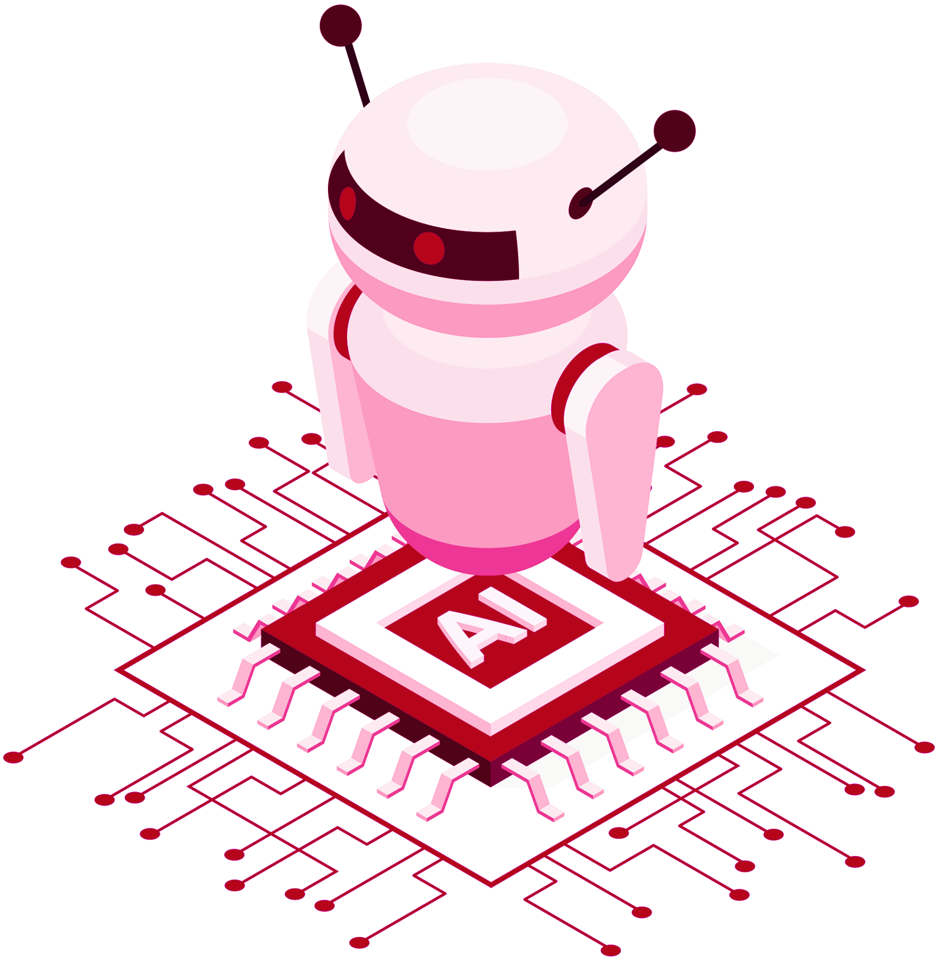


Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Homework 4:

Advanced Methods in RL

By:

[Ayeen Poostforoushan]

[401105742]

Spring 2025

Contents

Grading

The grading will be based on the following criteria, with a total of 100 points:

Task	Points
Task 1: PPO	25
Task 2: DDPG	20
Task 3: SAC	25
Task 4: Comparison between SAC & DDPG & PPO	20
Clarity and Quality of Code	5
Clarity and Quality of Report	5
Bonus 1: Writing your report in Latex	10

1 Task 1: Proximal Policy Optimization (PPO) [25]

1.1 Question 1:

What is the role of the actor and critic networks in PPO, and how do they contribute to policy optimization?

In PPO, the actor network is responsible for deciding which action to take given the current state (i.e., it represents the policy), while the critic network estimates the value function (how good it is to be in a certain state). The actor and critic work together in a loop: the actor proposes actions, and the critic provides feedback on how good those actions are. This feedback helps the actor adjust its policy to increase rewards over time. By having a separate critic, PPO can stabilize learning because it gets more reliable estimates of the action's long-term return, reducing the variance of policy updates.

1.2 Question 2:

PPO is known for maintaining a balance between exploration and exploitation during training. How does the stochastic nature of the actor network and the entropy term in the objective function contribute to this balance?

PPO uses a stochastic policy for its actor network, which means actions are sampled from a probability distribution instead of always picking the single best action. This randomness naturally encourages the agent to explore different actions. In addition, PPO's objective function includes an entropy term. Entropy measures how unpredictable or random the policy is. By rewarding higher entropy, the algorithm ensures the agent keeps trying out new actions (exploration) rather than converging too quickly on a single strategy (exploitation). This balance prevents the policy from getting stuck in suboptimal behavior.

