Project Two

Ryne Williams

Department of Computer Science, Southern New Hampshire University

CS 370: Current and Emerging Trends in Computer Science

Dr. Obafemi Balogun

10/12/2023

When humans try to solve a maze, they tend to take a heuristic approach to finding the end. While some humans will take a pseudo-random approach at a maze and follow whatever path happens to appeal to them, there are some approaches that are more structured. Some humans will attempt to take a breadth-first approach when solving a maze, by taking the right or left most path and following it until there is a dead end, back tracking and continuing in the same manner. However, humans are more likely to attempt to identify a path that seems to point toward the end goal and take that path, backtracking if it leads to a dead end. They will commit the already explored paths to memory and only try paths that have not been attempted before until reaching the end. Essentially, humans will try every possible path before finding one that works, then taking that path until it is no longer a viable option, which increases the amount of time that it would take.

This Artificial Intelligence model uses an epsilon-greedy policy to find the fastest, most efficient route to the end of the maze (to find the treasure). The model saves its most recent previous states, up to a specified number of states. It will use its history of states to determine the best possible action to take to move closer to the goal, committing the new state to memory and removing the oldest from memory to make room. If the epsilon value is greater than a randomly generated number, then the agent will choose to explore the maze in the next time step. Otherwise, the agent will determine the best possible action based on a fixed probability. As it completes the game, win or lose, the agent will learn which actions gained the most rewards and which actions the worst and apply that to the new game. This will continue until the maximum number of epochs has been reached or the agent has won enough games to obtain a win rate that has been set.

Both the human and AI approaches involve a certain amount of randomness to find the path to the end of a maze. However, the human approach involves more randomness than the AI approach. The difference in randomness is mostly dependent on the epsilon value in the epsilon-greedy policy of the AI model. An AI model will use significantly less randomness the lower the epsilon value, otherwise it will exploit what it has learned from previous states to determine the best possible action.

Exploration is when an AI agent randomly chooses the next action for the path to the end, not considering the previous actions taken. Exploitation, on the other hand, is when an AI agent uses probability distribution based on previous actions to determine the best path forward to the end goal. In this pathfinding problem, the ideal proportion of exploitation and exploration appeared to be an epsilon value of 0.2. This is the value that got the agent to obtain a win rate of 100% at its lowest of 538 epochs. The next lowest value was 0.05, which reached a 100% win rate after 802 epochs.

Reinforcement learning uses rewards to tell the agent whether they made a good decision or bad decision in the actions that they take in any given step. The agent is designed to want to get the highest reward it can and uses the positive rewards to learn what the right actions are at each state. Once the agent understands the layout of the maze it will begin to make decisions on the next action to take, whether exploitation or exploration, depending on the epsilon value. The agent will begin to build a path in memory that gives it the best rewards, thus leading it closer to the end goal, until it has found the best possible reward set from reaching the goal. This is the path that the reinforcement learning algorithm helped the agent to learn.

When developing the code to implement the Deep Q-Learning algorithm for this Treasure Hunt Game I first initialized the loss and n\_episodes variables. I then called multiple methods from the “TreasureMaze.py” file to set up the epoch for the agent to begin game to traverse the maze just before starting the epochs loop. Inside the loop is where I implemented the algorithm to have the agent make decisions on actions to search for the treasure using an epsilon-greedy policy, following the exploitation method until a randomly generated number was below the epsilon value, at which point the agent would explore for that state. Then I saved the state information in memory and trained the agent to the previous states and calculated the win rate for the episode to let the agent know when the win rate was high enough to end the training. At this point the epochs stop and the number of epochs that it took to reach the desired win rate is displayed to the console.

References

Botvinick, M., Ritter, S., Wang, J., Kurth-Nelson, Z., Blundell, C., Hassabis, D., (2019), Reinforcement learning: Fast and slow, DeepMind, Retrieved from: <https://www-sciencedirect-com.ezproxy.snhu.edu/science/article/pii/S1364661319300610?via%3Dihub>

Das, A., (2017), Introduction to Q-Learining, Towards Data Science, Retrieved from: <https://towardsdatascience.com/introduction-to-q-learning-88d1c4f2b49c>

Opperman, A., (2021), A deep dive into deep q-learing, Builtin, Retrieved from: <https://builtin.com/artificial-intelligence/deep-q-learning>

Wentworth, J., (2018), Problem solving with mazes and crayons, Lesswrong, Retrieved from: <https://www.lesswrong.com/posts/CPBmbgYZpsGqkiz2R/problem-solving-with-mazes-and-crayon>