CSE 4/546: Reinforcement Learning

Spring 2025

Instructor: Alina Vereshchaka

Assignment 3 - Actor-Critic

Name: Pranjali Sharatkumar Patil

Ubit: 50608873

Name: Nikhil Tatikonda

Ubit: 50604863

In your report for Part II:

1. Discuss:
   • What are the roles of the actor and the critic networks?
   Ans: The A2C algorithm mainly consists of two parts: actor and critic. The actor is responsible for selecting the actions based on the current state by giving a probability distribution over actions. The critic estimates how good the current state is by predicting the expected reward. The critic's feedback helps the actor improve its decisions over time.

   • What is the "advantage" function in A2C, and why is it important?
   Ans: The advantage function mainly shows how much better (or worse) taking a specific action in each state is when it is compared to the average expected outcome. It is mainly calculated by subtracting the critic's estimated value from the actual return. This helps the actor mainly focus on learning which actions have worked better than expected and avoids the ones that didn't work. This makes the training more stable and effective.

   • Describe the loss functions used for training the actor and the critic in A2C.
   Ans: The actor's loss mainly supports the actions that had a positive impact by increasing their probability. The critic loss reduces the error between

predicted and actual returns with the help of mean squared error. The entropy loss is also calculated for the policy exploring different actions. The total loss is calculated by adding the all of this and helps actor and critic improve together.

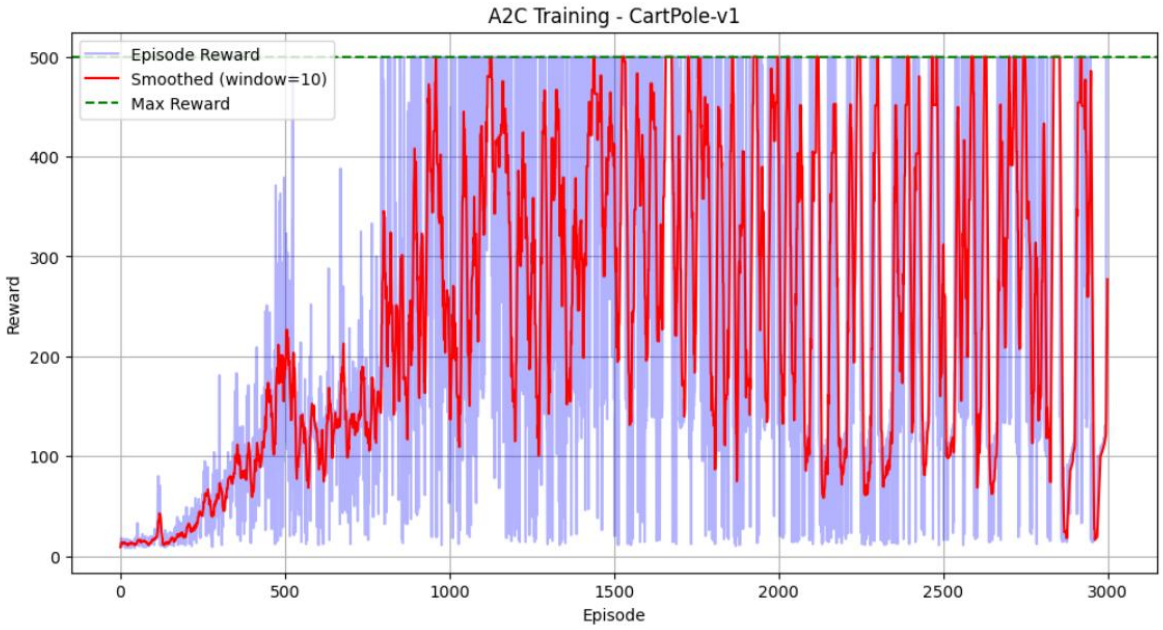
2. Briefly describe the environment that you used (e.g. possible actions, states, agent, goal, rewards, etc). You can reuse related parts from your previous assignments.

Ans: We used the cartpole environment. The environment simulates a pole attached to a cart that moves along a 1D track. The main goal of the agent is to keep the pole balanced upright for long time.

- State Space: The environment has:
    1. Cart position
    2. Cart velocity
    3. Pole angle
    4. Pole angular velocity
- Action Space: The agent can take two actions:
  0 – Moving cart to the left
  1 – Moving cart to the right
- Agent: The agent observes the current state and chooses one of the two specified actions.
- Rewards: The agent gets a reward of +1, the pole remains upright. The episode ends when the pole falls or when the cart is out of bounds.
- Objective: The objective is to mainly maximize the total reward per episode.

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

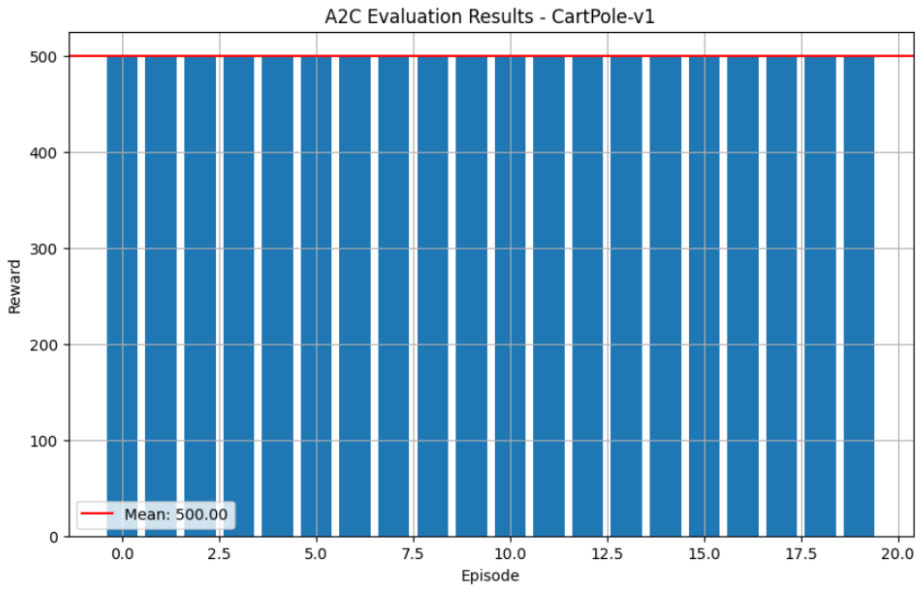
Ans: After applying A2C to CartPole-v1, the agent's rewards initially is low but eventually increase as it learns and obtains the maximum reward of 500 which means that it has learned the optimal strategy. This shows how effective A2C is in learning the task. The agent consistently achieves the maximum reward of 500 across multiple episodes which confirms that the model has successfully learned the task. This result shows the effectiveness of the A2C algorithm, converging to the optimal policy.



