CS6700 - Reinforcement Learning
Programming Assignment 2

March 13, 2022

1 Environments

This exercise aims to familiarize you with **DQN and Actor-Critic** methods. You will implement your algorithms on the following four OpenAI Gym environments.

- `Acrobot-v1`: The acrobot system includes two joints and two links, where the joint between the two links is actuated. Initially, the links are hanging downwards, and the goal is to swing the end of the lower link up to a given height.
- `CartPole-v1`: A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The system is controlled by applying a force of +1 or -1 to the cart. The pendulum starts upright, and the goal is to prevent it from falling over. A reward of +1 is provided for every timestep that the pole remains upright. The episode ends when the pole is more than 15 degrees from vertical, or the cart moves more than 2.4 units from the center.
- `MountainCar-v0`: A car is on a one-dimensional track, positioned between two "mountains". The goal is to drive up the mountain on the right; however, the car's engine is not strong enough to scale the mountain in a single pass. Therefore, the only way to succeed is to drive back and forth to build up momentum.

All the above three environments allow only discrete actions.

2 Tasks

DQN

For **DQN**, perform the following (for each environment):

- Start with the set of hyperparameters originally given in tutorial 4 and try solving the environment.
- To account for stochasticity, use the average of 10 runs (at least) for each variation.
- With the goal of improving the agent's performance (choose a metric - could be number of episodes to taken to solve the environment). Try changing a few hyperparameters. List down the new set of parameters and provide reasons for the choice made.
- For each variation of the agent, plot reward curves and print the number of steps to reach the goal in each episode. Use this to comment on the performance of the agent.

- Repeat this till you improve your agent 5 times [OR] try 12 different configurations (at least).
- Make inferences and provide conjectures from all your experiments and results.

NOTE: You are graded equally for the experiments, justifications and explanations provided. Some parameters you could play around with are - neural network architecture, learning rate, replay buffer size, batch size, update frequency of target network, truncation limit, discount factor and control parameter.

Actor-Critic

For **Actor-Critic**, experiment with the following aspects (for each environment) to improve the performance, by starting with the network provided in the tutorial:

- Variations of Actor-Critic methods [where $v(s)$ is a parametrized state value function]:
  1. One-step, where $\delta_t^{(1)} = R_{t+1} + \gamma v(s_{t+1}) - v(s_t)$
  2. Full returns, where $\delta_t^{(T)} = \sum_{t'=t}^{T} \gamma^{t'-t} R_{t'+1} - v(s_t)$; T being the maximum number of time steps considered.
  3. n-step returns, where $\delta_t^{(n)} = \sum_{t'=t}^{n+t-1} \gamma^{t'-t} R_{t'+1} + \gamma^n v(s_{n+t}) - v(s_t)$; you can implement for two values of n (chosen appropriately).
- Number and sizes of hidden layers
- Learning rate α
- Number of episodes

You can fix the discount factor $\gamma = 0.99$ for all Actor-Critic experiments. Use the average of at least 10 runs for each variation. Also, note the variances of total episode rewards across the 10 runs for each episode. Similar to DQN, plot reward curves and print the number of steps to reach the goal in each episode for each variation. Observe and comment on the differences in the performance of each of the three AC variations in terms of the number of steps to reach the goal and variance. Which AC method has the highest variance (you can determine this by plotting the variance of the episode reward for each episode)?

For the full returns AC scenario, the actor and critic loss functions are

$$L_{actor} = -\sum_{t=1}^{T} \log \pi_\theta(a_t|s_t) \delta_t^{(T)}$$

$$L_{critic} = \sum_{t=1}^{T} \delta_t^{(T)^2}$$

And for the n-step returns scenario, the actor and the critic loss functions are

$$L_{actor}^{(t)} = -\log \pi_\theta(a_t|s_t) \delta_t^{(n)}$$

$$L_{critic}^{(t)} = \delta_t^{(n)^2}$$

where $\delta_t^{(n)} = \sum_{t'=t}^{n+t-1} \gamma^{t'-t} R_{t'+1} + \gamma^n v(s_{n+t}) - v(s_t)$

3 Submission Instructions

You are required to submit both your report and your code on Gradescope. Please submit in **teams of two**. One submission per team will suffice. The due date for this programming assignment is **11:59 pm on Sunday, March 27th**.