CSE 4/546: Reinforcement Learning

Spring 2022.

Assignment 3 - Actor-Critic

Submission: May 1, Sun, 11:59pm.

Team 8:

Affan Ali Hasan Khan - 50432243

Hrishikesh Agrawal - 50428219

Q1) Discuss the algorithm you implemented.

➔ Actor-critic algorithms maintain two sets of parameters:

- Critic Updates action-value function parameters w
- Actor Updates policy parameters θ, in direction suggested by critic

➔ Actor-critic algorithms follow an approximate policy gradient:

$$\nabla_\theta J(\theta) \approx E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)]$$
$$\Delta\theta = \alpha \nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)$$

➔ The policy gradient has many equivalent forms:

$$\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) G_t] && \text{REINFORCE} \\
&= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)] && \text{Q Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) A_w(s,a)] && \text{Advantage Actor-Critic (A2C)} \\
&= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) \delta] && \text{TD Actor-Critic}
\end{aligned}$$

➔ The algorithm used for the assignment is 'Advantage Actor Crtitic' (A2C).

➔ The advantage function can significantly reduce variance of policy gradient.

➔ So the critic should really estimate the advantage function. For example, by estimating both Vπθ (s) and Qπθ (s, a).

➔ Using two function approximators and two parameter vectors,

$$V_v(s) \approx V_{\pi_\theta}(s)$$
$$Q_w(s,a) \approx Q_{\pi_\theta}(s,a)$$
$$A(s,a) = Q_w(s,a) - V_v(s)$$

➔ And updating both value functions by e.g. TD learning.

Q2) What is the main difference between the actor-critic and value based approximation algorithms?

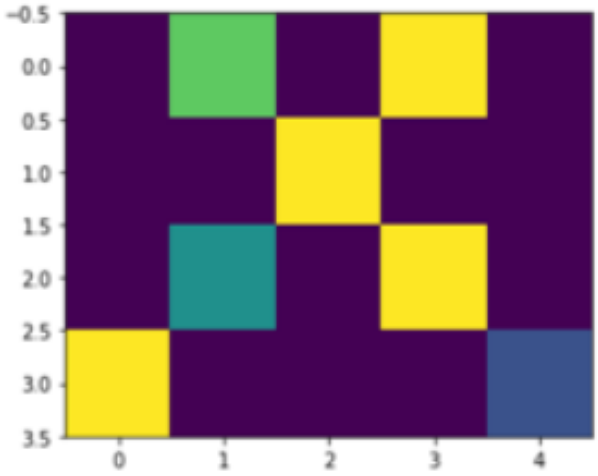
➔ **Value-based:** estimate value function or Q-function of the current policy (no explicit policy).

➔ **Actor-critic:** estimate value function or Q-function of the current policy, use it to improve the policy.

Q3) Briefly describe THREE environments that you used (e.g. possible actions, states, agent, goal, rewards, etc). You can reuse related parts from your Assignment 2 report.

(i) Grid environment:

a. Possible actions: Up, Down, right, left
b. Number of states: 20
c. Agent: the agent is moving in the environment trying to maximize the reward and to reach the goal. The agent starts from the start position [0,0].
d. Goal: The goal of the agent is to reach the goal position which is at [4,5].
e. Rewards: There are a total of 4 rewards:
(-5,0,+1,+10) the rewards corresponds to the states as shown.



(ii) Lunar Lander:

(a) Observations/states: The states/observations are represented in a form of vector having 8 elements.

a. X distance from target site
b. Y distance from target site
c. X velocity
d. Y velocity
e. Angle of ship
f. Angular velocity of ship
g. Left leg is grounded
h. Right leg is grounded

(b) Action Space: There are only 4 discrete actions: thrust, left, right, nothing.

(c) Goal: The goal of the agent is to successfully land on the landing pad. An episode score of 200 or more is considered a solution.

(d) Episode end/Done: The episode terminates after a successful land, crash, or 1000 steps.

(e) Rewards: After each step a reward is granted. The total reward of an episode is the sum of the rewards for all steps within that episode. The reward for moving from the top of the screen to landing pad with zero speed is awarded between 100 and 140 points. If the lander moves away from the landing pad it loses the same reward as moving the same distance towards the pad. The episode receives additional -100 or +100 points for crashing or landing, respectively. Grounding a leg is worth 10 points and thrusting the main engine receives -0.3 points.

(iii) OpenAI Ant:

(a) Observations/states: Observations will be an array of 111 numbers. The 111-dim observation space represents:

- z (height) of the Torso -> 1
- orientation (quarternion x,y,z,w) of the Torso -> 4
- 8 Joiint angles -> 8
- 3-dim directional velocity and 3-dim angular velocity -> 3+3=6
- 8 Joint velocity -> 8
- External forces (force x,y,z + torque x,y,z) applied to the CoM of each link (Ant has 14 links: ground+torso+12(3links for 4legs) for legs -> (3+3)*(14)=84
- **=> 1+4+8+6+8+84 = 111**

(b) Actions: Actions will be an array of 8 numbers. The 8-dim action space represents: Torque (value for -1 to +1) of each joint.

