CMPE 58Y - Robot Learning
Homework 3: Policy Gradient with Function Approximation

April 7, 2022

1 Introduction

In this homework you will implement policy gradient algorithm with a neural network for the cart pole task [1] in OpenAI Gym environment. As in previous homework, do not care about `done` variable. Terminate the episode after 500 iterations. You can consider the task is solved if you consistently get +450 reward.

2 Policy Gradient

As it is explained in the lecture, your RL agent can be a neural network. Since the environment is not complex, in this homework you will use a single layer with at most 4 neurons. (Our implementation has a single neuron and can solve the task approximately in 50 episodes where there are 50 roll-outs in each episode. Considering the state space, there are 4 weights and 1 bias for the neuron. Activation function is a sigmoid. Discount factor is 0.99, learning rate is 0.05. Average reward of the roll-outs is used as baseline. Remember to check the course website for the explanation of causality principle.)

In the lecture, we used a Gaussian distribution for the probability distribution of actions, but there are 2 discrete actions in cart pole task. Therefore, it is not reasonable to use a Gaussian distribution. Instead, a Bernoulli distribution will be used and the output of the network will be the probability(p) of pushing the cart either to the left or to the right. (The other probability is 1-p naturally)

Remember that:

$$\bigtriangledown_\theta J(\theta) = E_{\pi_\theta}[(\sum_{t=1}^{T} \bigtriangledown_\theta \log \pi_\theta(a_t|s_t)(\sum_{k=t}^{T} \gamma^{k-t} R(s_k, a_k)))] \tag{1}$$

Here (X is the random variable of Bernoulli distribution):

$$\pi_\theta(a_t|s_t) = p^x(1-p)^{1-x} \tag{2}$$

Where:

$$p = P_\theta(a_t) = sigmoid(\theta s + b) \tag{3}$$

So:

$$\log \pi_\theta(a_t|s_t) = [(x \log(p)) + ((1-x) \log(1-p))] \tag{4}$$

$$\bigtriangledown_\theta \log \pi_\theta(a_t|s_t) = x\frac{\bigtriangledown_\theta p}{p} + (1-x)\frac{\bigtriangledown_\theta (1-p)}{(1-p)} \tag{5}$$

Because of the property of sigmoid:

$$\bigtriangledown_\theta p = \bigtriangledown_\theta P_\theta(a_t) = p(1-p)s \tag{6}$$

Then;

$$\bigtriangledown_\theta \log \pi_\theta(a_t|s_t) = x(1-p)s + (1-x)(-p)s \tag{7}$$

If we evaluate the expectation with sampling (N is the rollout size):

$$\bigtriangledown_\theta J(\theta) = \sum_{i=1}^{N}[(\sum_{t=1}^{T} \bigtriangledown_\theta \log \pi_\theta(a_{i,t}|s_{i,t})(\sum_{k=t}^{T} \gamma^{k-t} R(s_{i,k}, a_{i,k})))] \tag{8}$$

To add baseline, let's say $G_{i,t}$ is the discounted reward received after the time 't' for the trajectory 'i':

$$G_{i,t} = (\sum_{k=t}^{T} \gamma^{k-t} R(s_{i,k}, a_{i,k})) \tag{9}$$

Here I denote the baseline with $b_{avg,t}$ the average reward received after the time 't' for all trajectories:

$$b_{avg,t} = \frac{\sum_{i=1}^{N} G_{i,t}}{N} \tag{10}$$

The gradient becomes:

$$\bigtriangledown_\theta J(\theta) = \frac{1}{N}\sum_{i=1}^{N}[(\sum_{t=1}^{T} \bigtriangledown_\theta \log \pi_\theta(a_{i,t}|s_{i,t})(G_{i,t} - b_{avg,t}))] \tag{11}$$

After you correctly calculate these gradients, you can update your parameters using stochastic gradient descent as in the second homework.

3 Deliverables

Plot the reward over episodes. Submit your work as a single zip file that contains a single jupyter notebook through moodle. For any questions regarding send an e-mail `muhammet.hatipoglu@boun.edu.tr`. There is a higher chance of getting a fast response if the title of your mail begins with "[CMPE58Y]". Cheating will be penalized by -200 points.

Deadline: Friday, April 22nd, 11:59 P.M. (You will be graded out of 100)
Late deadline: Monday, April 24th, 11:59 P.M. (You will be graded out of 80)

References

[1] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.