1.3 Question 3:

When analyzing the training results, what key indicators should be monitored to evaluate the performance of the PPO agent?

When looking at the PPO training results, the main indicators to watch are the average episodic reward (to see if it's improving over time), the policy loss (to ensure the policy is updating in a stable manner), and the value function loss (to make sure the critic is accurately estimating returns). It can also be helpful to track the entropy of the policy, because a drop in entropy might mean the policy is becoming too deterministic and not exploring enough.

2 Task 2: Deep Deterministic Policy Gradient (DDPG) [20]

2.1 Question 1:

What are the different types of noise used in DDPG for exploration, and how do they differ in terms of their behavior and impact on the learning process?

DDPG typically uses either Ornstein-Uhlenbeck (OU) noise or Gaussian noise to encourage exploration in continuous action spaces. OU noise is temporally correlated, which means the noise added to the action at one timestep influences the noise at the next timestep. This can help produce smoother action variations, which can be beneficial for physical control tasks. Gaussian noise, on the other hand, is independent at each timestep and can lead to more abrupt changes in actions. Both methods add randomness to the policy's output to promote exploration, but OU noise tends to provide more stable, correlated exploration in continuous control environments.

2.2 Question 2:

What is the difference between PPO and DDPG regarding the use of past experiences?

The key difference lies in how they use past experiences. PPO is an on-policy algorithm, meaning it relies on data generated by the current policy and updates the policy using that fresh data. DDPG is an off-policy algorithm, so it stores past experiences in a replay buffer and can continue to learn from them even after the policy has changed. This replay buffer approach helps stabilize learning by reusing experiences and breaking correlations in the training data, whereas PPO only learns from the latest set of trajectories collected by its current policy.

3 Task 3: Soft Actor-Critic (SAC) [25]

3.1 Question 1:

Why do we use two Q-networks to estimate Q-values?

In SAC, two Q-networks are used to address the overestimation bias common in value-based methods. By maintaining two independent estimators and taking the minimum of their outputs when calculating target Q-values, the algorithm becomes more conservative. This reduces the chance of overly optimistic value estimates, leading to more stable and reliable policy updates during training.

3.2 Question 2:

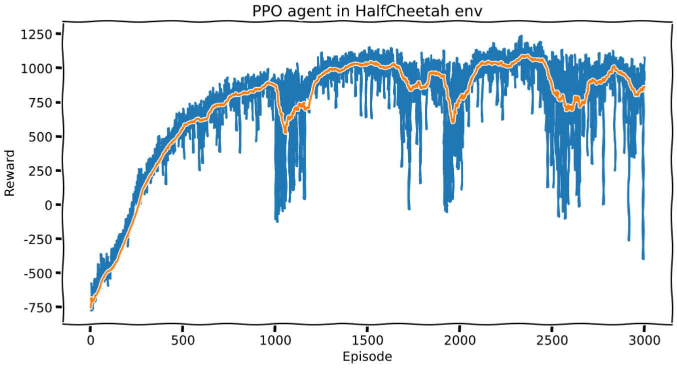
What is the temperature parameter α, and what is the benefit of using a dynamic α in SAC?

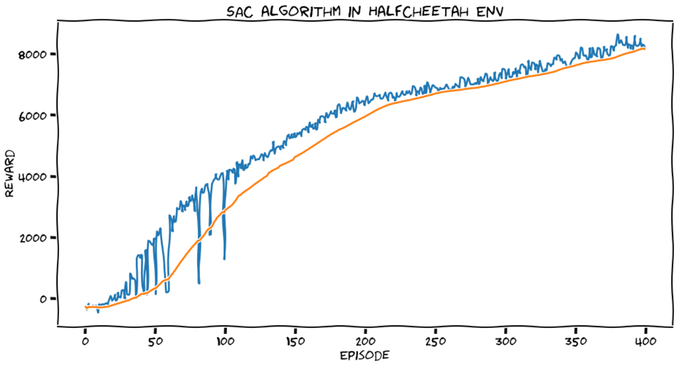
The temperature parameter, α, in SAC controls the trade-off between maximizing expected reward and maximizing policy entropy. A higher α encourages more exploration by weighting the entropy term more heavily, while a lower α shifts the focus toward reward maximization. Using a dynamic α allows the agent to adjust this balance automatically during training, which helps the agent explore sufficiently in the early stages and focus on exploiting learned behaviors as training progresses, leading to improved overall performance.

3.3 Question 3:

What is the difference between evaluation mode and training mode in SAC?

