

Mathematics for Machine Learning

— Continuous Optimization

Introduction to the Policy Gradient Trick

Joseph Chuang-Chieh Lin

Department of Computer Science & Engineering,
National Taiwan Ocean University

Fall 2025

Credits for the resource

- The slides are based on the textbooks and reference lectures:
 - *Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong: Mathematics for Machine Learning. Cambridge University Press. 2020.*
 - *Roger Grosse's Course Lectures on Neural Networks and Deep Learning*
(https://www.cs.toronto.edu/~rgrosse/courses/csc421_2019/).
- We could partially refer to the monograph:
Francesco Orabona: A Modern Introduction to Online Learning.
<https://arxiv.org/abs/1912.13213>

Outline

- 1 Markov Decision Process (MDP)
- 2 Policy Gradient

Outline

1 Markov Decision Process (MDP)

2 Policy Gradient

Reinforcement Learning (RL)

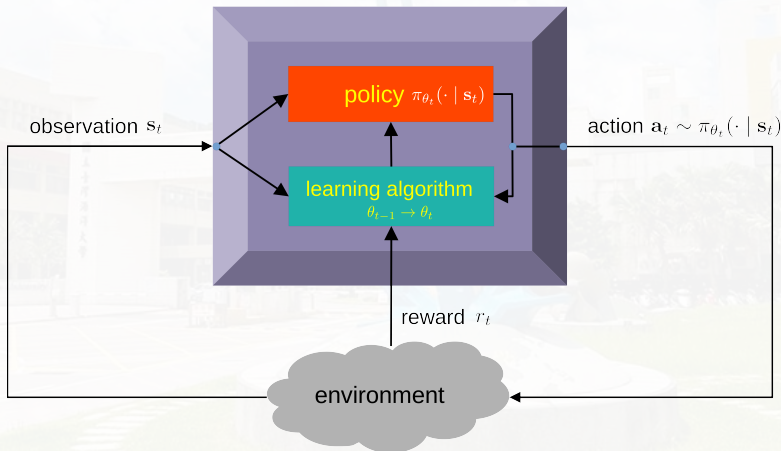
From *Wikipedia*:

- *Reinforcement learning (RL) is an interdisciplinary area of machine learning and optimal control concerned with how an intelligent agent should **take actions** in a **dynamic environment** in order to **maximize a reward signal**.*
- *Reinforcement learning is one of the three basic machine learning paradigms, alongside supervised learning and unsupervised learning.*
- Example of RL environments: [\[link\]](#).

RL Setting

- Each **agent** interacts with an **environment** (static or dynamic).
- In each time step t ,
 - the agent receives feedback or observations from the environment about the **state** \mathbf{s}_t .
 - the agent then takes an action \mathbf{a}_t which can affect the state ($\mathbf{s}_t \rightarrow \mathbf{s}_{t+1}$).
 - the agent receives the **reward** $r(\mathbf{s}_t, \mathbf{a}_t)$.
- Goal of the agent: learn a policy $\pi_{\theta}(\mathbf{a}_t \mid \mathbf{s}_t)$.
 - A distribution over the actions given the current state \mathbf{s}_t and the parameter θ .
 - θ : can be regarded as a machine learning model.

RL Setting



Markov Decision Process (MDP) (1/3)

- Markov decision process (MDP): an RL environment setting.
- Assumption: all information is encapsulated in the current state \mathbf{s}_t ; transitions are independent of past states.

MDP components

- initial state distribution $p(\mathbf{s}_0)$.
 - policy: $\pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t)$
 - transition prob.: $p(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t)$.
 - reward function: $r(\mathbf{s}_t, \mathbf{a}_t)$.
-
- We consider **fully observable** environment.
 - \mathbf{s}_t can be observed **directly**.

Markov Decision Process (MDP) (2/3)

- **Trajectory** or **rollout**: $\tau = (\mathbf{s}_0, \mathbf{a}_0, \mathbf{s}_1, \mathbf{a}_1, \dots, \mathbf{s}_T, \mathbf{a}_T)$
- Probability of a trajectory:

$$p(\tau) = p(\mathbf{s}_0) \pi_{\theta}(\mathbf{a}_0 \mid \mathbf{s}_0) p(\mathbf{s}_1 \mid \mathbf{s}_0, \mathbf{a}_0) \pi_{\theta}(\mathbf{a}_1 \mid \mathbf{s}_1) p(\mathbf{s}_2 \mid \mathbf{s}_1, \mathbf{a}_1) \cdots p(\mathbf{s}_T \mid \mathbf{s}_{T-1}, \mathbf{a}_{T-1}) \pi_{\theta}(\mathbf{a}_T \mid \mathbf{s}_T).$$

- Return for a trajectory: $r(\tau) = \sum_{t=0}^T r(\mathbf{s}_t, \mathbf{a}_t)$.
- **Goal**: Maximize $R := \mathbb{E}_{p(\tau)}[r(\tau)]$.
- The expectation is over the environment's dynamics and the policy, but we only have control over the **policy**.