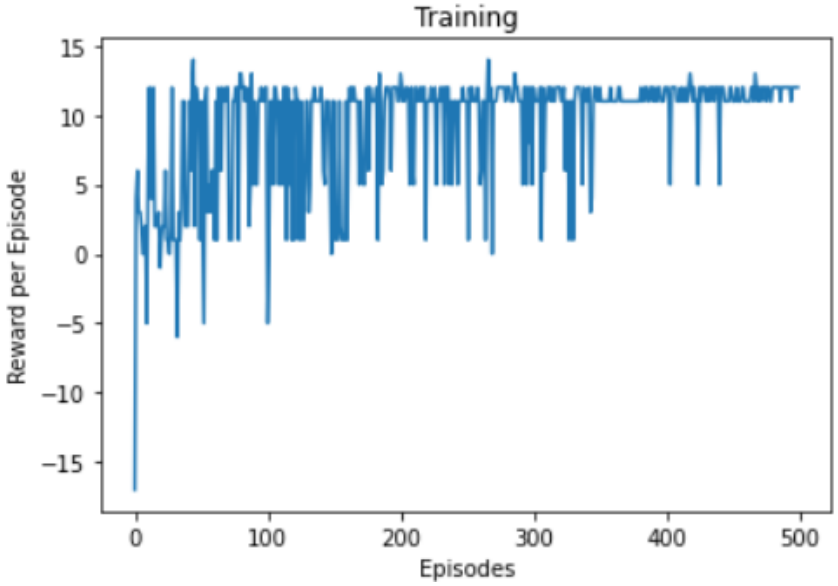
(c) Reward: Make a four-legged creature walk forward as fast as possible.

(d) Goal: To make the ant run as fast as possible and for as long as possible to maximize the rewards.

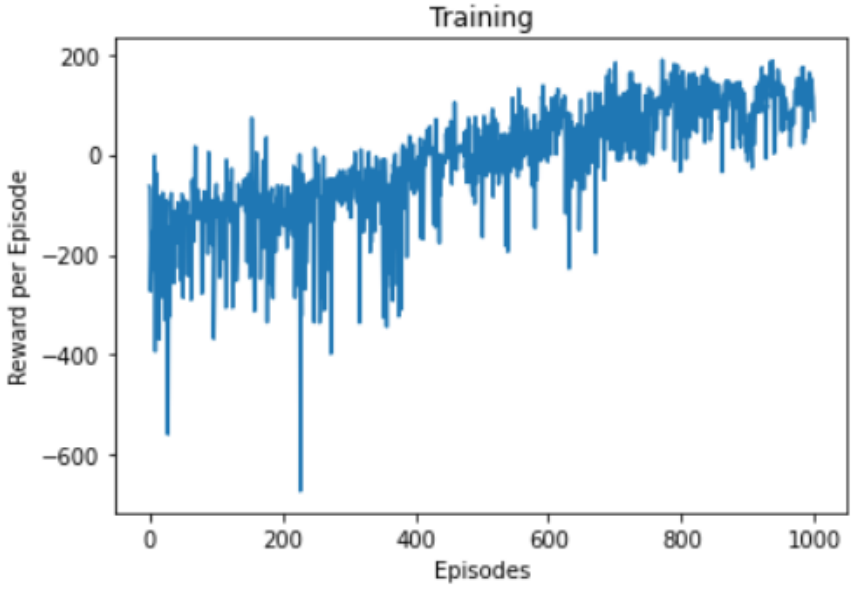
(e) Episode end: The episode ends once the ant falls and is unable to move.

Q4) Show and discuss your results after training your Actor-Critic agent on each environment. Plots should include the reward per episode for THREE environments. Compare how the same algorithm behaves on different environments while training.

