Oscar Rosa

CS 370

Southern New Hampshire University

02/18/2025

Project 2 Defense Design

### **Introduction**

This project explored how an AI-powered agent can effectively maneuver through a pirate-themed maze using Deep Q-Networks (DQN) in Python. The agent, depicted as a pirate, can move in four directions: forward, backward, left, and right. Positive reinforcement is given for reaching the treasure, while negative reinforcement is applied for revisiting locations, making invalid moves, or colliding with obstacles. By utilizing reinforcement learning, the agent continuously refines its strategy to optimize the shortest route through a combination of exploration and exploitation (Mnih et al., 2015).

### **Problem-Solving: Humans vs. Machines**

Humans and artificial intelligence solve navigation problems differently. Humans leverage prior knowledge, intuition, and reasoning when navigating a maze. By assessing the entire layout, they can predict potential dead ends and plan an optimal course. Pattern recognition helps them avoid unnecessary backtracking. Conversely, AI lacks intuition and must learn through trial and error. The agent gains experience by performing multiple attempts and receiving feedback through rewards or penalties. Unlike humans, who rely on deductive reasoning, the AI systematically tests various paths before identifying the most effective solution (Mnih et al., 2015).

### **How Humans Navigate a Maze**

A person solving a maze would follow these logical steps:

1. **Assess the Maze:** Determine the starting position and destination while identifying potential pathways.
2. **Develop a Strategy:** Mentally plan the shortest or least obstructed route.
3. **Proceed Through the Maze:** Follow the planned route while adjusting for obstacles.
4. **Retrace Steps if Necessary:** If an obstruction is encountered, backtrack and try another route.
5. **Reach the Goal:** Learn from mistakes and use reasoning to complete the maze efficiently.

### **How the AI Agent Navigates the Maze**

The AI followed a structured reinforcement learning process:

1. **State Representation:** The AI encodes its current location numerically.
2. **Decision-Making:** It selects movements using an epsilon-greedy policy, balancing exploration (testing new paths) and exploitation (using acquired knowledge) (Tokic & Palm, 2011).
3. **Movement Execution:** The AI moves based on the selected action.
4. **Feedback Reception:** It receives rewards for positive progress and penalties for ineffective moves.
5. **Updating Knowledge:** Q-values are adjusted using the Bellman equation to improve decision-making (Mnih et al., 2015).
6. **Adaptive Learning Rate:** Initially, exploration is prioritized, but over time, learned experiences guide movements (Haswani, 2020).

### **Exploration vs. Exploitation in Learning**

* **Exploration:** The AI tests unknown paths to collect information.
* **Exploitation:** It applies acquired knowledge to make informed decisions.

At the beginning of training, the agent explores different paths. Over time, it transitions toward making optimized decisions based on past experiences. The epsilon-greedy strategy facilitates this shift by gradually reducing randomness in decision-making (Tokic & Palm, 2011).

### **Deep Q-Learning Implementation**

The AI model incorporates several key components:

1. **Neural Network:** A deep learning framework predicts the most suitable action for a given state.
2. **Experience Replay:** The AI records past movements and learns from them.
3. **Training Process:**
   1. **Action Selection:** Uses an epsilon-greedy approach.
   2. **Data Storage:** Saves experiences in a replay buffer.
   3. **Learning from Past Moves:** Trains the model using stored data.
   4. **Optimizing Q-Values:** Improves predictions over time (Mnih et al., 2015).

A decay rate of 0.1 was chosen for the learning rate to ensure adequate exploration before transitioning to experience-based decision-making (Haswani, 2020).

### **Results and Observations**

Through reinforcement learning, the AI agent became increasingly proficient in solving the maze. After roughly 248 training cycles, it consistently determined the most efficient route to the treasure. Below is an example of how the reward system was structured:

def get\_reward(self):

pirate\_row, pirate\_col, mode = self.state

nrows, ncols = self.maze.shape

if pirate\_row == nrows-1 and pirate\_col == ncols-1:

return 1.0

if mode == 'blocked':

return self.min\_reward - 1

if (pirate\_row, pirate\_col) in self.visited:

return -0.25

if mode == 'invalid':

return -0.75

if mode == 'valid':

return -0.04

The decision-making algorithm worked as follows:

if 1 / (1 + epoch \* decayRate) < epsilon:

action = np.argmax(experience.predict(previous\_envstate))

else:

action = random.choice(validActions)

By the second training cycle, the AI successfully located the treasure. By roughly the 248th iteration, the agent achieved a 100% success rate (Mnih et al., 2015).

### **Conclusion**

### This project demonstrated how reinforcement learning enhances an AI agent’s ability to navigate a complex environment. By balancing exploration and exploitation, the agent gradually learned to determine the optimal route efficiently. Unlike humans, who depend on intuition and strategic foresight, the AI followed a trial-and-error approach. However, despite this limitation, DQNs allow the AI to acquire knowledge through repeated experiences, ultimately leading to consistent and effective pathfinding results.

### **References**

Haswani, V. (2020, September 3). Learning rate decay and methods in deep learning. *Medium.* <https://medium.com/analytics-vidhya/learning-rate-decay-and-methods-in-deep-learning-2cee564f910b>

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529–533. <https://doi.org/10.1038/nature14236>

Tokic, M., & Palm, G. (2011). Value-difference based exploration: Adaptive control between epsilon-greedy and softmax. *KI 2011: Advances in Artificial Intelligence*, 335–346. <https://link.springer.com/chapter/10.1007/978-3-642-24455-1_33>