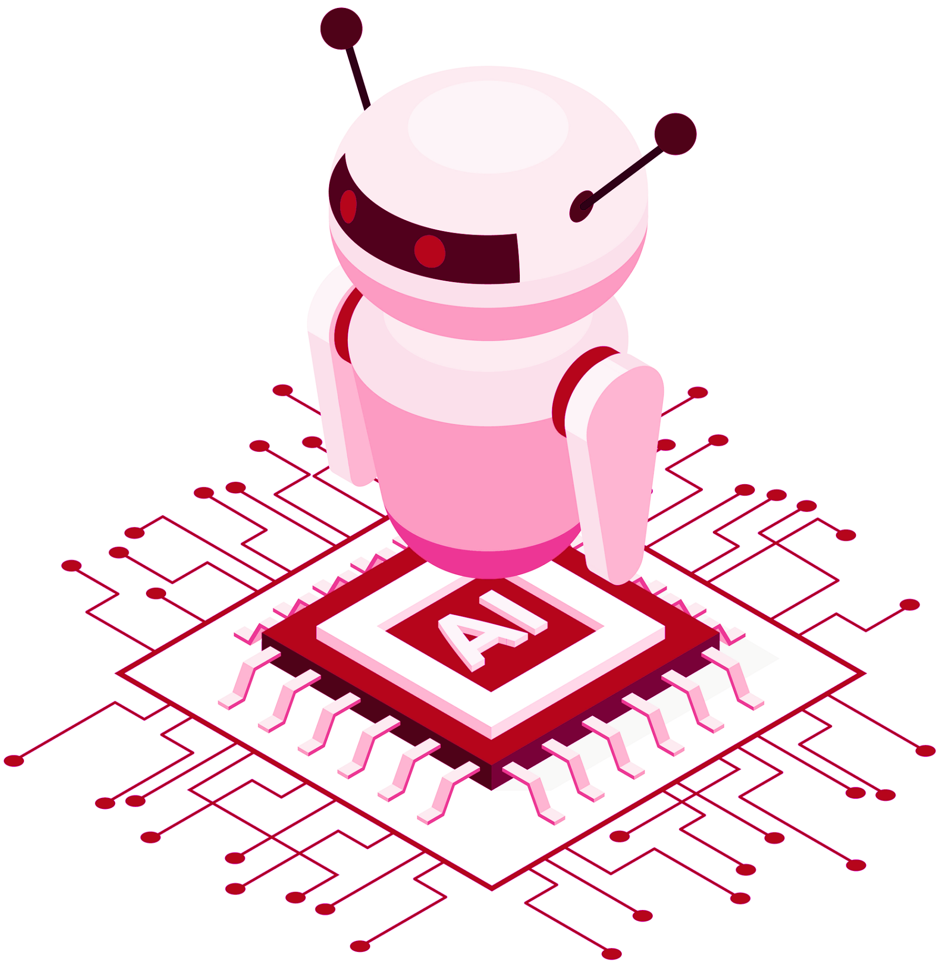

Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Homework 4:

Advanced Methods in RL

By:

Ashkan Majidi

400109984

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]

Experimantal setup (trainning configs) can be seen in notebooks.