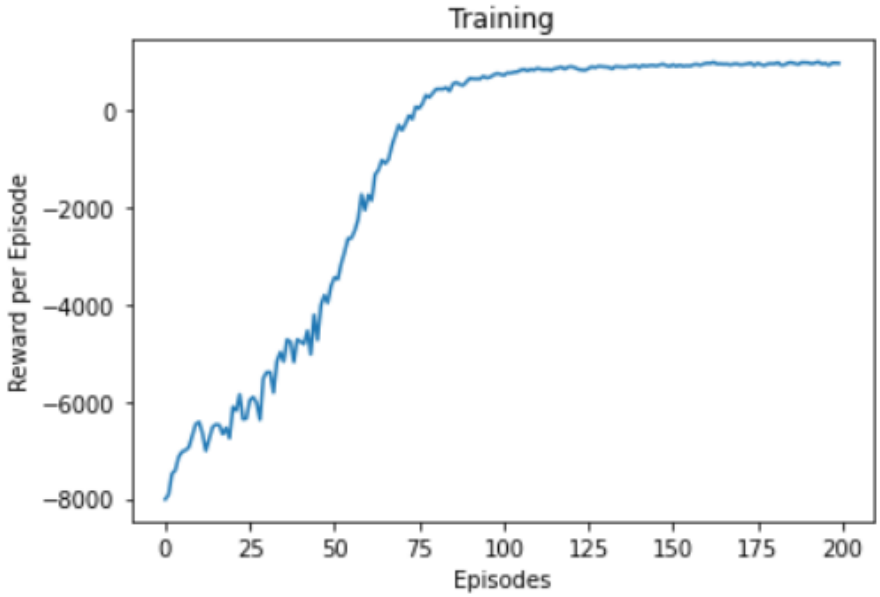
(i) Grid Environment:



(ii) Lunar Lander:



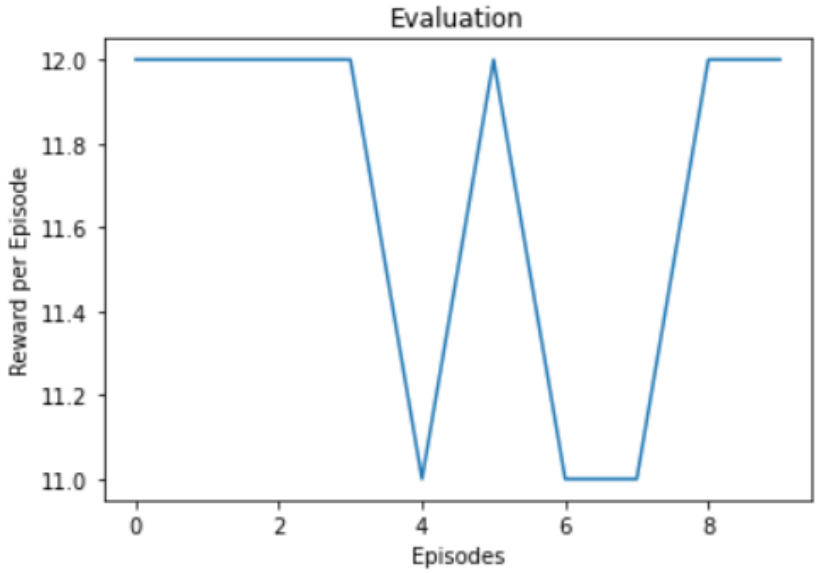
(iii) OpenAI Ant:



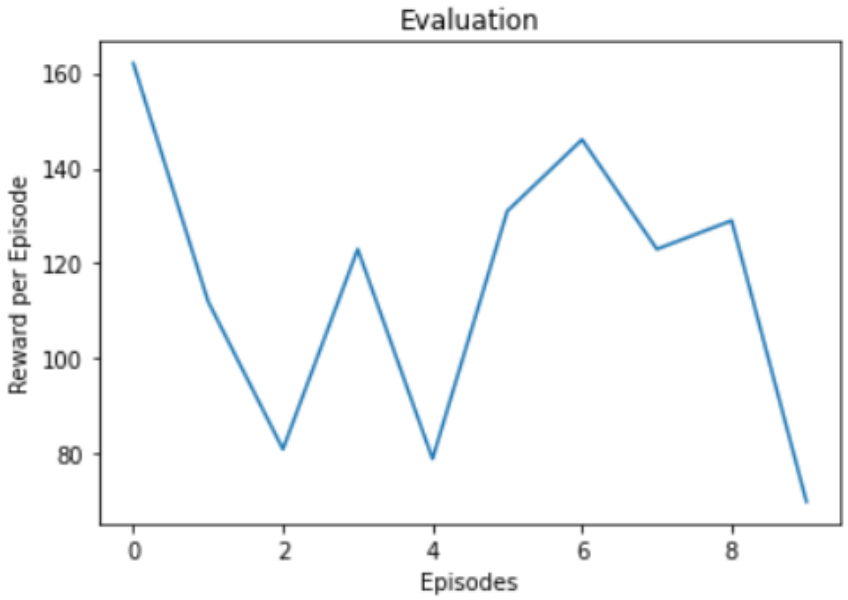
➔ As we can see, from the training graphs that all three environments after being trained are producing satisfactory results and the rewards in all the environment is converging to max cumulative rewards in an episode.

Q5) Provide the evaluation results for each environments that you used. Run your environments for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.

(i) Grid environment:



(ii) Lunar Lander:



(iii) OpenAI Ant: