnmower Grid World Environment
```



```
Current state: (2, 3)
Action: Down
Reward: 0
Lawnmower Grid World Environment
```



```
Current state: (2, 3)
Action: Right
Reward: 0
Lawnmower Grid World Environment
```

3. Safety in AI:

- The environment restricts the agent to stay within the defined grid boundaries, preventing it from going out of boundary.
- The environment also checks the validity of actions before applying them. If it is an invalid action, it will raise a ValueError.
- The environment also provides a transparent interface for actions through the ‘step’ and ‘reset’ methods. It also checks for terminal conditions to ensure the agent’s safety.
- The state space is discretized and the ‘state_to_index’ method makes sure that the agent navigates within the defined state space.
- The environment also enforces a maximum step limit, preventing prolonged interactions and enhancing the safety of our agent by ensuring the controlled exploration.

Part-II and Part - III

1. **Sarsa Agent:**

Update Function:

- This agent chooses actions based on an epsilon-greedy policy.
- This agent uses the SARSA (State Action Reward State Action) algorithm.
- The Q-values are updates using the formula –

  $Q(s,a) \leftarrow Q(s,a) + \text{alpha}[r + \text{gamma} * Q(s', a') – Q(s,a)]$

  where s is current state, a is current action, s' is next state, a' is next action, r is reward, alpha is learning rate and gamma is discount factor.

Key Features:

- Simple
- Easy to implement
- Balances exploration-exploitation using epsilon-greedy strategy
- Suitable for episodic tasks

Advantage:

- Converges for certain problems, is model-free, and well-suited for online learning.
- Guarateed Policy improvement
- On-policy learning ensures safety in real-world applications

Disadvantages:

- Slow convergency because of on-policy learning
- Sensitive to hyperparameter choices
- Exploration-exploitation trade-off challenges

Double Q-Agent:

Update Function:

- This agent chooses actions based on an epsilon-greedy policy.
- This agent uses the Double Q-Learning algorithm.
- The Q-values are updates using the formula –

  $Qi\ (s,a) \leftarrow Qi\ (s,a) + \text{alpha}[r + \text{gamma} * Qj\ (s', \text{argmax}\ a'\ Qi\ (s', a')) – Qi\ (s,a)]$

  where s is current state, a is current action, s' is next state, a' is next action, r is reward, alpha is learning rate, gamma is discount factor, i and j alternate between Q1 and Q2

Key Features:

- Improves stability
- Improves convergence

- Addresses overestimation bias present in traditional Q-Learning.

Advantages:

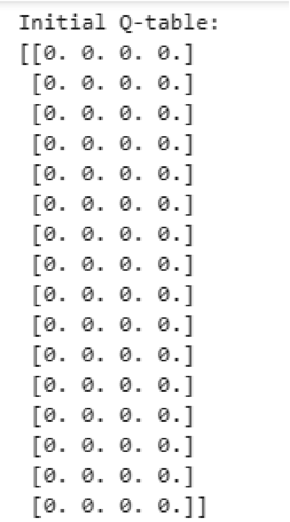
- Mitigates overestimation issues
- Enhances stability by using two sets of Q-values for better learning
- Mitigates risk of learning from noisy estimates
- Suitable for environments with high stochasticity

Disadvantages:

- Increased computational complexity due to maintaining two Q-tables
- Sensitivity to hyperparameter choices
- Requires more memory compared to standard Q-Learning

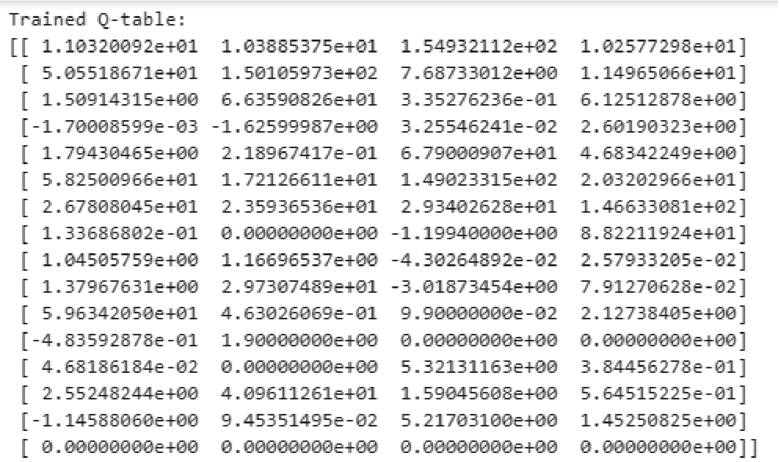
2. **Results after applying SARSA:**

Initial Q-table:



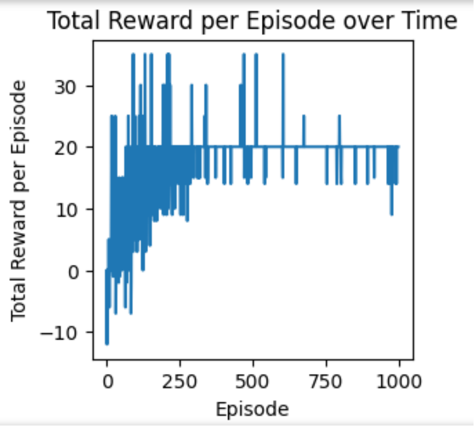
```
Initial Q-table:
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
```

Trained Q-table:

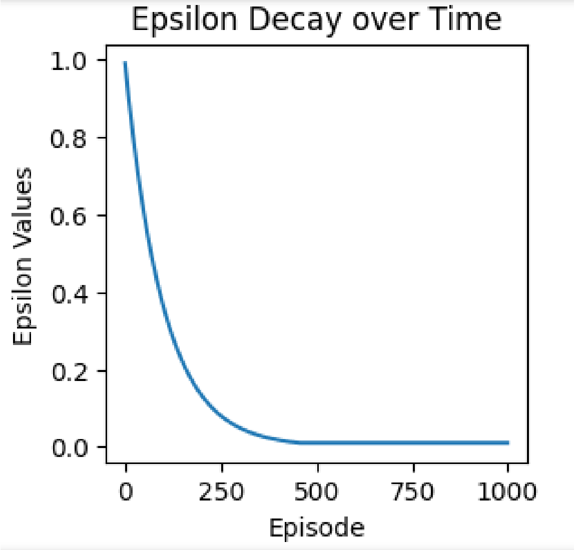


```
Trained Q-table:
[[ 1.10320092e+01  1.03885375e+01  1.54932112e+02  1.02577298e+01]
 [ 5.05518671e+01  1.50105973e+02  7.68733012e+00  1.14965066e+01]
 [ 1.50914315e+00  6.63590826e+01  3.35276236e-01  6.12512878e+00]
 [-1.70008599e-03 -1.62599987e+00  3.25546241e-02  2.60190323e+00]
 [ 1.79430465e+00  2.18967417e-01  6.79000907e+01  4.68342249e+00]
 [ 5.82500966e+01  1.72126611e+01  1.49023315e+02  2.03202966e+01]
 [ 2.67808045e+01  2.35936536e+01  2.93402628e+01  1.46633081e+02]
 [ 1.33686802e-01  0.00000000e+00 -1.19940000e+00  8.82211924e+01]
 [ 1.04505759e+00  1.16696537e+00 -4.30264892e-02  2.57933205e-02]
 [ 1.37967631e+00  2.97307489e+01 -3.01873454e+00  7.91270628e-02]
 [ 5.96342050e+01  4.63026069e-01  9.90000000e-02  2.12738405e+00]
 [-4.83592878e-01  1.90000000e+00  0.00000000e+00  0.00000000e+00]
 [ 4.68186184e-02  0.00000000e+00  5.32131163e+00  3.84456278e-01]
 [ 2.55248244e+00  4.09611261e+01  1.59045608e+00  5.64515225e-01]
 [-1.14588060e+00  9.45351495e-02  5.21703100e+00  1.45250825e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

Total Rewards per Episode plot:



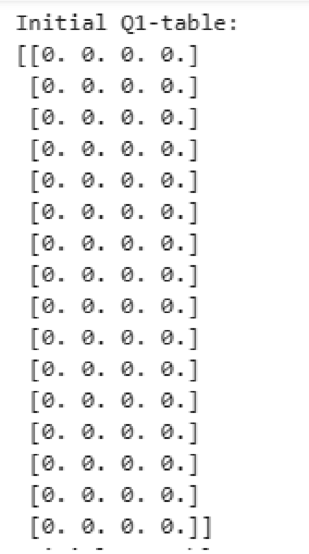
Epsilon Decay plot:



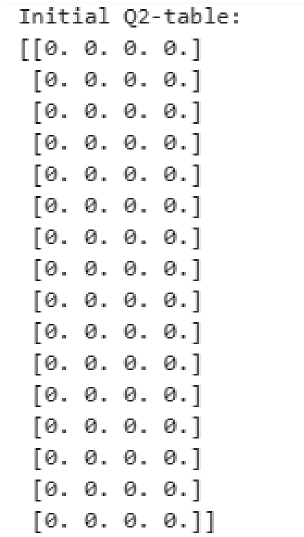
From the above results, it suggests that the SARSA agent successfully learned a policy that balances exploration and exploitation, leading to a Q-table that captures the expected rewards for different state-action pairs. The decreasing epsilon curve indicates a natural progression towards a more focused and refined policy over time. The positive Q-values in the various state-action pairs indicate that the agent has found effective strategy for navigating and achieveing rewards in the given environment.

