Quinnie Ho

Project Two: Design Defense

Human vs. Machine Intelligence Analysis

The first step a human being would take to solve this maze is to learn the basics about the game. This encompasses learning the objective of the game, rules of how to play, environment or setup, and who the players are and what they can do within the game. Then, the player would visually observe the entire maze to identify its boundaries and obstacles. This allows the player to visually map out possible paths from the start of the maze to the treasure using only the white or free spaces in the maze. A human player uses the concept of trial and error to learn from previous mistakes, such as wrong turns that they took, in order to avoid making the same errors and get to the treasure. A player who is solving the maze for the first time lacks experience and needs to depend on exploring the maze in order to get to the treasure. This tends to result in not the most optimized path that yields the best rewards. However, the more the player plays the game and gain experience with the maze, the more this player is familiar with the maze's environment and can memorize and learn from previous successful and unsuccessful paths. This process of learning from past experiences eventually allows the player to find the optimal path that yields the highest rewards.

The first step the intelligent agent takes to solve this pathfinding problem is inputting the basic data about the game, such as the setup of the maze, objective of the game, how to play it, and the rewards system. This is all the data that are in the TreasureMaze.py file. Then, the agent builds a neural network by passing the input data to hidden layers that create activation, optimizer, and loss functions. These functions are used to train the model. The agent learns using reinforcement learning, which highlights states and rewards for each action the agent takes. More specifically, the agent primarily learns by experience using exploitation, and it achieves this by storing its previous attempts and using experience replay to continuously learn and optimize its approach. However, the agent also learns by exploration to find new optimal paths that yield the best rewards.

Finally, the agent uses a deep Q-learning algorithm to find the best route to the treasure while maximizing its rewards.

The first similarity between the above two approaches is that both players take in inputs regarding the game's environment and objective. This is how the learning process starts. The next similarity is that both learn from experience by storing and pulling previous attempts from memory. The biggest difference between the two players is that the agent is more systematic, calculated, and precise, therefore, it can calculate the best path to the treasure that returns the highest rewards quicker and more efficiently than a human player. However, a human player can better visualize the possible paths to the treasure just by observing a bird's-eye view of the maze and before making a move.

Purpose of Intelligent Agent

Exploitation and exploration are both concepts within reinforcement learning. The difference between the two is exploitation uses known information to find the most optimal path forward that maximizes the rewards, whereas exploration is "finding more information about an environment" to obtain possibly better rewards or outcomes (Lamba, 2018). The ideal proportion of exploitation and exploration for this pathfinding problem is a low epsilon value of 0.1 for the exploration factor. This means that the agent attempts to learn by exploring a new path once out of every 10 attempts (the remaining nine attempts are learning by experience). This is the ideal proportion because the maze's environment is static. However, more exploration is beneficial if the maze's environment becomes dynamic, because then there are more opportunities for better outcomes for the agent to explore.

Reinforcement learning (RL) can help the agent determine the path to the goal by motivating the agent to learn and make optimal choices using a rewards system. More specifically, RL operates using a process cycle where the agent initializes the environment at the starting state, takes an action, is either rewarded or penalized for this action, and then is placed in a new state as a result (Beysolow, 2019). This cycle repeats until the agent arrives at the terminal state, which is finding the treasure. RL allows the agent to find the path to the treasure overtime by teaching it to not repeat actions that resulted in penalties and continue taking actions that earn rewards.

Evaluating Algorithms

I implemented deep Q-learning using neural networks for this game by creating a "qtrain" function that loops through each epoch. This loop starts with initializing the number of episodes and variable to accumulate loss within an epoch, always ensuring that the agent starts at a randomly selected free cell at the start of each epoch, and defining the environment's state. Then, the agent takes either exploiting or exploring actions while it is not at the terminal state. This leads to observing then updating the state of its new environment, and then recording the rewards or penalties of its action at the end of each epoch. Each episode is stored in an experience replay object in order to train the neural network model and evaluate loss.

The training data is inputted into the algorithm from the experience memory. Feedback on the loss, number of episodes, win rate, and amount of training time for each epoch is giving as the algorithm runs, and the Q-values are updated as the model trains. Training continues until a 100%-win rate is achieved. Then, the algorithm returns the total number of epochs and amount of time spent training. The outputted result is an optimal path for the agent to get to the treasure that maximizes its rewards.

References

Beysolow, I. T. (2019, August 24). *Applied reinforcement learning with python : With openAI gym, tensorflow, and keras* (pp. 7-9). Apress L. P.. https://doi.org/10.1007/978-1-4842-5127-0

Lamba, A. (2018, August 27). *A brief introduction to reinforcement learning*. Medium. https://medium.com/free-code-camp/a-brief-introduction-to-reinforcement-learning-7799af5840db