Steven Foltz

02/19/26

Project

One of the main challenges in artificial intelligence is the creation of intelligent entities that can navigate challenging settings. This design defense details the methodology used to develop a deep Q-learning pirate intelligent agent for a treasure hunt game. To find the treasure before the human player does, the agent must make their way through an 8x8 maze filled with obstacles. The goal of the intelligent agent in pathfinding is evaluated, the distinctions between human and machine problem-solving techniques are examined, and the application of deep Q-learning algorithms to this challenging problem is assessed.

An artificial intelligence agent and a person would approach solving a maze in very different ways. In order to quickly comprehend the maze pattern, humans rely on visual perception and spatial thinking. A human would quickly locate the treasure in the lower-right quadrant of the maze and mentally map out possible routes to it. In order to recognize dead ends, hallways, and barriers, humans adopt heuristics like "always turn right" or "avoid paths that look like they lead to walls." Memory is essential because it enables people to go back and attempt different routes by remembering which paths have been explored and which lead to dead ends.

The intelligent agent, on the other hand, approaches the pathfinding problem using statistics and mathematics. With each cell stored as either a wall (0.0) or free space (1.0), the agent depicts the maze as a numerical vector. The location of the treasure is invisible to the agent, who lacks visual comprehension of the maze. It must instead find the prize by trial and error, only getting reward signals when it succeeds. The agent approximates a Q-function that relates state-action pairs to anticipated future rewards using a neural network. By means of experience replay, the agent builds a statistical understanding of which behaviors result in success in various situations by storing and repeatedly learning from past events.

Learning from experience and striking a balance between exploration and exploitation are two ways that human and machine approaches are comparable. In order to explore the world, both humans and AI agents must take use of established, effective routes while also trying new ones. But the mechanics are very different. While AI systems need specific training data and quantitative calculations, humans employ intuitive reasoning and can generalize from similar scenarios. AI agents must learn the concept of "moving toward the treasure" solely through reward signals; they lack a semantic knowledge of direction or purpose, whereas humans comprehend this concept.

In order to effectively find the treasure, the intelligent agent must navigate the maze environment on its own. With positive reinforcement for obtaining the treasure and negative reinforcement for each step taken (promoting efficiency), the agent must learn a policy that maximizes cumulative reward in terms of reinforcement learning. The agent

must use the state representation to recall visited cells in its partially viewable environment, where it only knows its current cell.

Two essential tactics in reinforcement learning are exploration and exploitation. When an agent acts randomly to find new routes and learn more about its surroundings, this is known as exploration. The epsilon parameter (ε) in this version controls exploration; it starts at 1.0, which means the agent acts randomly at the start of training. This enables the agent to map out the surroundings, locate the wealth, and identify whether places have dead ends and walls. Exploitation happens when the agent chooses actions based on the Q-values predicted by the neural network and applies the knowledge it has learnt to maximize rewards. This is an example of the agent using what it has learnt to effectively accomplish its objective.

For this pathfinding task, the optimal ratio of exploration to exploitation follows a decaying schedule. High exploration (80–100%) is necessary during the first training phase (about the first 200 epochs) to guarantee the agent finds the treasure and explores the entire maze. The agent may converge on less-than-ideal routes or fail to find the treasure altogether if there is insufficient early research. A balanced strategy with 20–50% exploration during the middle phase (epochs 200–600) enables the agent to improve its knowledge while still finding possibly superior routes. As the agent concentrates on using its learnt policy to reliably locate the treasure in the late phase (epochs 600+), exploration should decline to 1–10%.

Through a number of important techniques, reinforcement learning aids in determining the route to the objective. Immediate feedback is provided via the reward signal, which encourages efficiency by giving the agent a little negative reward (-0.04) for each step and a +1 for reaching the treasure. The Bellman equation, which discounts future rewards by a factor gamma (γ=0.95), is used by the Q-values to propagate success backward through state-action pairs. This trains the agent to take long-term effects into account while placing a higher priority on short-term gains. By adjusting its estimations in response to the discrepancy between expected and actual rewards, temporal difference learning enables the agent to learn from partial episodes as well as full successes.

A neural network architecture created especially for this maze navigation problem was used to implement deep Q-learning. The flattened maze state, a 64-dimensional vector that contains the labyrinth layout and the agent's current position, is fed into the network. The maze's size is reflected in the architecture's 64-neuron input layer. There are 64 neurons with Parametric Rectified Linear Unit (PReLU) activation functions in each of the two hidden layers. Since some actions in Q-learning may have negative anticipated values, PReLU was selected over normal ReLU since it permits negative values. Four neurons in the output layer reflect the expected Q-values, one for each of the four possible actions (left, up, right, and down).

The GameExperience class, which saves previous experiences as tuples of [state, action, reward, next_state, game_status], was used to provide experience replay. A varied sampling of previous interactions is provided by setting the memory buffer size to eight times the maze size (512 experiences). In order to break temporal correlations and provide

more consistent learning updates, random batches of 32 events are sampled from this buffer during training. By repeatedly exploiting prior experiences, this method increases sample efficiency and keeps the agent from overfitting to recent encounters.

The problem of moving targets in Q-learning was resolved by implementing a target network. It can result in detrimental feedback loops where Q-values oscillate or diverge since the same network is used for both action selection and evaluation. Updated every ten epochs, the target network is a perfect replica of the primary network. Training stability and convergence are greatly enhanced by this separation, which gives the Bellman update stable goal values.

As previously mentioned, the epsilon-greedy exploration approach strikes a compromise between exploration and exploitation. A steady transition from exploration to exploitation is ensured by the implementation's 0.998 epsilon decay per epoch. Even after prolonged training, the agent is guaranteed to retain some degree of adaptability because to the minimal exploration rate of 0.01.

The agent's ability to properly navigate the maze from any starting place is confirmed by the completion check function. Because it verifies that the agent has actually learnt generalizable policies rather than just memorizing paths from frequent starting places, this test is more demanding than merely attaining a high win rate during training. The function runs a full game by iterating over all of the maze's available cells, resetting the environment to each point. The agent passes the completion check only if every position leads to victory.

An intelligent agent is successfully trained by the deep Q-learning implementation to navigate the maze and locate the treasure. In order to find the best routes, the agent balances exploration and exploitation through trial and error. In contrast to path memorization, the completion check guarantees that the answer generalizes to all beginning places. Experience replay, target networks, and decaying exploration are among the design decisions that adhere to recognized best practices in reinforcement learning and help the agent succeed. Using Double DQN to lessen Q-value overestimation and Prioritized Experience are two possible future enhancements. Dueling Network designs can be used to segregate state value estimate from action advantage estimation, or replay can be used to concentrate learning on significant transitions.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). MIT Press.

Watkins, C. J., & Dayan, P. (1992). Q-learning. Machine Learning, 8(3-4), 279-292.

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-learning. Proceedings of the AAAI Conference on Artificial Intelligence, 30(1), 2094-2100.

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. arXiv preprint arXiv:1511.05952.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Freitas, N. (2016). Dueling network architectures for deep reinforcement learning. International Conference on Machine Learning, 1995-2003.