**Pirate Intelligent Agent & Design Defense**

Guilherme Pereira

Southern New Hampshire University

CS-370-H7432

Dr. Obafemi Balogun

February 20, 2022

**Abstract**

In this paper, I discuss the difference in how a human and machine interact with a problem to reach a solution. I also draw the similarities between the two as the models for machines are mainly based off a mammal and human’s nervous system. The next section is to explain how exploitation and exploration are used together to find the best solution in a reasonable amount of time, as well as how the algorithms, specifically Q-learning, are used to help the agent make a better decision.

**Human and Machine Solution Approach**

Humans and machine both have similar and yet different ways in approach a problem. In the instance of the Pirate Maze game, both humans and machines are given the same resources from the start to beat the game. Both are placed in a random cell in the game and need to find the fastest route before the other. With humans, are approach is to first calculate the number of steps required to reach the goal, how many obstacles are in the way, and if there are any paths we can use to our advantage to exploit. With machines, they initially begin by rolling out different directions to reach a new state and then using that state to propel them into the next one. For this machine, we utilize a Q-learning algorithm and utilize episodic sequences to allow our machine to remember a random past event to make a better decision in the present.

Our machine continues its trials for the number of epochs we initialize it with. We then place the agent into a random cell and reset all its loss an episode counts for the current trial. After setting up the playing field, we run through the game loop until the game is over. From within this loop, we begin to count the episodes, actions, states, and rewards for the agent each time, so it is saved in memory. One important thing to note is when we check our epsilon value, which determines whether we allow the agent to explore a new path or resume exploiting its current trend. As explained by Gulli and Pal on page 279 of *Deep Learning with Keras:*

“We save the current state because we will need that for our experience replay queue, then decide what action signal to send the wrapped game. If we are in observation mode, we will just generate a random number corresponding to one of our actions, otherwise we will use -greedy exploration to either select a random action or use our neural network (which we are also training) to predict the action we should send”.

Upon saving our state, reward, and the status of the game, we train the neural network to evaluate what the next best action is for those given values. When then repeat this process until the agent has won, in which we append the win rate history to the agent’ track. Once the agent has passed the epsilon threshold that we set in the program, the game is considered over, and the agent has won enough times.

Like machines, humans also have the capacity to recall events from our past to complete an assessment. The major difference here are the memories in which humans invoke, as they may be distorted or warped compared to the original memory. Like machines, we decide to choose a memory or sometimes a memory is influenced by how strong the scenario is to the player, such as seeing a specific pattern or scene too many times.

**Pirate Pathfinding Purpose**

Exploration and exploitation are both used together in our daily lives to do what works best for us to achieve a goal, but the same principles also apply to machines attempting at a problem. Exploration is when a random or new action is performed to discover if the new action resulted in higher rewards. Sometimes going left in each scenario is better than going right, and vice-versa. The purpose of exploration is to help the agent make a decision that may allow them to reach higher rewards. Exploitation, on the other hand, is figuring out how we can optimize our current process to achieve better results. Unlike exploration, exploitation is not random and is generally in line with a given mindset or algorithm for the agent to follow.

From my personal testing with Keras, I believe the ideal proportion of exploration to exploitation is initially having a high exploration rate, referred to as a greedy epsilon, and a low exploitation rate. However, as the agent makes more progress and achieves higher rewards, the exploration rate should decrease in tandem with the rewards. As the rewards increase, the exploitation would align itself more with the actions performed to follow a better path and make better optimizations. Does this that exploration should be erased at some point? Unless achieving perfection (which is extremely unlikely), I believe there should always be a time in which exploration can be used to see better alternatives from the given path. In Keras, I set the minimum exploration value to 0.1, while the initial maximum value to 0.9. The discount each time is very small as to not explore at all, so I put my discount at 0.005 (this subtracts our exploration value each time we hit a reward).

Regarding reinforcement learning (RL), I believe it is a good way of helping give machines feedback to make a better decision for the next action. RL has different models as well such as Actor-Critic and Episodic models, each with a different approach but similar method of using RL to help the agent learn how to reach a solution. Humans are similar since we learn to solve many of our problems through trial and error, much like machines using RL.

Algorithms and Solutions

For this game, the main principle of Q-learning was utilized by saving data in the Experience replay object variable and recalling a random experience (state) from the object for each episode. Once the experience has been recalled, the neural network is trained on the current state of the agent and the cycle repeats itself until the agent exhausts all its epochs or it breaches the epsilon value with a greater win rate. We retrieve the agent’s input and position by getting the data from the current experience and creating a model that shows a history of how our agent has made decisions over time.

**References**

Gulli, A., & Pal, S. (2017). *Deep learning with keras: Implement neural networks with Keras on Theano and tensorflow*. Packt.