CS 370: Project 2

Jason Kremhelmer

Southern New Hampshire University

CS370: Current / Emerging Trends in CS

Roland Morales

10/20/2024

The task was to design an approach to solving a pirate maze. For this, a DQN (Deep Q Network) was used in Python. This drives the GameExperience and TreasureMaze classes. The movements were restricted to left, right, forward, and backward. If the pirate agent reached the treasure, there was a reward, and thus penalized if it was blocked, visited the same squares, or attempted an invalid move. The agent was trained through reinforcement learning on the optimum action for each state. The strategies fall into two categories, exploration, and exploitation.

In exploration, agents learning refers to a strategy of choosing actions it has not yet attempted, or it has reason to believe it will lead to new knowledge that can improve its long-term performance. Exploration focuses on acquiring new information that will lead to better decisions in the future.

The action of choosing actions the agent believes will lead to maximum immediate rewards is exploitation. Exploitation focuses on gaining the maximum short-term performance based on its current knowledge.

In order to have the agent explore a lot early on, but use the knowledge gained in later attempts, a learning rate decay was used (Haswani, 2020). An alpha value was calculated using the reciprocal of the epoch number multiplied by the decay rate. If the alpha was below the threshold, epsilon (ε), the agent stopped and used gained knowledge.

As the agent randomly wandered then relied on gained knowledge, a human may ineract in the same way. A person may start off just wandering around, but once it stared seeing a pattern, or familiar surroundings, it would use that knowledge to make better choices. While using this knowledge, they would then learn new pathways, both good and bad, and continue to gain knowledge. Eventually, with any luck, they would reach the end of the maze. A difference being, a human can see what lies ahead, while the agent cannot, it is limited to the square it is in. This is due to reinforced learning (RL) with DQN is model-free.

With no ‘model of the universe’ the agent ‘can be though of as [using] an “explicit” trial-and-error algorithm’ (Model-free (reinforcement learning), 2023). This simply means the agent cannot plan-ahead or think beyond its current move while in learning mode. This behavior does not pertain to humans, as we can learn and plan simultaneously.

As a good rule of thumb, in the beginning the agent should give more priority to exploration. This would allow the agent to explore different paths and learn about the environment. As the agent learns, it would slowly shift to exploration, as this would allow it to use the knowledge it has gained to progress through the maze.

In order to achieve a balance between exploration and exploitation, the decay rate was manipulated. A decay rate of 0.1 was what was settled on, this gave an epsilon value of 0.1. This allows for 90 epochs of random exploration (learning) before exploitation (experience-based) predictions were used. It is unknown what a human may or may not need in a situation where their visibility was restricted to a single move.

## *How can reinforcement learning help to determine the path to the goal (the treasure) by the agent (the pirate)?*

As stated earlier, the agent gains knowledge through actions. During each state the agent learns a policy that maps that state into an action. These states, in this case, correspond to different locations on the map, and thus the actions correlate to movements the agent can make to navigate to that location.

Once each action is completed, the agent is rewarded or punished with the goal of maximizing the total reward over time. As previously stated, exploration and exploitation are used to define the actions the agent can choose within the states.

In the learning process, the environment is explored by the agent, and it updates its action-value function based on the feedback it receives. Over time once the agent gains more experience, the action-value function converges into optimal values. This turns into the policy it learns to become more accurate and efficient at finding the correct path.

## *How did you implement deep Q-learning using neural networks for this game?*

DQN was used to map the action-state pairs. With each action a reward or punishment was determined using this code block:

*def get\_reward(self):*

*pirate\_row, pirate\_col, mode = self.state*

*nrows, ncols = self.maze.shape*

*if pirate\_row == nrows-1 and pirate\_col == ncols-1:*

*return 1.0*

*if mode == 'blocked':*

*return self.min\_reward - 1*

*if (pirate\_row, pirate\_col) in self.visited:*

*return -0.25*

*if mode == 'invalid':*

*return -0.75*

*if mode == 'valid':*

*return -0.04*

As the transition from exploration to exploitation occurs these rewards are the guide for the agent to make the next possible move. This was based on a decaying learning function. Above a threshold (ε) exploration is used, while below exploitation is used. The code is shown here:

*if 1 / (1 + epoch \* decayRate) < epsilon:*

*Action = np.argmax(experience.predict(previous\_envstate))*

*else:*

*Action = random.choice(validAction )#randomily pick from the validActions*

In the second epoch, the agent began ‘winning’ and by the second epoch (001) and by epoch 1043 had 100%-win rate. See image below for final path:

A black and white squares

Description automatically generated

# References

Haswani, V. (2020, September 3). *Learning Rate Decay and methods in Deep Learning*. Retrieved from Medium: https://medium.com/analytics-vidhya/learning-rate-decay-and-methods-in-deep-learning-2cee564f910b

*Model-free (reinforcement learning)*. (2023, December 20). Retrieved from Wikipedia: https://en.wikipedia.org/w/index.php?title=Model-free\_(reinforcement\_learning)&oldid%20=1142602733