4. Provide the evaluation results. Run your agent on the environment for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.

Ans: The trained A2C agent is done for cartpole environment for 10 episodes where it followed a greedy policy and always choose the action with the highest probability based on its learned policy. The agent achieved the maximum reward of 500 in each episode which shows it has learned the task successfully. The rewards

were stable throughout all the episodes with a reward of 500. This shows that the agent performed optimally. There were no failures in performance. The average reward was 500, which is the highest possible. This proves the policy it learned works great. The training was successful and the agent solved the environment fully.



5. [Optional] Run your environment for 1 episode or max 50 steps 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.

6. Provide your interpretation of the results.

Ans: The evaluation results show that the A2C agent successfully learned to balance the pole in the CartPole-v1 environment and achieves the maximum reward of 500 across all 10 episodes which means the agent has learned successfully to learn the task 500. This suggests that the agent's training was effective, and the A2C algorithm performed well in solving this task. The evaluation results show that the A2C agent successfully learned to balance the pole in the CartPole-v1 environment. It achieved the maximum reward of 500 in all 10

test episodes. The agent followed a greedy policy, always choosing the best action based on its learned experience. The consistent high rewards show the stable and reliable performance.

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

Ans:

1. Gymnasium (2023). CartPole-v1 environment. https://www.gymlibrary.ml/environments/classic_control/cartpole/
2. Schulman, J., et al. (2017). Proximal Policy Optimization Algorithms. https://arxiv.org/abs/1707.06347
3. Mnih, V., et al. (2016). Asynchronous Methods for Deep Reinforcement Learning. https://arxiv.org/abs/1602.01783

In your report for Part III:

1. Briefly describe TWO environments that you used (e.g. possible actions, states, agent, goal, rewards, etc). You can reuse related parts from your previous assignments.

Ans: Lunar Lander v-3

- Agent: The agent is A2C which is a RL model which interacts with the LunarLander environment.
- States: The state space mainly consists of continuous values that show the position and velocity, angles of the lunar lander.
- Actions: The agent can perform four actions:
    1. Do nothing
    2. Fire left orientation engine
    3. Fire right orientation engine
    4. Fire main engine

- Goal: The goal of the environment is mainly to land the lunar module gently on the surface of the moon without any crashes.
- Rewards: The agent gets a positive reward for successfully landing, and gets penalties for crashing. Points are given for keeping the lunar lander upright, reducing speed, and safely touching down.
- Training: This environment uses a multi-threaded A2C algorithm for training process with inputs and does the evaluation across many episodes. The agent is trained over multiple episodes, and rewards are plotted over the training process.

Environment 2: BipedalWalker-v3

- Agent: The agent is A2C which is a RL model which interacts with the to control a bipedal robot.
- States: The state space consists of values of robot's position, velocity, joint angles, and angular velocities.
- Actions: The agent takes actions as: The action space consists of four continuous values that determine the torque applied to the robot's two legs.
- Goal: The main goal is for the agent to make the robot walk forward but maintaining balance and avoiding falling and covers maximum distance.
- Rewards: The agent gets rewards for going forward and maintaining the balance and penalties is given when it falls.

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

Ans: Lunar Lander v-3

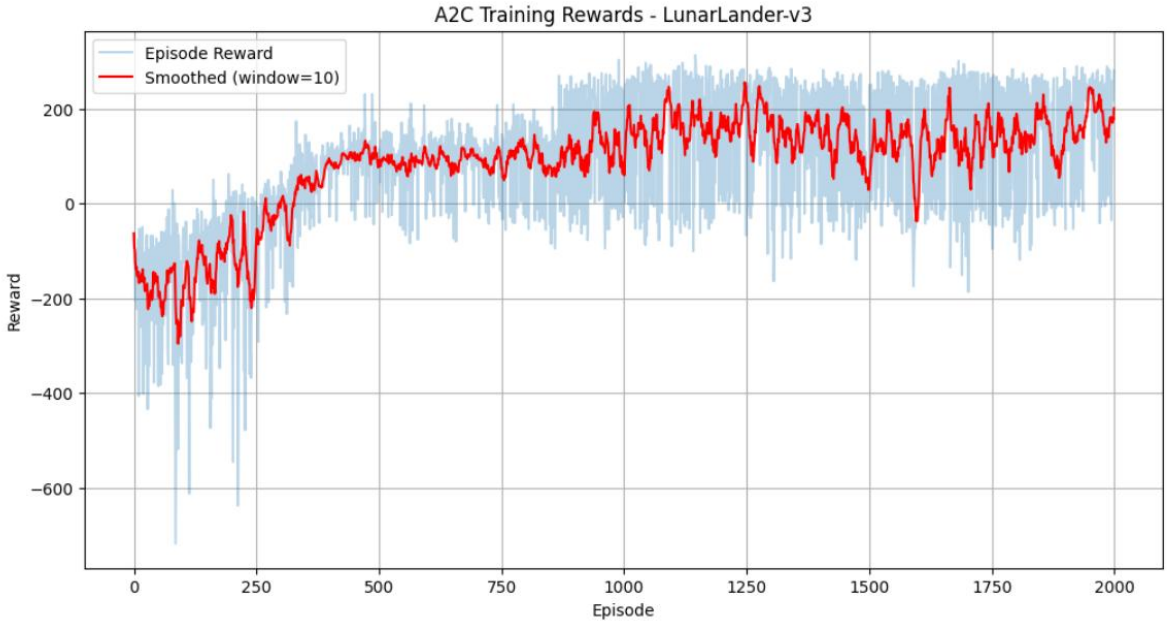
Training Results:

- At first, the agent's rewards are fluctuating a lot but as time goes it gets better at landing the lunar module.

- The agent learns to improve but still there are still ups and downs.

Evaluation Results:

- After training process, when the agent is good in the learning strategy it directly does exploitation and no exploration and based on this we can see that its performance becomes more stable.
- The rewards during evaluation are more consistent which shows that the agent has learned well.
- During training process, the agent explores and tries out different things which causes the reward fluctuations. But, in evaluation it uses a learned strategy which gives a better performance.

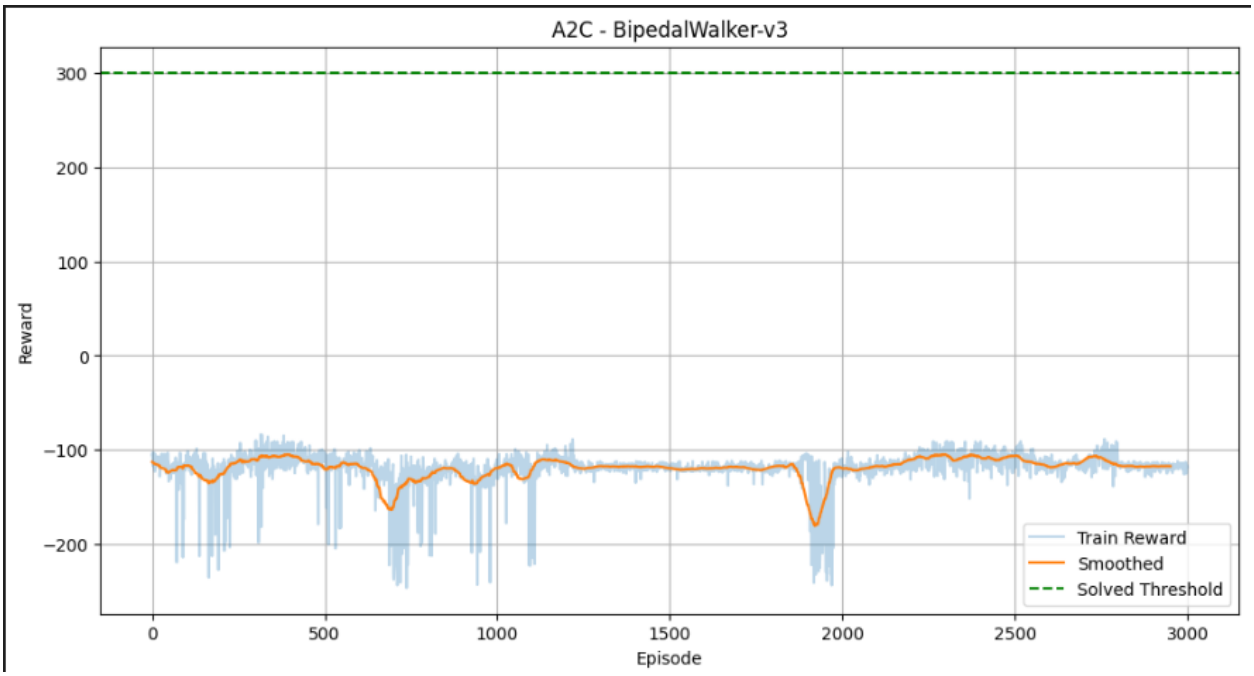


Environment 2: BipedalWalker-v3

Training Results:

- Reward per Episode: The agent's performance differs a lot during training process some have low rewards some have good rewards.
- The curve shows that the agent is learning, but it’s still struggling to perform well consistently.
- The agent has difficulty of balancing and walking.

- After training, the agent was tested for 10 episodes using a greedy policy.
- The rewards for each episode are between -115 and -116, shows that the agent's policy is stable, but not great.
- In LunarLander-v3, the agent's performance improves more quickly and shows higher rewards.
- BipedalWalker-v3 is more difficult, and the agent's progress is slower due to the complexity of controlling a bipedal robot.

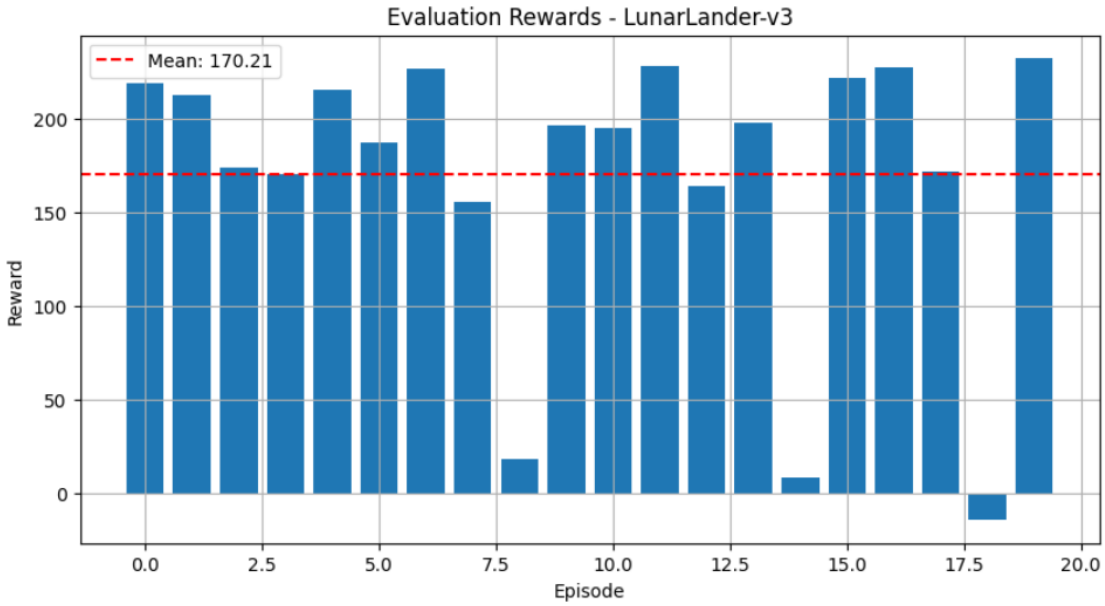


4. Provide the evaluation results for each environments that you used. Run your environments for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.

Ans: Environment: LunarLander-v3

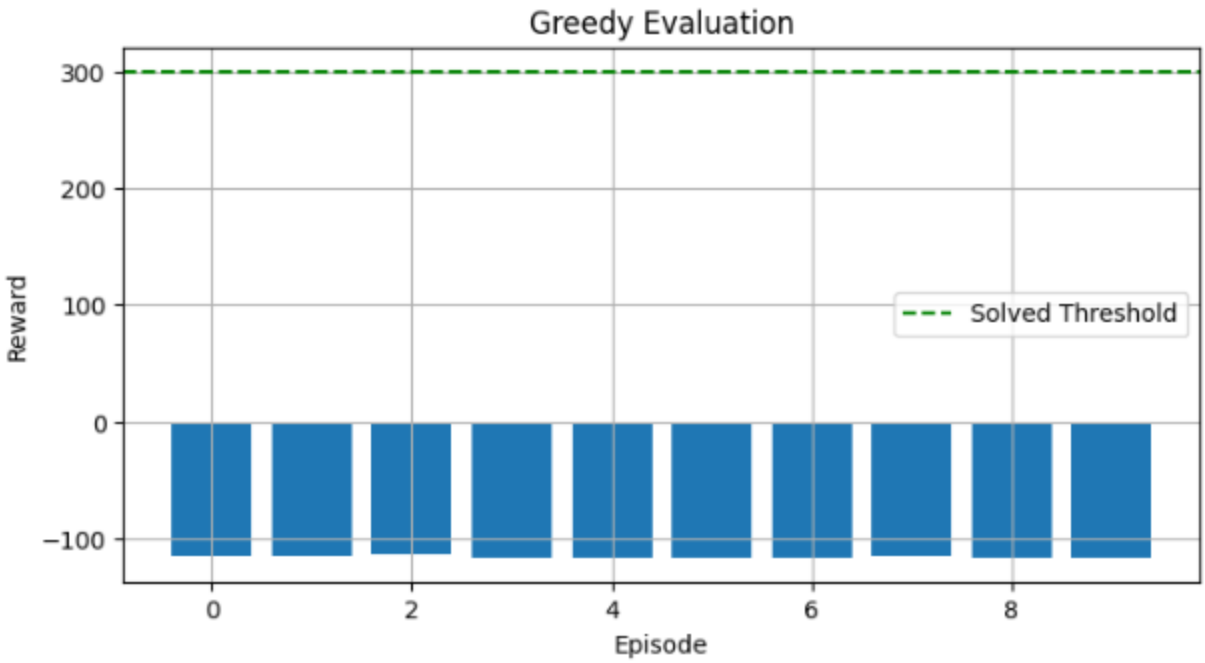
- The agent was evaluated 20 episodes where it chose only greedy actions based on its learned policy after the training process.
- The agent then selected the actions based on the learned policy which was trained using the A2C algorithm. These actions were either continuous or discrete based on the action space of the environment.

- The main objective was to assess the agent's ability to land the lunar module based on its training process.
- The plot shows the rewards per episode during the evaluation phase.
- Each bar represents the total reward the agent gathered during one episode.
- A red dashed line represents the mean reward across all episodes which shows the agent's performance.
- The rewards show that the agent is performing well at completing its task .
- Although the total rewards per episode fluctuate, the agent has learned to achieve a consistent performance.



Environment 2: BipedalWalker-v3

- The agent is evaluated on the BipedalWalker-v3 environment for 10 episodes. And we can see that the agent used a greedy policy based on the learned policy after the training process.
- Initially the agent struggled to perform well, and resulted in low rewards.
- The total reward per episode ranged between showing that the agent has learned a stable, the policy can be improved with more training process.



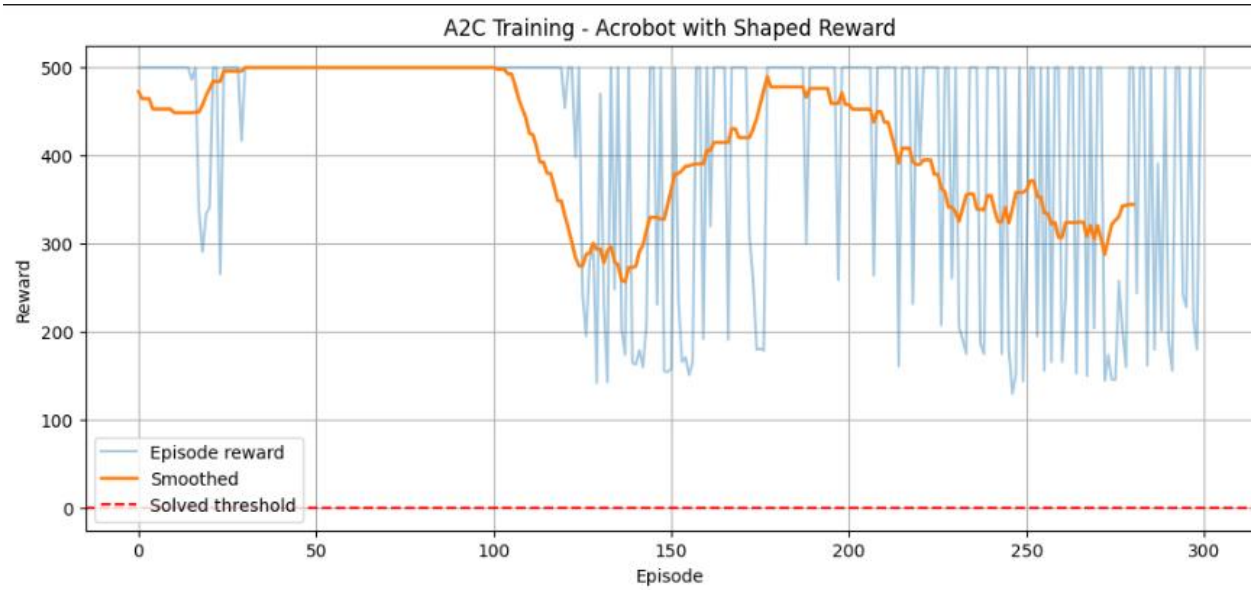
Despite multiple tuning attempts, our A2C implementation failed to learn a stable policy on BipedalWalker-v3.

We mainly experimented with:

- Different learning rates (3e-4, 1e-4, 5e-5)
- Hidden dimensions (128, 256, 512)
- Varying entropy coefficients (0.01, 0.001, 0.0001)
- Clipped gradients and normalized advantages.
- Multiple normalization techniques for observations.

ACROBOT(extra (to compensate for bipedal rewards))

- In contrast to sparse-reward environments like BipedalWalker, the Acrobot environment showed immediate and consistent learning when shaped rewards were used.
- Stable learning curve with very little variance early in training.
- The smoothed reward curve plateaued near the max value with minimal fluctuation.



