Nicole Penner

CS-370: Project 2

Design Defense

Human and machines are similar in how they learn, but also have many differences – the same can be said for solving problems. Humans navigating through a physical maze rely on strategic use of their eyes – this helps them to look at the route and figure out the best way to navigate through it (Zhao & Marquez, 2022). Humans will naturally just look for the shortest route that is between the start and end point – this is time saving, and helps to humans to find the most efficient route between two points (Bergan, Conversation, Uysal & Young, 2022). In general, being able to use vision to find this path is more specific to humans – teaching an AI to do the same would be difficult. Humans will also try to identify any shortcuts that are available to reach the end point faster (Bergan, Conversation, Uysal & Young, 2022). Visually a human would be able to see an occupied cell, and avoid choosing it unless the whole maze was covered and you were blindly choosing cells. An AI cannot look at the maze in the same way a human can. The AI in this algorithm starts at the top left and it can move up, down, left, and right. If the agent moves to an occupied cell or moves outside the matrix it will get a penalty. The idea is the pirate (agent) will have no prior knowledge of the map/environment. Using a deep Q-learning algorithm the pirate will be able to find the most efficient way to find the treasure. Therefore, the approaches between human and the AI differ greatly, though the end ultimately, they have the same goal – to find the most efficient way to the treasure on the map. The difference is based on that the AI/agent has to rely on learning without actual vision, while the user can actually see the course and make decisions based on that. The agent makes decisions based on what would be the best choice with the information it currently has about the environment. This is how machine learning and humans will differ to make their way through the maze to the treasure.

Exploitation and exploration are two differing concepts in reinforcement learning in this example of pathfinding. Exploration typically means a large portion of the environment is searched through. The hope of exploration is to attempt to find a promising solution from the data of the large are searched. Exploitation on the other hand typically involves searching through what is identified as a promising part of the environment (Koppula, 2020). Exploration relies on unrefined search results and a global search, while exploitation refines the search to find the more promising region of the environment through a local search (Koppula, 2020). There is a dilemma in using exploitation and exploration – finding a balance between the two. Ideal balance between the two would be to balance searching unknowns and exploiting the best-known strategy found (Jing, 2019). Finding the balance between exploitation and exploration is a key part of reinforcement learning because you want to gain more information that is the most useful for the algorithm to learn, but that involves sometimes too much exploration that could be considered a bad decision. There is also the idea that ideally, as the machine learns it will exploit what worked best (Jing, 2019). The epsilon greedy value dictates how often the algorithm with explore versus exploit. If the epsilon value is .01, that means that only 1% of the time will the agent choose to explore, and 99% of the time it will exploit (Jing, 2019). In my opinion, based on the win rate and times it took for the agent to learn to beat the player I think the epsilon value of .01 is not necessarily ideal – but works well for this problem. To truly find the most ideal value ranges between 5% and 10%, according to many articles I have read about this topic (Jing, 2019). For this problem it seems the balance that epsilon of .01, or 1% works very well based on the time to learn of the agent. Reinforcement learning typically is able to maximize a long-term reward. The agent in this case is able to use reinforcement learning, by learning through consequences of actions it has taken. The consequences could be positive or negative but it helps the agent to get feedback about the environment and learn so it is able to decide and take the path that will have the best result (Ribeiro, 2020). Basically, the pirate will take its experiences and the reward and consequence policy, and this will help the agent to learn how to best navigate the course successfully to beat its opponent to the treasure in all cases. Reinforcement learning helps the agent to learn the course efficiently, and even though it takes a while for it to learn – it will maximize the benefit long-term to beat the user more consistently.

Deep Q-learning using a neural network was key for getting this game to run properly. The difference between Q-learning and deep Q-learning is that deep-Q learning uses a neural network in place of the typical Q-table. Also, state-action pairs are not mapped to a q-value, the neural network will instead input the state to an action and Q-value pair. Deep Q-learning is better suited for use in environments that are considered “more complex”, and more actions by the agent are possible. Deep Q-learning is implemented because of the advantages of using it to solve a more complex problem, and this pathfinding game fits the description for that. In this game we are attempting to learn to efficiently predict the path to the treasure before the user, and since we are unsure of what action will be best to take at a current state – a more traditional method would not prove as effective for this treasure finding game. The Q-value uses the neural networks that have a similar architecture, but their weights differ. Every so many steps, weights are then copied from the main to the target neural network. Use of the neural network allows more stability in deep-Q learning, and the algorithm will also be able to learn more effectively. The input nodes include the states that are made up of the action and q-value pair. The output node of the networks represents an action an agent can take. This is therefore, how deep-Q learning makes use of neural networks in this game.

References

Bergan, B., Conversation, T., Uysal, C., & Young, C. (2022). Maze-Solving Artificial Intelligence Teaches Itself to Take Shortcuts. Retrieved 27 February 2022, from https://interestingengineering.com/maze-solving-artificial-intelligence-teaches-itself-to-take-shortcuts

Jing, H. (2019). Striking a Balance between Exploring and Exploiting. Retrieved 27 February 2022, from https://towardsdatascience.com/striking-a-balance-between-exploring-and-exploiting-5475d9c1e66e

Koppula, R. (2020). Exploration vs. Exploitation in Reinforcement Learning. Retrieved 27 February 2022, from https://www.manifold.ai/exploration-vs-exploitation-in-reinforcement-learning

Ribeiro, J. (2020). What is Reinforcement Learning and 9 examples of what you can do with it. Retrieved 27 February 2022, from https://medium.com/tech-cult-heartbeat/about-reinforcement-learning-2ff0dafe9b75

Zhao, M., & Marquez, A. (2022). Understanding Humans' Strategies in Maze Solving. Retrieved 27 February 2022, from https://www.researchgate.net/publication/250918101\_Understanding\_Humans'\_Strategies\_in\_Maze\_Solving