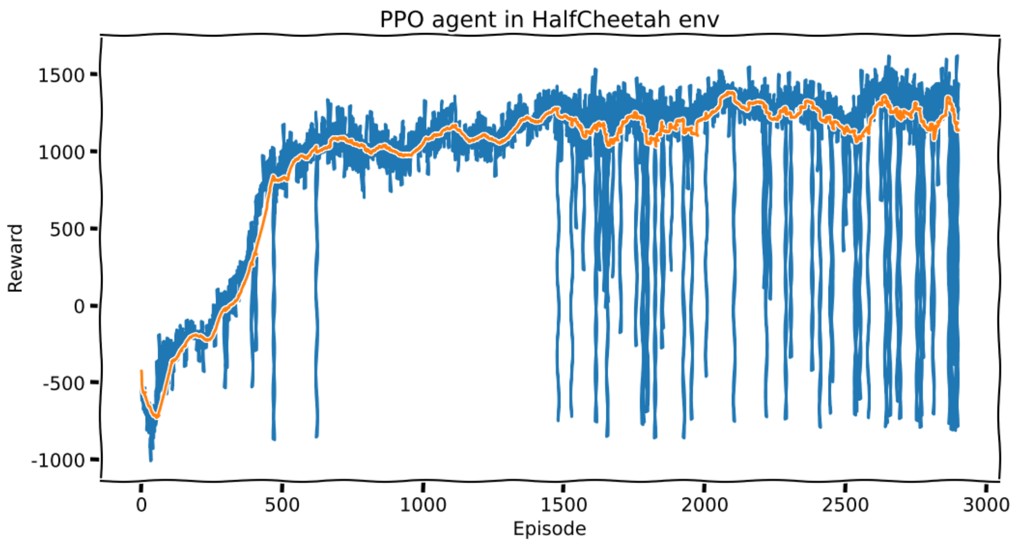

Figure 1: PPO (reward-to-go advantage) convergence curve

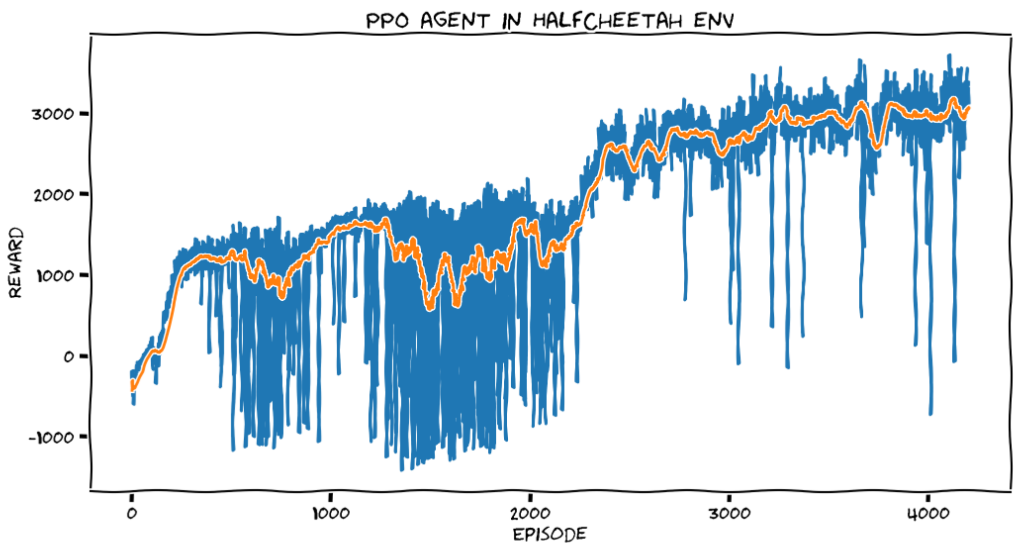

Figure 2: PPO (GAE advantage) convergence curve

1.1 Question 1:

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

Answer: In PPO, the actor and critic networks work together to optimize the policy. The actor network is responsible for selecting actions based on the current policy, which is parameterized by the network. It outputs a probability distribution over possible actions. The critic network evaluates the actions taken by the actor by estimating the value function, typically the expected return from a given state. The critic helps to reduce variance in the policy updates by providing a estimation of baseline. The actor is updated using the feedback from the critic, while the critic is updated based on the Temporal Difference (TD) error, which is the difference between the predicted value and the actual return.

As mentioned critic helps us lower the variance also note that we can apply ppo in different way because there are varient ways to compute advantage value in here we implemented two methods. First computing advantage in Generalised Advantage Estimation and second one computing using reward-to-go.

- **GAE**: it converges faster and achieves acceptable results. return around 3500.
- **reward-to-go**: it's convergence procedure is very slow in contrast to GAE.

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?

Answer: The stochastic nature of the actor network contributes to exploration by ensuring that actions are selected based on a probability distribution rather than being deterministic. This allows the agent to explore different states and actions during training. The entropy term in the objective function further encourages exploration by penalizing low entropy in the action distribution, which would otherwise indicate deterministic behavior. By maximizing the entropy, PPO encourages the agent to maintain a diverse set of policies and prevent it from prematurely converging to a suboptimal deterministic strategy. Together, the stochasticity of the actor and the entropy term allow PPO to balance exploration and exploitation effectively.

1.3 Question 3:

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

Answer: Key indicators to monitor when evaluating the performance of the PPO agent include:

- **Cumulative reward:** The total reward accumulated by the agent over time, which provides insight into the overall performance.
- **Value function loss:** The loss associated with the critic network, indicating how well the value function is approximating the true return.
- **Policy loss:** The loss from the actor network, showing how well the policy is being updated. This includes the clipped objective function to ensure stability.
- **Entropy:** The entropy of the action distribution, which helps to track the balance between exploration and exploitation. Low entropy indicates the policy is becoming deterministic.
- **Episode length:** The average number of steps per episode, which helps assess whether the agent is solving the task or failing early.
- **Learning rate:** The rate at which the agent's parameters are updated. Monitoring this ensures the agent is learning at an appropriate pace.

These indicators together help assess the stability and performance of the PPO agent over time.

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

As you can see DDPG algorithm has much more stable learning curve in contrast to PPO. Additionaly, it converges much faster than PPO. Experimental setup can be found in notebooks.

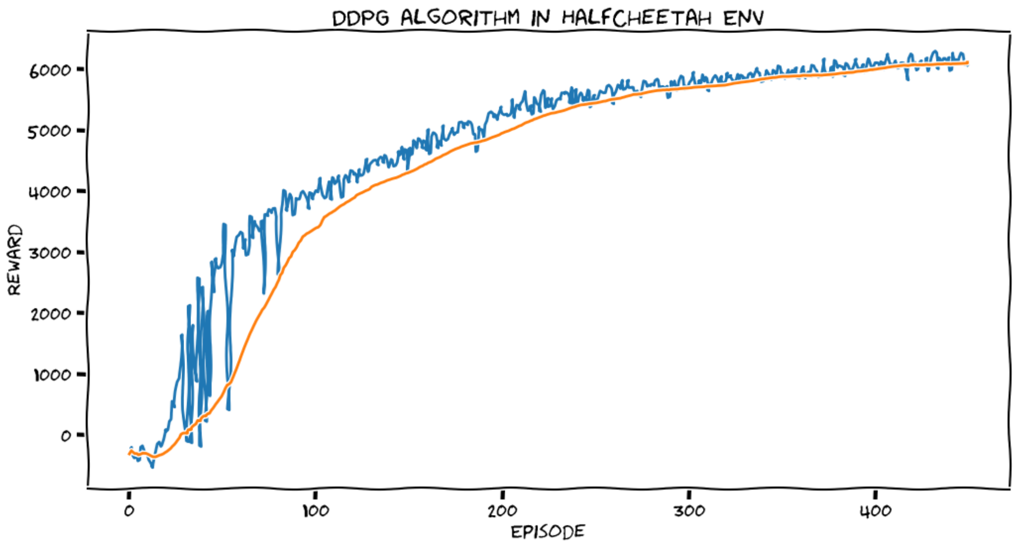

Figure 3: DDPG learning curve

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?

Answer: In DDPG, two types of noise are commonly used to encourage exploration:

- **Ornstein-Uhlenbeck noise:** This noise is specifically designed for continuous action spaces and is used to add temporally correlated noise to the action selection. The noise is generated by an Ornstein-Uhlenbeck process, which introduces a form of smooth randomness that encourages exploration in environments where actions have momentum. This helps the agent avoid getting stuck in local optima and improves exploration in high-dimensional spaces.
- **Gaussian noise:** Gaussian noise is added to the action by sampling from a normal distribution. This noise is independent at each time step and is typically used for environments where uncorrelated noise is required. While it is simple to implement, Gaussian noise does not maintain temporal correlation, which can be less effective in certain environments.

Both types of noise help the agent explore the action space, but Ornstein-Uhlenbeck noise is generally preferred in environments with continuous action spaces due to its temporally correlated nature, which better matches the dynamics of many real-world problems.

2.2 Question 2:

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

Answer: The main difference between PPO and DDPG in terms of using past experiences lies in how they handle experience replay and the learning process:

- **DDPG:** DDPG uses an experience replay buffer to store past experiences, allowing the agent to sample and learn from them multiple times. This approach helps break the correlation between consecutive experiences, which improves the stability and efficiency of the learning process. Additionally, DDPG utilizes target networks (both for the actor and critic) to stabilize training, which further leverages past experiences by providing more stable value estimates.
- **PPO:** PPO does not use experience replay. Instead, it uses on-policy learning, where each batch of data comes from the most recent interactions with the environment. This means that the agent learns from experiences gathered in a more recent time frame, which can lead to more stable and robust updates. PPO updates the policy through a clipped objective function to ensure that the policy changes are not too large.

Thus, DDPG benefits from the reuse of past experiences through experience replay, while PPO relies on learning from the most recent interactions, emphasizing stability through on-policy updates.

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

Among the implemented algorithms SAC have best performance in terms of stability, convergence speed and sample efficiency. Experimental setup (training configurations) can be found in notebook.

