**CS-370 Project Two**

Zachary N. Locke

Southern New Hampshire University

CS-370: Current/Emerging Trends

Mohammad Habibi

October 26, 2025

**What I built**

I trained a Deep Q-Network (DQN) to guide the pirate to the treasure. The community takes a flattened snapshot of the maze and returns four numbers, one for each circle. During schooling, it learns to align those numbers (Q-values) with the long-term return you would get by taking that pass now and playing properly afterward. Replay reminiscence: every step (state, action, reward, next\_state, done) is saved in a buffer. Instead of training on consecutive steps (which might be rather correlated), I sample random mini-batches. That simple trick stabilizes learning and lets the network revisit valuable experiences later. Target network. The bootstrapped label uses a second network that lags the main one. Periodically, I copy the leading network’s weights into the target. This continues the goal of chasing a moving goalpost and reduces comment loops. For movement choice, I used ε-greedy: with probability ε, the agent chooses a random action; otherwise, it selects the action with the best Q-value. I started with an extraordinarily high ε (around 0.35) and slowly decayed it, so the agent spends time exploring before settling into exploitation. That slow decay mattered: with sparse rewards, the agent needs time to stumble into full paths before it can commit to them.

**Why is this maze trickier than it looks**

This is a sparse-reward, long-horizon problem. Most moves carry minor penalties; the only significant positive reward is at the treasure. Some start cells are many steps away, which means the good signal arrives late. That combination demands many episodes and generous step caps; otherwise, the agent never experiences full successes from the far corners. Given the time limit, I prioritized getting a firm, working policy from the top-left start (the single test) and, as time allowed, added training with random starts to broaden coverage. I also raised the discount factor to γ = 0.99 (so success at the end influences earlier choices) and allowed more steps per episode, so long routes do not get cut off.

**What I tried, what happened, and what I kept**

Episodes. I moved from quick smoke tests to multi-thousand-episode runs. More episodes noticeably improved consistency from distant starts. Exploration schedule. ε started 0.30-0.35 and decayed slowly (0.998-0.9993), with a floor around 0.05-0.10. Slower decay meant the agent kept discovering paths instead of locking in too early. Steps per episode. Increased from 4 times the maze size to 8-9 times. That reduced “fake losses” where the agent was on a good route but ran out of time. Replay and batch size. Replay grew to 12-16 times the maze size with batch sizes of 64-128. Larger, more varied batches calmed the bootstrapped target and reduced regressions. Target sync, every 100 episodes worked well; more frequent syncs did not help.

**Dead ends and fixes**

Early versions checked a game status string of "not\_over" while the environment returned "running". That could let episodes drift forever. I updated the loop to accept either value and added a per-episode step cap. TensorFlow’s GPU warnings cluttered the output in Codio. I forced CPU and silenced logs so training feedback was readable. When ε decayed too quickly, the agent plateaued. Resetting ε before long runs and using a slower decay fixed that. I used a DQN with replay and a target network under an ε-greedy policy. Because rewards are sparse and some paths are long, I kept exploration alive longer, raised the step cap, and trained for more episodes. Under the deadline, I ensured a working single-start policy first and then pushed generalization, documenting the settings and results along the way.

**References**

Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529–533. <https://doi.org/10.1038/nature14236>

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