Results after applying Double Q-Learning:

Initial Q1 and Q2 table:

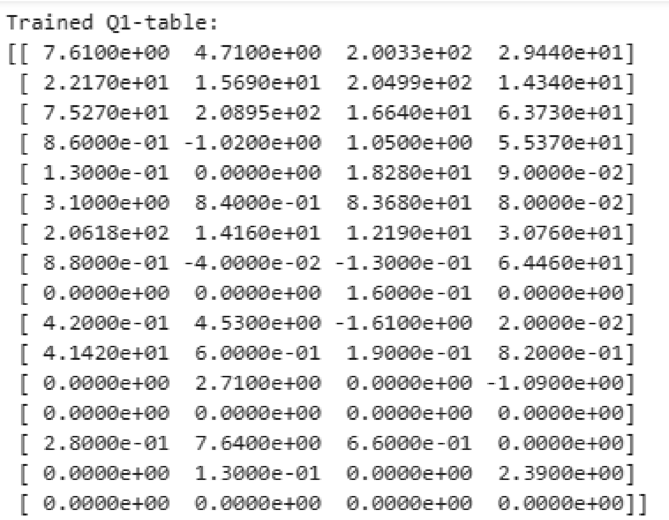


```
Initial Q1-table:
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
```



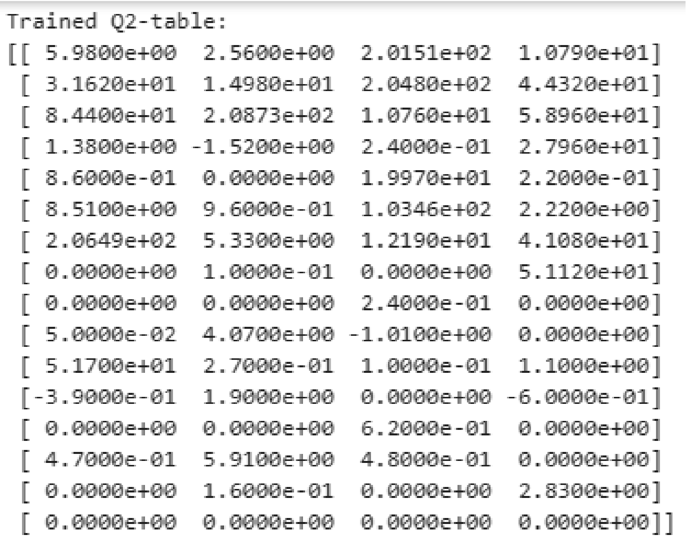
```
Initial Q2-table:
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
```

Trained Q1-table:



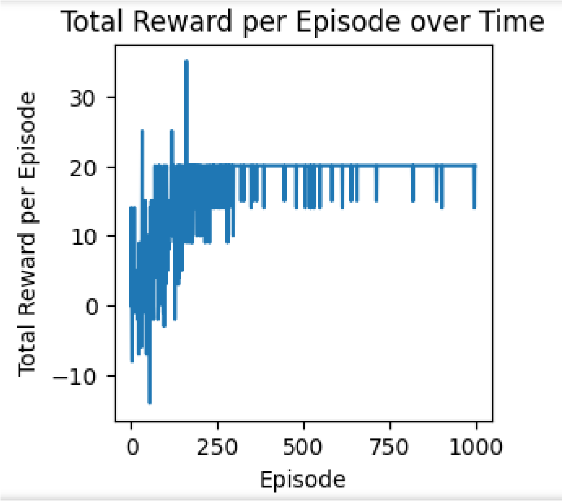
```
Trained Q1-table:
[[ 7.6100e+00  4.7100e+00  2.0033e+02  2.9440e+01]
 [ 2.2170e+01  1.5690e+01  2.0499e+02  1.4340e+01]
 [ 7.5270e+01  2.0895e+02  1.6640e+01  6.3730e+01]
 [ 8.6000e-01 -1.0200e+00  1.0500e+00  5.5370e+01]
 [ 1.3000e-01  0.0000e+00  1.8280e+01  9.0000e-02]
 [ 3.1000e+00  8.4000e-01  8.3680e+01  8.0000e-02]
 [ 2.0618e+02  1.4160e+01  1.2190e+01  3.0760e+01]
 [ 8.8000e-01 -4.0000e-02 -1.3000e-01  6.4460e+01]
 [ 0.0000e+00  0.0000e+00  1.6000e-01  0.0000e+00]
 [ 4.2000e-01  4.5300e+00 -1.6100e+00  2.0000e-02]
 [ 4.1420e+01  6.0000e-01  1.9000e-01  8.2000e-01]
 [ 0.0000e+00  2.7100e+00  0.0000e+00 -1.0900e+00]
 [ 0.0000e+00  0.0000e+00  0.0000e+00  0.0000e+00]
 [ 2.8000e-01  7.6400e+00  6.6000e-01  0.0000e+00]
 [ 0.0000e+00  1.3000e-01  0.0000e+00  2.3900e+00]
 [ 0.0000e+00  0.0000e+00  0.0000e+00  0.0000e+00]]
```

Trained Q2-table:

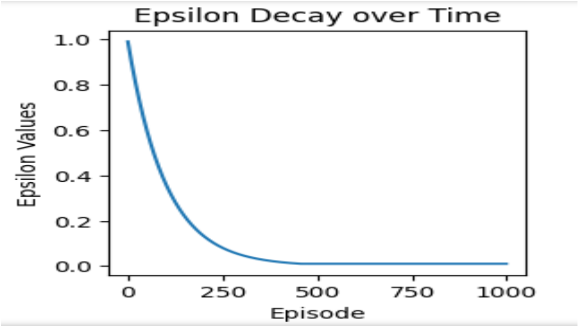


```
Trained Q2-table:
[[ 5.9800e+00  2.5600e+00  2.0151e+02  1.0790e+01]
 [ 3.1620e+01  1.4980e+01  2.0480e+02  4.4320e+01]
 [ 8.4400e+01  2.0873e+02  1.0760e+01  5.8960e+01]
 [ 1.3800e+00 -1.5200e+00  2.4000e-01  2.7960e+01]
 [ 8.6000e-01  0.0000e+00  1.9970e+01  2.2000e-01]
 [ 8.5100e+00  9.6000e-01  1.0346e+02  2.2200e+00]
 [ 2.0649e+02  5.3300e+00  1.2190e+01  4.1080e+01]
 [ 0.0000e+00  1.0000e-01  0.0000e+00  5.1120e+01]
 [ 0.0000e+00  0.0000e+00  2.4000e-01  0.0000e+00]
 [ 5.0000e-02  4.0700e+00 -1.0100e+00  0.0000e+00]
 [ 5.1700e+01  2.7000e-01  1.0000e-01  1.1000e+00]
 [-3.9000e-01  1.9000e+00  0.0000e+00 -6.0000e-01]
 [ 0.0000e+00  0.0000e+00  6.2000e-01  0.0000e+00]
 [ 4.7000e-01  5.9100e+00  4.8000e-01  0.0000e+00]
 [ 0.0000e+00  1.6000e-01  0.0000e+00  2.8300e+00]
 [ 0.0000e+00  0.0000e+00  0.0000e+00  0.0000e+00]]
```

Total Reward per Episode plot:



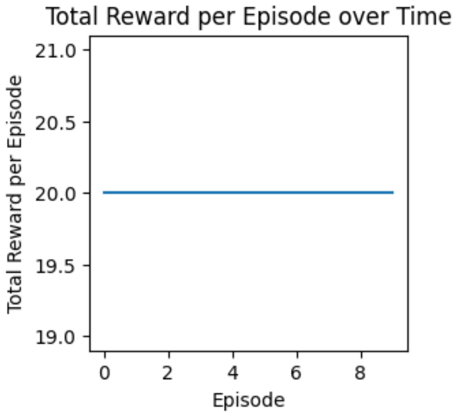
Epsilon Decay plot:



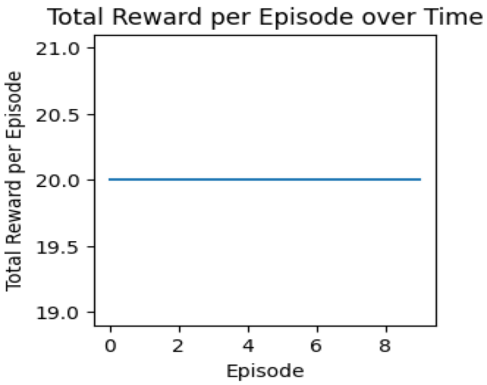
From the above results, we can understand that the Double Q-Learning agent successfully learned a stable policy by maintaining two Q-tables. The Q-values reflects the agent's understanding of the effective actions in different states, and the epsilon plot indicates gradual shift towards more exploitation behaviour. The similarity in the epsilon curve to SARSA suggests that both algorithms exhibit a similar exploration-exploitation tradeoff during training.

Evaluation results for both SARSA and Double Q-learning where the agent chooses only greedy actions from the learned policy:

SARSA:



Double Q-Learning:



As we can see from above, both algorithms learned an optimal policy for the given environment as they are achieving maximum reward for each episode during evaluation. Each episode results in the maximum achievable total reward, indicating that the agents consistently make optimal decisions, maximizing their cumulative reward.

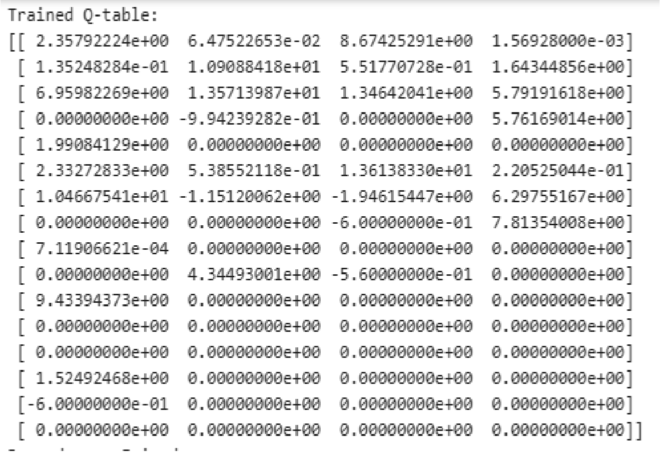
3. Hyperparameter tuning:

SARSA:

Hyperparameters used for tuning:
gamma_values = [0.8, 0.9, 0.95]
epsilon_decay_rates = [0.95, 0.99, 0.999]

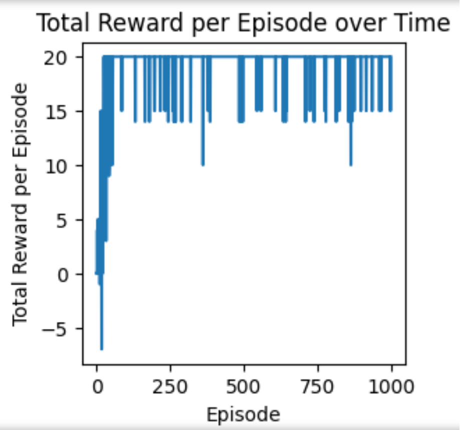
Evaluation for Hyperparameters: (gamma=0.8, epsilon_decay=0.95, num_episodes=1000)

Trained Q-Table:

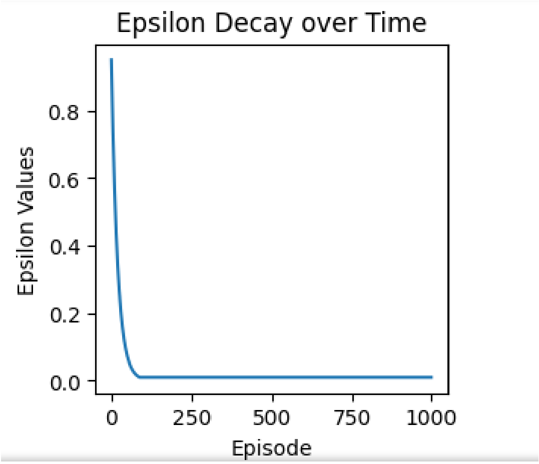


```
Trained Q-table:
[[ 2.35792224e+00  6.47522653e-02  8.67425291e+00  1.56928000e-03]
 [ 1.35248284e-01  1.09088418e+01  5.51770728e-01  1.64344856e+00]
 [ 6.95982269e+00  1.35713987e+01  1.34642041e+00  5.79191618e+00]
 [ 0.00000000e+00 -9.94239282e-01  0.00000000e+00  5.76169014e+00]
 [ 1.99084129e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 2.33272833e+00  5.38552118e-01  1.36138330e+01  2.20525044e-01]
 [ 1.04667541e+01 -1.15120062e+00 -1.94615447e+00  6.29755167e+00]
 [ 0.00000000e+00  0.00000000e+00 -6.00000000e-01  7.81354008e+00]
 [ 7.11906621e-04  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  4.34493001e+00 -5.60000000e-01  0.00000000e+00]
 [ 9.43394373e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 1.52492468e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-6.00000000e-01  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

Rewards per Episode:



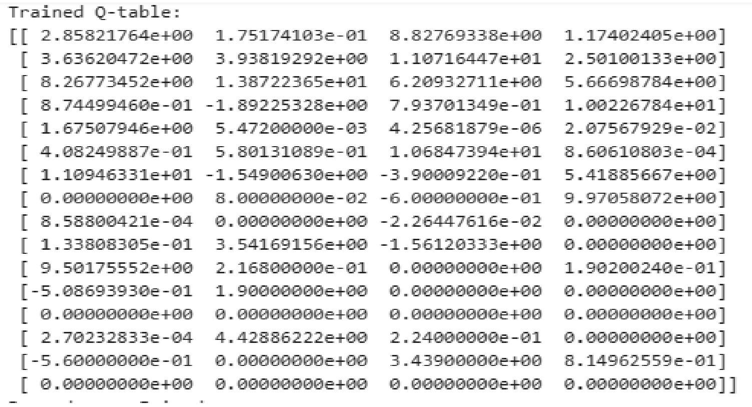
Epsilon Decay:



From the above results, we can see that the total reward per episode consistently increased, indicating effective learning. However, the abrupt epsilon decrease suggests a need for balancing exploration-exploitation.

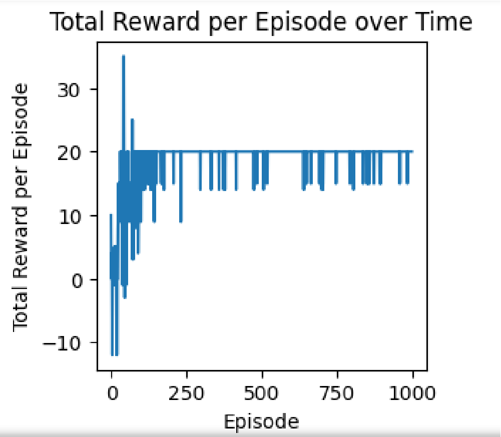
Evaluation for Hyperparameters: (gamma=0.8, epsilon_decay=0.98, num_episodes=1000)

Trained Q-Table:

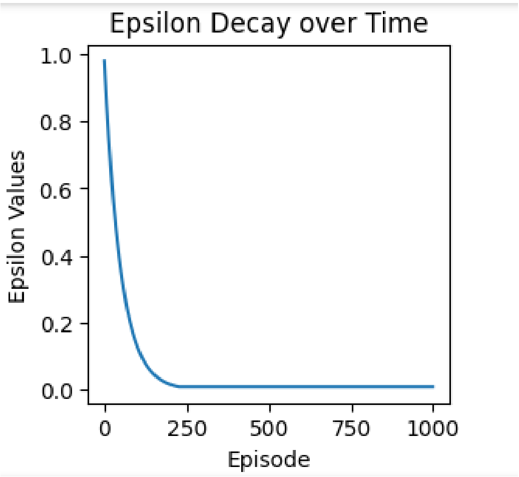


```
Trained Q-table:
[[ 2.85821764e+00  1.75174103e-01  8.82769338e+00  1.17402405e+00]
 [ 3.63620472e+00  3.93819292e+00  1.10716447e+01  2.50100133e+00]
 [ 8.26773452e+00  1.38722365e+01  6.20932711e+00  5.66698784e+00]
 [ 8.74499460e-01 -1.89225328e+00  7.93701349e-01  1.00226784e+01]
 [ 1.67507946e+00  5.47200000e-03  4.25681879e-06  2.07567929e-02]
 [ 4.08249887e-01  5.80131089e-01  1.06847394e+01  8.60610803e-04]
 [ 1.10946331e+01 -1.54900630e+00 -3.90009220e-01  5.41885667e+00]
 [ 0.00000000e+00  8.00000000e-02 -6.00000000e-01  9.97058072e+00]
 [ 8.58800421e-04  0.00000000e+00 -2.26447616e-02  0.00000000e+00]
 [ 1.33808305e-01  3.54169156e+00 -1.56120333e+00  0.00000000e+00]
 [ 9.50175552e+00  2.16800000e-01  0.00000000e+00  1.90200240e-01]
 [-5.08693930e-01  1.90000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 2.70232833e-04  4.42886222e+00  2.24000000e-01  0.00000000e+00]
 [-5.60000000e-01  0.00000000e+00  3.43900000e+00  8.14962559e-01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

Rewards per Episode:

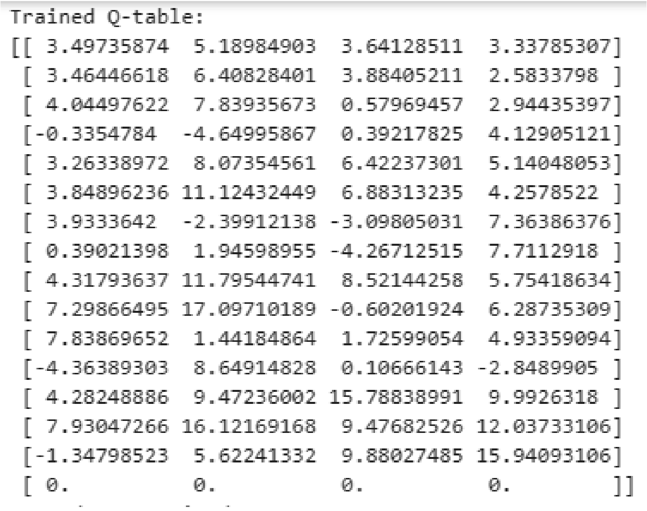


Epsilon Decay:



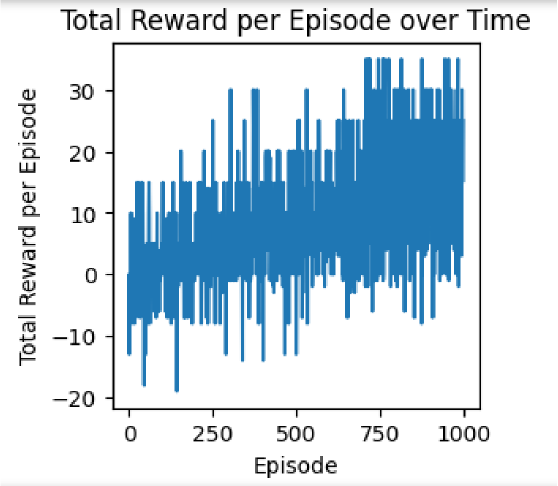
From the above results, we can see that the total reward per episode consistently increased, indicating effective learning. The epsilon decay curve displayed a consistent downward trend, refkecting a gradual shift from exploration to exploitation (balance). The overall performance seems promising.

Evaluation for Hyperparameters: (gamma=0.8, epsilon_decay=0.999, num_episodes=1000)
Trained Q-Table:

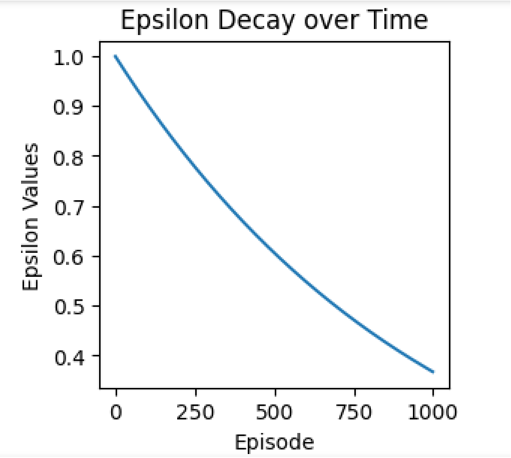


```
Trained Q-table:
[[ 3.49735874  5.18984903  3.64128511  3.33785307]
 [ 3.46446618  6.40828401  3.88405211  2.5833798 ]
 [ 4.04497622  7.83935673  0.57969457  2.94435397]
 [-0.3354784  -4.64995867  0.39217825  4.12905121]
 [ 3.26338972  8.07354561  6.42237301  5.14048053]
 [ 3.84896236 11.12432449  6.88313235  4.2578522 ]
 [ 3.9333642  -2.39912138 -3.09805031  7.36386376]
 [ 0.39021398  1.94598955 -4.26712515  7.7112918 ]
 [ 4.31793637 11.79544741  8.52144258  5.75418634]
 [ 7.29866495 17.09710189 -0.60201924  6.28735309]
 [ 7.83869652  1.44184864  1.72599054  4.93359094]
 [-4.36389303  8.64914828  0.10666143 -2.8489905 ]
 [ 4.28248886  9.47236002 15.78838991  9.9926318 ]
 [ 7.93047266 16.12169168  9.47682526 12.03733106]
 [-1.34798523  5.62241332  9.88027485 15.94093106]
 [ 0.          0.          0.          0.        ]]
```

Rewards per Episode:

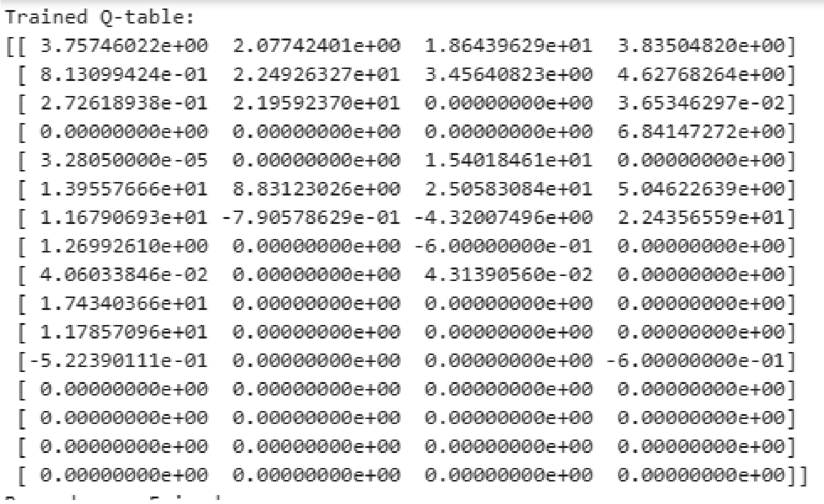


Epsilon Decay:



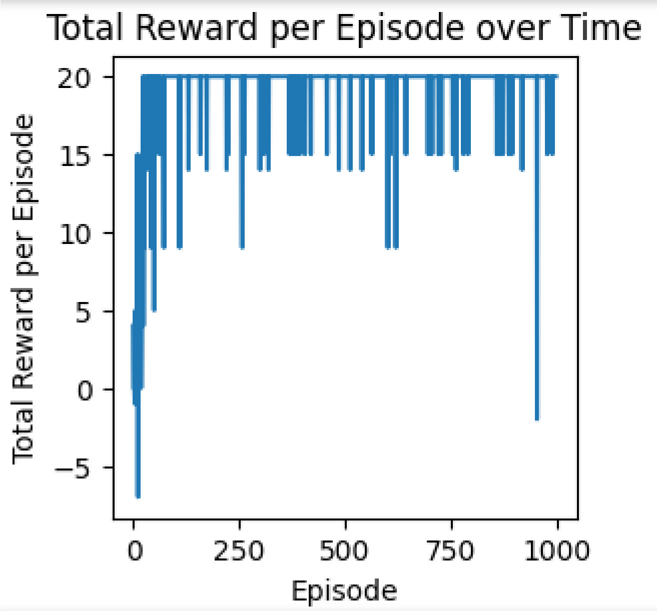
From the above results, the epsilon curve exhibits a decrease from top left to the bottom right, suggesting a slow and steady decay in exploration over the training. This pattern indicates that the agent reduces its exploration rate consistently moving more towards the exploitation. A moderate decay will stabilize the learning as it will ensure that the exploration is essential to find optimal policy.

Evaluation for Hyperparameters: (gamma=0.9, epsilon_decay=0.95, num_episodes=1000)
Trained Q-Table:

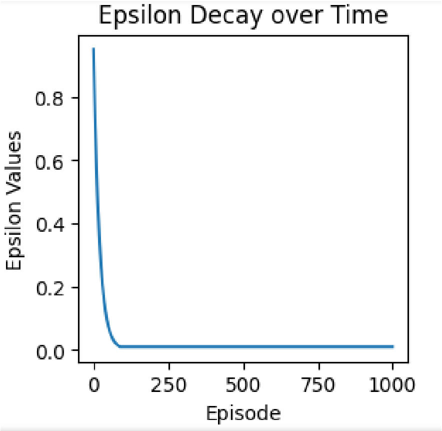


```
Trained Q-table:
[[ 3.75746022e+00  2.07742401e+00  1.86439629e+01  3.83504820e+00]
 [ 8.13099424e-01  2.24926327e+01  3.45640823e+00  4.62768264e+00]
 [ 2.72618938e-01  2.19592370e+01  0.00000000e+00  3.65346297e-02]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  6.84147272e+00]
 [ 3.28050000e-05  0.00000000e+00  1.54018461e+01  0.00000000e+00]
 [ 1.39557666e+01  8.83123026e+00  2.50583084e+01  5.04622639e+00]
 [ 1.16790693e+01 -7.90578629e-01 -4.32007496e+00  2.24356559e+01]
 [ 1.26992610e+00  0.00000000e+00 -6.00000000e-01  0.00000000e+00]
 [ 4.06033846e-02  0.00000000e+00  4.31390560e-02  0.00000000e+00]
 [ 1.74340366e+01  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 1.17857096e+01  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-5.22390111e-01  0.00000000e+00  0.00000000e+00 -6.00000000e-01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

Rewards per Episode:

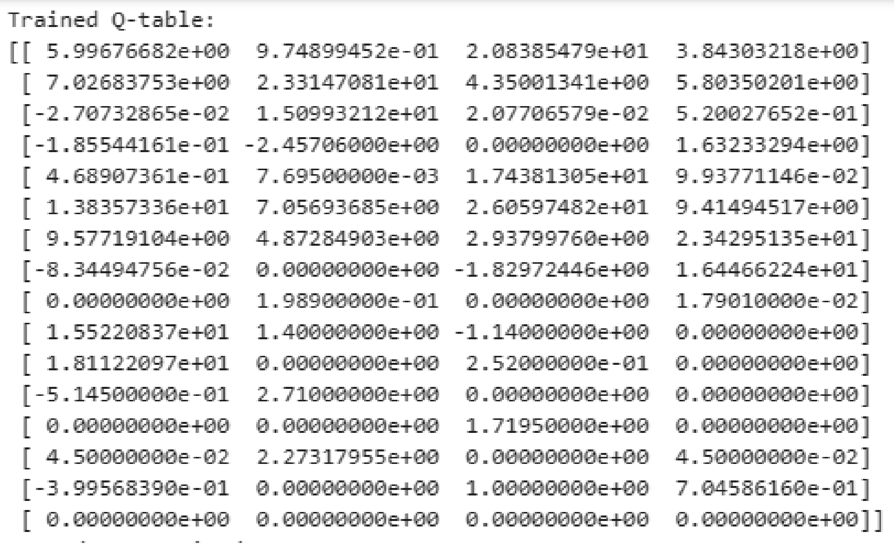


Epsilon Decay:



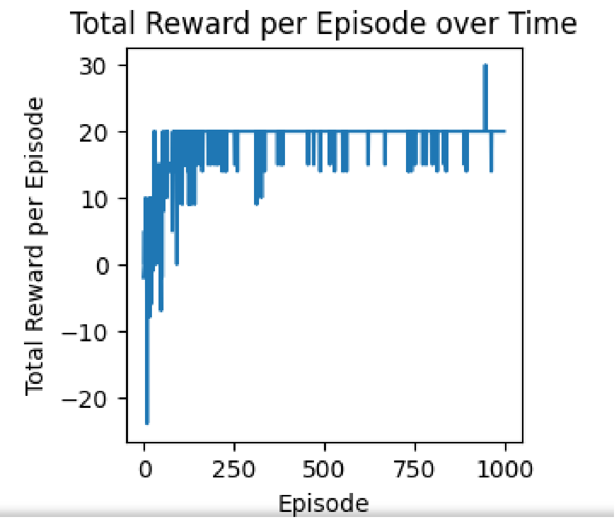
From the above results, the discount factor (gamma) value of 0.9 shows the importance of future rewards in the agent's decision making. The epsilon decay curve implies a steady redution in exploration striking a balance between exploration and exploitation. This shows an overall good performance.

Evaluation for Hyperparameters: (gamma=0.9, epsilon_decay=0.98, num_episodes=1000)
Trained Q-Table:

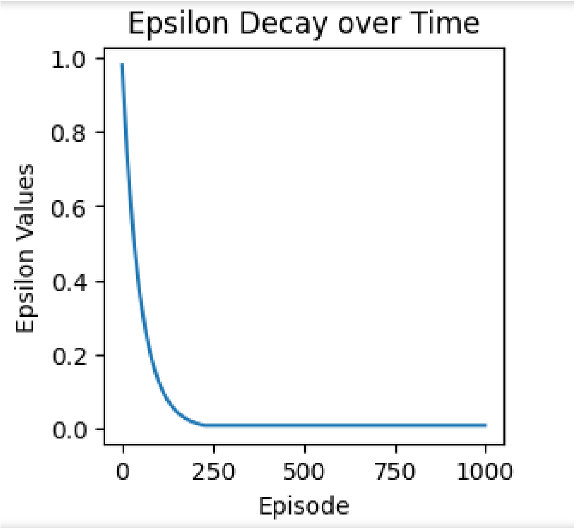


```
Trained Q-table:
[[ 5.99676682e+00  9.74899452e-01  2.08385479e+01  3.84303218e+00]
 [ 7.02683753e+00  2.33147081e+01  4.35001341e+00  5.80350201e+00]
 [-2.70732865e-02  1.50993212e+01  2.07706579e-02  5.20027652e-01]
 [-1.85544161e-01 -2.45706000e+00  0.00000000e+00  1.63233294e+00]
 [ 4.68907361e-01  7.69500000e-03  1.74381305e+01  9.93771146e-02]
 [ 1.38357336e+01  7.05693685e+00  2.60597482e+01  9.41494517e+00]
 [ 9.57719104e+00  4.87284903e+00  2.93799760e+00  2.34295135e+01]
 [-8.34494756e-02  0.00000000e+00 -1.82972446e+00  1.64466224e+01]
 [ 0.00000000e+00  1.98900000e-01  0.00000000e+00  1.79010000e-02]
 [ 1.55220837e+01  1.40000000e+00 -1.14000000e+00  0.00000000e+00]
 [ 1.81122097e+01  0.00000000e+00  2.52000000e-01  0.00000000e+00]
 [-5.14500000e-01  2.71000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  1.71950000e+00  0.00000000e+00]
 [ 4.50000000e-02  2.27317955e+00  0.00000000e+00  4.50000000e-02]
 [-3.99568390e-01  0.00000000e+00  1.00000000e+00  7.04586160e-01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

Rewards per Episode:

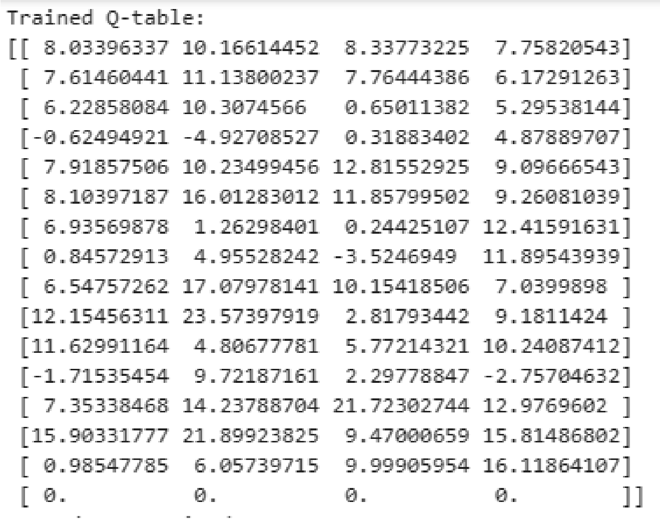


Epsilon Decay:



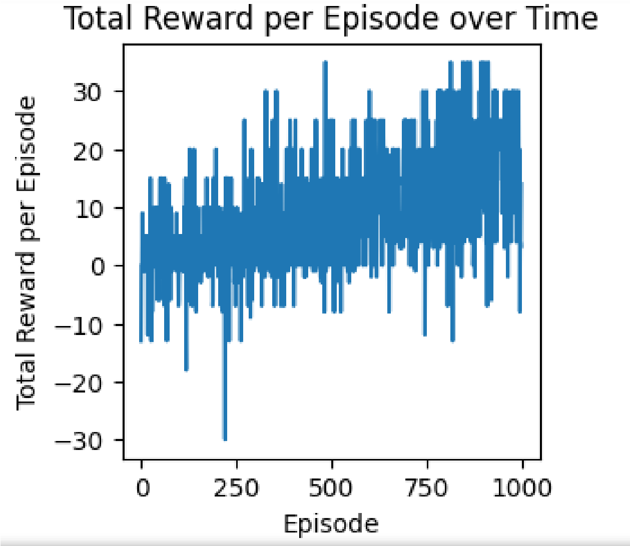
From the above results, the discount factor (gamma) value of 0.9 shows the importance of future rewards in the agent's decision making. The epsilon decay curve implies a steady redution in exploration striking a balance between exploration and exploitation. This shows an overall better performance than before one. I think these are the most efficient set of hyperparameters.

