Project Two: Design Defense

Kyle Dale

SNHU: CS-370-Current/Emerging Trends in CS

Professor Alexander

December 10, 2023

Difference Between Human and Machine Learning:

Although I am a human, and can describe how I would go about solving the maze, I cannot speak for all humans, as our minds are as varied as our appearance. With that stipulation considered, I would approach the maze initially with a few considerations. First, do I have a birds-eye view like in the code example, or am I in the maze. Additionally, what do my senses deduce? Can I see a difference in the blocks that I should not touch versus the ones I can? Is the task simply walking the open path, and if so, it would be incredibly trivial. After surveying the initial scene, I would progress with any information gleaned, or previously held about the situation, and cautiously proceed through the maze. Assuming a one-to-one interaction with the maze, a human would merely need to observe the environment and proceed along a clear path.

With a basic understanding of how a human may approach solving the maze, we can begin to analyze and compare the differences in the steps taken when the intelligent system solved the maze. The specific algorithm implemented utilizes Q-learning, a value-based approach to reinforcement learning that aims to estimate the value function, also known as the Q-value, which represents the expected cumulative reward for taking a specific action in a specific state (Gulli et al, 2017). The basic steps that Q-learning rely on can be summarized in the following:

Initialization: The first step is to initialize the Q-table as a means to facilitate function. The Q-table is made up of two distinct matrices, the Q-matrix and the R-matrix. The Q-matrix holds the accumulated knowledge about the environment, initialized to 0 and updated after each step in the environment, and is responsible for maximizing the rewards obtained (Beysolow, 2019). The R-matrix, on the other hand, represents the environment, with rows indicating states and columns denoting rewards for transitioning to other states

(Beysolow, 2019). Put simply, the Q-table is the map to the environment and the R-table is the world itself.

Observation: During this step, the agent observes the current environment state in preparation for analysis and selection of an action.

Action: During this stage, the agent (pirate, in this example) chooses an action. The choice of an action is either determined with a random action among potential actions, selected with probability epsilon, or by selecting the action with the highest Q-value.

Update: After an action is taken, the Q-table is updated with the results. The Q-learning update rule can be seen as follows:

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha\left[\left(r(s_t, a_t) + \gamma . \max\left\{Q(s_{t+1}, a_{t+1})\right\}\right) - Q(s_t, a_t)\right]$$

where s represents the current state, a the chosen action, r the reward received, s′ the next state, a′ the action with the maximum Q-value, α the learning rate, and γ the discount factor (Gulli et al, 2017).

Repeat: The previous steps are repeated until the designated number of epochs are completed, or the system reaches convergence and automatically exits operation.

Although there are similarities between the method by which the intelligent system would solve the maze, and the way I would, there remain intrinsic differences. The main difference is that the intelligent system receives a very literal translation of the environment, as well as very literal rules and restrictions for operating within the maze. All of the decisions within the system are based on predefined rules and functions to determine value and reinforce best practices. Although at a high-level, this operation is the same as described by myself, humans have access to much more ambiguous information such as sensory information, preconceived notions or experiences, as well as the capability of innovation and abstract thought. All of these factors are relevant when completing a task like a maze, and are wholly absent within the AI.

Purpose of Intelligent Agent in Pathfinding:

As mentioned in the prior response, actions had the possibility to be chosen at random with a probability of epsilon, as Q-learning employs epsilon-greedy exploration to facilitate stable and effective learning in a dynamic environment. To conceptualize this, it is important to recognize the exploration-exploitation trade-off, where the first possible path to a solution may not be the best path, however, it is also true that it is unlikely that every time you continue searching you will find a better solution, therefore in effect abstaining from a solution. This conundrum is controlled by the epsilon parameter, which aims to solve the problem of choosing between a variety of options with the end-goal of maximizing reward (Beysolow, 2019). This is ultimately done with the goal of maximizing the cumulative reward through optimal action choice, effectively solving the exploration-exploitation conundrum.

