Reinforcement Learning
Assignment 3(Final Report)
CSE 546

Team members-
Sannihith Kilaru (Ubit name-sannihit)
Sabariis Venugopal Sankaranaryanan (Ubit name-sabariis)

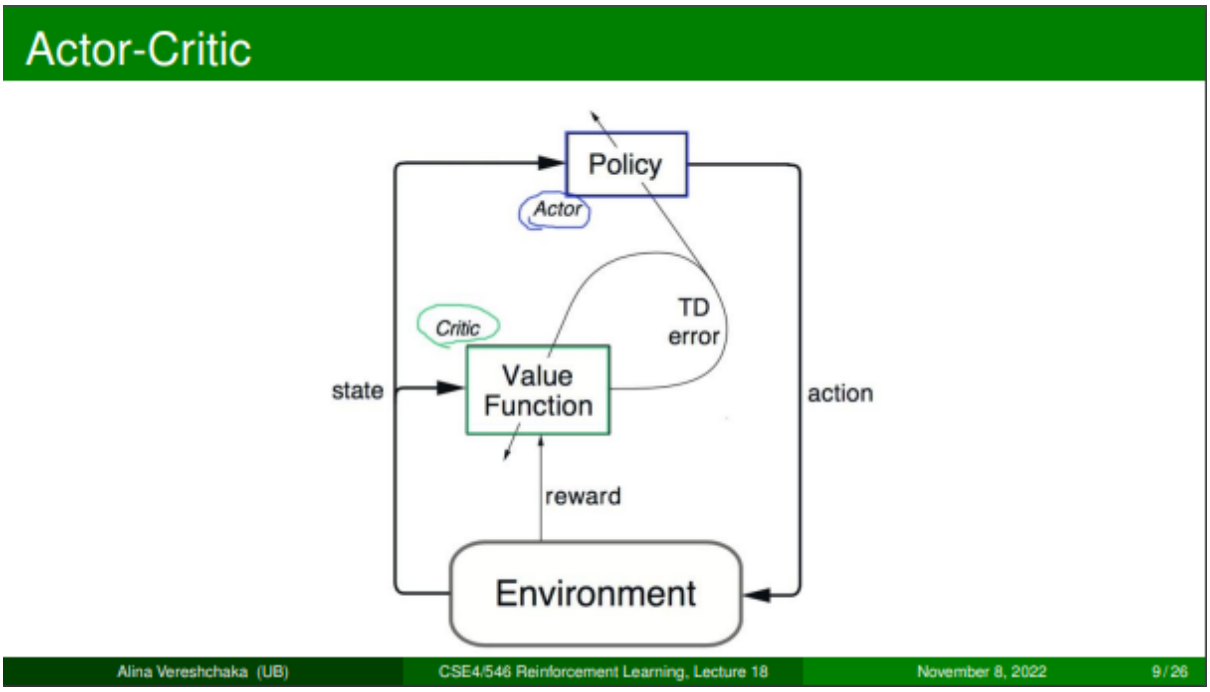
1. Discuss the algorithm you implemented.

Actor-Critic algorithms

- Actor-critic algorithms maintain two sets of parameters:
  - **Actor:** Updates policy parameters theta in direction suggested by critic.
  - **Critic:** Updates action-values function parameters w
- They follow an approximate policy gradient, we use the TD error to compute the policy gradient.

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

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)\delta_{\pi_\theta}]$$

- 

A2C

- For the true value function Vπθ (s), the TD error δπθ is

$$\delta_{\pi_\theta} = r + \gamma V_{\pi_\theta}(s') - V_{\pi_\theta}(s)$$

- Estimate of the advantage function (predicted advantage) is

$$\begin{aligned}\mathbb{E}_{\pi_\theta}[\delta_{\pi_\theta}|s,a] &= \mathbb{E}_{\pi_\theta}\left[r + \gamma V_{\pi_\theta}(s')|s,a\right] - V_{\pi_\theta}(s) \\ &= Q_{\pi_\theta}(s,a) - V_{\pi_\theta}(s) \\ &= A_{\pi_\theta}(s,a)\end{aligned}$$

