CSE 446 - REINFORCEMENT LEARNING - ASSIGNMENT 3

1. Discuss
    a. What are the roles of the actor and the critic networks?

Roles of the Actor and the Critic Networks
In reinforcement learning algorithms like A2C (Advantage Actor-Critic), two distinct components, the actor and the critic, play important roles:

Actor: The actor is responsible for selecting actions based on the current policy. It maps the environment's state to a probability distribution over the possible actions, from which actions are sampled. The actor updates its policy to maximize the expected future reward by taking actions that lead to higher rewards. Essentially, the actor determines which actions to take at each step.

Critic: The critic evaluates the actions taken by the actor. It does this by estimating the value function, which represents the expected future return (reward) from a given state. The critic helps the actor by providing feedback on how good or bad the actions were, allowing the actor to refine its policy. It estimates the value of states or state-action pairs, giving the actor a measure of the desirability of its actions.

The actor and critic work together in the A2C framework: the actor explores the environment by selecting actions, while the critic evaluates those actions to guide the actor's policy improvements.

    b. What is the "advantage" function in A2C, and why is it important?

The **"Advantage"** Function in A2C and Its Importance
The advantage function in A2C is used to improve the training efficiency of the actor. It is defined as the difference between the Q-value (expected return) and the value function (state value), i.e., the advantage of an action in a particular state. Mathematically, it is expressed as:

$A(s,a)=Q(s,a)-V(s)$
Where:

A(s,a) is the advantage function for a state s and action a

Q(s,a) is the action-value function, representing the expected return from taking action

V(s) is the value function, representing the expected return from state s (the critic's estimate of the state's value).

Importance of the Advantage Function:

Stabilization of Training: The advantage function is crucial for improving the stability and efficiency of the policy gradient updates. Without the advantage function, the updates would be based on absolute values, which could be noisy and inefficient. By using the advantage, the actor is updated in a way that highlights the relative benefit of specific actions over others, allowing it to learn more effectively.

Reduced Variance: The advantage function helps reduce the variance of the gradient estimates by focusing on the difference between the action value and state value, rather than just the raw reward. This reduces the effect of high-variance rewards, resulting in more stable and efficient learning.

c. Describe the loss functions used for training the actor and the critic in A2C.

In A2C, the actor and critic networks are trained simultaneously with different loss functions:

Actor Loss: The actor loss is based on the policy gradient method and is defined as the negative of the log probability of the taken action multiplied by the advantage. This encourages the actor to take actions that lead to higher rewards (positive advantage) and avoid actions that lead to lower rewards (negative advantage). The formula is:

$$\mathcal{L}_{\text{actor}} = -\log(\pi(a_t|s_t)) \cdot A(s_t, a_t)$$

Where:

$\pi(a_t|s_t)$ is the probability of taking action a_t at state s_t under the current policy,

$A(s_t, a_t)$ is the advantage, which indicates how good the action a_t is compared to the baseline value $V(s_t)$

Critic Loss: The critic's loss function is typically based on the mean squared error (MSE) between the predicted value function $V(s_t)$ and the target return (the value target). The value target is the discounted sum of rewards, which can be computed from the observed trajectory:

$$\mathcal{L}_{\text{critic}} = (V(s_t) - R_t)^2$$

Where: $V(s_t)$ is the predicted value of state s_t

R_t is the return (the discounted sum of rewards) for state s_t

This loss function forces the critic to learn an accurate value function that helps guide the actor's learning process.

Total Loss: The total loss for training both the actor and critic is the sum of the actor loss and critic loss:
$\mathcal{L} = \mathcal{L}_{\text{actor}} + 0.5 \cdot \mathcal{L}_{\text{critic}} - \beta \cdot H(\pi)$

Where $H(\pi)$ is the entropy of the policy, which is used as a regularization term to encourage exploration. The term β controls the strength of the entropy regularization, helping to balance exploration and exploitation.

In summary, the actor loss drives the actor to improve its policy by increasing the likelihood of actions that lead to positive advantages, while the critic loss helps the critic refine its value estimates, providing feedback to the actor on the value of different states. The combination of these losses allows the A2C algorithm to learn both the optimal policy and accurate value estimates efficiently.

2. Briefly describe the environment that you used (e.g. possible actions, states, agent, goal, rewards, etc). You can reuse related parts from your previous assignmnets.

Environment: CartPole-v1

Agent: The agent in the CartPole-v1 environment is tasked with balancing a pole that is attached to a cart. The agent has control over the cart's movement along a one-dimensional track.

