Question 1: Markov Decision Process and Q-Learning

GridWorld

A screenshot of a computer code

Description automatically generated

Markov Decision Process

A computer screen shot of a program

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

Question 2: Model-Based vs Model-Free Reinforcement Learning

Objective: Deepen the understanding of the difference between Model-Based and ModelFree Reinforcement Learning.

Tasks:

1. In the Markov Decision Process (MDP) notebook, modify the code to compare the

execution time and convergence between a Model-Based approach (e.g., Policy Iteration or

Value Iteration) and a Model-Free approach (e.g., Q-Learning). A screenshot of a computer code

Description automatically generated

2. Explain the difference between Model-Based and Model-Free algorithms briefly.

1. Model-Based Algorithms

These algorithms have access to or learn the environment’s transition dynamics.

They use the model to simulate future states and rewards to determine optimal actions. Examples include Policy Iteration and Value Iteration, where the agent calculates the value of states or actions by iterating through possible future outcomes.

2. Model-Free Algorithms

These algorithms do not require knowledge of the environment's dynamics. Instead, they learn directly from experience (interacting with the environment).

The agent learns optimal policies by interacting with the environment and improving its action-value estimates (Q-values) over time. Examples include Q-Learning and SARSA.

Question 3: Introduction to Deep Q-Learning (DQN)

Objective: Transition from traditional Q-Learning to Deep Q-Learning, building on the

foundational knowledge gained in the previous tasks.

Tasks:

1. Extend your implementation from Question 1 to develop a Deep Q-Learning (DQN)

model. Use a neural network to approximate the Q-values instead of using a lookup

table. 

2. Implement an epsilon-greedy strategy for action selection in the DQN, testing different

epsilon values (e.g., 0.1, 0.5, 0.9) to analyze the balance between exploration and

exploitation. A screenshot of a computer code

Description automatically generated

3. Compare the convergence and performance of the DQN model at different epsilon

values and plot the results.

Based on the graph, we can observe varying performance of the DQN model across different epsilon values (0.1, 0.5, and 0.9), which control the balance between exploration and exploitation. With epsilon = 0.1, the agent primarily exploits its learned knowledge, leading to relatively stable performance and quicker convergence. The cumulative reward rises early on, and although there are some fluctuations, the line stabilizes over time. This indicates that the agent effectively selects actions based on past experiences and achieves higher rewards. However, this comes at the cost of reduced exploration, meaning the agent might miss discovering better strategies or policies in more complex environments.

For epsilon = 0.5, which offers a balance between exploration and exploitation, the agent demonstrates moderate variability in performance. The cumulative reward fluctuates more than with epsilon = 0.1, as the agent explores new actions while still leveraging what it has learned. Over time, the agent shows steady improvement, benefiting from both exploration and exploitation. Finally, with epsilon = 0.9, which prioritizes exploration, the agent experiences substantial fluctuations in cumulative reward, especially in the early episodes. This is due to frequent exploratory actions, which result in inconsistent rewards as the agent tries different strategies. While convergence is slower, the agent eventually improves its performance as it balances exploration over time. Ultimately, lower epsilon values lead to faster convergence, while higher epsilon values encourage broader exploration but with slower learning.