- 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 and updating both value functions

$$V_v(s) \approx V_{\pi_\theta}(s)$$
$$Q_w(s,a) \approx Q_{\pi_\theta}(s,a)$$
$$A(s,a) = Q_w(s,a) - V_v(s)$$
$$\mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) A_w(s,a)] \qquad \text{Advantage Actor-Critic (A2C)}$$

PPO

PPO is an actor-critic algorithm. We created 2 different networks for both the actor and the critic to remove the added complication of entropy. We also implemented a replay buffer to store experience which would be used for updating the networks. The main difference between PPO and the other AC algorithms is the actor's loss function which is given by:

$$L^{CLIP}(\theta) = \hat{E}_t[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)\,]$$

- θ is the policy parameter
- $\hat{E}_t$ denotes the empirical expectation over timesteps
- r_t is the ratio of the probability under the new and old policies, respectively
- $\hat{A}_t$is the estimated advantage at time t
- ε is a hyperparameter, usually 0.1 or 0.2

The clipping helps in reducing the instability caused by AC algorithms.

2. What is the main difference between the actor-critic and value-based approximation algorithms?

In value based approximation algorithms, we get the Q-values for a state using a neural net and the action is chosen using a epsilon-greedy policy. In actor critic algorithms the Q values are learnt by the Critic neural network and the action chosen is not based on an epsilon greedy policy, it has another neural network, the Actor neural network, which learns the optimal policy.

3. 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.

Grid Environment

The main objective of the agent in the deterministic environment that is defined for the 4X4 grid is to reach the goal state which is at (3,3) with an equal probability of an agent moving to another state. The objective of the stochastic environment is to perform well and get the reward. The neighborhood of a node consists of 16 states represented as 4x4 square. In the square the agent gets the rewards of [+1, +3, +5 and +10]. The rewards are perceived at a square if the agent is at that square where the rewards are awarded. The sequence of actions causes the environment to go through a sequence of states, if the sequence is desirable the agent is performed well.

An agent can do the following actions:

- An agent can take one step at a time.
- The agent can do the following actions- {move(north), move(south), move(east), move(west).

The start square is (0,0) and the goal state is (3,3) and +10 points are awarded when the agent reaches the goal state.

The number of episodes considered for this environment are 1000

Cartpole-

The CartPole-v1 environment from OpenAI is characterized as follows: an unactuated joint connects a pole to a cart that drives along a frictionless track. Controlling the mechanism is as simple as delivering a force of +1 or -1 to the cart. The goal is to keep the pendulum upright and from falling over. When the pole is more than 15 degrees from vertical or the cart is more than 2.4 units from the center, the episode terminates. Every timestep that the pole remains erect results in a +1 reward. CartPole-v1 defines "solving" as receiving an average reward of 500 over 500 episodes. There are two actions for this environment. 0 pushes the cart to the left and 1 pushes the cart to the right. We have four observations which are cart position, cart velocity, pole angle and pole angular velocity. The number of episodes considered for this environment are 8000.