Evaluation for Hyperparameters: (gamma=0.9, epsilon_decay=0.999, num_episodes=1000)
Trained Q-Table:

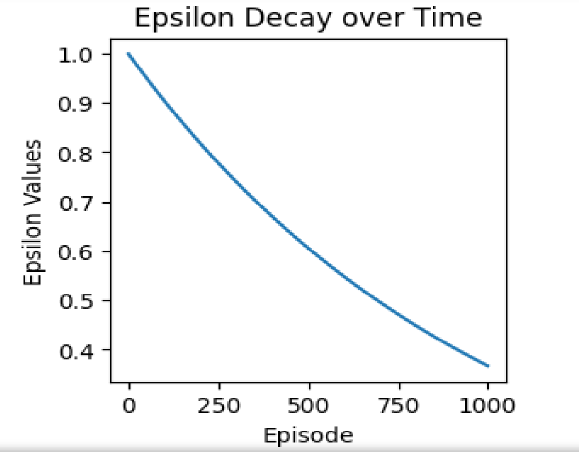


```
Trained Q-table:
[[ 8.03396337 10.16614452  8.33773225  7.75820543]
 [ 7.61460441 11.13800237  7.76444386  6.17291263]
 [ 6.22858084 10.3074566   0.65011382  5.29538144]
 [-0.62494921 -4.92708527  0.31883402  4.87889707]
 [ 7.91857506 10.23499456 12.81552925  9.09666543]
 [ 8.10397187 16.01283012 11.85799502  9.26081039]
 [ 6.93569878  1.26298401  0.24425107 12.41591631]
 [ 0.84572913  4.95528242 -3.5246949  11.89543939]
 [ 6.54757262 17.07978141 10.15418506  7.0399898 ]
 [12.15456311 23.57397919  2.81793442  9.1811424 ]
 [11.62991164  4.80677781  5.77214321 10.24087412]
 [-1.71535454  9.72187161  2.29778847 -2.75704632]
 [ 7.35338468 14.23788704 21.72302744 12.9769602 ]
 [15.90331777 21.89923825  9.47000659 15.81486802]
 [ 0.98547785  6.05739715  9.99905954 16.11864107]
 [ 0.          0.          0.          0.        ]]
```

Rewards per Episode:



Epsilon Decay:



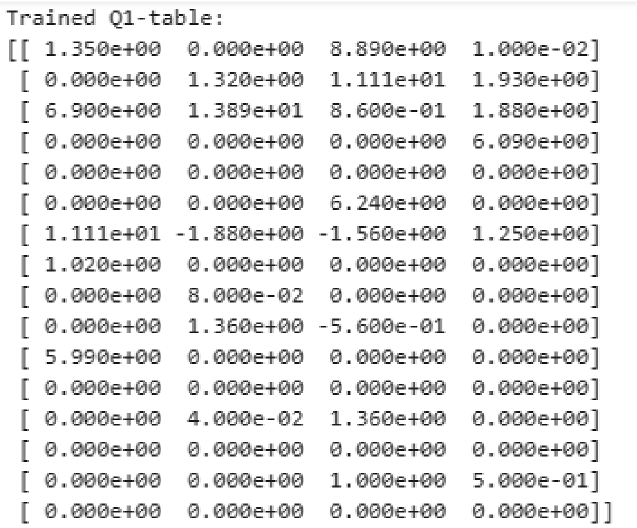
From the above results, the epsilon curve exhibits a decrease from top left to the bottom right, suggesting a slow and steady decay in exploration over the training. This pattern indicates that the agent reduces its exploration rate consistently moving more towards the exploitation. A moderate decay will stabilize the learning as it will ensure that the exploration is essential to find optimal policy.

Double Q-Learning:

Hyperparameters used for tuning:
gamma_values = [0.8, 0.9, 0.95]
epsilon_decay_rates = [0.95, 0.99, 0.999]

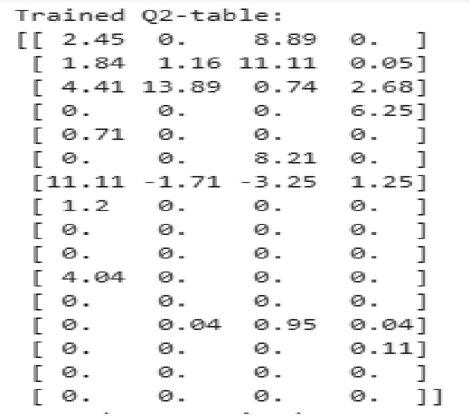
Evaluation for Hyperparameters: (gamma=0.8, epsilon_decay=0.95, num_episodes=1000)

Trained Q1-Table:



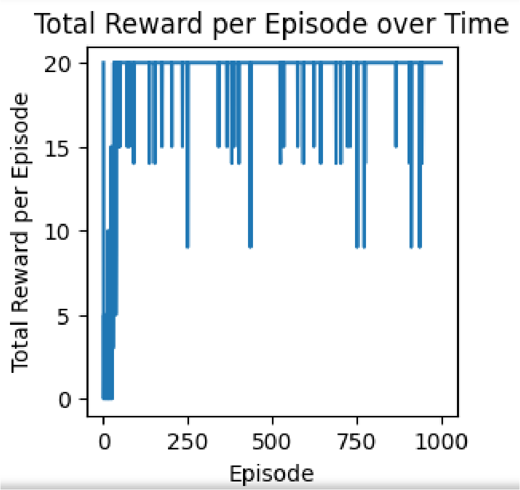
```
Trained Q1-table:
[[ 1.350e+00  0.000e+00  8.890e+00  1.000e-02]
 [ 0.000e+00  1.320e+00  1.111e+01  1.930e+00]
 [ 6.900e+00  1.389e+01  8.600e-01  1.880e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  6.090e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  6.240e+00  0.000e+00]
 [ 1.111e+01 -1.880e+00 -1.560e+00  1.250e+00]
 [ 1.020e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  8.000e-02  0.000e+00  0.000e+00]
 [ 0.000e+00  1.360e+00 -5.600e-01  0.000e+00]
 [ 5.990e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  4.000e-02  1.360e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  1.000e+00  5.000e-01]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]]
```

Trained Q2-Table:

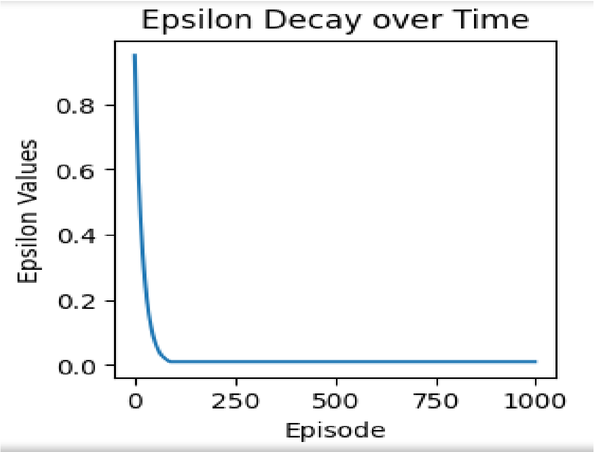


```
Trained Q2-table:
[[ 2.45  0.    8.89  0.  ]
 [ 1.84  1.16 11.11  0.05]
 [ 4.41 13.89  0.74  2.68]
 [ 0.    0.    0.    6.25]
 [ 0.71  0.    0.    0.  ]
 [ 0.    0.    8.21  0.  ]
 [11.11 -1.71 -3.25  1.25]
 [ 1.2   0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 4.04  0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.04  0.95  0.04]
 [ 0.    0.    0.    0.11]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]]
```

Rewards per Episode:



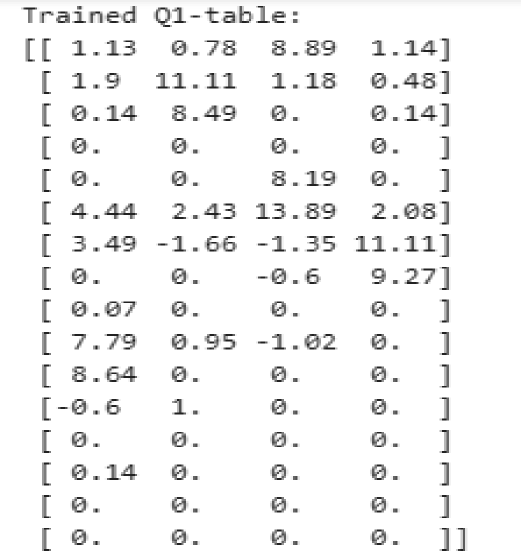
Epsilon Decay:



From the above results, The epsilon decay curve implies a steady redution in exploration striking a balance between exploration and exploitation. Also, the rewards graph shows a good performance over time. This shows an overall better performance. Also, double q-learning address overestimation bias for more stable learning.

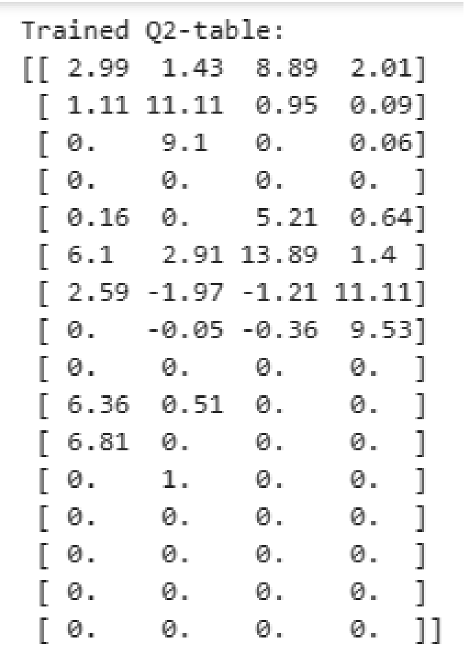
Evaluation for Hyperparameters: (gamma=0.8, epsilon_decay=0.98, num_episodes=1000)

Trained Q1-Table:



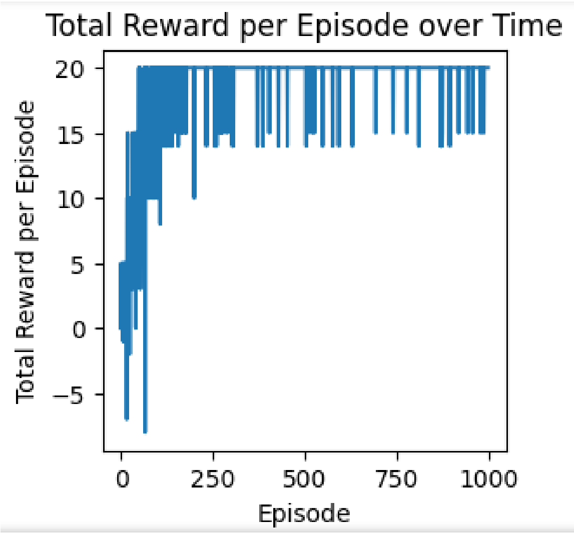
```
Trained Q1-table:
[[ 1.13  0.78  8.89  1.14]
 [ 1.9  11.11  1.18  0.48]
 [ 0.14  8.49  0.    0.14]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    8.19  0.  ]
 [ 4.44  2.43 13.89  2.08]
 [ 3.49 -1.66 -1.35 11.11]
 [ 0.    0.   -0.6   9.27]
 [ 0.07  0.    0.    0.  ]
 [ 7.79  0.95 -1.02  0.  ]
 [ 8.64  0.    0.    0.  ]
 [-0.6   1.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.14  0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]]
```

Trained Q2-Table:

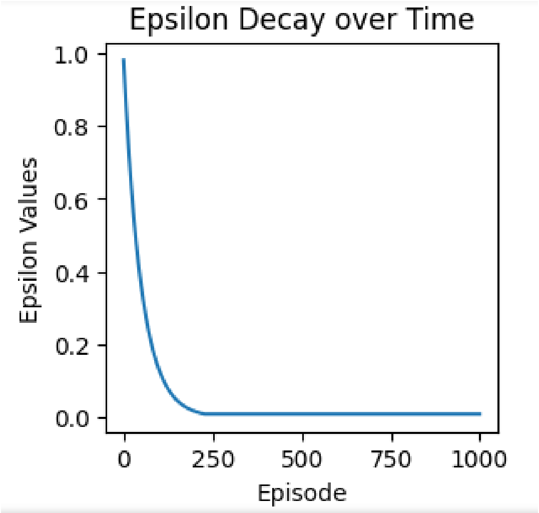


```
Trained Q2-table:
[[ 2.99  1.43  8.89  2.01]
 [ 1.11 11.11  0.95  0.09]
 [ 0.    9.1   0.    0.06]
 [ 0.    0.    0.    0.  ]
 [ 0.16  0.    5.21  0.64]
 [ 6.1   2.91 13.89  1.4 ]
 [ 2.59 -1.97 -1.21 11.11]
 [ 0.   -0.05 -0.36  9.53]
 [ 0.    0.    0.    0.  ]
 [ 6.36  0.51  0.    0.  ]
 [ 6.81  0.    0.    0.  ]
 [ 0.    1.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]]
```

Rewards per Episode:

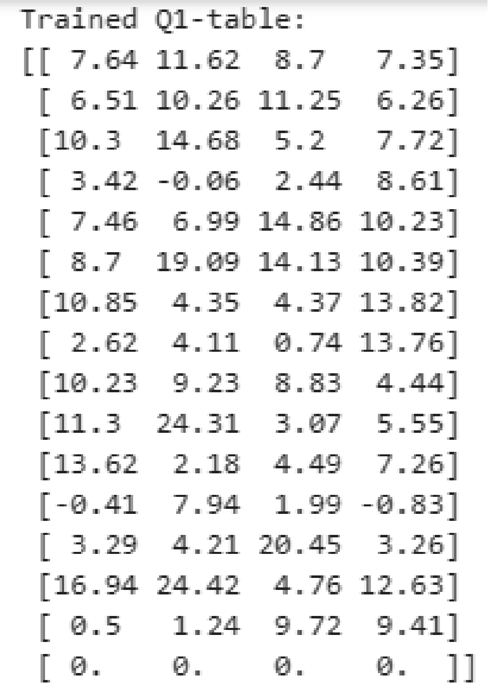


Epsilon Decay:



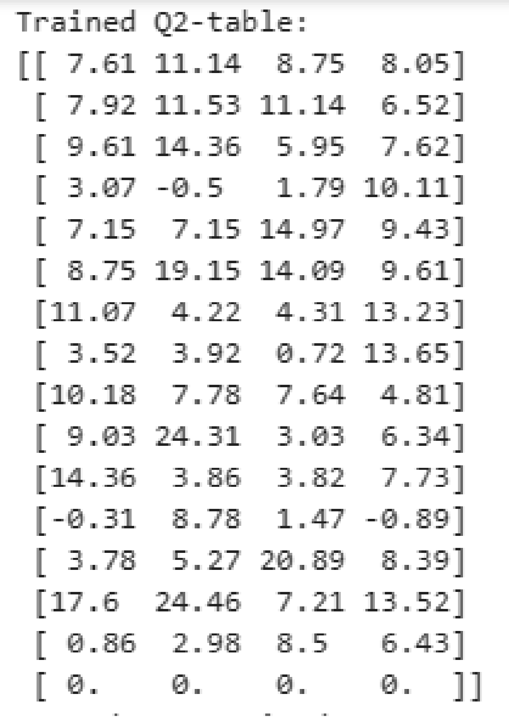
From the above results, The epsilon decay curve implies a steady redution in exploration striking a balance between exploration and exploitation. Also, the rewards graph shows a good performance over time. This shows an overall better performance almost same as previous parameters.

Evaluation for Hyperparameters: (gamma=0.8, epsilon_decay=0.999, num_episodes=1000)
Trained Q1-Table:



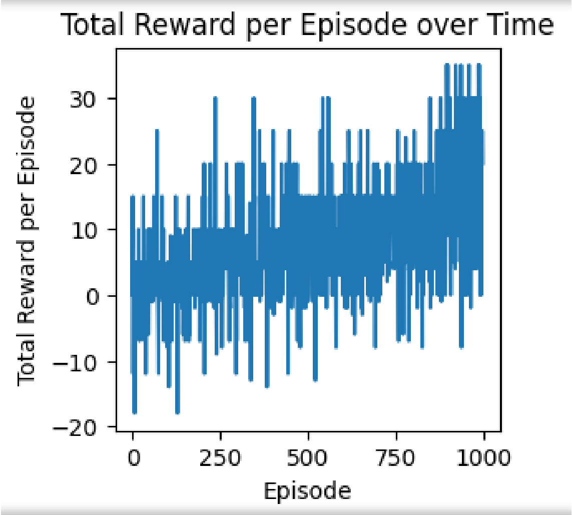
```
Trained Q1-table:
[[ 7.64 11.62  8.7   7.35]
 [ 6.51 10.26 11.25  6.26]
 [10.3  14.68  5.2   7.72]
 [ 3.42 -0.06  2.44  8.61]
 [ 7.46  6.99 14.86 10.23]
 [ 8.7  19.09 14.13 10.39]
 [10.85  4.35  4.37 13.82]
 [ 2.62  4.11  0.74 13.76]
 [10.23  9.23  8.83  4.44]
 [11.3  24.31  3.07  5.55]
 [13.62  2.18  4.49  7.26]
 [-0.41  7.94  1.99 -0.83]
 [ 3.29  4.21 20.45  3.26]
 [16.94 24.42  4.76 12.63]
 [ 0.5   1.24  9.72  9.41]
 [ 0.    0.    0.    0.  ]]
```

Trained Q2-Table:

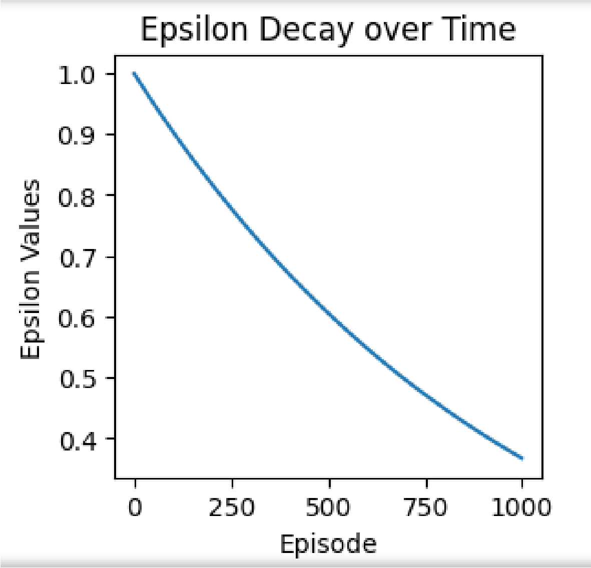


```
Trained Q2-table:
[[ 7.61 11.14  8.75  8.05]
 [ 7.92 11.53 11.14  6.52]
 [ 9.61 14.36  5.95  7.62]
 [ 3.07 -0.5   1.79 10.11]
 [ 7.15  7.15 14.97  9.43]
 [ 8.75 19.15 14.09  9.61]
 [11.07  4.22  4.31 13.23]
 [ 3.52  3.92  0.72 13.65]
 [10.18  7.78  7.64  4.81]
 [ 9.03 24.31  3.03  6.34]
 [14.36  3.86  3.82  7.73]
 [-0.31  8.78  1.47 -0.89]
 [ 3.78  5.27 20.89  8.39]
 [17.6  24.46  7.21 13.52]
 [ 0.86  2.98  8.5   6.43]
 [ 0.    0.    0.    0.  ]]
```

Rewards per Episode:



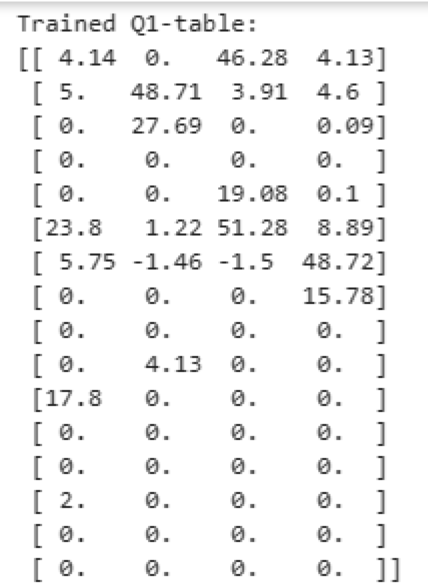
Epsilon Decay:



From the above results, we can observe the epsilon decay graph indicates a strong emphasis on exploration with a slow decay of epsilon. This setting might lead to prolonged exploration, potentially capturing more nuanced state-action spaces but risking longer convergence times.

