**7-3 Project Two**

Amanda Purnhagen

Southern New Hampshire University

CS 370: Current and Emerging Trends in Computer Science

Professor Timothy Alexander

June 23, 2024

**7-3 Project Two**

When it comes to the Treasure Hunt Game that I was introduced to last week to demonstrate a real-world application of the Q-learning algorithm, humans and machines would absolutely approach and solve this problem using different methods. Depending on the human’s ability, they would use vision to observe the treasure maze. Since it is a 2D, top-down maze image, the human can clearly see which cells are obstacles, where the pirate is, and where the treasure is. Humans past a certain age will simply draw a path to the treasure. Humans below a certain age might need some sort of supervision (a critic) to learn when they have hit an obstacle and how to win. On the other hand, the machine understands this problem through a series of episodes, each of which is a set including the current state of the game board, the potential actions, which available action will earn the highest reward, and the next state of the game board after the machine has completed this action. There is a start state, various transition states, and an end state. The machine attempts to solve the maze with its repertoire of available actions, and it has a memory of past actions that it can draw from for insight.

As we can see, the way a human approaches this problem is far more organic in nature and quite intuitive for people to understand. The way that a machine approaches this problem may not seem obvious and feel clunky at first glance. The machine’s approach has a foundation set in logic and mathematics. The human’s approach takes advantage of their eyesight, motor abilities, intuition, and evolved reasoning skills.

The machine also uses the concepts of exploration and exploitation to solve the problem. Exploration is just that, random moves that the machine takes to learn more about the environment (Gulli & Pal, 2017, p. 270). Exploitation takes advantage of the information learned through exploration by making more deliberate pathing choices that it remembers were successful in the past (p. 270). I ended up experimenting with many different values for epsilon (the ratio of exploitation to exploration), number of epochs, maximum memory, and maximum data size. In the end, I reverted to the original values. Regardless, my algorithm never reached the target win rate for this exercise. In fact, it never exceeded 35.5%. In theory, a good proportion of exploitation and exploration is one that starts high (e.g., 1.0) in exploration and gradually gets as lower (e.g., 0.1), giving the artificial agent the opportunity to exploit the information it learned (Gulli & Pal, 2017, p. 270).

In the treasure hunt game, reinforcement learning is used to determine the pirate’s successful path to the treasure. The machine is not just blindly trying a random path every episode. Each cell holds a reward or punishment in the form of positive or negative points, respectively. If the pirate’s point total reaches the minimum, the pirate loses, and the episode ends. The pirate wins the episode by staying above the minimum point threshold and reaching the treasure. This reinforcement learning algorithm punishes the pirate (subtracts game points) for hitting an obstacle or attempting to leave the maze’s boundary. The algorithm rewards the pirate (adds game points) for reaching a previously untouched empty cell. Of course, the most points are rewarded when the pirate reaches the treasure, signifying its win.

Deep Q-learning was implemented in this treasure hunt game using neural networks. Q-learning alone is the use of a Q-function to calculate the optimal action, maximizing the short-term reward, for each state in a finite set of possibilities (Gulli & Pal, 2017, p. 267). To turn Q-learning into deep Q-learning, the decision to either explore or exploit is introduced (Beysolow, 2019, p. 65). Experience replay (in our case, the pirate’s memory) is used to make predictions on where to move as opposed to manually calculating the potential reward of every action every time (Beysolow, 2019, pp. 65-66).

**References**

Beysolow, T. (2019). *Applied Reinforcement Learning with python: With Openai Gym, tensorflow and keras*. Apress.

Gulli, A., & Pal, S. (2017). *Deep learning with keras: Implementing deep learning models and neural networks with the power of python*. Packt Publishing.