LunarLander-

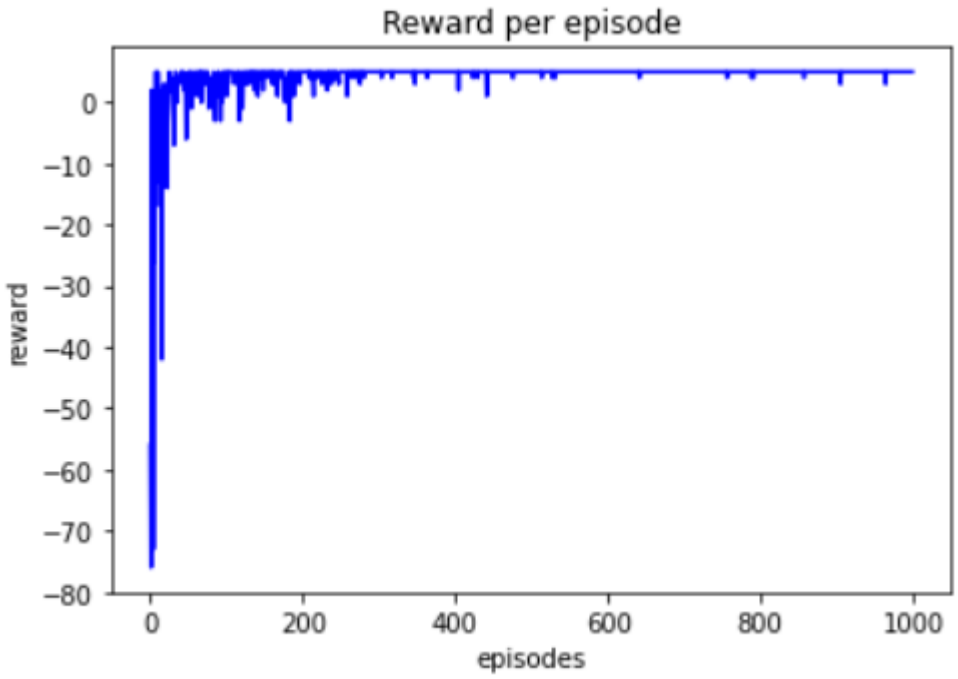
The second complex environment that we chose to design is the Lunar Lander with discrete actions. This environment is a classic rocket trajectory optimization problem. According to Pontryagin's maximum principle, it is optimal to fire the engine at full throttle or turn it off. This is the reason why this environment has discrete actions: engine on or off.

There are two environment versions: discrete or continuous. The landing pad is always at coordinates (0,0). The coordinates are the first two numbers in the state vector. Landing outside of the landing pad is possible. Fuel is infinite, so an agent can learn to fly and then land on its first attempt. There are four discrete actions available: do nothing, fire left orientation engine, fire main engine, fire right orientation engine. The state is an 8-dimensional vector: the coordinates of the lander in x & y, its linear velocities in x & y, its angle, its angular velocity, and two booleans that represent whether each leg is in contact with the ground or not. Reward for moving from the top of the screen to the landing pad and coming to rest is about 100-140 points. If the lander moves away from the landing pad, it loses reward. If the lander crashes, it receives an additional -100 points. If it comes to rest, it receives an additional +100 points. Each leg with ground contact is +10 points. Firing the main engine is -0.3 points each frame. Firing the side engine is -0.03 points each frame. Solved is 200 points.

4. 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.