State Space: The state space consists of four variables that represent the current state of the environment:

1. Cart position: The horizontal position of the cart (float).
2. Cart velocity: The velocity of the cart (float).
3. Pole angle: The angle of the pole with respect to the vertical (float).
4. Pole velocity: The angular velocity of the pole (float).

These four variables together form the observation space, which is a continuous space of four dimensions.

Action Space: The action space is discrete and consists of two actions:

1. Move left: The agent applies a force to the left to move the cart.

2. Move right: The agent applies a force to the right to move the cart.

Goal: The goal of the agent is to balance the pole on the cart for as long as possible, by applying forces to the cart to prevent the pole from falling.

Rewards: The agent receives a reward of +1 for every timestep that the pole remains balanced. The episode terminates when the pole falls beyond a certain angle (typically 15 degrees from the vertical), or when the cart moves too far away from the center of the track. When the episode ends, the total reward is the number of timesteps the agent successfully balanced the pole.

Termination Condition: The episode ends if:

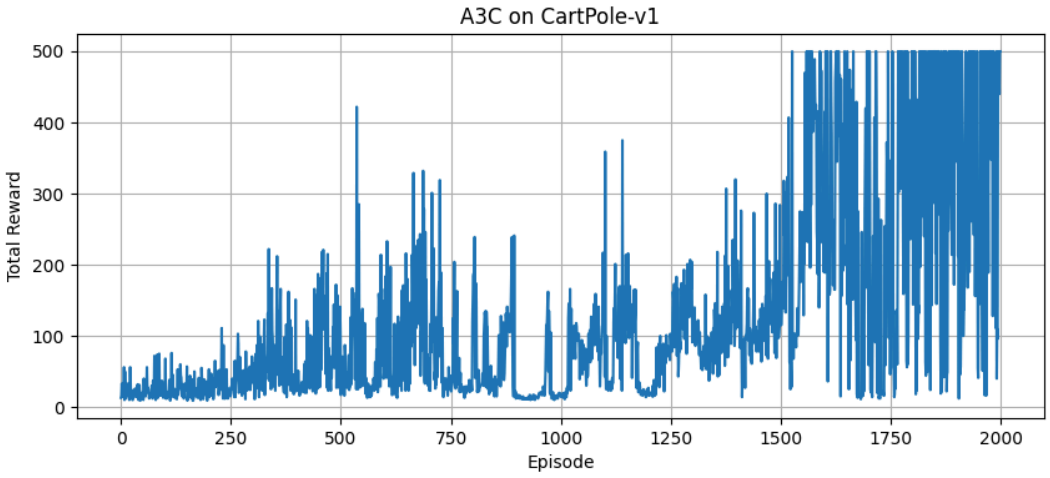
1. The pole's angle exceeds 15 degrees from the vertical.

2. The cart moves too far from the center (typically beyond 2.4 units).

Average Rewards to consider the environment solved: 475

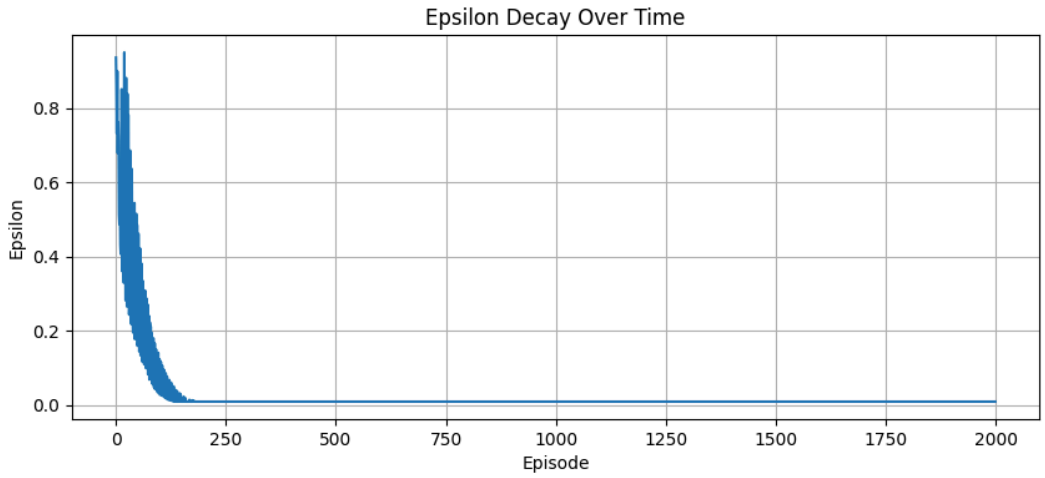
3. Show and discuss your results after applying your A2C/A3C implementation on the environments.
Plots should include epsilon decay and the total reward per episode.