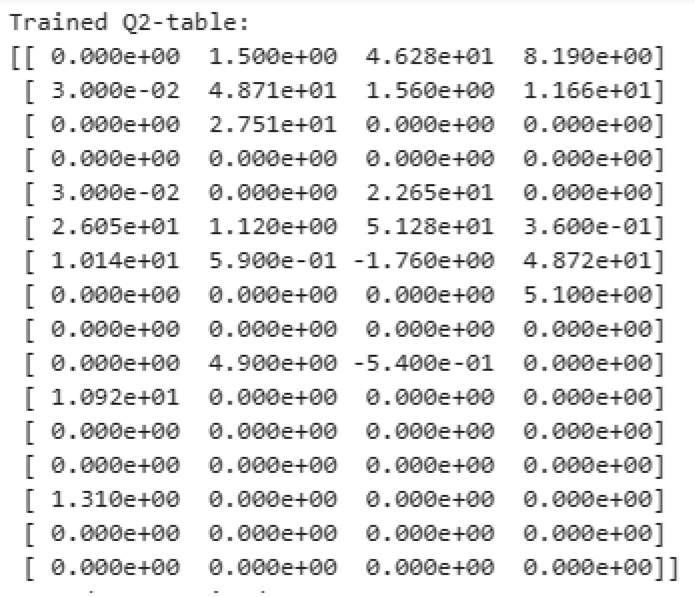
Evaluation for Hyperparameters: (gamma=0.95, epsilon_decay=0.95, num_episodes=1000)

Trained Q1-Table:



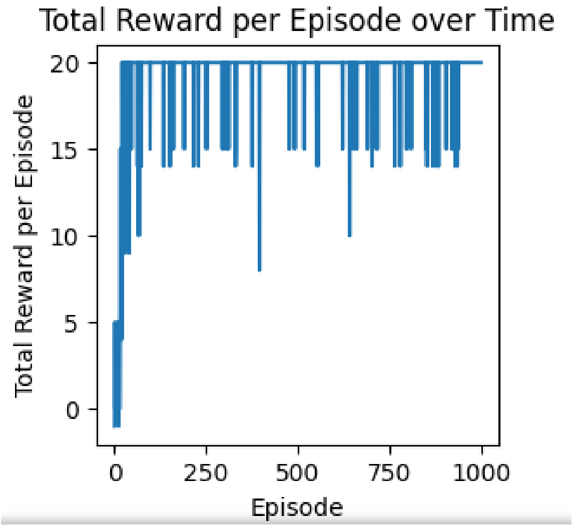
```
Trained Q1-table:
[[ 4.14  0.   46.28  4.13]
 [ 5.   48.71  3.91  4.6 ]
 [ 0.   27.69  0.    0.09]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.   19.08  0.1 ]
 [23.8   1.22 51.28  8.89]
 [ 5.75 -1.46 -1.5  48.72]
 [ 0.    0.    0.   15.78]
 [ 0.    0.    0.    0.  ]
 [ 0.    4.13  0.    0.  ]
 [17.8   0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 2.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.  ]]
```

Trained Q2-Table:

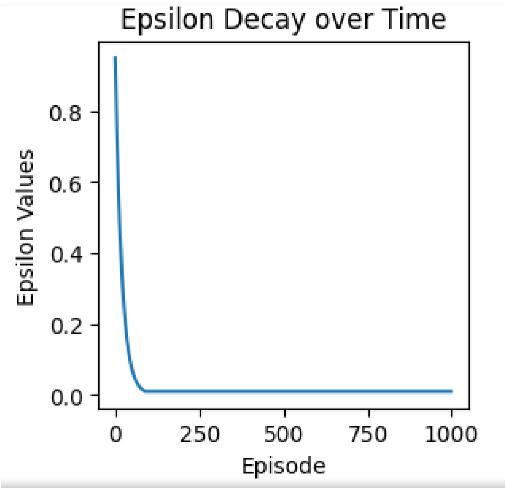


```
Trained Q2-table:
[[ 0.000e+00  1.500e+00  4.628e+01  8.190e+00]
 [ 3.000e-02  4.871e+01  1.560e+00  1.166e+01]
 [ 0.000e+00  2.751e+01  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 3.000e-02  0.000e+00  2.265e+01  0.000e+00]
 [ 2.605e+01  1.120e+00  5.128e+01  3.600e-01]
 [ 1.014e+01  5.900e-01 -1.760e+00  4.872e+01]
 [ 0.000e+00  0.000e+00  0.000e+00  5.100e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  4.900e+00 -5.400e-01  0.000e+00]
 [ 1.092e+01  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 1.310e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]]
```

Rewards per Episode:

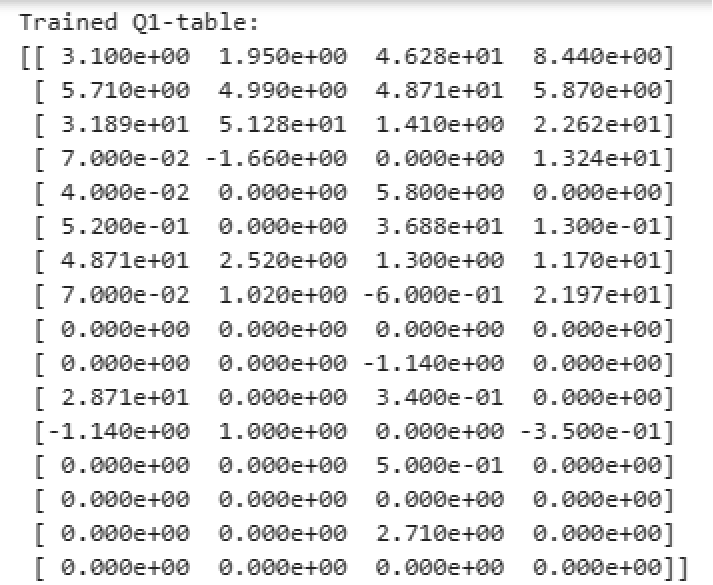


Epsilon Decay:



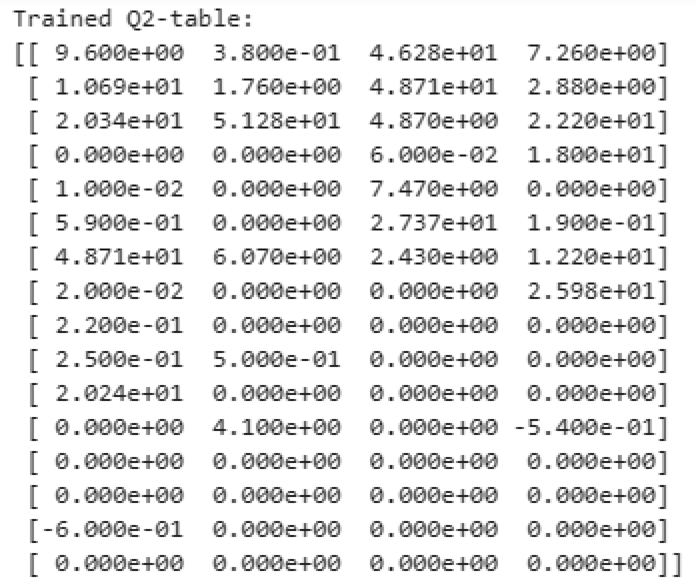
From the above results, we can observe that there is a balance between maintaining exploration and emphasis on future rewards. The graphs suggests a steady exploration rate and shows a resonable trade-off between exploitation and exploration.

Evaluation for Hyperparameters: (gamma=0.95, epsilon_decay=0.98, num_episodes=1000)
Trained Q1-Table:



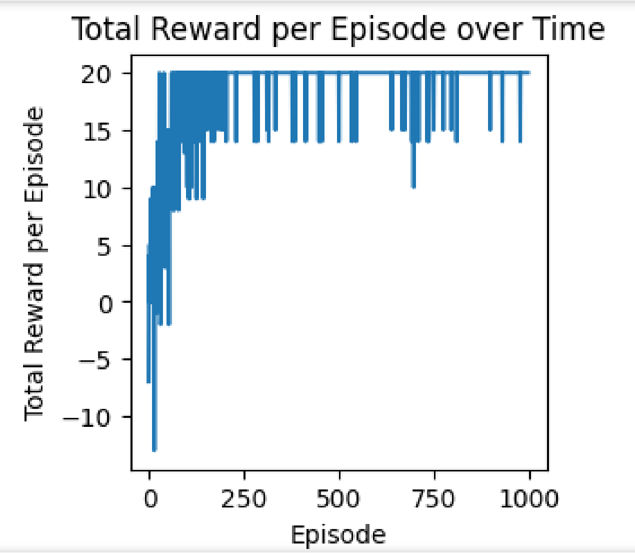
```
Trained Q1-table:
[[ 3.100e+00  1.950e+00  4.628e+01  8.440e+00]
 [ 5.710e+00  4.990e+00  4.871e+01  5.870e+00]
 [ 3.189e+01  5.128e+01  1.410e+00  2.262e+01]
 [ 7.000e-02 -1.660e+00  0.000e+00  1.324e+01]
 [ 4.000e-02  0.000e+00  5.800e+00  0.000e+00]
 [ 5.200e-01  0.000e+00  3.688e+01  1.300e-01]
 [ 4.871e+01  2.520e+00  1.300e+00  1.170e+01]
 [ 7.000e-02  1.020e+00 -6.000e-01  2.197e+01]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00 -1.140e+00  0.000e+00]
 [ 2.871e+01  0.000e+00  3.400e-01  0.000e+00]
 [-1.140e+00  1.000e+00  0.000e+00 -3.500e-01]
 [ 0.000e+00  0.000e+00  5.000e-01  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  2.710e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]]
```

Trained Q2-Table:

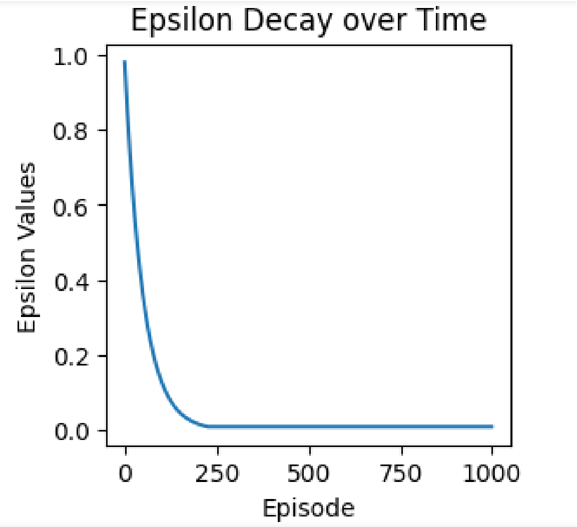


```
Trained Q2-table:
[[ 9.600e+00  3.800e-01  4.628e+01  7.260e+00]
 [ 1.069e+01  1.760e+00  4.871e+01  2.880e+00]
 [ 2.034e+01  5.128e+01  4.870e+00  2.220e+01]
 [ 0.000e+00  0.000e+00  6.000e-02  1.800e+01]
 [ 1.000e-02  0.000e+00  7.470e+00  0.000e+00]
 [ 5.900e-01  0.000e+00  2.737e+01  1.900e-01]
 [ 4.871e+01  6.070e+00  2.430e+00  1.220e+01]
 [ 2.000e-02  0.000e+00  0.000e+00  2.598e+01]
 [ 2.200e-01  0.000e+00  0.000e+00  0.000e+00]
 [ 2.500e-01  5.000e-01  0.000e+00  0.000e+00]
 [ 2.024e+01  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  4.100e+00  0.000e+00 -5.400e-01]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [-6.000e-01  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00]]
```

