CS229 Lecture notes

Tengyu Ma

Part XV
Policy Gradient (REINFORCE)

We will present a model-free algorithm called REINFORCE that does not require the notion of value functions and Q functions. It turns out to be more convenient to introduce REINFORCE in the finite horizon case, which will be assumed throughout this note: we use $\tau = (s_0, a_0, \ldots, s_{T-1}, a_{T-1}, s_T)$ to denote a trajectory, where $T < \infty$ is the length of the trajectory. Moreover, REINFORCE only applies to learning a **randomized policy**. We use $\pi_\theta(a|s)$ to denote the probability of the policy π_θ outputting the action a at state s. The other notations will be the same as in previous lecture notes.

The advantage of applying REINFORCE is that we only need to assume that we can sample from the transition probabilities $\{P_{sa}\}$ and can query the reward function $R(s, a)$ at state s and action a,[1] but we do not need to know the analytical form of the transition probabilities or the reward function. We do not explicitly learn the transition probabilities or the reward function either.

Let s_0 be sampled from some distribution μ. We consider optimizing the expected total payoff of the policy π_θ over the parameter θ defined as.

$$\eta(\theta) \triangleq \mathrm{E}\left[\sum_{t=0}^{T-1} \gamma^t R(s_t, a_t)\right] \tag{1}$$

Recall that $s_t \sim P_{s_{t-1}a_{t-1}}$ and $a_t \sim \pi_\theta(\cdot|s_t)$. Also note that $\eta(\theta) = \mathrm{E}_{s_0 \sim P}\left[V^{\pi_\theta}(s_0)\right]$ if we ignore the difference between finite and infinite horizon.

[1] In this notes we will work with the general setting where the reward depends on both the state and the action.

We aim to use gradient ascent to maximize $\eta(\theta)$. The main challenge we face here is to compute (or estimate) the gradient of $\eta(\theta)$ without the knowledge of the form of the reward function and the transition probabilities.

Let $P_\theta(\tau)$ denote the distribution of τ (generated by the policy π_θ), and let $f(\tau) = \sum_{t=0}^{T-1} \gamma^t R(s_t, a_t)$. We can rewrite $\eta(\theta)$ as

$$\eta(\theta) = \mathrm{E}_{\tau \sim P_\theta}[f(\tau)] \tag{2}$$

We face a similar situations in the variational auto-encoder (VAE) setting covered in the previous lectures, where the we need to take the gradient w.r.t to a variable that shows up under the expectation — the distribution P_θ depends on θ. Recall that in VAE, we used the re-parametrization techniques to address this problem. However it does not apply here because we do know not how to compute the gradient of the function f. (We only have an efficient way to evaluate the function f by taking a weighted sum of the observed rewards, but we do not necessarily know the reward function itself to compute the gradient.)

The REINFORCE algorithm uses an another approach to estimate the gradient of $\eta(\theta)$. We start with the following derivation:

$$\begin{aligned}
\nabla_\theta \mathrm{E}_{\tau \sim P_\theta}[f(\tau)] &= \nabla_\theta \int P_\theta(\tau) f(\tau) d\tau \\
&= \int \nabla_\theta (P_\theta(\tau) f(\tau)) d\tau \quad \text{(swap integration with gradient)} \\
&= \int (\nabla_\theta P_\theta(\tau)) f(\tau) d\tau \quad \text{(becaue } f \text{ does not depend on } \theta) \\
&= \int P_\theta(\tau) (\nabla_\theta \log P_\theta(\tau)) f(\tau) d\tau \\
&\qquad\qquad \text{(because } \nabla \log P_\theta(\tau) = \tfrac{\nabla P_\theta(\tau)}{P_\theta(\tau)}\text{)} \\
&= \mathrm{E}_{\tau \sim P_\theta}[(\nabla_\theta \log P_\theta(\tau)) f(\tau)] \qquad (3)
\end{aligned}$$

Now we have a sample-based estimator for $\nabla_\theta \mathrm{E}_{\tau \sim P_\theta}[f(\tau)]$. Let $\tau^{(1)}, \ldots, \tau^{(n)}$ be n empirical samples from P_θ (which are obtained by running the policy π_θ for n times, with T steps for each run). We can estimate the gradient of $\eta(\theta)$ by

$$\begin{aligned}
\nabla_\theta \mathrm{E}_{\tau \sim P_\theta}[f(\tau)] &= \mathrm{E}_{\tau \sim P_\theta}[(\nabla_\theta \log P_\theta(\tau)) f(\tau)] \qquad (4) \\
&\approx \frac{1}{n} \sum_{i=1}^{n} (\nabla_\theta \log P_\theta(\tau^{(i)})) f(\tau^{(i)}) \qquad (5)
\end{aligned}$$