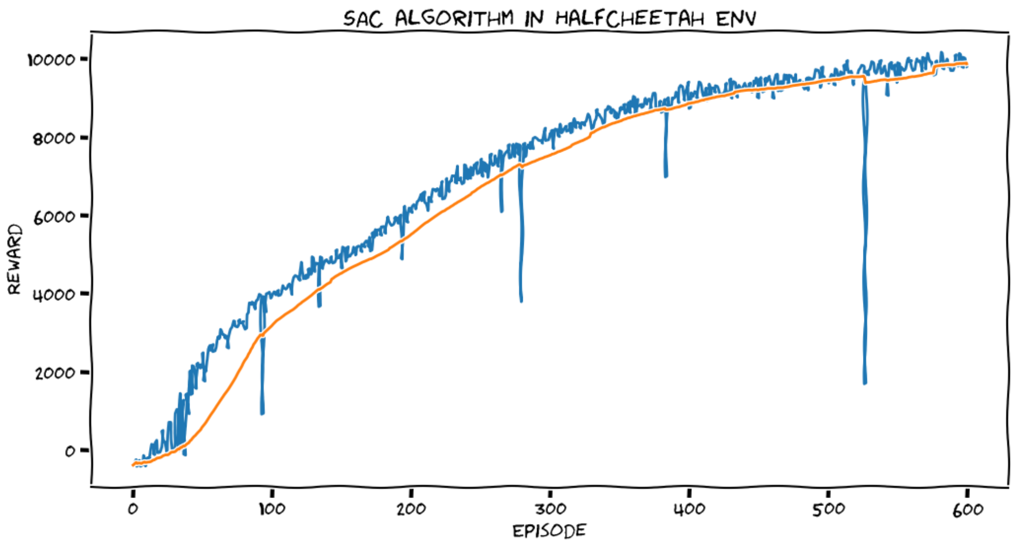

Figure 4: SAC learning curve

3.1 Question 1:

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

Answer: In SAC, two Q-networks (denoted as Q_1 and Q_2) are used to mitigate the overestimation bias of the Q-value estimates. This bias can arise when a single Q-network is used and leads to the agent overestimating the value of certain actions, which can negatively impact learning and stability. By using two Q-networks and taking the minimum of their estimates, SAC reduces the chance of this overestimation bias, leading to more accurate value function estimates. This technique helps improve the stability of the algorithm and leads to better performance in continuous action spaces.

3.2 Question 2:

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

Answer: The temperature parameter α in SAC controls the trade-off between exploration and exploitation by adjusting the importance of the entropy term in the objective function. A higher value of α encourages more exploration by assigning a higher weight to entropy, whereas a lower value encourages more exploitation by focusing on maximizing the reward. The benefit of using a dynamic α is that it allows the agent to adjust the balance between exploration and exploitation during training. Initially, a higher α promotes exploration, but as training progresses, α can decrease, encouraging the agent to focus more on exploiting the learned policy. This dynamic adjustment helps improve the learning process and ensures better convergence to an optimal policy.

3.3 Question 3:

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

Answer: In SAC, the difference between evaluation mode and training mode primarily lies in how the agent interacts with the environment and the updates to its networks:

- **Training Mode:** During training, the agent uses the stochastic policy and entropy regularization to encourage exploration. The policy network is updated based on the gradient of the objective function that includes both the expected return and the entropy term. The Q-networks and value function are updated using Temporal Difference (TD) error. Exploration is encouraged by adding stochasticity to the policy, and the agent interacts with the environment to collect experiences.
- **Evaluation Mode:** In evaluation mode, the agent uses the deterministic policy (the action with the highest value based on the current state) and does not actively encourage exploration. The networks are not updated during evaluation. This mode is used to assess the agent's performance and see how well it performs the task based on the learned policy without any further exploration or randomness.

The primary goal of evaluation mode is to assess the final performance of the agent, while training mode focuses on learning and improving the policy.

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?

Answer: In the `HalfCheetah` environment, SAC generally outperforms PPO and DDPG in terms of overall accumulated reward, training stability, and convergence speed. The reasons for this are as follows:

- **Training Stability:** SAC tends to be more stable during training due to its off-policy nature and the use of entropy regularization. This encourages exploration while ensuring stable policy improvement. It also uses a target value network and two Q-value networks, which further enhance stability.
- **Convergence Speed:** SAC converges faster than DDPG because it uses both value-based and policy-based methods, benefiting from a better balance between exploration and exploitation. This enables faster learning and more efficient exploration of the state space compared to DDPG, which relies on deterministic policies.
- **Accumulated Reward:** SAC typically achieves higher accumulated rewards than both PPO and DDPG. This is partly due to its exploration strategy and entropy maximization, which allows the agent to explore the environment more effectively.