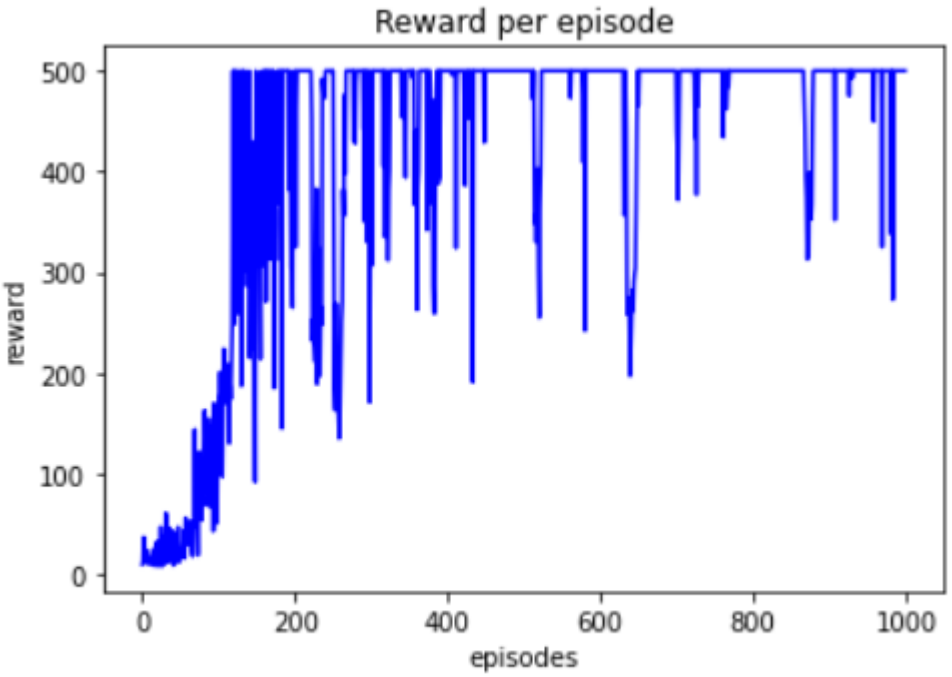
A2C

Grid World



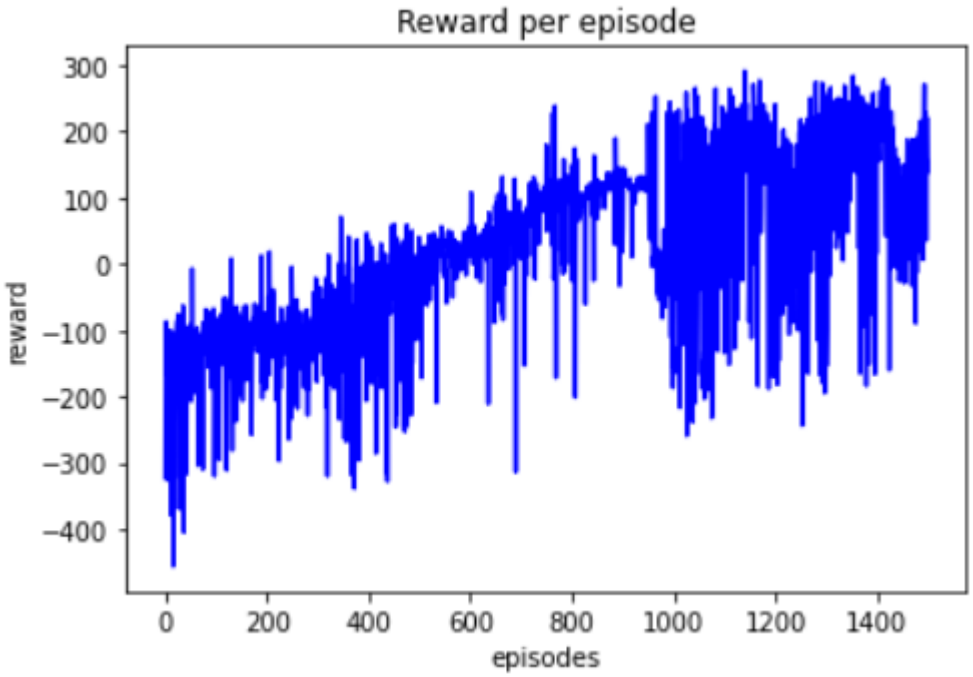
Complete

CartPole



Complete

LunarLander

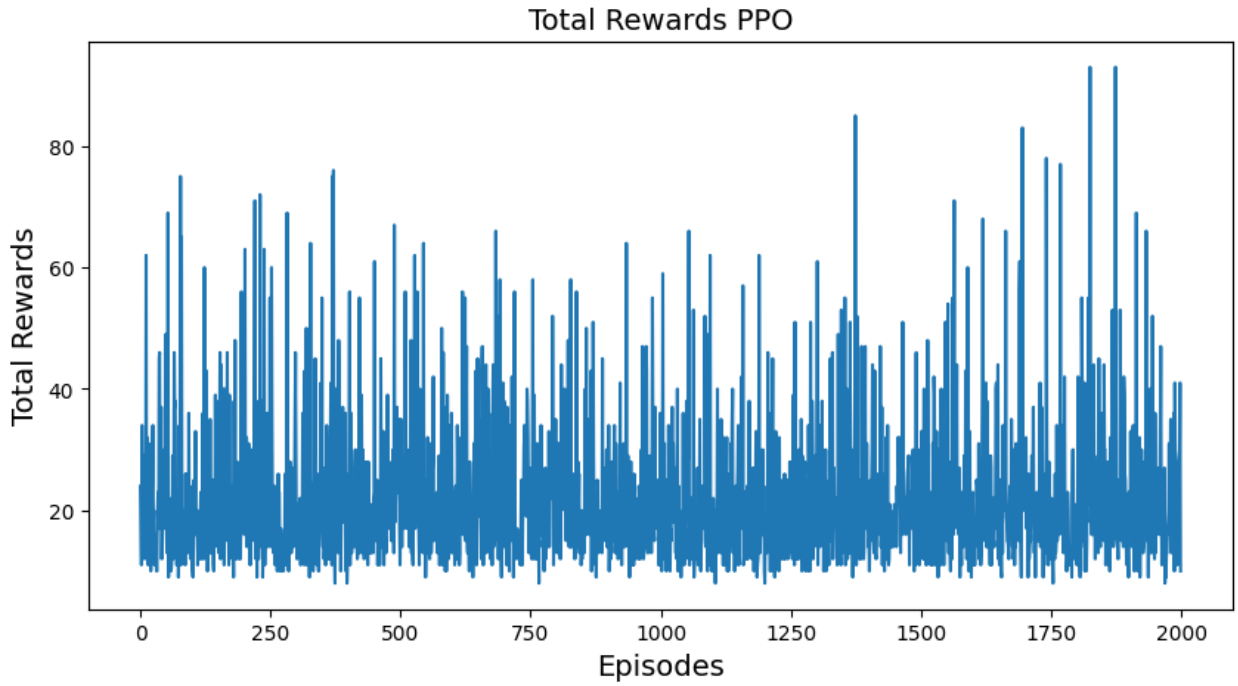


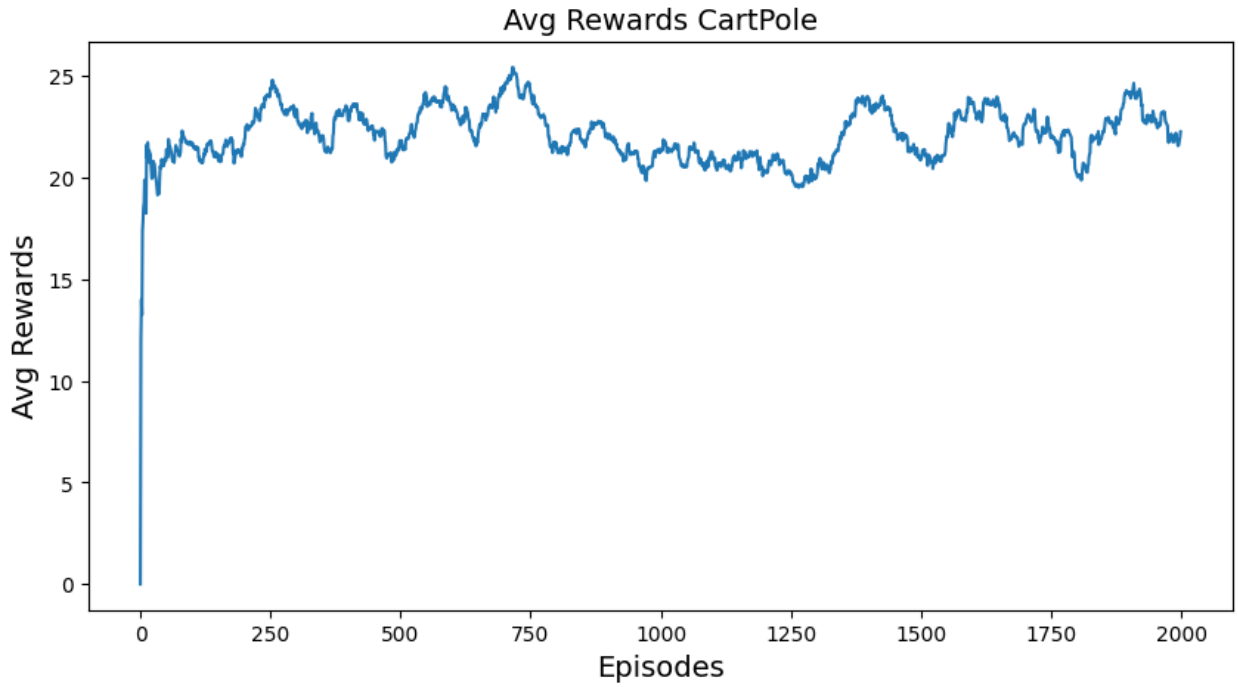
From the above graphs we can infer that the oscillations in rewards per episode increases and the number of episodes needed to converge increases as the complexity of the environment increases.

