Describing training in image

Training Loop:

The process begins with the Training Loop, which iterates through several steps:

a. Choose random/policy action: The agent either explores (random action) or exploits (policy action) based on its current policy.

b. Sample environment: The chosen action is applied to the environment, generating a new state and reward.

c. Record memory update: The experience (state, action, reward, next state) is stored in the Replay Memory.

d. Optimize: The agent learns from past experiences.

e. Occasional target net update: The Target Network is periodically updated to match the Policy Network.

Replay Memory:

This is a buffer that stores past experiences (state, action, reward, next state tuples).

Sampling from replay memory makes the actions i.i.d.

Policy Net:

This is the main neural network that represents the agent's current policy.

It takes the current state as input and outputs Q-values for each possible action.

The action with the highest Q-value is chosen (with some exploration).

Target Net:

This is a separate neural network with frozen weights.

This network is periodically updated to match the policy network. This process stabilizes training and enables convergence.

Optimization Process:

A random batch of experiences is sampled from the Replay Memory.

These experiences are used to compute the loss between the predicted Q-values (from Policy Net) and the target Q-values (from Target Net).

The Policy Net is then updated with gradient descent.

Network Architecture:

Convolutional layers:

conv1: 32 filters, 8x8 kernel, stride 4

conv2: 64 filters, 4x4 kernel, stride 2

conv3: 64 filters, 3x3 kernel, stride 1

Fully connected layers:

linear1: 64 * 7 * 7 input features to 512 neurons

linear2: 512 to number of actions (6 actions)

Optimizer and Loss:

Uses Adam optimizer with the provided learning rate.

Uses Mean Squared Error (MSE) loss function.

Forward Pass (forward):

Applies the convolutional layers with ReLU activations.

Flattens the output of convolutional layers.

Applies fully connected layers with ReLU activation on the first one.

Returns the final output (Q-values for each action ~ 6 actions).

List of hyperparameters:

Name: seed
Value: 42
Description: The random seed used for reproducibility of results.
Name: env
Value: "PongNoFrameskip-v4"
Description: The name of the Atari game environment used for training.

Name: replay-buffer-size
Value: 5000
Description: The size of the replay buffer, which stores past experiences for training.
Name: learning-rate
Value: 0.0001 (1e-4)
Description: The learning rate used by the Adam optimizer for updating the neural network weights.
Name: discount-factor
Value: 0.99
Description: The discount factor (gamma) used in the Q-learning algorithm to balance immediate and future rewards.
Name: num-steps
Value: 1,000,000 (1e6)
Description: The total number of steps to run the environment for during training.
Name: batch-size
Value: 256
Description: The number of transitions sampled from the replay buffer for each optimization step.
Name: learning-starts
Value: 10000
Description: The number of steps to take before starting the learning process, allowing the replay buffer to fill with some experiences.
Name: learning-freq
Value: 5
Description: The number of iterations between every optimization step.
Name: use-double-dqn
Value: False
Description: Whether to use double deep Q-learning, an improvement over standard DQN to reduce overestimation of Q-values.
Name: target-update-freq
Value: 1000
Description: The number of iterations between every update of the target network, used to stabilize learning.
Name: eps-start
Value: 1.0
Description: The starting value for epsilon in the epsilon-greedy exploration strategy.
Name: eps-end
Value: 0.01
Description: The final value for epsilon in the epsilon-greedy exploration strategy.
Name: eps-fraction
Value: 0.1

Description: The fraction of total steps over which epsilon is annealed from eps-start to eps-end.
Name: print-freq
Value: 10
Description: The frequency (in episodes) at which to print training progress information.