Rewards per Episode:

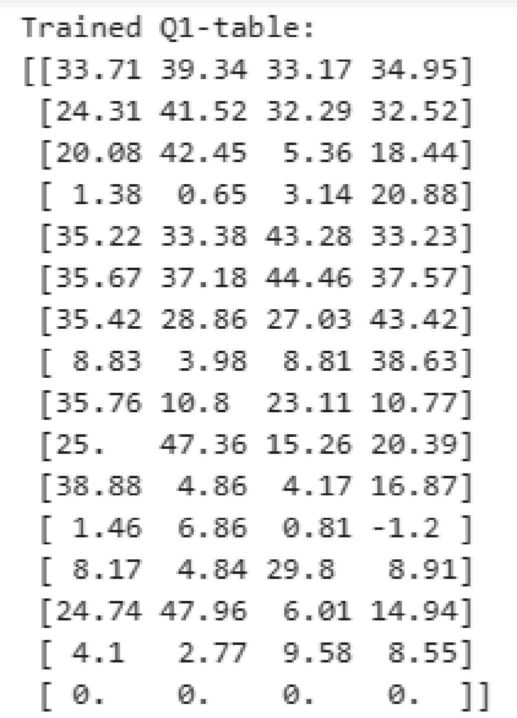


Epsilon Decay:



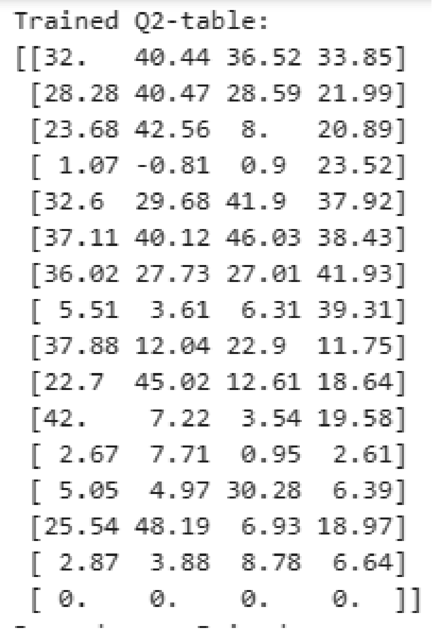
From the above results, The epsilon decay curve implies a steady redution in exploration striking a balance between exploration and exploitation. Also, the rewards graph shows a good performance over time. This shows an overall good performance. I think these are most efficient hyperparameter values.

Evaluation for Hyperparameters: (gamma=0.95, epsilon_decay=0.999, num_episodes=1000)
Trained Q1-Table:



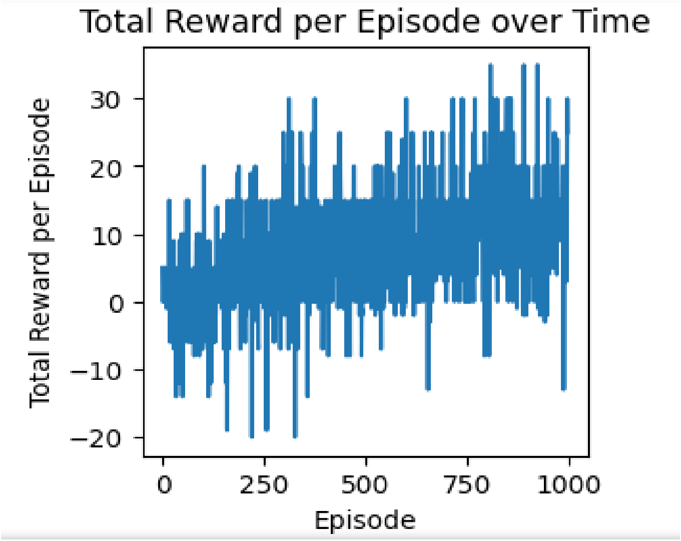
```
Trained Q1-table:
[[33.71 39.34 33.17 34.95]
 [24.31 41.52 32.29 32.52]
 [20.08 42.45  5.36 18.44]
 [ 1.38  0.65  3.14 20.88]
 [35.22 33.38 43.28 33.23]
 [35.67 37.18 44.46 37.57]
 [35.42 28.86 27.03 43.42]
 [ 8.83  3.98  8.81 38.63]
 [35.76 10.8  23.11 10.77]
 [25.   47.36 15.26 20.39]
 [38.88  4.86  4.17 16.87]
 [ 1.46  6.86  0.81 -1.2 ]
 [ 8.17  4.84 29.8   8.91]
 [24.74 47.96  6.01 14.94]
 [ 4.1   2.77  9.58  8.55]
 [ 0.    0.    0.    0.  ]]
```

Trained Q2-Table:

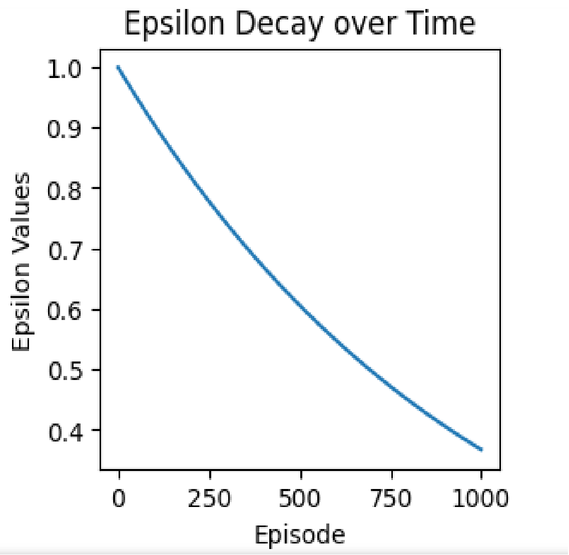


```
Trained Q2-table:
[[32.   40.44 36.52 33.85]
 [28.28 40.47 28.59 21.99]
 [23.68 42.56  8.   20.89]
 [ 1.07 -0.81  0.9  23.52]
 [32.6  29.68 41.9  37.92]
 [37.11 40.12 46.03 38.43]
 [36.02 27.73 27.01 41.93]
 [ 5.51  3.61  6.31 39.31]
 [37.88 12.04 22.9  11.75]
 [22.7  45.02 12.61 18.64]
 [42.    7.22  3.54 19.58]
 [ 2.67  7.71  0.95  2.61]
 [ 5.05  4.97 30.28  6.39]
 [25.54 48.19  6.93 18.97]
 [ 2.87  3.88  8.78  6.64]
 [ 0.    0.    0.    0.  ]]
```

Rewards per Episode:

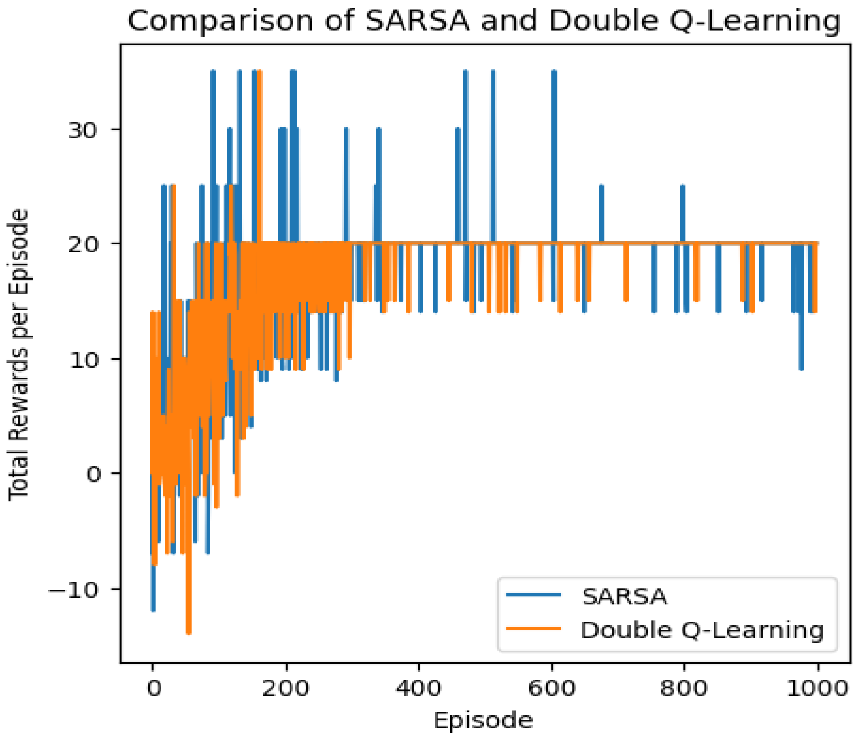


Epsilon Decay:



From the above results, we can observe the epsilon decay graph indicates a strong emphasis on exploration with a slow decay of epsilon. This setting might lead to prolonged exploration, potentially capturing more nuanced state-action spaces but risking longer convergence times.

4. Comparing the performance of both algorithms on the same environment:



From the above graph, we can see that Double Q-Learning is more stable and learns better than SARSA, especially as episodes increase. Looking at the rewards, Double Q-Learning consistently performs a bit better, indicating it is more effective in learning and performing better in terms of cumulative rewards over time.

References:

1.https://ubuffalo-my.sharepoint.com/personal/avereshc_buffalo_edu/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Favereshc%5Fbuffalo%5Fedu%2FDocuments%2F2023%5FFall%5FML%2F%5Fpublic%2FCourse%20Materials%2FRL%20Environment%20Visualization%20by%20Nitin%20Kulkarni&ga=1

2.https://ubuffalo-my.sharepoint.com/personal/avereshc_buffalo_edu/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Favereshc%5Fbuffalo%5Fedu%2FDocuments%2F2023%5FFall%5FAI%2F%5Fpublic%2FCourse%20Materials%2FRL%20Environment%20Visualization%20by%20Nitin%20Kulkarni&ga=1

Contribution Form:

Naga Venkata Sahithya Alla – 100%