Joshua Morris

CS370 Project 2 Design Document

For this project, I was required to create an artificial intelligence program that was capable of navigating a simple maze. The agent, represented as a pirate in the context of the game, had to search a maze and locate the end goal (the treasure, located at the bottom right corner) within a fixed timeframe. To solve this problem, I created an intelligent agent based on the Deep Q Learning algorithm, using the Keras library in Python. The resulting agent was able to achieve a win rate of over 90% after a 25 epoch training period.

For a human, solving a maze is often a simple matter of trial and error. When confronted with a similar problem, most human agents would attempt to map out the maze, avoid backtracking, and arrive at the goal through the process of elimination. Other techniques exist, such as the ‘hand on the wall’ method, but all rely on a basic system of exploration (Dawson & Dawson, 2022).

Our pirate agent uses the same basic principle, except that the untrained agent begins with no knowledge whatsoever of either the environment, or the techniques necessary to interact with it (Silver & Hassabis, 2017). This means that, unlike a human, the AI agent does not begin the game with a plan. Instead, the agent relies on a process called Reinforcement Learning. Reinforcement learning is a process by which an AI agent can be trained to interact with an environment in a way which maximizes its reward (Yoon, 2021). The agent starts off taking random actions, and gradually learns through trial and error which actions result in the greatest rewards (Doshi, 2021)

While the two approaches are similar in some ways, there are important differences. We have already noted that a human agent would likely start the game with both a basic knowledge of the rules (find the treasure = reward, less steps = better) and a plan of action. The AI agent, however, knows only its available actions (move up, down, left, right), and that its goal is to maximize reward (achieved by finding the treasure using a minimum number of steps, but the agent does not know that). The human’s approach relies on problem solving, while the AI approaches the game (at least initially) using a combination of Exploration and Exploitation.

Exploration refers to the process of taking random actions with the goal of discovering new avenues towards reward. Actions which result in reward are prioritized in future iterations, while actions which do not will be assigned a lower value. The initial phase of the training process is heavily weighted towards exploration, as we want the agent to gather as much information as possible about its environment. Exploitation, on the other hand, denotes actions that the agent takes because its policy (or governing logic) associates them with potential reward. These are the actions towards which the agent’s training is intended to steer it, with the goal of eventually eliminating the need for further exploration (Nova & Nova, 2023).

In our model, the agent begins with its movements heavily weighted towards exploration. Since we want our agent to have as much information about the environment as possible, we set a high ‘exploration value’ or percent chance that the agent will take a random, non-exploitive action. We then program the agent to reduce this value after every epoch, meaning that the more the agent learns, the more it will utilize its gained knowledge.

Our implementation of the Deep Q algorithm for this problem was based on simple logic. Every time the agent takes an action, it is given a reward based upon the result. Reaching the goal results in a large reward, while running into a wall incurs a penalty. By the same logic, entering a space that has already been explored is assigned a lower value than entering an unexplored space. These values were implemented using Pythons Keras library.

Doshi, K. (2021, December 26). Reinforcement Learning Explained Visually (Part 6): Policy Gradients, step-by-step. *Medium*. <https://towardsdatascience.com/reinforcement-learning-explained-visually-part-6-policy-gradients-step-by-step-f9f448e73754>

Nova, & Nova. (2023, July 16). Deep Dive into Reinforcement Learning: Solving the CartPole Problem with PyTorch - AITechTrend. *AITechTrend - Further into the Future*. <https://aitechtrend.com/deep-dive-into-reinforcement-learning-solving-the-cartpole-problem-with-pytorch/>

Ruth Dawson & Nick Dawson, Northumbria University & Ruth Dawson & Nick Dawson, Northumbria University. (2022, February 24). How maths can help you escape a maze. *BBC Future*. <https://www.bbc.com/future/article/20170201-how-maths-can-help-you-escape-a-maze>

Silver, D., & Hassabis, D. (2017, October 18). *AlphaGo Zero: Starting from scratch*. Google DeepMind. <https://deepmind.google/discover/blog/alphago-zero-starting-from-scratch/>

Surma, G. (2021, October 13). Cartpole - Introduction to Reinforcement Learning (DQN - Deep Q-Learning). *Medium*. <https://gsurma.medium.com/cartpole-introduction-to-reinforcement-learning-ed0eb5b58288>

Yoon, C. (2021, December 7). Deriving policy gradients and implementing REINFORCE. *Medium*. <https://medium.com/@thechrisyoon/deriving-policy-gradients-and-implementing-reinforce-f887949bd63>