Landing on the moon with Asynchronous Advantage Actor Critic
Lunar Lander v2

Naveen Prashanth G

Department of Electronics Engineering
Anna University (MIT Campus)
Chennai, India

1 Introduction

In this project, we have chose to try to solve the lunar lander[1] problem from OpenAI gym. To do so, I have tried two methods, the first one, by adapting code found[2] for an other problem, which use Deep Convolutional Q-learning. For the second one I tried to implement A3C algorithm by myself with some help.

2 A3C algorithm

We have tried to implement the Asynchronous Advantage Actor-Critic introduced by Google DeepMind Group. This algorithm is a evolution of Deep Convolutonal Q-learning algorithm. It uses multiple asynchronous agents that interact with the environment to get multiple experience that are independent from each other. It also combine the main advantages of policy gradient and Q-learning in a single algorithm. I wanted to implement this method because it is one of the state of art one in reinforcement learning. I chose to implement this algorithlm as it doesn't require complex multi-threading.

2.1 Eligibility trace

The eligibilty trace consist of taking in account more of the pasts reward than the normal Q-learning. In traditional Q-learning, one compute the temporal difference as $TD = r_{k+1} + \gamma max\hat{Q}(x_{k+1}, u)\hat{Q}(x_k, u_k)$. In the case of Eligibility Trace, one use

$$TD^{(n)} = r_{k+1} + \gamma r_{k+1} + \gamma^2 r_{k+2} + ... + \gamma^{n-1} r_{k+n-1} + \gamma^n max\hat{Q}(x_{k+1}, u)\hat{Q}(x_k, u_k)$$

This allow to take more in account what happend in the past. Thus, the neural network uses the predicted Q values by it output and the computed one to compute the mean squared error loss to update its weights.

2.2 Experience replay

For each action taken, the algorithm push to a buffer, this allows to have the benefits of experience replay, by sampling this buffer in order to train the neural network.

[1] `https://gym.openai.com/envs/LunarLander-v2/` [2] `https://www.superdatascience.com/artificial-intelligence/`

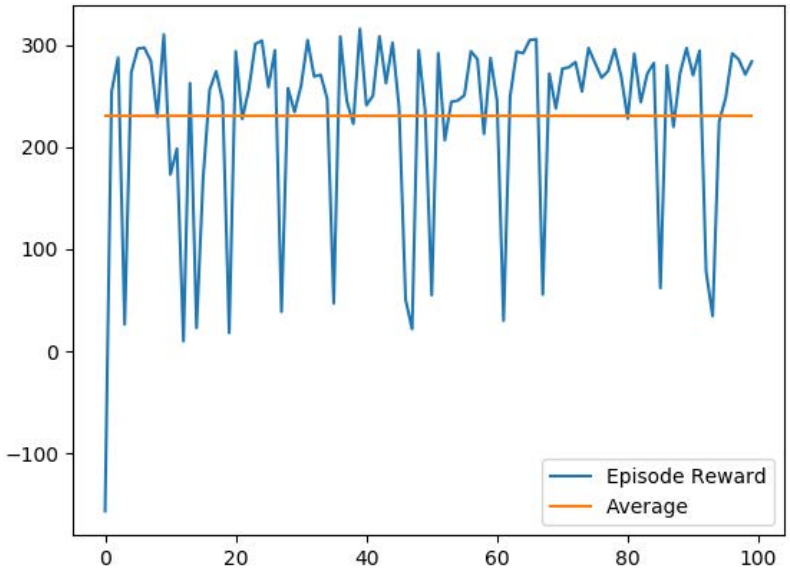

Figure 1

2.3 Performances

After a few adjustments in code, the average reward is touching peaks, and it seems that it has understood that it has to land between the flags, but it seems that it takes longer to understand how to manoeuvring to land there.

As we can see at **figure 1**, the performance is quite noisy, but still respects the conditions for it to be considered solved. The noisy solution may be due to early stopping, but it seems that the reward structure does not translate itself into landings that would be considered safe. For example, landing with just a leg inside the landing zone seems to be as good as landing with both legs inside. Landing in very few time steps is also considered better, encouraging maneuvers performed with high acceleration.

2.4 Installation

In order to run the code, it is needed to install first openAI gym lunar lander v2 environement, then tensorflow is needed for the A3C part.