Overall, SAC is better suited for the `HalfCheetah` environment, where continuous control tasks with high-dimensional action spaces require efficient exploration and stable training.

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?

Answer: The exploration strategies in PPO, DDPG, and SAC differ significantly in terms of how they balance exploration and exploitation:

- **PPO:** PPO uses stochastic policies and encourages exploration through entropy regularization. The objective function includes an entropy term, which penalizes deterministic behavior and promotes a diverse set of actions. This approach helps PPO maintain exploration during training while still allowing for exploitation of learned strategies.
- **DDPG:** DDPG uses deterministic policies and relies on noise injection to promote exploration. It typically adds Ornstein-Uhlenbeck noise to the action selection process to encourage exploration in continuous action spaces. This noise helps the agent explore different action sequences, but the deterministic policy can limit the diversity of the agent's behavior.
- **SAC:** SAC uses stochastic policies and incorporates entropy regularization as part of its objective. The agent is encouraged to explore by maximizing the entropy of its policy, which leads to more

diverse action selection. This exploration strategy is more robust than DDPG's noise injection, and it allows SAC to better balance exploration and exploitation, especially in continuous action spaces.

SAC's approach to exploration is more sophisticated and robust, allowing for better exploration of the state-action space compared to PPO and DDPG, which rely on entropy and noise injection, respectively.

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?

Answer: The key advantages and disadvantages of each algorithm in terms of sample efficiency and stability are as follows:

- **PPO:**
  - **Advantages:** PPO is highly stable due to its clipped objective function, which ensures that policy updates do not change too drastically. This prevents large updates that could destabilize training.
  - **Disadvantages:** PPO is less sample-efficient than off-policy algorithms like DDPG and SAC, as it relies on on-policy data, meaning each experience can only be used once.
- **DDPG:**
  - **Advantages:** DDPG is more sample-efficient than PPO due to its off-policy nature and the use of experience replay, which allows for reusing past experiences.
  - **Disadvantages:** DDPG is less stable compared to PPO and SAC. The deterministic policy can lead to issues with exploration, and the algorithm can be sensitive to hyperparameters such as the learning rate.
- **SAC:**
  - **Advantages:** SAC is the most sample-efficient and stable of the three. It uses off-policy learning, experience replay, and entropy regularization to ensure stable learning and efficient exploration. Its stochastic policy and entropy maximization allow it to avoid getting stuck in local optima.
  - **Disadvantages:** SAC's complexity in terms of the number of networks and the need for careful tuning of the temperature (entropy coefficient) can make it more challenging to implement and tune.

In terms of trade-offs, PPO offers better stability at the cost of sample efficiency, DDPG is more sample-efficient but less stable, and SAC strikes a balance, being both stable and efficient but at the cost of complexity.

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?

Answer: In terms of ease of tuning:

- **PPO:** PPO is generally easier to tune compared to DDPG and SAC. It has fewer hyperparameters to optimize, and its stability due to the clipped objective function helps reduce the impact of poor hyperparameter choices. Critical hyperparameters for PPO include the learning rate, the clipping parameter (epsilon), and the entropy coefficient.
- **DDPG:** DDPG is more sensitive to hyperparameter choices and can be harder to tune. The most critical parameters include the learning rate, the noise scale for exploration, the discount factor, and the size of the experience replay buffer. Fine-tuning these parameters is essential to stabilize training.
- **SAC:** SAC is also sensitive to hyperparameters, particularly the learning rate, the entropy coefficient (temperature), and the target smoothing coefficient. Tuning the temperature is crucial, as it controls the trade-off between exploration and exploitation. SAC also requires careful tuning of the Q-function and value function learning rates.

Common tuning strategies include:

- **For PPO:** Start with default hyperparameters and gradually adjust the learning rate and epsilon. Monitoring the entropy term can help ensure sufficient exploration.
- **For DDPG:** Start with a small learning rate and gradually increase it. Pay attention to the noise scale, as too much noise can make the agent overly exploratory.
- **For SAC:** Carefully tune the entropy coefficient to maintain a balance between exploration and exploitation. Use a small learning rate and gradually increase the size of the experience replay buffer.

Overall, PPO is the easiest to tune, while DDPG and SAC require more careful attention to hyperparameters to ensure stable and efficient training.