CS370

Jose Munoz

This design defense outlines the methodology and rationale used to implement a deep Q-learning agent tasked with solving a path finding problem in a maze. The agent, represented by a pirate, was trained to navigate a matrix-based environment and reach a goal state — the treasure — while maximizing its reward.

**Human vs. Machine Approaches to Solving Problems**

A human solving a maze typically observes the environment visually, identifies paths and obstacles, and makes decisions based on logic, memory, or heuristics. They might backtrack if they reach a dead end and can adapt based on visual cues and reasoning.  
  
The intelligent agent, in contrast, follows a trial-and-error process powered by reinforcement learning. It receives feedback through rewards or penalties and updates a neural network to estimate the optimal action-value (Q-value) for each possible move. Over many episodes, the agent learns which actions maximize the reward.  
  
While both approaches rely on exploration and feedback, humans use intuition and abstract reasoning, whereas the agent relies on numerical updates and past experiences stored in memory.

**Purpose of the Intelligent Agent in Pathfinding**

The pirate agent is designed to autonomously discover a successful strategy to reach the treasure. It does so by balancing two behaviors: exploration (trying new paths) and exploitation (reusing known successful paths).  
  
Exploration helps the agent avoid local optima and uncover better routes, while exploitation enables it to reinforce high-value behaviors. An ideal balance in this maze was achieved using an epsilon-greedy policy with ε = 0.1 — meaning 90% exploitation and 10% exploration — which provided enough randomness to avoid overfitting while maintaining convergence efficiency.  
  
Reinforcement learning allows the agent to learn from interaction, gradually improving its policy by maximizing cumulative rewards without explicit instruction.

**Evaluating the Use of Algorithms to Solve Complex Problems**

Deep Q-learning is a powerful algorithm that combines reinforcement learning with neural networks. In this project, the agent uses a fully connected network to approximate Q-values for each action given a state (the flattened maze grid). It uses experience replay to avoid overfitting and stabilize training. During training, the agent observes states, chooses actions, receives rewards, and stores the episodes in memory. It samples batches of experiences to update the network using stochastic gradient descent and mean squared error loss.  
  
This model is effective because it generalizes across unseen states and enables long-term learning. The combination of Q-learning and neural networks makes it suitable for complex, dynamic problems like this treasure hunt maze.