Reginald True

Professor Vellanki

CS 370

17 October 2025

In this project, I built a smart computer helper called a pirate agent. The pirate learns how to find treasure inside a maze. I used a method called deep Q learning to train it. This means the pirate learns by trying things, making mistakes, and getting better over time. The goal was to finish the missing part of the code so the pirate could always reach the treasure. The pirate uses rewards for good moves and small punishments for bad ones to figure out the best path.

When a person solves a maze, they look at the paths, try one, and turn around if it doesn’t work. People remember where they have been and use logic to guess the right way. The pirate agent doesn’t actually see the maze like we do. It gets numbers that tell it where it is and what moves it can make. It learns by testing different moves and seeing what happens. If it finds treasure, it gets a reward. If it runs into a wall, it gets a small penalty. After many tries, it learns which moves help it win. People and the pirate both learn from experience, but people use thinking and memory. The pirate uses math and data to make better choices each time

The pirate’s job is to find the treasure as fast as possible. Each turn, it decides whether to go left, right, up, or down. Every move gives it feedback.

The pirate uses two types of learning:

* Exploration – trying new moves to learn more.
* Exploitation – using the moves it already knows work best.

At the start, the pirate explores more so it can learn the maze. As it gets better, it starts to exploit more by using the best path it has found. I started the random exploration at 10% (ε = 0.10) and later lowered it to 5% ( e= 0.05) once the pirate began winning most of the time. This balance helps it stay smart and flexible

The pirate learns using something called reinforcement learning. Each time it moves, it remembers:

1. The current state,
2. The move it made,
3. The reward it got,
4. The new state it moved to.

The computer uses this memory to figure out which moves lead to treasure and which don’t. The learning rule adds the reward plus the best possible next reward. Over time, the pirate understands which moves have the highest value.

It takes in numbers that show what the maze looks like and gives four numbers back one for each move . The pirate picks the move with the biggest number unless it’s exploring. I used a method called epsilon greedy, where the pirate sometimes makes a random move to keep exploring. I also used a memory system to store past games. The program trains by looking at these stored memories to improve its guesses.

This project helped me see how computers can teach themselves by practicing, just like people do. It showed that even simple programs can learn to solve problems like finding treasure in a maze.

This project taught me how deep Q learning helps computers learn by trying and improving. The pirate agent started with no knowledge but learned to find the treasure using feedback and practice. Humans use logic and vision, but the agent uses math and rewards. Both get better the more they practice.

Using a mix of exploration, exploitation, and memory, the pirate can now reach the treasure almost every time. This project helped me understand how reinforcement learning works and how smart systems can solve real world problems.