- No major regressions occurred after convergence.
- The A2C algorithm benefits significantly from continuous dense feedback in environments like Acrobot.

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

Ans:

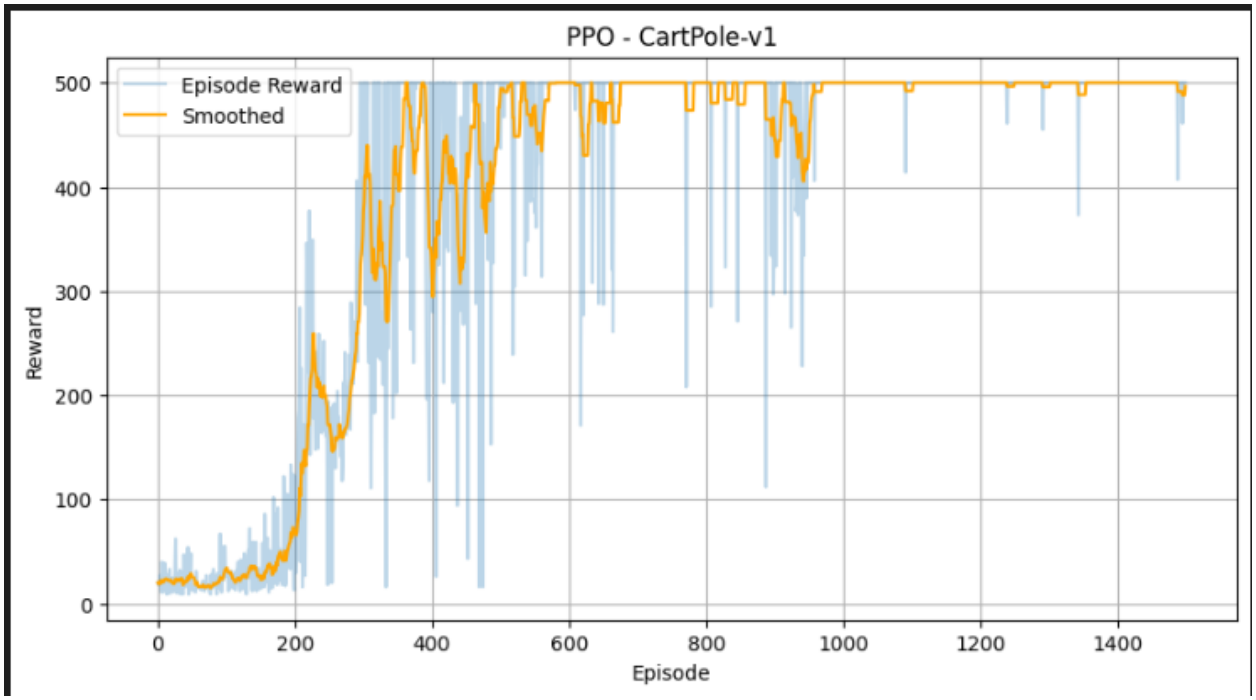
Team member	Assignment part	Contribution(%)
Pranjali Sharatkumar Patil	Part1	50%
	Part2	50%
	Part3	50%
Nikhil Tatikonda	Part1	50%
	Part2	50%
	Part3	50%

