Embedded
Machine
Intelligence
Lab

PPO and SAC Algorithms

Reinforcement Learning Presentation

Saman Khamesian - 03/26/2025

PPO and SAC Algorithms

Hierarchy of Reinforcement Learning Algorithms

- Reinforcement Learning (RL) is a branch of machine learning where agents learn by interacting with an environment.

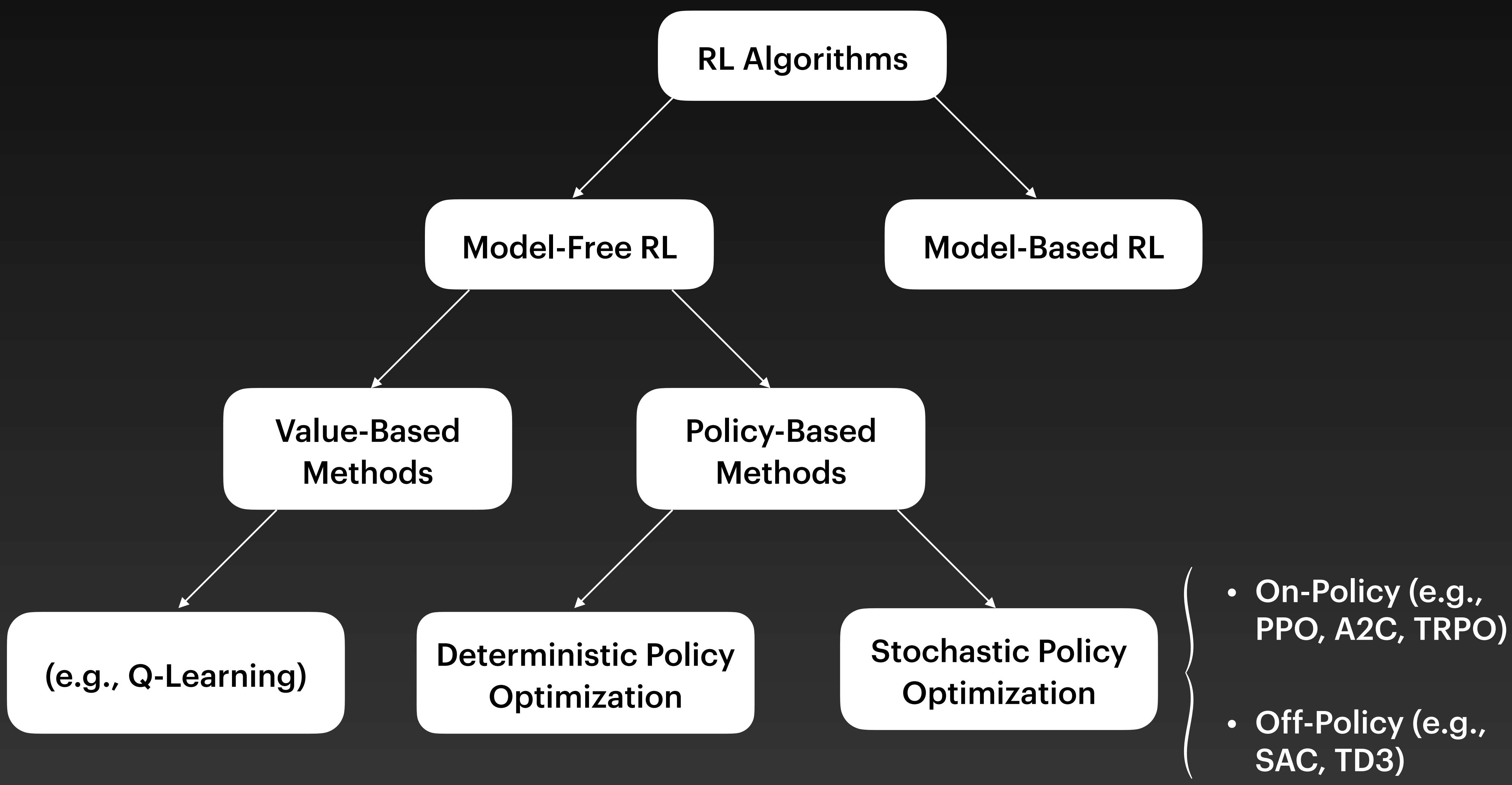
- RL algorithms can be broadly classified into:

  1. Model-Based RL (e.g., Dyna-Q, Monte Carlo Tree Search)

  2. Model-Free RL

PPO and SAC Algorithms

Hierarchy of Reinforcement Learning Algorithms



PPO and SAC Algorithms

Where Policy-Based Methods Work?

- Policy-based methods are well-suited for:

    1. Continuous or high-dimensional action spaces (e.g., robotics, autonomous vehicles)

    2. Learning a stochastic policy is beneficial (e.g., games, finance, dynamic decision-making)

    3. Complex simulations where value-based methods struggle.

    4. Smooth and gradual policy updates (e.g., healthcare, industrial automation)

PPO and SAC Algorithms

What is PPO?

- Proximal Policy Optimization (PPO) is a reinforcement learning algorithm designed for stability and efficiency.
- It belongs to the policy gradient family, meaning it directly optimizes the policy without relying on a value function.
- Introduces clipping in its loss function to prevent large updates, ensuring a smooth learning process.
- It is used in robotics, gaming, self-driving simulations, and healthcare.

PPO and SAC Algorithms

How PPO Works?

- The RL agent interacts with the environment and collects a batch of experiences:

$$(s_t, a_t, r_t, s_{t+1})$$

- where:
  - s_t = state at time t (e.g., robot position, stock prices, etc.).
  - a_t = action at time t (e.g., move left, buy stock, etc.).
  - r_t = reward received after taking the action.
  - s_{t+1} = new state after the action is applied.