In regards to this specific pathfinding problem, I utilized an epsilon level of 0.1. Although this is a fairly standard value, I believe the AI system would have benefited from a slightly increased epsilon level (ChessProgrammer, 2020). As seen in the submitted images, the agent took 242 epochs to reach ten wins. However, by epoch 590 it had accrued 258 wins, and by the final epoch, 656 wins, and a near perfect win rate. As one can see from this output, the agent took a substantial amount of time to find and converge on a winning strategy. With a higher epsilon value, the agent will take more random actions, and will be more likely to find a winning strategy faster. As a possible mitigation for the exploration-exploitation conundrum, a scaling epsilon value could be implemented to reduce randomness as the system accrued wins.

Reinforcement learning, specifically the Q-learning algorithm in this instance, are essential to the agent's completion of the maze. As explored in previous responses, the agent utilizes the Q-value to predict and take the action most likely to maximize reward, and utilizes this experience to fine-tune and update the algorithm for better performance. Without the ability to reinforce upon prior learning, the agent would be randomly selecting a different path each time, with no real direction. By striking a balance between exploration and exploitation, we can ensure our agents experiment to find new solutions, while relying upon prior experience to inform current attempts.

Algorithms in Solving Complex Problems:

For this assignment, deep Q-learning was implemented using a neural network at various steps. Although the neural network itself was predefined, I implemented Q-learning in the following ways:

Initialization: As previously discussed, the epsilon parameter was initiated to 0.1, but a slightly higher (more random) value would have likely been beneficial. Additionally, the n_epoch parameter was defined, where epochs represent the total iterations through the maze. To further bolster reinforcement learning, experience replay was utilized, with max_memory and data_size being defined. Max_memory and data_size are relevant to experience replay, in that max_memory represents the total number of experiences (episodes) the agent will store in its replay memory, while data_size represents the number of experiences samples from replay memory for each training iteration (deeplizard, 2018).The inclusion of experience replay allows the system to benefit from prior experience in a way that is absent in traditional Q-learning, and thus allows this implementation to be classified as deep Q-learning.

Training: The largest personal effect on the implementation of Q-learning algorithm on this project was centered around the implementation of training for the algorithm. In concordance with the requirements and traditional steps outlined for Q-learning prior, my implementation for training begins with a for loop surrounding each epoch where the pirates starting position is defined. After this, the agent observes the environment (envstate) of the maze. Next, variables for loss, episode numbers, and history tracking are initialized. A while loop follows the variable initialization, dictating that while the game is not over, the agent will choose an action based on epsilon-greedy strategy, execute the action, receive a reward, observe the next state and store the experience, train utilizing the new and prior experiences, evaluate loss of the model, and track number of episodes. After this loop, the training implementation will check if the pirate has won or lost the game, and subsequently update win history; if the win rate is above the threshold, the model passes completion check and training breaks. Epoch information is subsequently printed to the console, reporting loss, episode count, win count, and win rate. This implementation utilizes the normal functionality of Q-learning in combination with experience replay, to provide more adequate training and faster convergence. As mentioned previously, the only additional, potential change I can see to make would be to slightly increase the epsilon value.

Ultimately deep Q-learning is utilized in this assignment to train an agent (pirate) to navigate to a goal (treasure). Through the use of deep Q-learning the agent was able to attain a win rate near 100% by the end of training. Although there were not enough epochs to raise the overall win-rate to 90%, this implementation was much more successful in operation than the milestone. I am glad I was able to solve the errors present in my previous assignment, and present a functioning pirate for the final.

References

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

Chessprogrammer. (2020, November 26). *What should the value of epsilon be in the Q-learning?*. Artificial Intelligence Stack Exchange. https://ai.stackexchange.com/questions/24857/what-should-the-value-of-epsilon-be-in-the-q-learning.

Deeplizard. (2018, November 3). *Replay memory explained - experience for DEEP Q-network training*. YouTube. https://www.youtube.com/watch?v=Bcuj2fTH4_4.

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.

Kerner, S. M. (2023, May 22). *What is Q-learning?*. Enterprise AI. https://www.techtarget.com/searchenterpriseai/definition/Q-learning.