Southern New Hampshire University

7-3 Project 2: Design Defense

Preston Burkhardt

Michael Susalla, MBA-IS

CS-370-T1204 Current/Emerging Trends in Computer Science

October 15, 2022

Humans vs. Machine Approaches

When it comes to solving a maze, humans and machines can take similar or completely different approaches. For humans, we should be specific on the type of maze that is being solved as it will dictate most people's approach to solving the maze. For example, a human solving a maze that is drawn out on paper can see the whole maze and make decisions based on their view of the entire maze. A human completing something such as a corn maze where they might not be able to see other parts of the maze will take a different approach to solving said maze. Let us look at how a human solves a maze that is drawn out on a piece of paper. Usually, someone starts at the beginning and attempts to find a path in the direction of the end of the maze. If the start is in the top left and the end in the bottom right, the initial search would be diagonal from top left to bottom right. Sometimes there is no viable route along this heading, so then the human would select a different heading from the start (either top left to top right or top left to bottom left). One thing to note is that with this "top down" view over the entire maze, a human can quickly and easily see if a specific path leads to dead ends and then discount that path from possibly being a viable solution. Another method that some people use is to work backwards from the end of the maze. Sometimes this is done until they reach the beginning, but most will work backwards partially in order to get the general direction of where they should travel from the start, then attempt to go from the start to where they left off when working backwards. For example, you may work backwards from the end of the maze in the bottom right and find that the correct path goes from the end to the top right. From here, you'd go to the start of the maze in the top left and attempt to find a path to the right that leads to where you left off during your "reverse path" phase of solving. When posed with a maze where a human doesn't have an overhead view of the maze's layout, such as with a corn maze, either a brute force method is

used to solve the maze, or a more exploratory method is used. The brute force method is simple, but can be time consuming. This is where the person just runs their hand along one wall of the maze, left or right, and goes forward until they reach the end. The exploratory method can be quicker, but also more difficult if the maze is very complex and/or the person doesn't have good memory. This is where a person just tries random paths until they find the end. They may go down one path to find only dead ends, then try the path not taken yet. Issues arise when someone cannot differentiate between a path taken previously and one that has not taken yet, then travels down already explored paths. Eventually though, they should begin to recognize paths/layouts that they have taken multiple times and make the connection that the only reason said path has been traversed multiple times is because it doesn't lead to the end of the maze.

For our project, we used Q-learning to teach the agent how to solve the maze. The agent is in a similar position to a human that is in a corn maze as it cannot see the overall layout of the maze in order to make decisions about which overall direction to travel in or to quickly discount paths, that when viewed from above, clearly do not lead to the end of the maze. The agent finds its way through the maze by using a function to calculate the Q-values of actions available in a current state. A Q-value is a representation of possible rewards by selecting an action. The higher the Q-value an action has, the more likely it is that choosing that action at that specific time leads to a reward (completing the maze). The agent is fed Q-values for the actions available to it from a neural network that is fed a sample of previous actions taken by the agent in previous solve attempts. The agent aims to complete the maze by choosing the action with the highest Q-value (potential for reward) at each time interval/step. As time passes and more attempts are made by the agent to solve the maze, the neural network gets better at estimating Q-values, which in turn makes the agent better at choosing the correct path to solve the maze.

The way that a human and the agent (using Q-learning) solves the maze can be both similar and dissimilar because there are different methods that humans use to solve the maze. When a human is tasked with getting through a corn maze, the exploratory method is the most similar to Q-learning in that both the human and agent use data about previous actions to guide them on choosing their next action. If the human uses the brute force method (hand on the wall), it is nothing like the agent's method to solving the maze. No actions are taken except "move forward." No learning needs to occur in order for the human to be able to solve the maze using this method. There is also not much of a similarity between a human solving a maze drawn on paper, where they have an overhead view of the entire maze, and the way that an agent solves a maze. This is due to the fact that the human can see all aspects of the maze while the agent cannot. The only similarity is that the human may base their actions on previous knowledge like the agent does, but the human's knowledge is much more broad in that they usually discount entire sections of the maze at a time rather than individual paths.

