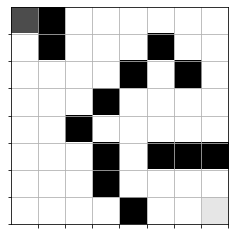
# CS - 370

Project Two

David France

david.france@snhu.edu

Southern New Hampshire University

 For this project we trained an intelligent agent to solve the problem of successfully finding a route to exit a maze. If a human were solving the same maze, it would be a relatively straightforward process. The maze only has one major branch, right after the start in the upper left. If the human follows the straight path down the maze, they will quickly hit a dead-end, turn around, and follow the correct branch to the player’s left of the start. That branch follows straight to the finish. After one successful attempt, most humans would remember to take the left branch at the start and continue to the finish. It should be noted that this is a simple maze. In a more complex scenario, with many branches and decisions, humans will often retrace their own steps, get disoriented, and forget the right choice to make at any given branch.

Our intelligent agent solves the maze in much the same way, initially exploring the new world, then using that knowledge to find a successful route. At each square, the agent either picks a direction to go at random or reaches into its neural network for the best possible choice at that square. The decision to explore or exploit is based on a programmer-decided variable to determine the percentage of the time the agent spends on either choice. After each decision, the agent is rewarded for its choice – the better the choice, the higher the reward. Eventually, the agent has explored every square and knows the best choice to make at each one to maximize total reward, therefore developing a winning route.

While the human appears to take an intuitive approach compared to the agent’s methodical exploration of the whole maze, the strategies are more similar than different. In both, the maze-goer receives a reward for completing the maze – either programmatically or mentally – and negative rewards for making bad moves – expressed as an actual negative reward value for the agent and as frustration for the human. Both the agent and human store experiences to refer to later. One difference is the agent will keep playing until it has explored every square, most humans will stop after one successful run. A second difference is the human can explore more than one square at a time by using their eyes to scan many squares while standing in one. This advantage would vanish in larger mazes. Finally, the agent has perfect memory, so once it has visited every square, it always knows the best route. This would be exhibited in a more complex maze where the human’s imperfect memory would be a detriment.

For the agent, a major part of the process is the exploration v. exploitation decision. Exploration is manifested in the agent making random moves to eventually visit every cell in the maze, then storing the reward from the result. Exploitation means the agent uses their neural network model to determine the best available move for a given cell. Ideally, the agent will spend much of the early game exploring the maze, and much of the later stages exploiting it. To that end, I designed my agent to have an 80% chance of exploring early in the process, with a 5% decrease in exploration chance for each epoch. This can be seen in a starting epsilon of 0.8, and a learning\_rate value of 0.95. This resulted in finishing training in 154 epochs, as opposed to 225 for the epsilon of 0.1 and no learning rate used in the project two milestone. Experimenting with different epsilon and learning\_rate values did not yield an improvement, so I ended up with 0.8 and 0.95 respectively.

Reinforcement learning is a simple way of helping the agent determine the best path to the goal. Much like a human, the agent will seek out the path with the greatest reward. With a little guidance in terms of how often to explore the world before making its own decisions, the agent can explore every cell, making a calculation to determine the best move for each one. When using exploitation, the agent calculates the cumulative reward for each move and chooses the one with the highest value (Surma, 2018).

Deep Q-learning means using a neural network to aid the intelligent agent in its decisions. The model is put into use when the algorithm determines to use exploitation to determine the next move. The Q value is updated by calculating the cumulative updated reward (Surma, 2018). This is as opposed to using a Q-table that stores the best move for each cell. The neural network is a sequential model with three dense layers, PReLu activation, and Adam optimizer. It then uses the predict() function in the exploitation branch of the algorithm to calculate the next step to take. The end result is an agent that knows the best possible move at any given square in the maze.

# References

Surma, G. (2018, September 26). *Cartpole - Introduction to Reinforcement Learning (DQN - Deep Q-Learning)*. From Medium: https://gsurma.medium.com/cartpole-introduction-to-reinforcement-learning-ed0eb5b58288