Policy Gradient Methods

Sunmook Choi felixchoi@korea.ac.kr

August 30, 2023

1 Introduction

Let's first recall the goal of solving MDPs. The goal of solving MDPs is to find an optimal policy so that the agent can make a decision at each state based on the policy. So far, we obtained an optimal policy indirectly from the estimated action-value function $Q(s, a)$. For example, after an iterative algorithm finding an optimal action-value function, an optimal policy is obtained as follow:

$$\pi(a|s) = \arg\max_a Q(s, a) \tag{1.1}$$

In particular, DQN uses a neural network in order to approximate the optimal action-value function based on Q-learning. In this section, on the other hand, we consider methods that learn a probability distribution (or a stochastic function) that approximates policy directly. By using the notation $\theta \in \mathbb{R}^d$ for the policy's parameter vector, we write the parameterized policy as

$$\pi_\theta(a|s) = Pr[A_t = a \,|\, S_t = s, \theta_t = \theta]. \tag{1.2}$$

Such methods are called *policy gradient methods*, since the parameters of policy is updated directly using gradient *ascent* method. The goal of policy gradient methods is to *maximize the total reward* $r(\tau)$, where τ is a sampled trajectory obtained by the policy π_θ. To be specific, a sampled trajectory τ can be written as below:

$$\tau = (s_0, a_0, r_1, s_1, a_1, r_2, s_2, \ldots, s_{T-1}, a_{T-1}, r_T, s_T) \tag{1.3}$$

For the trajectory τ, the total reward is $r(\tau) = \sum_{t=1}^{T} r_t$. Moreover, by using Markov property, the probability of the trajectory τ can be reduced as below:

$$p_\theta(\tau) = p(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t)\, p(s_{t+1}|s_t, a_t). \tag{1.4}$$

The objective function $J(\theta)$ would be the expected total reward which should be maximized:

$$J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)}[r(\tau)] \tag{1.5}$$

We can update the parameter vector using gradient-ascent method with small step size α:

$$\theta \leftarrow \theta + \alpha \cdot \nabla_\theta J(\theta) \tag{1.6}$$

2 Policy Gradient Theorem

To use the gradient ascent method, it should be easy to differentiate the objective function. The following theorem enables us to compute the gradient of $J(\theta)$ and to approximate it easily by Monte Carlo methods.

Theorem 1 (Policy Gradient Theorem)**.**

$$\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_{\tau\sim p_\theta(\tau)}[r(\tau)] = \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[r(\tau)\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t|s_t)\right] \tag{2.1}$$

Proof. First, observe the following equation:

$$\nabla_\theta \log p_\theta(\tau) = \nabla_\theta\Big(\log p_0(s_0) + \sum_{t=0}^{T-1}\log \pi_\theta(a_t|s_t) + \sum_{t=0}^{T-1}\log p(s_{t+1}|s_t,a_t)\Big) = \sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t|s_t)$$

where $p_0(s_0)$ is the probability of an episode that starts at s_0. Then, we obtain the following:

$$\begin{aligned}
\nabla_\theta\, \mathbb{E}_{\tau\sim p_\theta(\tau)}[r(\tau)] &= \nabla_\theta \int r(\tau)\, p_\theta(\tau)\, d\tau && (2.2)\\
&= \int \nabla_\theta p_\theta(\tau)\, r(\tau)\, d\tau && (2.3)\\
&= \int \frac{\nabla_\theta p_\theta(\tau)}{p_\theta(\tau)} p_\theta(\tau)\, r(\tau)\, d\tau && (2.4)\\
&= \int p_\theta(\tau)\, r(\tau)\, \nabla_\theta \log p_\theta(\tau)\, d\tau && (2.5)\\
&= \mathbb{E}_{\tau\sim p_\theta(\tau)}\big[r(\tau)\,\nabla_\theta \log p_\theta(\tau)\big] && (2.6)\\
&= \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[r(\tau)\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t|s_t)\right] && (2.7)
\end{aligned}$$

This completes the proof. □

This theorem implies that the gradient of the objective can be approximated by sampling a large number of trajectories and taking their average (Monte Carlo estimation). To be specific, after executing M trajectories, say $\tau_1, \tau_2, \ldots, \tau_M$, the approximated gradient from these trajectories can be computed as

$$g_\theta = \frac{1}{M}\sum_{i=1}^{M} r(\tau_i)\sum_{t=0}^{T_i-1}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i).$$

Notice that the estimator is indeed unbiased. Hence, the update equation of θ would be

$$\theta \;\leftarrow\; \theta + \alpha\, g_\theta \;\approx\; \theta + \alpha\, \nabla_\theta J(\theta).$$

3 Reducing Variance

We have found a way to approximate the parameterized policy via policy gradient theorem. However, there is a big disadvantage of this approximation. The total reward $r(\tau)$ would add high

variance because the length of each episode varies resulting in very oscillating total rewards (erratic trajectories). Therefore we need to find a way to reduce the variance.

3.1 Causality Problem

Recall the gradient of our objective. For a sampled trajectory $\tau = (s_0, a_0, r_1, s_1, \ldots, s_T)$, the total sum of rewards is $r(\tau) = \sum_{t=0}^{T-1} r_{t+1}$ and the gradient of our objective is

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[r(\tau) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \right].$$

Notice that there is a summation over time step t. In the summation, at each time step t, the total reward is always multiplied to the gradient of the log-policy. However, while a trajectory is simulated, policy at time t' cannot affect reward at time t when $t < t'$. That is, $\sum_{t=0}^{k} r_{t+1}$ does not affect the actions $a_{k+1}, a_{k+2}, \ldots, a_{T-1}$. This property is called causality problem. Using this property, we can reduce the variance of gradient estimates as below:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\left(\sum_{t=0}^{T-1} r_{t+1} \right) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \right] \tag{3.1}$$

$$= \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=0}^{T-1} \left(\sum_{t'=t}^{T-1} r_{t'+1} \right) \nabla_\theta \log \pi_\theta(a_t|s_t) \right] \tag{3.2}$$

We should notice that it is nor related to Markov property.

3.2 Discount Factor γ

Moreover, from Eq. 3.2, by introducing discount factor $\gamma \in [0, 1]$, we can further reduce the variance. That is, we are slightly changing, or redefining, the goal of policy gradient method.

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=0}^{T-1} \left(\sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'+1} \right) \nabla_\theta \log \pi_\theta(a_t|s_t) \right]$$

$$= \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=0}^{T-1} G_t \nabla_\theta \log \pi_\theta(a_t|s_t) \right]$$

Recall that the return G_t is the discounted total reward defined by $G_t = \sum_{k=t}^{T-1} \gamma^{k-t} r_{k+1}$.

4 REINFORCE

REINFORCE is a reinforcement learning algorithm based on Monte Carlo policy gradient methods. It updates the policy parameters using the gradient defined in Section 3.2. The pseudocode of REINFORCE is described in Algorithm 4.1.

Algorithm 4.1 REINFORCE: Monte-Carlo Policy Gradient Control (episodic)

Input: a differentiable policy parameterization π_θ

Parameter: step size $\alpha > 0$

Initialize policy parameter $\theta \in \mathbb{R}^d$

1: **while True** (for each episode) **do**

2: Generate an episode $S_0, A_0, R_1, S_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$ following π_θ

3: Approximate the gradient of the objective function $J(\theta)$

$$g_\theta := \sum_{t=0}^{T-1} G_t \nabla_\theta \log \pi_\theta(a_t|s_t) \approx \nabla_\theta J(\theta)$$

4: Update policy parameters to maximize $J(\theta)$

$$\theta \leftarrow \theta + \alpha g_\theta \approx \theta + \alpha \nabla_\theta J(\theta)$$

5: **end while**

5 REINFORCE with Baseline

In this section, we want to reduce the variance of the estimator of $\nabla_\theta J(\theta)$. Consider an arbitrary *baseline* $b(s)$, which is any function that does not vary with actions.

Theorem 2. *For any baseline $b(\cdot)$ that does not depend on actions, we have*

$$\mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=0}^{T-1} b(s_t) \nabla_\theta \log \pi_\theta(a_t|s_t) \right] = 0. \tag{5.1}$$

Furthermore, we obtain another unbiased estimator of $\nabla_\theta J(\theta)$:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=0}^{T-1} \Big(G_t - b(s_t) \Big) \nabla_\theta \log \pi_\theta(a_t|s_t) \right]. \tag{5.2}$$

Proof. In the proof, we use the following trick:

$$\begin{aligned} \mathbb{E}[f(X,Y)] &= \iint p(x,y)\, f(x,y)\, dxdy \\ &= \iint f(x,y)\, p(x)\, p(y|x)\, dxdy \\ &= \int \left(\int f(x,y)\, p(y|x)\, dx \right) p(x)\, dx \\ &= \mathbb{E}_X \left[\mathbb{E}_{Y|X}[f(x,Y) \mid X = x] \right] \end{aligned} \tag{5.3}$$

For brevity, let $g_\theta(\tau) = \sum_{t=0}^{T-1} b(s_t) \nabla_\theta \log \pi_\theta(a_t|s_t)$. Since $p_\theta(\tau)$ is a joint probability of a trajectory

τ, we have

$$\mathbb{E}_{\tau\sim p_\theta(\tau)}\big[g_\theta(\tau)\big] = \mathbb{E}_{s_0\sim\rho_0(\cdot)}\big[\mathbb{E}_{a_0\sim\pi_\theta(\cdot|s_0)}\big[\mathbb{E}_{(s_1,a_1,r_2,\dots)\sim p_\theta(\cdot|s_0,a_0)}\big[g_\theta(\tau)\mid s_0,a_0\big]\mid s_0\big]\big] \tag{5.4}$$

$$= \cdots$$

$$= \mathbb{E}_{s_0}\big[\mathbb{E}_{a_0|s_0}\big[\mathbb{E}_{s_1|s_0,a_0}\big[\mathbb{E}_{a_1|s_1}\cdots\big[\mathbb{E}_{s_{T-1}|s_{T-2},a_{T-2}}\big[\mathbb{E}_{a_{T-1}|s_{T-1}}\big[g_\theta(\tau)\mid s_{T-1}\big]\mid s_{T-2},a_{T-2}\big]\cdots\mid s_1\big]\mid s_0,a_0\big]\mid s_0\big] \tag{5.5}$$

Consider the term $\mathbb{E}_{s_{T-1}|s_{T-2},a_{T-2}}\big[\mathbb{E}_{a_{T-1}|s_{T-1}}\big[g_\theta(\tau)\mid s_{T-1}\big]\mid s_{T-2},a_{T-2}\big]$. It is equal to the following:

$$\mathbb{E}_{s_{T-1}|s_{T-2},a_{T-2}}\left[\sum_{t=0}^{T-2} b(s_t)\nabla_\theta\log\pi_\theta(a_t|s_t) + b(s_{T-1})\,\mathbb{E}_{a_{T-1}|s_{T-1}}\big[\nabla_\theta\log\pi_\theta(a_{T-1}|s_{T-1})\mid s_{T-1}\big]\mid s_{T-2},a_{T-2}\right] \tag{5.6}$$

In the above equation, we first focus on the term $\mathbb{E}_{a_{T-1}|s_{T-1}}\big[\nabla_\theta\log\pi_\theta(a_{T-1}|s_{T-1})\mid s_{T-1}\big]$.

$$\mathbb{E}_{a_{T-1}|s_{T-1}}\big[\nabla_\theta\log\pi_\theta(a_{T-1}|s_{T-1})\mid s_{T-1}\big] = \sum_{a_{T-1}}\pi_\theta(a_{T-1}|s_{T-1})\nabla_\theta\log\pi_\theta(a_{T-1}|s_{T-1}) \tag{5.7}$$

$$= \sum_{a_{T-1}}\nabla_\theta\,\pi_\theta(a_{T-1}|s_{T-1}) \tag{5.8}$$

$$= \nabla_\theta\sum_{a_{T-1}}\pi_\theta(a_{T-1}|s_{T-1}) \tag{5.9}$$

$$= \nabla_\theta 1 = 0. \tag{5.10}$$

Therefore, the term (5.6) reduces to

$$\mathbb{E}_{s_{T-1}|s_{T-2},a_{T-2}}\left[\sum_{t=0}^{T-2} b(s_t)\nabla_\theta\log\pi_\theta(a_t|s_t)\mid s_{T-2},a_{T-2}\right] = \sum_{t=0}^{T-2} b(s_t)\nabla_\theta\log\pi_\theta(a_t|s_t) \tag{5.11}$$

because it does not contain s_{T-1}. By repeating this process, we can remove all expectations and the summations. Therefore, we conclude that $\mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\sum_{t=0}^{T-1} b(s_t)\nabla_\theta\log\pi_\theta(a_t|s_t)\right] = 0$. □

The theorem gives us another unbiased estimator so that we need to find a good baseline $b(\cdot)$. We can think of the state-value function $V(s)$ as a baseline because it does not depend on actions. Moreover, since the discounted return G_t is an unbiased estimator of $V(s_t)$, the state-value function would be a good baseline, reducing the variance of Eq. 5.2. The model, REINFORCE with baseline, becomes a fundamental algorithm for actor-critic methods. In actor-critic method, there are two networks: the actor network and the critic network. The actor network corresponds to the policy network as in policy gradient algorithms. The critic network corresponds to the value network, which is the approximation of the state-value function working as a baseline. We also describe the pseudocode of REINFORCE with baseline in Algorithm 5.1.

References

[1] Andrew Barto and Richard S. Sutton, *Reinforcement Learning: An Introduction* (2nd ed.), The MIT Press, 2018.

Algorithm 5.1 REINFORCE with baseline

Input: a differentiable policy parameterization π_θ, a baseline b

Parameter: step size $\alpha > 0$

Initialize policy parameter $\theta \in \mathbb{R}^d$

1: **while True** (for each episode) **do**

2: Generate an episode $S_0, A_0, R_1, S_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$ following π_θ

3: Approximate the gradient of the objective function $J(\theta)$

$$g_\theta := \sum_{t=0}^{T-1} \big(G_t - b(s_t)\big) \nabla_\theta \log \pi_\theta(a_t|s_t) \approx \nabla_\theta J(\theta)$$

4: Update policy parameters to maximize $J(\theta)$

$$\theta \leftarrow \theta + \alpha g_\theta \approx \theta + \alpha \nabla_\theta J(\theta)$$

5: **end while**