The next question is how to compute $\log P_\theta(\tau)$. We derive an analytical formula for $\log P_\theta(\tau)$ and compute its gradient w.r.t θ (using auto-differentiation). Using the definition of τ, we have

$$P_\theta(\tau) = \mu(s_0)\pi_\theta(a_0|s_0)P_{s_0a_0}(s_1)\pi_\theta(a_1|s_1)P_{s_1a_1}(s_2)\cdots P_{s_{T-1}a_{T-1}}(s_T) \tag{6}$$

Here recall that μ to used to denote the density of the distribution of s_0. It follows that

$$\begin{aligned}\log P_\theta(\tau) = \log \mu(s_0) + \log \pi_\theta(a_0|s_0) + \log P_{s_0a_0}(s_1) + \log \pi_\theta(a_1|s_1) \\ + \log P_{s_1a_1}(s_2) + \cdots + \log P_{s_{T-1}a_{T-1}}(s_T)\end{aligned} \tag{7}$$

Taking gradient w.r.t to θ, we obtain

$$\nabla_\theta \log P_\theta(\tau) = \nabla_\theta \log \pi_\theta(a_0|s_0) + \nabla_\theta \log \pi_\theta(a_1|s_1) + \cdots + \nabla_\theta \log \pi_\theta(a_{T-1}|s_{T-1})$$

Note that many of the terms disappear because they don't depend on θ and thus have zero gradients. (This is somewhat important — we don't know how to evaluate those terms such as $\log P_{s_0a_0}(s_1)$ because we don't have access to the transition probabilities, but luckily those terms have zero gradients!)

Plugging the equation above into equation (4), we conclude that

$$\begin{aligned}\nabla_\theta \eta(\theta) = \nabla_\theta \mathrm{E}_{\tau\sim P_\theta}\left[f(\tau)\right] &= \mathrm{E}_{\tau\sim P_\theta}\left[\left(\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t|s_t)\right)\cdot f(\tau)\right] \\ &= \mathrm{E}_{\tau\sim P_\theta}\left[\left(\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t|s_t)\right)\cdot\left(\sum_{t=0}^{T-1}\gamma^t R(s_t,a_t)\right)\right]\end{aligned} \tag{8}$$

We estimate the RHS of the equation above by empirical sample trajectories, and the estimate is unbiased. The vanilla REINFORCE algorithm iteratively updates the parameter by gradient ascent using the estimated gradients.

Interpretation of the policy gradient formula (8)**.** The quantity $\nabla_\theta P_\theta(\tau) = \sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t|s_t)$ is intuitively the direction of the change of θ that will make the trajectory τ more likely to occur (or increase the probability of choosing action $a_0, \ldots, a_{t-1}$), and $f(\tau)$ is the total payoff of this trajectory. Thus, by taking a gradient step, intuitively we are trying to improve the likelihood of all the trajectories, but with a different emphasis or weight for each τ (or for each set of actions $a_0, a_1, \ldots, a_{t-1}$). If τ is very rewarding (that is, $f(\tau)$ is large), we try very hard to move in the direction

that can increase the probability of the trajectory τ (or the direction that increases the probability of choosing $a_0, \ldots, a_{t-1}$), and if τ has low payoff, we try less hard with a smaller weight.

An interesting fact that follows from formula (3) is that

$$\mathrm{E}_{\tau \sim P_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\right] = 0 \tag{9}$$

To see this, we take $f(\tau) = 1$ (that is, the reward is always a constant), then the LHS of (8) is zero because the payoff is always a fixed constant $\sum_{t=0}^{T} \gamma^t$. Thus the RHS of (8) is also zero, which implies (9).

In fact, one can verify that $\mathrm{E}_{a_t \sim \pi_\theta(\cdot|s_t)} \nabla_\theta \log \pi_\theta(a_t|s_t) = 0$ for any fixed t and s_t.[2] This fact has two consequences. First, we can simplify formula (8) to

$$\begin{aligned}
\nabla_\theta \eta(\theta) &= \sum_{t=0}^{T-1} \mathrm{E}_{\tau \sim P_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \left(\sum_{j=0}^{T-1} \gamma^j R(s_j, a_j)\right)\right] \\
&= \sum_{t=0}^{T-1} \mathrm{E}_{\tau \sim P_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \left(\sum_{j \geq t}^{T-1} \gamma^j R(s_j, a_j)\right)\right]
\end{aligned} \tag{10}$$

where the second equality follows from

$$\begin{aligned}
&\mathrm{E}_{\tau \sim P_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \left(\sum_{0 \leq j < t} \gamma^j R(s_j, a_j)\right)\right] \\
&= \mathrm{E}\left[\mathrm{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)|s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t\right] \cdot \left(\sum_{0 \leq j < t} \gamma^j R(s_j, a_j)\right)\right] \\
&= 0 \qquad \text{(because } \mathrm{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)|s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t\right] = 0\text{)}
\end{aligned}$$