PPO and SAC Algorithms

How PPO Works?

- PPO uses the advantage function to determine whether an action was better than expected:

$$A_t = Q(s_t, a_t) - V(s_t)$$

- where:
  - $Q(s_t, a_t)$ = expected return after taking action
  - $V(s_t)$ = expected return from state s_t
  - If $A_t > 0$, the action was beneficial and should be repeated.

PPO and SAC Algorithms

How PPO Works?

- PPO prevents large policy updates by clipping the objective function:

$$L(\theta) = \mathbb{E}_t \left[A_t \times \min \left(r_t(\theta),\ \text{clip}(r_t(\theta),\ 1-\epsilon,\ 1+\epsilon) \right) \right]$$

- where:
  - $r_t(\theta) = \dfrac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$ is the probability ratio of taking an action.
  - Clipping range $(1-\epsilon,\ 1+\epsilon)$ restrict updates:
    - If $r_t(\theta)$ moves too far from 1, the update is clipped.
    - This prevents the policy from changing too drastically in a single step.

PPO and SAC Algorithms

How PPO Works?

- Intuition Behind This:
    - If an action is much better than expected, PPO allows a policy update but caps it to a reasonable extent.
    - If an action is much worse, PPO discourages updating too aggressively.
    - This smooths out learning and prevents the policy from becoming too optimistic or too conservative.
    - This reduces aggressive exploration, keeping the policy updates within a safe zone.

PPO and SAC Algorithms

What is SAC?

- Soft Actor-Critic (SAC) is an off-policy reinforcement learning algorithm that focuses on continuous action control.
- Introduces entropy maximization to encourage exploration.
- Unlike PPO, SAC is an actor-critic method that maintains:
  - **Two Critic Networks:** to estimate Q-values accurately.
  - **One Policy (Actor) Network:** to generate actions.
  - **Temperature Parameter:** to balance exploration vs. exploitation.

PPO and SAC Algorithms

How SAC Works?

- The SAC Objective Function is defined as:

$$J(\pi) = \mathbb{E}_t \left[Q(s_t, a_t) - \alpha \log(\pi(a_t \mid s_t)) \right]$$

- where:
  - $Q(s_t, a_t)$ is the critic estimate of action value.
  - α is the temperature parameter, controlling the tradeoff between exploration and exploitation.
  - $\log(\pi(a_t \mid s_t))$ encourages diverse action selection to prevent premature convergence.
- This formula updates the **policies** using the current **Q-values.**

PPO and SAC Algorithms

How SAC Works?

- SAC maintains two Q-value (critic) networks to reduce overestimation bias commonly found in Q-learning approaches.
- The target Q-value is computed as:

$$y = r_t + \gamma \left(\min(Q_1(s_{t+1}, a_{t+1}), Q_2(s_{t+1}, a_{t+1}) - \alpha H(\pi)) \right)$$

- r_t is the reward at time t
- γ is the discount factor.
- Q_1 and Q_2 are the two critic networks, reducing overestimation bias.
- The entropy term $H(\pi)$ prevents the agent from being overly deterministic, encouraging exploration by making sure actions remain varied.
- This formula updates **Q-values** based on **current policies**.

PPO and SAC Algorithms

What is SAC?

- In RL, the Q-function $Q(s, a)$ estimates the expected return (reward) for taking an action a in state s.
- If the Q-values are overestimated, the policy may select suboptimal actions, leading to poor performance.
- Traditional Q-learning (like in DQN) can suffer from this because the agent always picks the maximum estimated Q-value, even when it is overestimated.
- Since one Q-network might overestimate, taking the minimum value prevents the policy from trusting an overoptimistic Q-value.
- This technique makes learning more stable by reducing bias.

PPO and SAC Algorithms

Key Differences Between PPO and SAC

Feature	PPO	SAC
Learning Type	On-Policy	Off-Policy
Exploration Strategy	Epsilon-Greedy	Entropy Maximization
Policy Type	Deterministic	Stochastic
Best for	Stable Training	Continuous Action Control
Computation Cost	Lower	Higher

PPO and SAC Algorithms

Examples of PPO Applications

Application	Description	Organization
OpenAI Five	Used PPO to train Dota 2 AI that defeated professional human players.	OpenAI
Humanoid Control (Robotics)	Trained humanoid robots to walk and balance using PPO.	DeepMind, OpenAI
Self-Driving Simulation	Used in autonomous driving simulations for decision-making.	Wayve AI
Healthcare: Personalized Treatment	Applied PPO to optimize treatment plans in dynamic healthcare settings.	MIT AI Lab
Autonomous Drones	PPO was used to train drones to navigate safely in complex environments.	Google AI

PPO and SAC Algorithms

Examples of SAC Applications

Application	Description	Organization
Tesla's Autopilot System (Simulation)	Used SAC in reinforcement learning simulations to improve driving policies.	Tesla
Dexterous Robotic Hands	SAC was used to train robotic hands to manipulate objects more naturally.	OpenAI, DeepMind
Energy Optimization in Smart Grids	SAC was used to optimize energy distribution dynamically.	Google DeepMind
Autonomous Trading Bots	SAC was implemented to train financial trading algorithms.	JP Morgan AI Research
Insulin Dosing for Diabetes	SAC was tested in simulated environments for adaptive insulin dose recommendations.	Harvard Medical AI Research

PPO and SAC Algorithms

Choosing Between PPO and SAC

✅ Use PPO when:

- You need a stable and reliable RL agent.
- Your action space is discrete or small continuous.
- Computation efficiency is important.

✅ Use SAC when:

- You need to control continuous actions with high precision.
- You want better exploration and adaptability.

Thank you