**CS-370 7-2 Project 2 Design Defense**

Joseph Farrish

joseph.farrish@snhu.edu

Instructor: Professor Travis Smith

[t.smith16@snhu.edu](mailto:t.smith16@snhu.edu)

**Analyzation of differences between human and machine learning approaches to solving problems.**

A black and white square tiles

Description automatically generated The trial-and-error nature of machine learning is not too dissimilar to the manner in which humans learn. In certain respects, humans are more efficient learners, at least initially. When considering a maze, such a machine will, on initial attempts, go out of bounds and assess the rules on each move. In contrast, a human would intuitively know to stay within bounds and avoid dark squares where negative rewards would be the implications. However, with more complex tasks, a machine can teach itself over many iterations, surpassing human capabilities, such as in games such as Chess, Go and the many video game titles used to train emerging AI technologies. Considering the output of the Treasure hunt game, one can observe that in earlier Epochs, the Win count is zero. The win count will increase as the agent progresses, except when exploration occurs. A human attempting to minimize steps will first consider the most efficient route to get to the finish. Afterward, a human would consistently take that route without needing exploration. In contrast, the implemented machine learning algorithm takes some percentage of epochs to explore where it is potentially taking a destructive course, where the goal is to gain more insight into the surrounding environment.

**Assessing the purpose of the intelligent agent in pathfinding.**

After initial epochs are assessed, and a sufficient path is realized, an agent may take a path based on memory to obtain a higher reward and complete the course quickly. This process is called exploitation. At some point, an agent will want to know if there is a quicker path, and this step is called exploration for machine learning algorithms. The percentage of epochs an agent will explore is based on the value set for epsilon. In exploratory epochs, the algorithm will choose random directions and assess the reward by evaluating the environment state, reward, and game status. Based on that evaluation, the algorithm may select the next course of action. Training of the neural network will occur where later explorations will render more favorable paths. The purpose of pathfinding is to allow the agent to become aware of its environment.

When considering Reinforcement Learning the Actor-Critic solution uses value-based approximation and policy-based algorithm functions, where the actor takes the state as an input and outputs the best action based on optimal policy. The critic evaluates the action by computing the value function. With each epoch, both the actor and the critic become more proficient at predicting the best course of action (Karagiannakos, 2018b).

**Evaluation of Q-learning implementation.**

The implemented code does well with the random value generation needed during initial epochs, the values generated using an 8 x 8 matrix implemented using a NumPy array and the agent then attempts to traverse the maze based on random direction values. Enumerated values are randomly selected for up, left, down and right directions. The agent begins the maze at position 0,0. Randomly generated values are evaluated against predictions where the randomly generated value is greater than epsilon the predicted values are assigned to the action else a randomly generated direction is used. Next the environment state, reward value, game status is populated by an action function from the maze object these values are packed as an episode along with the action and previous environment state and remembered by agent in this case the Experience object. The Model object trains the neural network and evaluates loss.

**References**

Karagiannakos, S. (2018b, November 17). The idea behind Actor-Critics and how A2C and A3C improve them | AI Summer. AI Summer. <https://theaisummer.com/Actor_critics/>