Total Reward Over Episodes:



The total rewards increase significantly over time, with certain episodes having sharp increases in rewards, followed by periods of lower reward. This is a sign of an agent refining its policy over time, with some episodes showing more efficient actions than others. By the final stages of training, the agent consistently achieves higher rewards, suggesting convergence towards an effective policy.

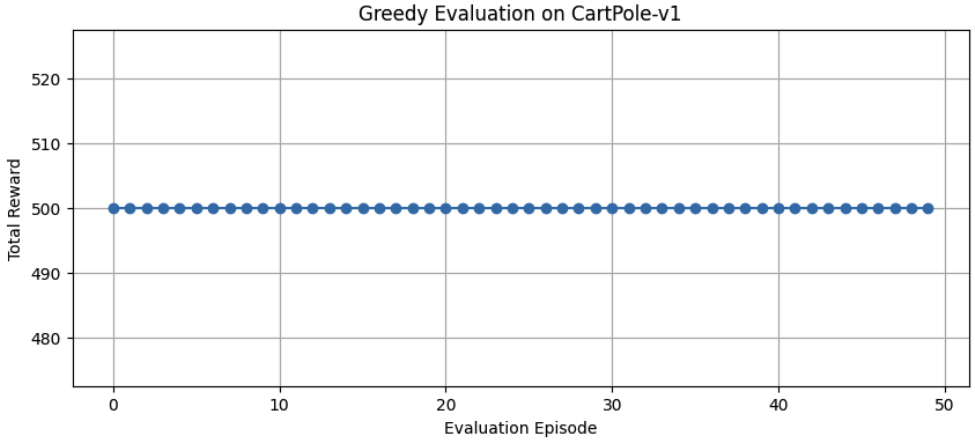
Epsilon Decay Over Time:
Initially, epsilon is set to 1.0, allowing for maximum exploration. As the episodes progress, epsilon decays significantly, reducing the amount of exploration and encouraging the agent to exploit the learned policy. The curve shows a rapid decay at the beginning, which eventually stabilizes. This is a typical pattern to ensure that the agent explores enough initially before focusing on exploiting the learned policy.



Reward per Episode for Each Worker:

The rewards fluctuate significantly across episodes, with some workers showing larger fluctuations than others. Over time, there is an observable trend of increasing rewards, as the workers begin to learn the optimal policy. The divergence between workers' rewards indicates that each worker may be exploring slightly different policies due to the asynchronous nature of A3C, but they are all converging towards higher rewards as the training progresses.

4. Provide the evaluation results. Run your agent on the three 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.




```
Average Reward across 50 episodes : 500.00
Environemt Solved!!
```

The agent was evaluated on 50 episodes using my trained agent that uses only greedy strategy, meaning it always selects the action with the highest probability as per its learned policy.
The total reward remains consistent at 500 across all evaluation episodes, demonstrating that the agent has solved the CartPole-v1 environment. A reward of 500 signifies that the agent has mastered balancing the cartpole. The average reward across 50 episodes is 500, average reward more than 475 means that the environment is solved.

5. Run your environment for 1 episode where the agent chooses only greedy actions from the learned policy. Render each step of the episode and verify that the agent has completed all the required steps to solve the environment. Save this render and include it in your report as clearly ordered screenshots or as a clearly named video file in your submission.

Video simulation for 1 episode is attached in the zip.

6. Provide your interpretation of the results.

The evaluation results show that the A3C agent has effectively learned to solve the CartPole-v1 environment, consistently achieving the maximum reward of 500 across all 50 evaluation episodes. This indicates that the agent has mastered the task and is reliably selecting the right actions based on the learned policy. The fact that it performs so well in a greedy evaluation, choosing the best possible actions every time, suggests that the agent has not only learned but also generalized the task well. Overall, these results highlight the success of the A3C algorithm in solving the CartPole problem and show that the agent is both stable and effective.

7. Include all the references that have been used to complete this part

References:

1. https://gymnasium.farama.org/environments/classic_control/cart_pole/
2. https://www.geeksforgeeks.org/actor-critic-algorithm-in-reinforcement-learning/
3. https://stackoverflow.com/questions/77042526/how-to-record-and-save-video-of-gym-environment
4. https://arxiv.org/abs/1602.01783
5.

CONTRIBUTION:

1. SHAURYA MATHUR - 50%
2. SHISHIR HEBBAR - 50%