Ryan G. Cooper

Professor Richard

CS-370 Module 7 Project Two

April 21st, 2024

**Project Two**

Humans approach maze-solving using a variety of strategies. One such strategy is by trial and error, where they will try each path until the path that leads to the exit or treasure is found. Another way is making rules like always turning right and then using trial and error till the end is found. Lastly, there is backtracking, when encountering a dead-end, humans backtrack to the last decision point to try a different path. Humans rely heavily on visual cues and spatial memory, and the process is often iterative and based on accumulating knowledge from previous attempts.

My intelligent agent uses a Deep Q-learning model which uses reinforcement learning to train the agent to make decisions. This is done through the agent exploring the environment, learning from rewards and optimizing actions. In the beginning of training, the agent will randomly explore the maze which helps it to learn about the states and possible actions. Then the agent will receive rewards or penalties based on its actions, which it uses to update its policy on how it selects actions. With this the agent will begin to predict the outcome of each action, with the goal of maximizing its reward. Both humans and machines use a form of trial and error, but the machine’s methods are more structured and quantifiable through the use of algorithms and reward systems. Humans like to use diverse strategies based on intuition and experience that are hard to quantify and model precisely in a machine.

In pathfinding, the intelligent agent aims to find the most efficient route from start to goal. This efficiency is not solely about finding the shortest path but also involves learning to avoid paths that lead to dead ends or require backtracking, thus optimizing the path over time based on continuous learning from the environment. Exploitation involves using known information to make decisions, while exploration involves seeking out new information that might lead to better decisions in the future. For this maze, the ideal ratio exploitation and exploration for this maze is having a high exploration rate in the beginning and then decreasing it as the number of trials increases. High exploration helps the agent discover various states and understand the consequences of different actions, including finding potential shortcuts and identifying dead ends. Without sufficient exploration, the agent might repeatedly follow a suboptimal path that seems best based only on initial limited experiences and never discover a shorter or faster route. As the training progresses and the Q-values the model predicts become more accurate, the agent can rely more on this knowledge to navigate the maze efficiently. Over time, as the model's predictions stabilize and the agent becomes familiar with the environment, it benefits from exploiting known good actions to reduce the number of steps to reach the goal, thus solving the maze more quickly and efficiently. The ideal proportion thus starts with higher exploration and transitions to higher exploitation. Reinforcement learning could help the agent by helping it make decisions based on previous experiences. Based on the feedback of each reward and penalty the agent will learn how to move through the environment.

Deep Q-Learning was implemented using a neural network that approximates the Q-value function. The network predicts the value of taking each action in each state, guiding the agent to make decisions that maximize the end reward. Training involves storing each transaction, such as current state, action, reward, next state in memory. Then the model evaluates what it has learned to then improve on the next run.

**References**

Juliani, A. (2017, June 30). Simple reinforcement learning with tensorflow part 8: Asynchronous actor-critic agents (A3C). Medium. <https://medium.com/emergent-future/simple-reinforcement-learning-with-tensorflow-part-8-asynchronous-actor-critic-agents-a3c-c88f72a5e9f2>

OpenAI. (n.d.). Part 2: Kinds of RL algorithms¶. Part 2: Kinds of RL Algorithms - Spinning Up documentation. <https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html#a-taxonomy-of-rl-algorithms>