Markov Decision Process (MDP) (3/3)

- ★ What's the issue when we compute $p(\tau)$ and R ?

Markov Decision Process (MDP) (3/3)

- ★ What's the issue when we compute $p(\tau)$ and R ?
- Each long trajectory could happen with extremely low probability.

Markov Decision Process (MDP) (3/3)

- ★ What's the issue when we compute $p(\tau)$ and R ?
- Each long trajectory could happen with extremely low probability.
- Problematic to derive $\frac{dR}{d\theta}$.

Outline

- 1 Markov Decision Process (MDP)
- 2 Policy Gradient

The Log-derivative Trick

Log-derivative Trick

$$\frac{\partial}{\partial \theta} \log p(\tau) = \frac{1}{p(\tau)} \frac{\partial}{\partial \theta} p(\tau).$$

- Hence, the gradient of the expected return turns out to be

$$\begin{aligned} \frac{\partial}{\partial \theta} \mathbb{E}_{p_{\theta}(\tau)}[r(\tau)] &= \frac{\partial}{\partial \theta} \sum_{\tau} r(\tau) p_{\theta}(\tau) = \sum_{\tau} r(\tau) \frac{\partial p_{\theta}(\tau)}{\partial \theta} \\ &= \sum_{\tau} r(\tau) p_{\theta}(\tau) \frac{\partial}{\partial \theta} \log p_{\theta}(\tau) \\ &= \mathbb{E}_{p_{\theta}(\tau)} \left[r(\tau) \frac{\partial}{\partial \theta} \log p_{\theta}(\tau) \right]. \end{aligned}$$

Estimate of the gradient

$$\frac{\partial}{\partial \theta} \mathbb{E}_{p_{\theta}(\tau)}[r(\tau)] = \mathbb{E}_{p_{\theta}(\tau)} \left[r(\tau) \frac{\partial}{\partial \theta} \log p_{\theta}(\tau) \right].$$

- Sampling the trajectories and rewards to have its estimate.

Estimate of the gradient

$$\frac{\partial}{\partial \theta} \mathbb{E}_{p_{\theta}(\tau)}[r(\tau)] = \mathbb{E}_{p_{\theta}(\tau)} \left[r(\tau) \frac{\partial}{\partial \theta} \log p_{\theta}(\tau) \right].$$

- Sampling the trajectories and rewards to have its estimate.
- Let's unpack the gradient of $\log p_{\theta}(\tau)$:

Estimate of the gradient

$$\frac{\partial}{\partial \theta} \mathbb{E}_{p_{\theta}(\tau)}[r(\tau)] = \mathbb{E}_{p_{\theta}(\tau)} \left[r(\tau) \frac{\partial}{\partial \theta} \log p_{\theta}(\tau) \right].$$

- Sampling the trajectories and rewards to have its estimate.
- Let's unpack the gradient of $\log p_{\theta}(\tau)$:

$$\begin{aligned} \frac{\partial}{\partial \theta} \log p_{\theta}(\tau) &= \frac{\partial}{\partial \theta} \log \left[p(\mathbf{s}_0) \prod_{t=0}^T \pi_{\theta}(\mathbf{a}_t \mid \mathbf{s}_t) \prod_{t=1}^T p(\mathbf{s}_t \mid \mathbf{s}_{t-1}, \mathbf{a}_{t-1}) \right] \\ &= \frac{\partial}{\partial \theta} \log \left(\prod_{t=0}^T \pi_{\theta}(\mathbf{a}_t \mid \mathbf{s}_t) \right) \\ &= \sum_{t=0}^T \frac{\partial}{\partial \theta} \log(\pi_{\theta}(\mathbf{a}_t \mid \mathbf{s}_t)). \end{aligned}$$

Update after T steps

- Let a trajectory be $\tau = (\mathbf{s}_0, \mathbf{a}_0, \dots, \mathbf{s}_T, \mathbf{a}_T)$ and define the episode return

$$r(\tau) = \sum_{k=0}^T r(\mathbf{s}_k, \mathbf{a}_k).$$

Since we have the gradient

$$\nabla_{\theta} \mathbb{E}_{p_{\theta}(\tau)}[r(\tau)] = \mathbb{E}_{p_{\theta}(\tau)} \left[r(\tau) \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t) \right].$$

- Issue:**
 - How to perform the expectation?