Note that here we used the law of total expectation. The outer expectation in the second line above is over the randomness of $s_0, a_0, \ldots, a_{t-1}, s_t$, whereas the inner expectation is over the randomness of a_t (conditioned on $s_0, a_0, \ldots, a_{t-1}, s_t$.) We see that we've made the estimator slightly simpler. The second consequence of $\mathrm{E}_{a_t \sim \pi_\theta(\cdot|s_t)} \nabla_\theta \log \pi_\theta(a_t|s_t) = 0$ is the following: for any value $B(s_t)$ that only depends on s_t, it holds that

$$\begin{aligned}
&\mathrm{E}_{\tau \sim P_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot B(s_t)\right] \\
&= \mathrm{E}\left[\mathrm{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)|s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t\right] B(s_t)\right] \\
&= 0 \qquad \text{(because } \mathrm{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)|s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t\right] = 0\text{)}
\end{aligned}$$

[2] In general, it's true that $\mathrm{E}_{x \sim p_\theta}[\nabla \log p_\theta(x)] = 0$.

Again here we used the law of total expectation. The outer expectation in the second line above is over the randomness of $s_0, a_0, \ldots, a_{t-1}, s_t$, whereas the inner expectation is over the randomness of a_t (conditioned on $s_0, a_0, \ldots, a_{t-1}, s_t$.) It follows from equation (10) and the equation above that

$$\begin{aligned}\nabla_\theta \eta(\theta) &= \sum_{t=0}^{T-1} \mathrm{E}_{\tau \sim P_\theta} \left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \left(\sum_{j \geq t}^{T-1} \gamma^j R(s_j, a_j) - \gamma^t B(s_t) \right) \right] \\ &= \sum_{t=0}^{T-1} \mathrm{E}_{\tau \sim P_\theta} \left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \gamma^t \left(\sum_{j \geq t}^{T-1} \gamma^{j-t} R(s_j, a_j) - B(s_t) \right) \right] \end{aligned} \tag{11}$$

Therefore, we will get a different estimator for estimating the $\nabla \eta(\theta)$ with a difference choice of $B(\cdot)$. The benefit of introducing a proper $B(\cdot)$ — which is often referred to as a **baseline** — is that it helps reduce the variance of the estimator.[3] It turns out that a near optimal estimator would be the expected future payoff $\mathrm{E}\left[\sum_{j \geq t}^{T-1} \gamma^{j-t} R(s_j, a_j) | s_t\right]$, which is pretty much the same as the value function $V^{\pi_\theta}(s_t)$ (if we ignore the difference between finite and infinite horizon.) Here one could estimate the value function $V^{\pi_\theta}(\cdot)$ in a crude way, because its precise value doesn't influence the mean of the estimator but only the variance. This leads to a policy gradient algorithm with baselines stated in Algorithm 1.[4]

[3] As a heuristic but illustrating example, suppose for a fixed t, the future reward $\sum_{j \geq t}^{T-1} \gamma^{j-t} R(s_j, a_j)$ randomly takes two values $1000 + 1$ and $1000 - 2$ with equal probability, and the corresponding values for $\nabla_\theta \log \pi_\theta(a_t|s_t)$ are vector z and $-z$. (Note that because $\mathrm{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\right] = 0$, if $\nabla_\theta \log \pi_\theta(a_t|s_t)$ can only take two values uniformly, then the two values have to two vectors in an opposite direction.) In this case, without subtracting the baseline, the estimators take two values $(1000 + 1)z$ and $-(1000 - 2)z$, whereas after subtracting a baseline of 1000, the estimator has two values z and $2z$. The latter estimator has much lower variance compared to the original estimator.

[4] We note that the estimator of the gradient in the algorithm does not exactly match the equation 11. If we multiply γ^t in the summand of equation (13), then they will exactly match. Removing such discount factors empirically works well because it gives a large update.

Algorithm 1 Vanilla policy gradient with baseline

for $i = 1, \cdots$ **do**

Collect a set of trajectories by executing the current policy. Use $R_{\geq t}$ as a shorthand for $\sum_{j \geq t}^{T-1} \gamma^{j-t} R(s_j, a_j)$

Fit the baseline by finding a function B that minimizes

$$\sum_{\tau} \sum_{t} (R_{\geq t} - B(s_t))^2 \tag{12}$$

Update the policy parameter θ with the gradient estimator

$$\sum_{\tau} \sum_{t} \nabla_\theta \log \pi_\theta(a_t | s_t) \cdot (R_{\geq t} - B(s_t)) \tag{13}$$