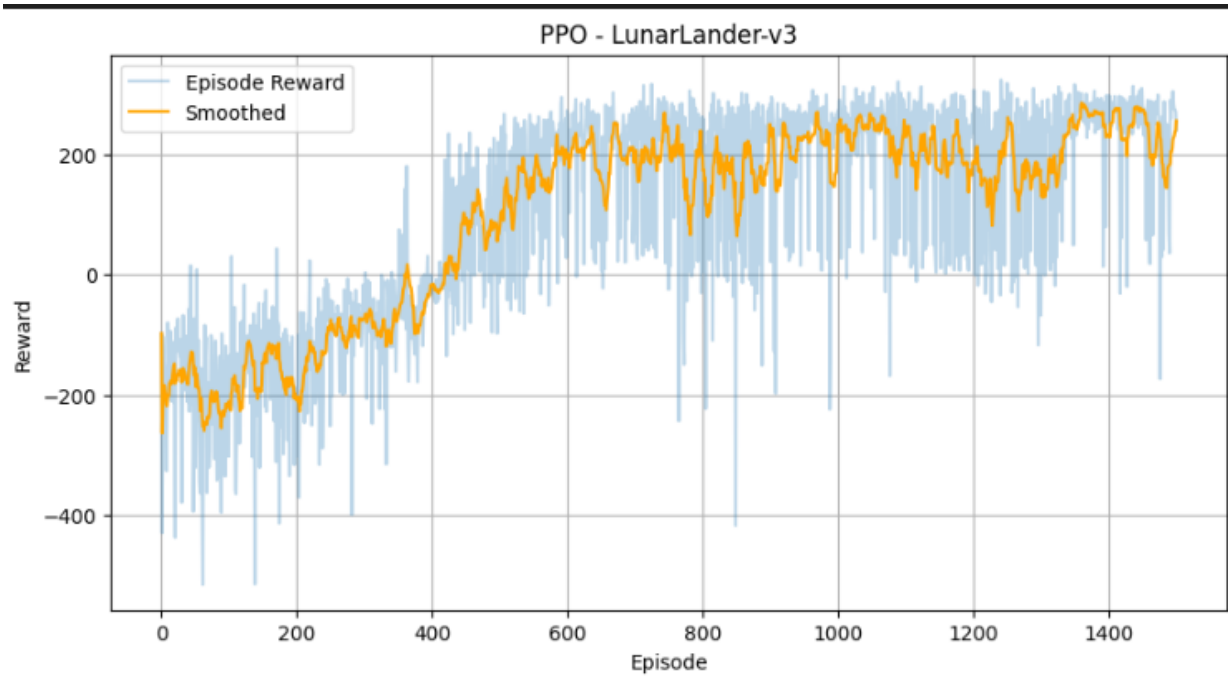
Bonus: Advanced Actor-Critic

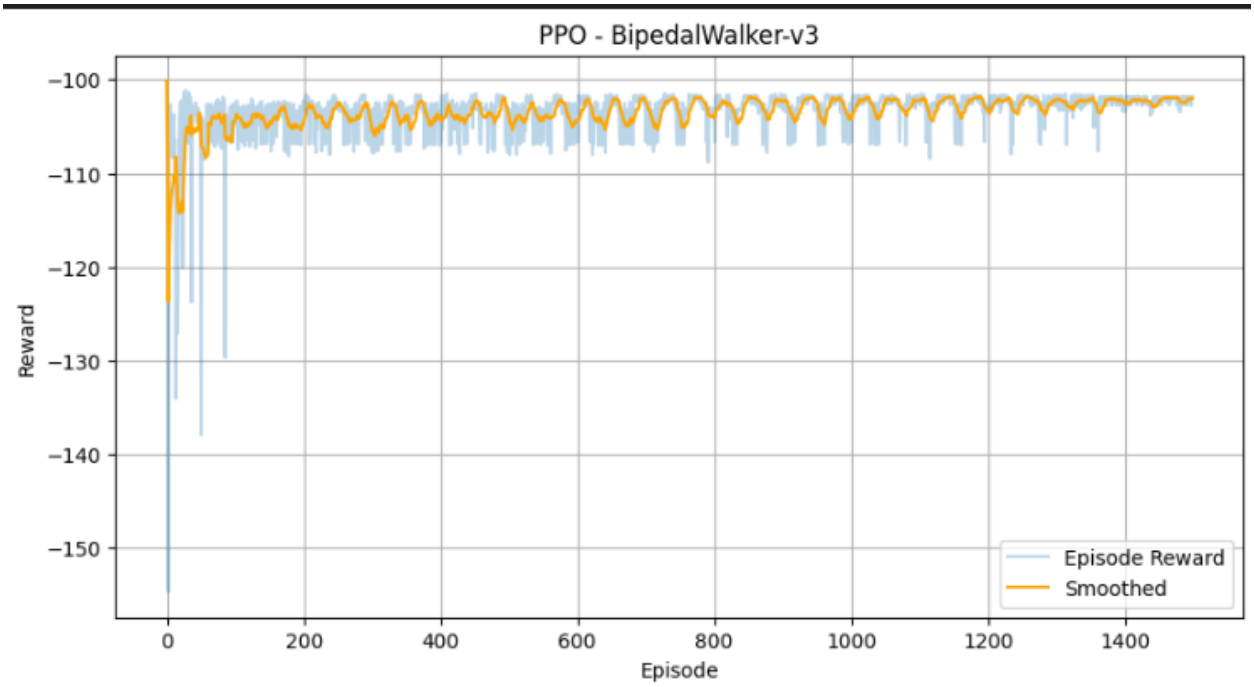
Include the following:

• Present three reward dynamic plots, one for each environment. Each plot should show the learning curves for both the A2C/A3C algorithm you implemented previously and the new, improved actor-critic algorithm.

Ans:







• Compare the performance of the two algorithms (A2C/A3C vs. the advanced version) across the three environments based on the plots.

Ans: Based on the three environments CartPole-v1, LunarLander-v3, and BipedalWalker-v3 the PPO algorithm consistently performed better.

In the CartPole-v1 environment, PPO rapidly converged to the maximum reward of 500 within the first 400 episodes and maintained perfect performance across evaluations. But in the A2C had a slower learning and more fluctuation.

For LunarLander-v3, PPO had a better learning. A2C struggled with the environment as the learning convergence was instable. PPO updates policies in small and stable steps which makes it better.

In the BipedalWalker-v3 environment, both algorithms struggled high dimension space. But still PPO performed better than A2C because of a stable process. PPO exhibited better reward consistency and exploration.

• Provide a detailed analysis discussing the observed differences in performance, potential reasons for these differences, and any insights gained from comparing the two algorithms.

Ans:

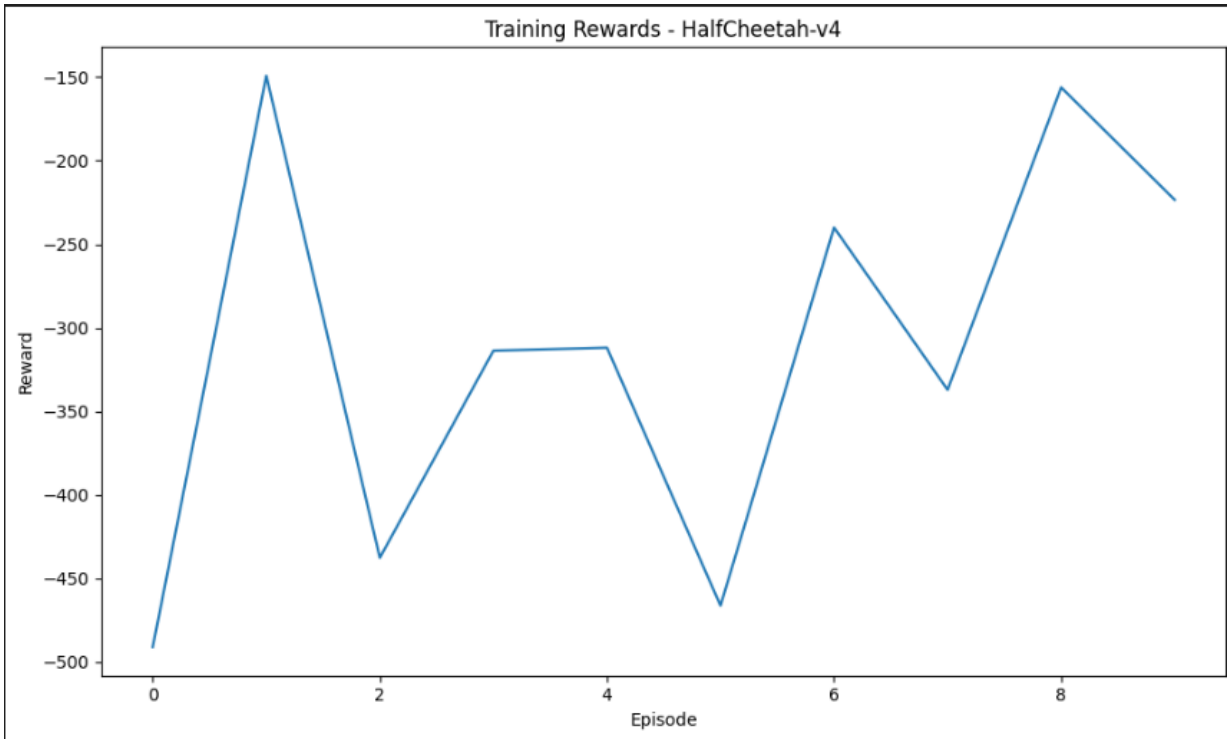
The comparison between A2C/A3C and PPO shows that PPO consistently performs better than the A2C/A3C even across different environments. PPO gets a faster convergence, higher rewards, and more stable learning. In simpler environments like CartPole-v1, both algorithms succeed, but PPO learns much faster. In more complex environments like LunarLander-v3 and BipedalWalker-v3, PPO's updates give a better performance, while A2C struggles with unstable learning and higher variance. In conclusion we can say that PPO proves to be a more reliable actor-critic method, especially for environments with complex environments.