Update after a sequence of N trajectories

- Given trajectories $\tau^{(1)}, \tau^{(2)}, \dots, \tau^{(N)}$, each consists of T_i steps.

Update after a sequence of N trajectories

- Given trajectories $\tau^{(1)}, \tau^{(2)}, \dots, \tau^{(N)}$, each consists of T_i steps.

$$\nabla_{\theta} \mathbb{E}_{p_{\theta}(\tau)}[r(\tau)] \approx \frac{1}{N} \sum_{i=1}^N r(\tau^{(i)}) \sum_{t=0}^{T_i} \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t^{(i)} \mid \mathbf{s}_t^{(i)}).$$

Update after a sequence of N trajectories

- Given trajectories $\tau^{(1)}, \tau^{(2)}, \dots, \tau^{(N)}$, each consists of T_i steps.

$$\nabla_{\theta} \mathbb{E}_{p_{\theta}(\tau)}[r(\tau)] \approx \frac{1}{N} \sum_{i=1}^N r(\tau^{(i)}) \sum_{t=0}^{T_i} \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)}).$$

update rule: (η : the step-size)

$$\theta \leftarrow \theta + \eta \frac{1}{N} \sum_{i=1}^N r(\tau^{(i)}) \sum_{t=0}^{T_i} \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)}).$$

or a time-step-averaged alternative:

$$\theta \leftarrow \theta + \eta \frac{1}{\sum_{i=1}^N (T_i + 1)} \sum_{i=1}^N \sum_{t=0}^{T_i} r(\tau^{(i)}) \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)}).$$

A Monte-Carlo algorithm REINFORCE

REINFORCE [Ronald J. Williams 1992]

- While true
 - Sample a trajectory $\tau = (\mathbf{s}_0, \mathbf{a}_0, \dots, \mathbf{s}_T, \mathbf{a}_T)$ and define the episode return

$$r(\tau) = \sum_{k=0}^T r(\mathbf{s}_k, \mathbf{a}_k).$$

- perform the **single-episode update**: (η : the step-size)

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \underbrace{\eta r(\tau) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t | \mathbf{s}_t)}_{\text{accumulated with } t \text{ as } \Delta \boldsymbol{\theta}} \quad \text{for } t = 0, 1, \dots, T.$$

★ This resembles the stochastic gradient ascent on R .

- Recall the gradient is

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{p_{\boldsymbol{\theta}}(\tau)}[r(\tau)] = \mathbb{E}_{p_{\boldsymbol{\theta}}(\tau)} \left[r(\tau) \sum_{t=0}^T \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t | \mathbf{s}_t) \right].$$

Credit only future rewards (still unbiased)

Split the total return at time t into past and future parts:

$$r(\tau) = \underbrace{\sum_{k=0}^{t-1} r(\mathbf{s}_k, \mathbf{a}_k)}_{P_t} + \underbrace{\sum_{k=t}^T r(\mathbf{s}_k, \mathbf{a}_k)}_{F_t =: r_t(\tau)}.$$

Then

$$\mathbb{E}[P_t \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t)] = 0, \quad \text{since}$$

$$\begin{aligned} \mathbb{E}_{\mathbf{a}_t \sim \pi_{\theta}(\cdot | \mathbf{s}_t)}[\nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t)] &= \sum_{\mathbf{a}} \pi_{\theta}(\mathbf{a} | \mathbf{s}_t) \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t)] \\ &= \nabla_{\theta} \sum_{\mathbf{a}} \pi_{\theta}(\mathbf{a} | \mathbf{s}_t) = 0. \end{aligned}$$

Hence we may drop P_t to have the gradient without bias.

Update rule: $\theta \leftarrow \theta + \alpha r_t(\tau) \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t), \quad t = 0, 1, \dots, T.$

Discussions on the policy gradient approach

- Model-free for the environment.
 - The agent doesn't try to fit *the model of the environment*.
- If a trajectory happens to be *good*, all the actions get reinforced.
- A trajectory is viewed as a random walk.
- One can consider time-discount reward $G_t := \sum_{k=t}^{T-1} \gamma^{k-t} r_{k+1}$, that is,
 $G_t = r_{t+1} + \gamma G_{t+1}$, $G_T = 0$.

Discussions