Lab 4: Reinforcement Learning

Environment and Model: MountainCar-v0

The Mountain Car environment is straightforward yet challenging because the car's engine is underpowered, requiring the agent to learn a strategy of oscillating back and forth to build momentum. The Mountain Car focused on experimentation with techniques such as reward shaping, exploration strategies, and function approximation methods.

Specific Changes

Improved Network Architecture

The Deep Q-Network with ReLU activation functions which help to mitigate the vanishing gradient problem was altered to increase the layer size to (256, 512). This allowed the network to learn more complex features as it goes deeper.

Hyperparameter Tuning

In regards to hyperparameter tuning, five key hyperparameters were changed:

1. Learning Rate: In the Mountain Car environment, because exploration is crucial, a high learning rate allows for faster convergence.
2. Replay Buffer: A large replay buffer allows for better sampling of experiences, improving the stability of learning.
3. Gamma: The high discount factor emphasizes long term rewards. In this case, immediate rewards are sparse and this is crucial for the MountainCar problem.
4. Epsilon Decay: The goal state in this environment is not easily reachable thus, the slow decay rate allows for extended exploration.
5. Batch Size: A moderate batch size of 64 helps to balance between computational efficiency and learning stability.

Advanced Techniques

1. Double DQN: The decoupling of the action selection from the action evaluation in the Double DQN helps to mitigate the overestimation of Q values that can occur in standard DQN.
2. Custom Reward Function: The function implements reward shaping which can significantly speed up learning in sparse reward environments.
3. Extended Episode Length: The maximum episode steps are increased to 1000, allowing the agent more time to explore and reach the goal.
4. Target Network Updates: The target network is updated every 1000 episodes, which stabilizes learning by reducing the moving target problem.
5. Frequent Training: The agent trains every 4 steps, allowing for rapid learning from recent experiences.

Environment and Model: Acrobot-v1

The Acrobot environment is a pendulum system that swings the end-effector (the free end of the second link) to a height at least the length of one link above the base. The task requires the agent to learn effective long-term planning and precise control to overcome the limited actuation and the effects of gravity.

Specific Changes

Improved Network Architecture

The DQN classes use a sequential model with three linear layers and ReLU activations to result in better feature extraction and representation learning.

Hyperparameter Tuning

1. Learning rate (1e-4): To achieve more stable learning, a lower learning rate is used in more complex environments such as Acrobot-v1.
2. Epsilon decay (0.99): In environments where the optimal policy is not obvious, the slower decay rate allows for extended exploration.
3. Epsilon minimum (1e-4): A very low minimum epsilon ensures that the agent continues to explore.

Advanced Techniques

1. Extended Training Duration: The training loop ran for 5,000 episodes to allow for more time to learn and improve its policy.
2. Large Replay Buffer: A larger buffer with a maxlen of 1,000,000 allows for diverse experience sampling, which leads to more stable and efficient learning.
3. Solving Condition: The code checks for a solving condition (avg_reward > -92.95), which is a reasonable benchmark for good performance in the Acrobot-v1 environment.