1. During training, why is it necessary to act according to an ε-greedy policy instead of a greedy policy (withrespect to Q)?

A greedy policy just chooses the action based on short term reward gains, but the ε-greedy policy chooses best action with probability 1 – ε and a random action with probability ε. The action can either be random or an action based on the stored actions that maximise the reward. This can be paired with SoftMax policy to select the random action with probabilities proportional to their current value. It is certainly better than a normal greedy policy.

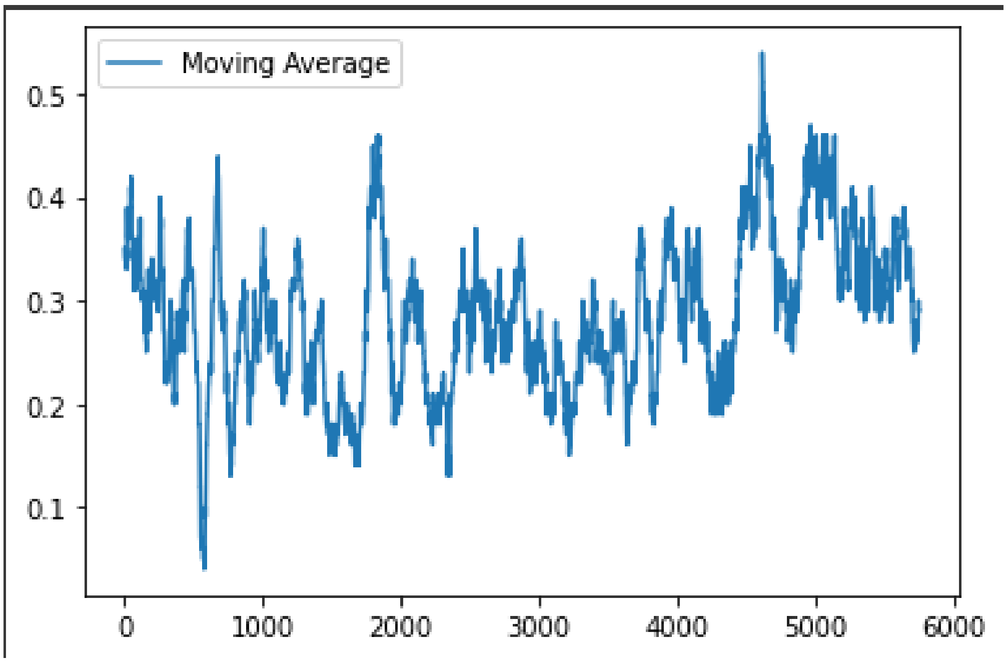
2. How do the authors of the paper [Mnih et al., 2015] explain the need for a target Q-network in addition to an online Q-network?

The modification that is used a technique that is called experience replay where the agent's experience at each time skip is stored. It is more efficient as it is used in many weight updates. It averages the behaviour distribution of many previous states. The second is that a neural network generates targets to offset the oscillations and divergence.

3. Explain why the one-step return for each state in a batch is computed incorrectly by the baseline implementation and compare it to the correct implementation.

The updated implementation uses the cumulative sum of the rewards whereas the baseline implementation uses the immediate reward. This can lead to high bias in the data.

4. Plot a moving average of the returns that you stored in the list episode reward history.



5. Several OpenAI Gym wrappers were employed to transform the original Atari Breakout environment. Briefly explain the role of each of the following wrappers: MaxAndSkipEnv, EpisodicLifeEnv, WarpFrame, ScaledFloatFrame, ClipRewardEnv, and FrameStack.

MaxAndSkipEnv – Skips intermediate frames where to speed up training of the neural network. In the case of our game, at every 4th frame.

EpisodicLifeEnv – Helps in estimating future policies by modifying the environment. Reset on true game over and not the episodic game over.

WarpFrame – aids in feature extraction by converting frames to greyscale and warp frames to 84x84 as is done in the nature publication.

ScaledFloatFrame – Normalizes observation and scales down the resolution.

ClipRewardEnv – Clips the reward to {-1, 0, 1} depending on its sign.

FrameStack – This helps in stacking the frames to monitor the motion of objects which can improve the reaction of agents according to future states.