Summary of the DQN Algorithm

Group Members

Lisa Godwin: 2437980

Nihal Ranchod: 2427378

Brendan Griffiths: 2426285

Konstantinos Hatzipanis: 2444096

Training Process

The training process for the DQN algorithm. Here's a summary of the key steps:

1. **Environment Setup:** The Atari environment is created and wrapped with various preprocessing wrappers to prepare the state for the DQN.
2. **Agent Initialization:** A DQN Agent is created with the specified hyperparameters.
3. **Training Loop:** The agent interacts with the environment for a specified number of steps. For each step:
   a. An action is chosen using an epsilon-greedy policy.
   b. The action is executed in the environment, and the next state, reward, and done flag are observed.
   c. The experience (state, action, reward, next_state, done) is added to the replay buffer.
   d. If enough steps have passed, the agent performs a learning update:
      - A batch of experiences is sampled from the replay buffer.
      - The TD loss is calculated and optimized.
   e. Periodically, the target network is updated with the current network's weights.
4. **Evaluation:** After training, the agent's performance is evaluated by running it in a test environment and recording a video of its gameplay.

Q Network Architecture

   a. Convolutional Layers:
      - Two convolutional layers with ReLU activations.
      - The first layer has 16 filters with kernel size 8, stride 4, and padding 2.
      - The second layer has 32 filters with kernel size 4, stride 2, and padding 1.
   b. Fully Connected Layers:
      - The output of the convolutional layers is flattened.
      - Two fully connected layers with ReLU activation in between.
      - The final output layer has a size equal to the number of actions in the environment.

Hyperparameters

1. seed: 42
   Random seed for reproducibility

2. replay-buffer-size: 5000
   Maximum number of experiences stored in the replay buffer

3. learning-rate: 1e-4
   Learning rate for the Adam optimizer

4. discount-factor: 0.99
   Discount factor (gamma) for future rewards

5. num-steps: 1000000
   Total number of steps to run the environment for during training

6. batch-size: 256
   Number of experiences sampled from the replay buffer for each learning update

7. learning-starts: 256
   Number of steps before learning begins

8. learning-freq: 5
   Number of steps between each learning update

9. use-double-dqn:
   True whether to use double DQN algorithm

10. target-update-freq: 1000
    Number of steps between each target network update

11. eps-start: 1.0
    Starting value for epsilon in the epsilon-greedy policy

12. eps-end: 0.01
    Final value for epsilon in the epsilon-greedy policy

13. eps-fraction: 0.1
    Fraction of total steps over which epsilon is annealed

14. print-freq: 10
    Frequency of printing training progress