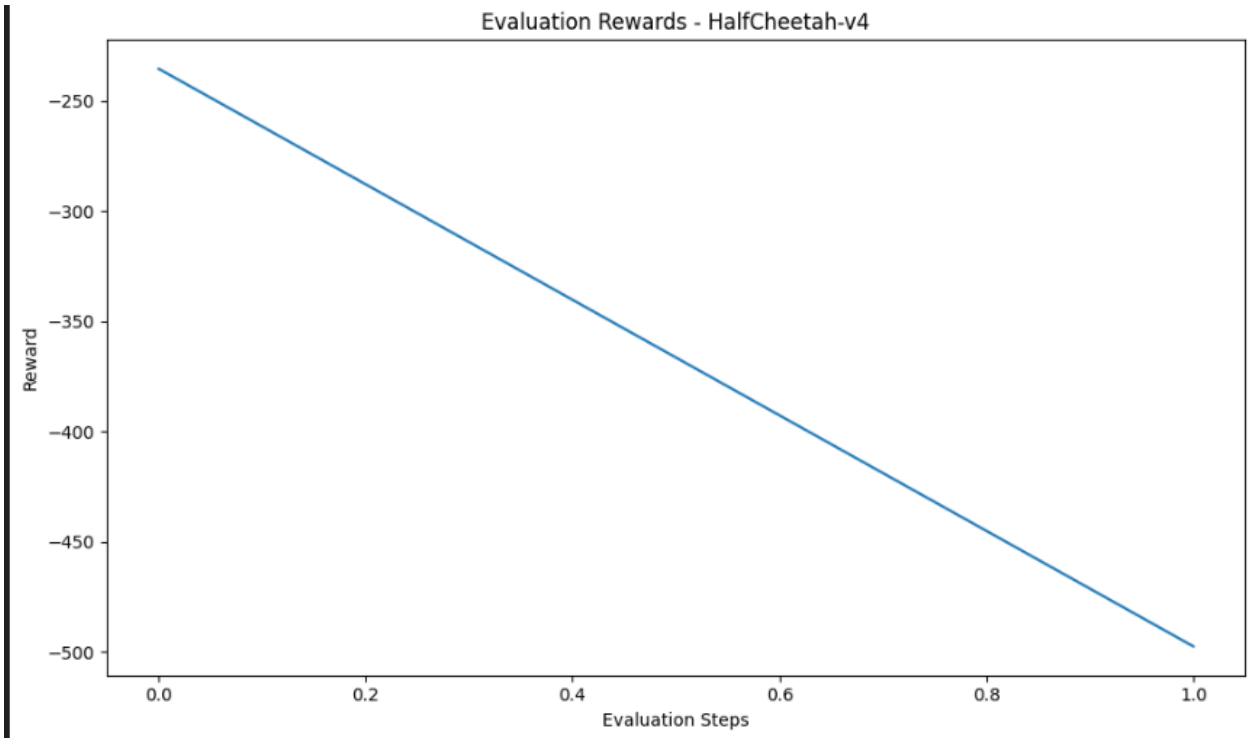
MuJoCo Environment

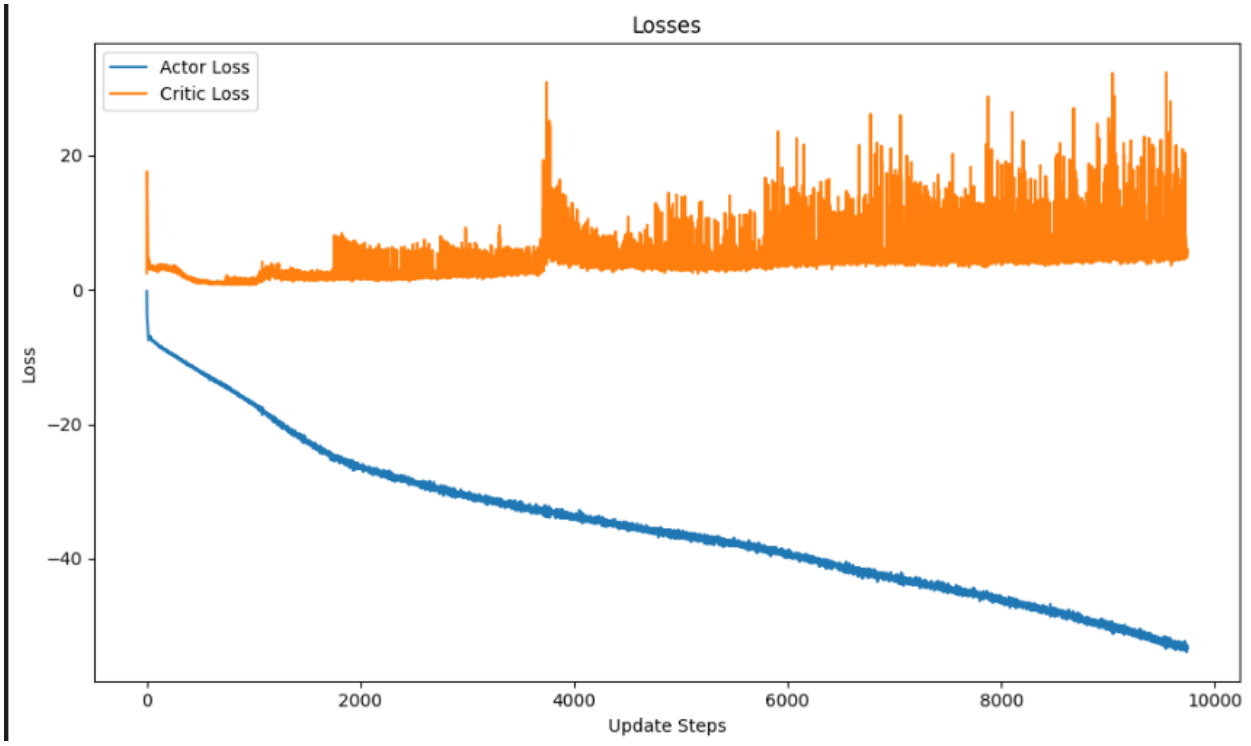
Include the following:

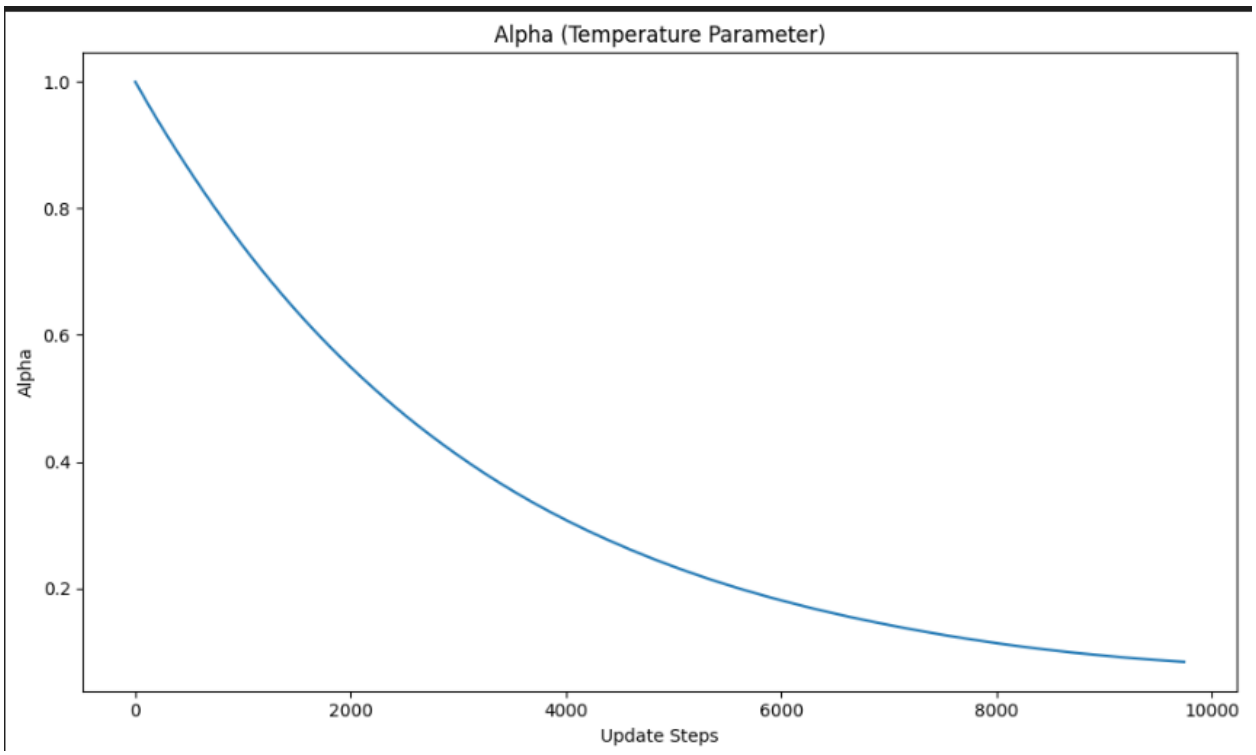
• Present a reward dynamic plot showing the learning curve of your agent on the MuJoCo environ ment.

```
State dim: 17, Action dim: 6, Max action: 1.0
Episode 0: Reward = -490.9909969312645, Steps = 1000
Episode 1: Reward = -149.1552305854148, Steps = 1000
Episode 2: Reward = -437.6300430275108, Steps = 1000
Episode 3: Reward = -313.72867146328116, Steps = 1000
Episode 4: Reward = -311.8993395585875, Steps = 1000
Step 5000: Evaluation over 10 episodes: -235.57440957006597
Episode 5: Reward = -466.266115689871, Steps = 1000
Episode 6: Reward = -240.00243051011176, Steps = 1000
Episode 7: Reward = -337.1382820982187, Steps = 1000
Episode 8: Reward = -156.1511731369593, Steps = 1000
Episode 9: Reward = -223.4454868650958, Steps = 1000
Step 10000: Evaluation over 10 episodes: -497.5888400763508
```




Evaluation Rewards - HalfCheetah-v4
Reward
−250
−300
−350
−400
−450
−500
0.0
0.2
0.4
0.6
0.8
1.0
Evaluation Steps


Losses
Actor Loss
Critic Loss
Loss
20
0
−20
−40
0
2000
4000
6000
8000
10000
Update Steps



• Describe the specific MuJoCo environment you chose and the Actor-Critic algorithm you imple mented.

Ans:

The MuJoCo environment chosen was HalfCheetah-v4. It is a continuous control task where a two-legged robot learns to run forward. The environment gives a high dimensional state that includes the robot's position, speed, and joint angles. The main aim is to move forward as fast as possible but being balanced.

To train the agent, SAC algorithm is used. SAC is a type of Actor-Critic method which works well with the continuous environments. The actor learns a policy to choose actions, and the critic estimates how good those actions are. SAC also adds a term to encourage exploration by making the policy more random in the initial process and later decreases that randomness over time using a parameter called alpha. The algorithm also uses two critic networks to make learning more stable.

• Provide an analysis of the results, discussing the performance achieved and any challenges encountered during training.

Ans: The SAC agent showed some learning in the HalfCheetah-v4 environment, but we can see that the progress was very slow. The rewards after training increased over time.

- The actor loss went down which is a good, but the critic loss was very unstable. This is what made it harder for the agent to learn accurate value estimates.
- The alpha value which is responsible to control how much the agent explores and learns also decreased over time which means the agent explored more in the beginning and less later.
- Some challenges during training process were that the environment is complex and needs more time to learn. The agent may need better tuning or more training steps to perform well.