This analysis is accurate because the neural network used for all three environments is the same as well as the hyperparameters. The only change is the number of episodes. Complexity: Grid world < Cart Pole < Lunar Lander. To observe if my agent was training I printed the mean reward while training. In all three environments the agent has mean rewards greater than the expected mean rewards by OpenAI. Therefore all three environments have been successfully trained for the A2C algorithm.

PPO

Results For CartPole





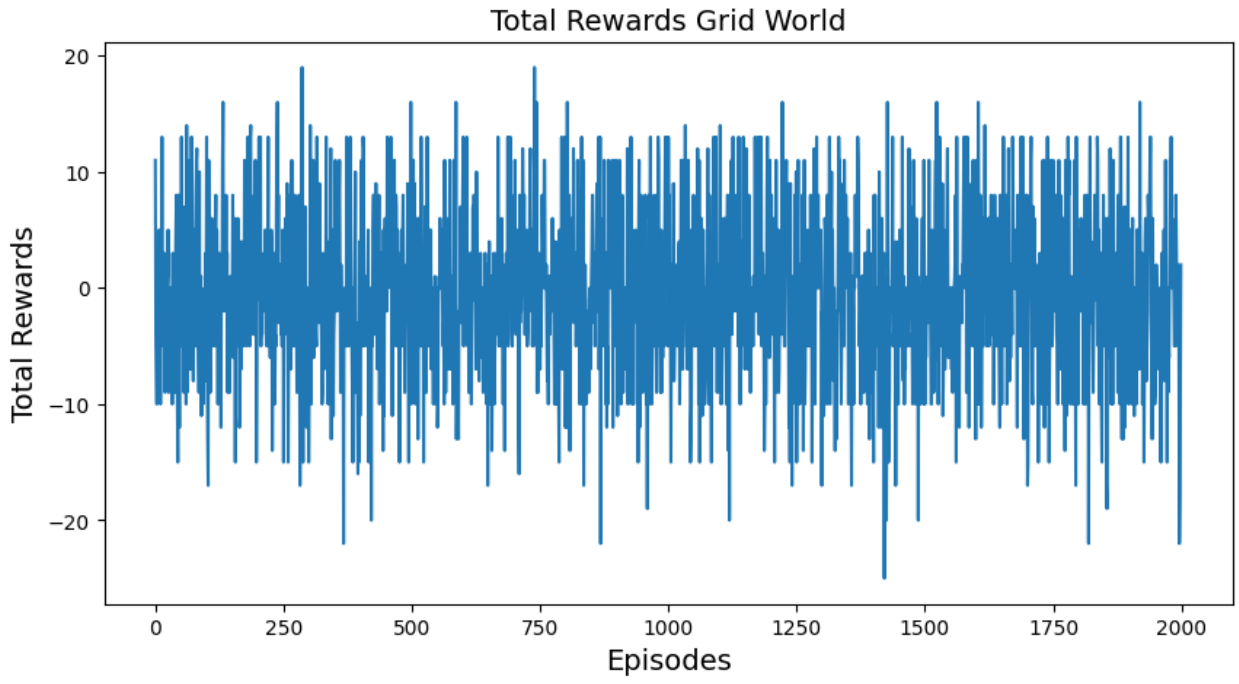
States = 4

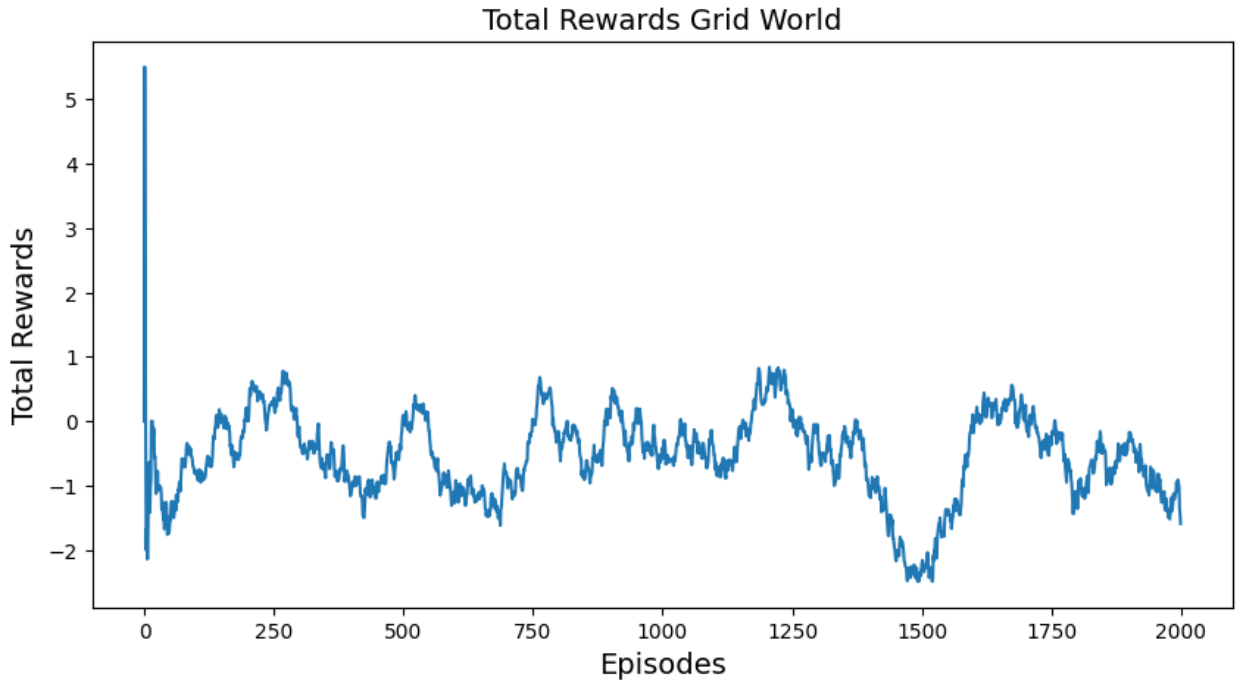
Actions = 2

Max Timesteps = 500

Reward = 1

Results for Grid World





States = 16

Actions = 4

Max Timesteps = 12

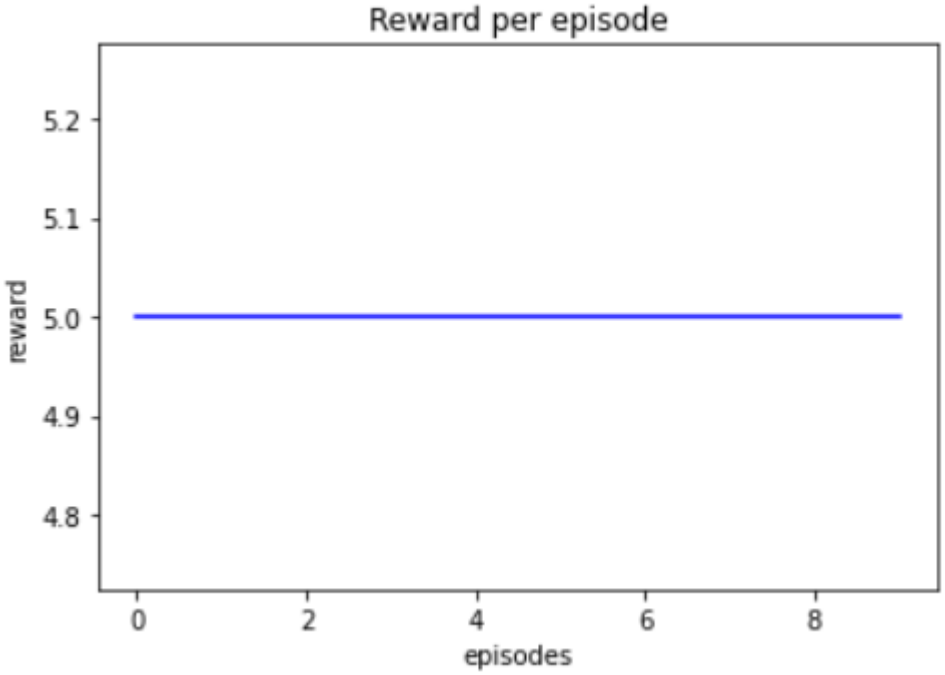
Rewards = 10,5,1,-3

5. Provide the evaluation results for each environment 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

A2C Evaluation

Grid World

```
Complete
```



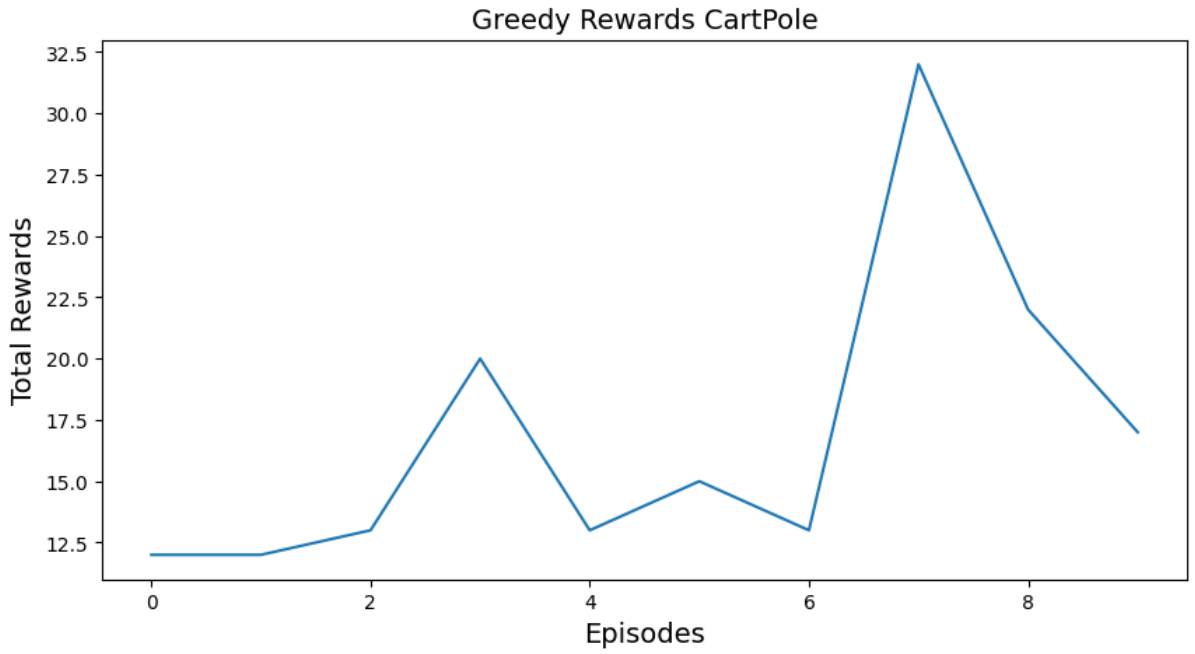
Cart Pole

Complete

Lunar Lander

Complete

PPO Evaluation



6. If you are working in a team of two people, we expect equal contribution for the assignment. Provide contribution summary by each team member.

Team Member	**Assignment Part**	Contribution (%)
sannihit	• Implemented A2C on Grid World, Cart Pole, Lunar Lander. • Report.	50%
sabariis	• Implemented PPO on Grid World, Cart Pole, Lunar Lander. • Report.	50%

References:

https://www.gymlibrary.dev/environments/classic_control/cart_pole/

https://www.gymlibrary.dev/environments/box2d/lunar_lander/

https://piazza.com/buffalo/fall2022/cse4546/resources