Neuroscience of Learning, Memory, and Cognition
Reinforcement Learning Project

Parisa Toumari
99101857

Introduction

Reinforcement Learning (RL) is a type of machine learning where an agent learns to make decisions by interacting with an environment. The agent learns to achieve a goal in an uncertain, potentially complex environment. One common RL algorithm is Q-learning, which is employed in my code.

Grid Environment

The code defines a grid environment loaded from an Excel file. The grid represents a problem where the agent starts at the bottom-left cell and needs to reach the top-right cell. Each cell has a reward associated with it, and the agent's goal is to find the optimal path that maximizes the cumulative reward.

Q-Learning

The Q-learning algorithm is employed to learn the optimal action for each state. Q-values are initialized to zeros, and the agent iteratively updates these values based on its interactions with the environment. The algorithm uses an epsilon-greedy strategy to balance exploration and exploitation.

- Learning Rate (alpha): The learning rate determines the impact of new information on the existing Q-values. A higher learning rate emphasizes recent experiences over past ones.
- Discount Factor (gamma): The discount factor influences the agent's preference for immediate rewards over future rewards. A lower gamma values short-term rewards, while a higher gamma considers long-term rewards.
- Epochs: The number of iterations or epochs determines how many times the agent explores and exploits the environment to learn an optimal policy.

Learning Policy and Path

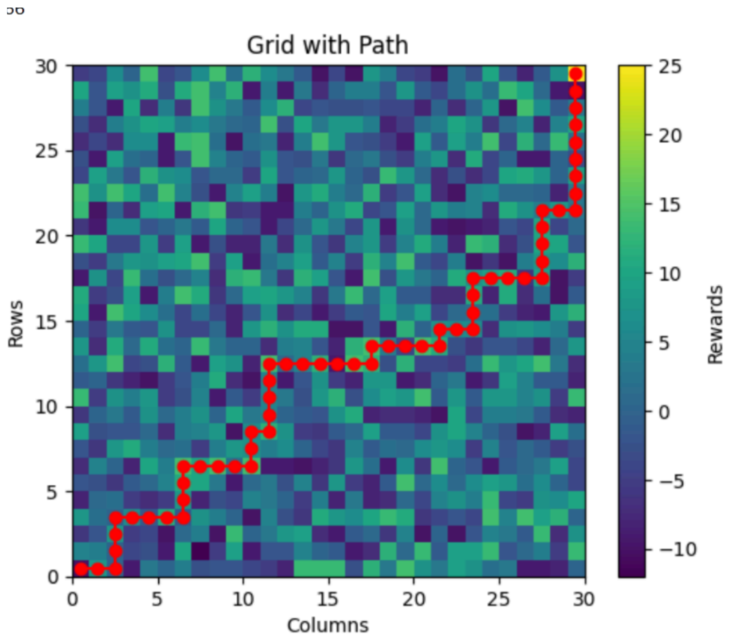
After training, the code extracts the learned policy, which is the optimal action for each state. It then finds the path that the agent would take based on this learned policy, starting from the bottom-left to the top-right.

Visualization

The code uses matplotlib to visualize the grid environment, highlight the learned path, and display the cumulative rewards along the path.

Results

The obtained path and total cumulative reward for the learned policy are printed (366), providing insights into the performance of the trained agent.



```
Obtained Path:
[(29, 0), (29, 1), (29, 2), (28, 2), (27, 2), (26, 2), (26, 3), (26, 4),
(26, 5), (26, 6), (25, 6), (24, 6), (23, 6), (23, 7), (23, 8), (23, 9),
(23, 10), (22, 10), (21, 10), (21, 11), (20, 11), (19, 11), (18, 11), (17,
11), (17, 12), (17, 13), (17, 14), (17, 15), (17, 16), (17, 17), (16, 17),
(16, 18), (16, 19), (16, 20), (16, 21), (15, 21), (15, 22), (15, 23), (14,
23), (13, 23), (12, 23), (12, 24), (12, 25), (12, 26), (12, 27), (11, 27),
(10, 27), (9, 27), (8, 27), (8, 28), (8, 29), (7, 29), (6, 29), (5, 29),
(4, 29), (3, 29), (2, 29), (1, 29), (0, 29)]
Total Cumulative Reward for Obtained Path: 366
```

Conclusion

Reinforcement Learning, specifically Q-learning, is demonstrated in this code to solve a grid navigation problem. The agent learns to navigate the environment, balancing exploration and exploitation, and discovers an optimal policy to achieve its goal. The visualization aids in understanding the learned policy and the path the agent takes to maximize cumulative rewards. Further enhancements could involve experimenting with different hyperparameters or extending the approach to more complex environments.

The code is attached to my file.