Name: VINAYAK AABUJ,

TUSHAR VERMA

Project Report: Q-Learning for Maze Navigation

Problem Statement

The objective of this project is to implement a Q-Learning agent capable of navigating a grid-based maze to find the shortest path from a given start point to a goal while avoiding obstacles. This task showcases the practical application of reinforcement learning in solving pathfinding problems in dynamic and constrained environments.

Real-World Relevance

1. **Autonomous Navigation: The principles used in this project are applicable in designing navigation systems for robots or drones in unknown terrains.**
2. **Logistics and Supply Chain: Optimized route planning in constrained warehouse spaces.**
3. **Gaming and Simulation: AI agents in games and simulations can benefit from similar reinforcement learning techniques.**

Approach and Methodology

Environment Design

- **The maze is modeled as a 2D grid where:**
    - **0 represents free cells.**
    - **1 represents obstacles.**
    - **The agent can start from a specific cell and must reach the goal cell.**

State Representation

- **Each state corresponds to the agent's current grid cell.**

Action Space

- **The agent can choose from four possible actions: up, down, left, right.**

Reward System

- **+100 for reaching the goal.**
- **-10 for hitting obstacles.**
- **-1 for each step to encourage shorter paths.**

Q-Learning Algorithm

1. **Initialization:**
   - Initialize a Q-table with zeros for all state-action pairs.
2. **Policy:**
   - Use an ε-greedy policy to balance exploration (choosing random actions) and exploitation (choosing actions based on the Q-table).
3. **Update Rule:**
   - The Q-value is updated using the formula:
   - Q(s,a)←Q(s,a)+α[r+γ������ ��Q(s′,a)−Q(s,a)]
   - where:
     - α\alpha is the learning rate.
     - γ\gamma is the discount factor.
     - r is the reward for the action.
     - s′ is the next state.
4. **Training:**
   - The agent learns by iteratively updating the Q-table over multiple episodes.

Visualization

- A plot of the maze with the agent's optimal path is generated to visualize the learned policy.

Results and Observations

Training Results

- After training for 1000 episodes:
  - The agent successfully learned an optimal path to the goal.
  - The Q-table values converged, indicating a stable policy.

Path Visualization

- The agent’s learned path avoids obstacles and reaches the goal in the shortest time possible, as observed in the generated plots.

Performance Analysis

- The agent performed well in mazes with clear pathways.

- For mazes with narrow corridors or high obstacle density, convergence required more episodes.

Challenges Faced

1. **Sparse Rewards:**
   - The agent initially struggled to find the goal due to sparse positive rewards in the environment.
2. **Exploration-Exploitation Tradeoff:**
   - Balancing exploration (to discover new paths) and exploitation (to use known good paths) was challenging.
3. **Hyperparameter Tuning:**
   - Parameters like the learning rate (α), discount factor (γ), and exploration rate (ϵ) required careful tuning for optimal performance.

Potential Improvements

1. **Dynamic Exploration Rate:**
   - Gradually reducing ϵ\epsilon over episodes could allow more exploration early on and more exploitation later.
2. **Reward Shaping:**
   - Providing intermediate rewards for moving closer to the goal could accelerate learning.
3. **Complex Environments:**
   - Testing the algorithm on larger mazes with more complex obstacle patterns.
4. **Deep Reinforcement Learning:**
   - Extending the project to use neural networks to approximate the Q-values for environments with larger state spaces.