During training, SAC uses a stochastic policy that samples actions from a probability distribution and includes entropy in the objective to encourage exploration. In contrast, evaluation mode typically employs a deterministic policy that selects the action with the highest probability or value, without adding noise. This deterministic approach in evaluation mode provides a clearer measure of the agent's performance by eliminating the randomness that is beneficial for exploration during training.



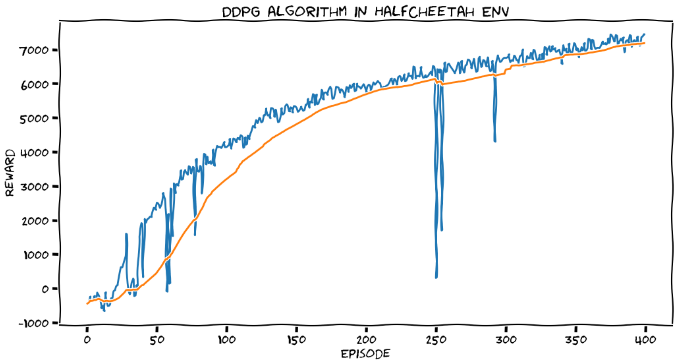


4 Task 4: Comparison between SAC & DDPG & PPO [20]

4.1 Question 1:

Which algorithm performs better in the `HalfCheetah` environment? Why?
Compare the performance of the PPO, DDPG, and SAC agents in terms of training stability, convergence speed, and overall accumulated reward. Based on your observations, which algorithm achieves better results in this environment?

Based on our observations, SAC tends to outperform both PPO and DDPG on HalfCheetah. SAC shows more stable training and converges faster to higher accumulated rewards. In contrast, our PPO agent exhibited unusual behavior—the cheetah ended up moving forward upside down—indicating instability in its learned policy, while DDPG often struggles with consistency due to its reliance on deterministic policies and sensitivity to noise.



4.2 Question 2:

How do the exploration strategies differ between PPO, DDPG, and SAC?
Compare the exploration mechanisms used by each algorithm, such as deterministic vs. stochastic policies, entropy regularization, and noise injection. How do these strategies impact learning in environments with continuous action spaces?

The exploration strategies vary notably between these algorithms. PPO uses a stochastic policy, naturally sampling actions from a probability distribution and further encourages exploration through an entropy bonus, which helps prevent premature convergence to suboptimal policies. DDPG, on the other hand, is deterministic and relies on adding external noise (like Ornstein-Uhlenbeck or Gaussian noise) to its actions during training, which can sometimes lead to less smooth exploration. SAC blends the benefits of both by using a stochastic policy with entropy regularization that not only encourages exploration but also helps maintain a balance between exploring new actions and exploiting known good ones, particularly important in continuous action spaces.

4.3 Question 3:

What are the key advantages and disadvantages of each algorithm in terms of sample efficiency and stability?
Discuss how PPO, DDPG, and SAC handle sample efficiency and training stability. Which algorithm is more sample-efficient, and which one is more stable during training? What trade-offs exist between these properties?

When considering sample efficiency and stability, DDPG is generally more sample efficient because it learns off-policy from a replay buffer, but this comes at the cost of training stability—small changes in hyperparameters or noise settings can lead to erratic behavior. PPO, as an on-policy algorithm, tends to be more stable but is less sample efficient since it discards old experiences once an update is made. SAC strikes a good balance between the two; it is off-policy and sample efficient while also incorporating entropy maximization to maintain stability during training. The trade-off here is that while SAC can be robust, it might require careful tuning of the entropy temperature to get the right balance.

4.4 Question 4:

Which reinforcement learning algorithm—PPO, DDPG, or SAC—is the easiest to tune, and what are the most critical hyperparameters for ensuring stable training for each agent?
How sensitive are PPO, DDPG, and SAC to hyperparameter choices, and which parameters have the most significant impact on stability? What common tuning strategies can help improve performance and prevent instability in each algorithm?

PPO is often considered the easiest to tune due to its clipping mechanism, which limits policy updates and reduces the sensitivity to hyperparameter changes. The most critical hyperparameters for PPO are the learning rate and the clipping parameter. DDPG is more challenging because it is highly sensitive to the choice of noise parameters, learning rates, and the design of the replay buffer. SAC tends to be relatively robust, but tuning its temperature parameter (which governs the balance between reward maximization and exploration) along with the learning rates for the actor and critic remains crucial. In general, a common tuning strategy involves starting with default parameters known to work well in similar environments and then adjusting the learning rates and noise or temperature settings gradually while monitoring both the stability of the training curves and the achieved rewards.