Kerrian Offermann

CS-370: Current/Emerging Trends in CS

7-3 Project Two: Design Defense

April 13, 2023

Typically, when a human being is asked to solve a puzzle of any kind, they are given a set of rules that they have to follow before making the attempt. It is then that each person takes their own unique approach to the puzzle. In the case of this treasure hunting game, a human being would more than likely rely on their eyes to examine the entire field, assess where they can and cannot cross, and then visualize a path that can be traced from the start to the end. When it comes to AIs, though, there is no dependence on visual aids. Instead, the AI/pirate agent in this program uses deep machine Q-learning. This is essentially the pirate agent selecting actions that will maximize the rewards it received (Gulli & Pal, 2017). Due to this, it does not need to use visual cues like human beings. It can instead “see” what spots on the field will add or deduct points, and learn to make decisions that will minimize deductions and maximize additions to reward points.

In a way, deep machine Q-learning is a lot like the techniques human being use to learn. We know, for example, that if we pour water in a glass for too long then it will spill and if we pour too little then we will not have enough to drink; therefore, we pour with an understanding of the penalties in mind of pouring too much or too little. The pirate agent approaches the treasure hunt in the same way by approaching a goal with an understanding of the penalties in mind. A major difference in how humans approach solving a maze versus how a machine learns, though, is the amount of time and resources it takes to learn. It takes an AI an immense amount of data and training just to operate on a level that could be comparable to a human being (Botvinick et al., 2019). Some might view this as machines learning “slower” or taking a longer time to grasp things that humans can learn in a matter of minutes, hours, days, etc.

When the intelligent agent (aka the pirate) uses deep reinforcement learning to determine the best path to the treasure, it relies on predictions to decide where to move. Relying on random predictions alone is not entirely efficient, so the agent will use a method known as exploitation and exploration to make better predictions. Exploration makes choices at random beyond what is learned from Q-values to see if it is possible to get more rewards outside of a path already established to be a rewarding one (Dutta, 2018). Meanwhile, exploitation is sticking to the path that is already rewarding without taking any chances with random actions that might fail (Dutta, 2018). This is the key difference between exploitation and exploration: one encourages learning by trying an unexplored path despite the risk of being penalized while another encourages sticking to the paths that are “safe” since they offer definite reward. When it comes to this pathfinding problem, it seems ideal to start with exploration and then rely on exploitation once enough of the maze has been explored. The reason this approach might be best is because it is similar to how humans learn to solve problems. When we do not know what we are doing, we try; when we try enough to understand the consequences of specific actions, we stick to the actions that will lead to less difficulties. At the start of the game, the agent has no idea where to go and thus it is expected that it will make more mistakes in the beginning. It is more towards the end of the puzzle that we expect the agent to understand the best paths to take and make better decisions as a result.

In this game, deep Q-learning using neural networks is implemented by the use of a Q-matrix that pinpoints where the agent will be rewarded or penalized during its movements. The agent if left to explore the maze and build predictions based on the most rewarding paths that it finds during exploration. The learning aspect takes place through the various epochs and episodes where the agent attempts to build a path to treasure. For every 9 attempts of learning based on the experience it gained, the agent will attempt a new path one time to see if there is possibly a path with a better reward than the ones learned.

**References**

Beysolow, I. T. (2019). *Applied reinforcement learning with python : With openai gym, tensorflow, and keras*. Apress L. P..

Botvinick, M., Ritter, S., Wang, J. X., Kurth-Nelson, Z., Blundell, C., & Hassabis, D. (2019). Reinforcement learning, fast and slow. *Trends in cognitive sciences*, *23*(5), 408-422. <https://doi.org/10.1016/j.tics.2019.02.006>

Dutta, S. (2018). *Reinforcement learning with tensorflow : A beginner's guide to designing self-learning systems with tensorflow and openai gym*. Packt Publishing, Limited.

Gulli, A., & Pal, S. (2017). *Deep learning with keras : Get to grips with the basics of keras to implement fast and efficient deep-learning models*. Packt Publishing, Limited.