Purpose of the Agent

For the maze problem the Agent has two purposes. The main one is to solve the maze by finding the best route from the start to the end of the maze. The second is to gather information about the maze will be fed to the neural network, which gives information to the agent on which actions it should choose at a given step/state. This is where the balance of exploration and exploitation comes into play. Exploration is where the agent doesn't always take the action with the highest reward potential with the intent that it either finds a better action or a better reward (Deeplizard, 2018). Initially, the agent has no knowledge of the layout of the maze or potential rewards found in the maze. It must explore the maze to build up its knowledge base. Exploitation is when the agent chooses the actions based on their reward potential. For example, if it finds a path to a reward, it will exploit its knowledge of said path in order to take that path each time to get the reward (Deeplizard, 2018). It's as if you have a favorite restaurant and you know the path from your house to said restaurant. If you take an "exploit" approach, you travel that known path to the restaurant to get the reward of a good meal. Despite knowing the route to the restaurant, you may want to use an "exploration" approach to either find a better restaurant (better meal/better reward) or a quicker route to your current favorite restaurant. This example also highlights the importance of exploration versus exploitation. Therefore, I feel that a specific proportion of exploration to exploitation isn't the best solution for training our agent to solve a maze. I propose a method in which the agent initially favors exploration heavily and then transitions to an exploitation approach over time. This is referred to as an Epsilon-Greedy strategy (Deeplizard, 2018). It ensures that the agent gains knowledge of the maze and rewards initially, then transfers to actions that maximize the rewards gained later. It is preferred that the

agent never goes completely for exploitation as it does not normally have knowledge of the entire maze and therefore cannot say for certain that no other rewards exist within the maze.

Reinforcement learning helps determine the path to the goal for the agent by providing a basis for the agent to make decisions on what actions to take at a particular time/step. As the agent makes correct decisions on what actions to take, it is rewarded. This is the basis of what reinforcement learning is. Good actions are rewarded, bad actions are penalized (Osiński & Budek, 2018). Over time, the agent is informed of the correct actions to take at a specific time/step by an algorithm that calculates the reward potential for each action at that given time/step based on the rewards and/or penalties the agent received for those actions in previous attempts. Eventually, the algorithm should be able to tell the agent the best action to take at each step in the maze so that the agent can find its way through maze in the most efficient manner.

Neural Networks and Deep Q-Learning

For this project, we implemented deep Q-learning by creating a neural network that handles the estimation of Q-values for state/action pairs. Q-learning involves the assignment of values to actions available at specific states based on their potential to lead the agent to a reward for taking said action. This can be done via calculations and a Q-value table, but becomes very compute intensive when there are many actions and states. To get around this, we implement a neural network to estimate the Q-values of actions at a given state. The neural network looks at the rewards gained when the agent performed previous actions and uses that information to estimate Q-values for the current actions available to the agent (TensorFlow, 2021). For example, if the agent is five steps into maze and then has a choice to go right or left, the neural network will take a sample of previous actions taken by the agent and surmise that when the agent went left at this fork, they solved the maze 90% of the time, but when they went right, they solved the maze 5% of the time. The neural network would then estimate that the Q-value (potential for reward/completing the maze) for going left is higher than the Q-value for going right. The agent then looks at these Q-values and makes a decision on which action to take. In basic terms, the neural network's purpose it to estimate the Q-values of state/action pairs by analyzing the agent's past experiences because calculating the Q-values would take and excessive amount of time and power.

References

Deeplizard. (2018, October 11). *Exploration Vs. Exploitation - Learning The Optimal Reinforcement Learning Policy*. Retrieved from Deeplizard: https://deeplizard.com/learn/video/mo96Nqlo1L8

Osiński, B., & Budek, K. (2018, July 05). *What is reinforcement learning? The complete guide*. Retrieved from DeepSense.ai: https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/

TensorFlow. (2021, December 08). *Introduction to RL and Deep Q Networks*. Retrieved from TensorFlow: https://www.tensorflow.org/agents/tutorials/0_intro_rl