Comparing Policy Gradient Algorithm

Tianyu Li

April 2017

1 Preliminary

Let us now consider the standard reinforcement learning framework under a Markov decision process. The *sate*, *action* and *reward* at each time $t \in 0, 1, 2, \cdots$ are denoted as $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$ and $r_t \in \mathcal{R}$. The environment is described with the *state transition probabilities* $P^a_{ss'} = Pr\{s_{t+1} = s' | s_t = s, a_t = a\}$, and *expected rewards* $\tilde{R}^a_s = E\{r_t + 1 | s_t = s, a_t = a\}$ and a *policy* $\pi(s, a, \theta)$ on which the agent is operating, where θ is a parameter vector.

Under this setting, the agent's objective can be characterized as maximize the expected reward that it will receive under certain policy π, for which we define $J(\pi) = \lim_{n\to\infty} \frac{1}{n} E\{r_1 + r_2 + \cdots + r_n | \pi\}$, and it has been shown that the following equation holds:

$$J(\pi) = \sum_s d^\pi(s) \sum_a \pi(s, a) \tilde{R}^a_s \tag{1}$$

where we have $d^\pi(s) = \lim_{t\to\infty} Pr(s_t = s | s_0, \pi)$ is the stationary distribution under certain policy. With this setting, we will also define *return* as

$$R_t = \sum_{t=1} (r_t - J(\pi)) \tag{2}$$

For policy gradient method, instead of taking the argmax to the action value function, it parametrize the policy with θ, then by taking derivative and applying gradient ascent, it is trying to find the optimal θ which will guide the policy the maximize the expected reward (objective). Thus, by taking the gradient of equation (1), we will have:

$$\bigtriangledown_\theta J = \sum_s d^\pi(s) \sum_a Q^\pi(s, a) \bigtriangledown_\theta \pi(s, a) \tag{3}$$

where $Q^\pi(s, a)$ is the regular action value function:(Proof can be found in [2][3][4])

$$Q^\pi(s, a) = E_\pi\{R_t | s_t = s, a_t = a\} \tag{4}$$

Assuming now for the state s and action a, we sample them from the policy distribution π. This will then give us the form:

$$\bigtriangledown_\theta J = E_{s,a\sim\pi}\{\frac{Q^\pi(s, a)}{\pi(s, a)} \bigtriangledown_\theta \pi(s, a)\} \tag{5}$$

Therefore, our updating rule for θ should take the form of:

$$\triangle\theta_t = \alpha \frac{\hat{Q}^{\pi}(s_t, a_t)}{\pi(s_t, a_t)} \bigtriangledown_{\theta} \pi(s_t, a_t) \tag{6}$$

For the form in equation(3) we can see the gradient of the policy function π is easy to compute, if we impose some form of the function, i.e. linear functions. Also, the stationary distribution is easy to estimate, as we can simply apply the empirical probability of the states' occurrence. The tricky part is the state action function $Q^{\pi}(s, a)$.

To approximate $^{\pi}(s, a)$, we can take the following three strategies:

Using Return

First we can just use the direct return R_t as the estimator, and it is in fact a unbiased estimator since $E(R_t) = Q^{\pi}$. Moreover, we can also use the discounted return $\hat{Q}^{\pi}(s_t, a_t) = \frac{1}{\gamma}(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots)$, where γ is chosen based on the mixing time of MDP.

Truncated Return

In the truncated return, we don't directly sum up all the discounted future returns, but instead, we stop at some point and then using an approximator of the state action function, for k-truncated return we have:

$$R_t^{(k)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{k-1} r_{t+k} + \gamma^k q(s_{t+k}, a_{t+k}) \tag{7}$$

where $q(s_{t+k}, a_{t+k})$ is the approximator for action value function.

Approximator

For this approach, we completely ditch the information provided by the reward, just use the approximator of action value function.

2 Unbiased values don't help

Naturally we would think the these three methods should in the order of increasing bias, as we gradually introduce approximator of action value function. However, it is not really the case.

In [5] and [6], for the approximator q, they proposed a form of linear combination of the synthetic features:

$$q(s, a) = w^T \frac{\bigtriangledown_{\theta} \pi(s, a)}{\pi(s, a)} \tag{8}$$

In practice, in order to train the parameter vector w, we will try to minimize the difference between q and the unbiased estimator $\hat{Q}^{\pi}(s_t, a_t)$. This gives us

the following objective function:

$$w = \arg\min_{w} \sum_{t} (\hat{Q}_t^{\pi} - q(s_t, a_t))^2 \tag{9}$$

However, given a batch of data $\mathcal{D}$, if we take the first order derivative w.r.t. w of the right hand side objective function, and due to w is minimizing the function, thus the derivative should be zero. Therefore, we will have:

$$0 = \sum_{t \in \mathcal{D}} (\hat{Q}_t^{\pi} - q(s_t, a_t)) \frac{\bigtriangledown_{\theta} \pi(s, a)}{\pi(s, a)} \tag{10}$$

$$\sum_{t \in \mathcal{D}} (\hat{Q}_t^{\pi}) \frac{\bigtriangledown_{\theta} \pi(s, a)}{\pi(s, a)} = \sum_{t \in \mathcal{D}} (q(s_t, a_t)) \frac{\bigtriangledown_{\theta} \pi(s, a)}{\pi(s, a)} \tag{11}$$

This is rather surprising, because if we set $\hat{Q}_t^{\pi} = R_t$, then the gradient updating rule that we obtained is just the REINFORCE algorithm proposed in []. The message here to send is that the approximate values don't introduce any bias, but neither do they reduce the variance relative to simpler REINFORCE algorithm, as the updating rule is essentially the same.

3 Value Baselines

Another common trice that people do to improve the efficiency of REINFORCE is to introduce the baseline function $b : \mathcal{S} \to \mathcal{R}$. Therefore, our expression of equation (5) and (6) can be then rewritten as:

$$\bigtriangledown_{\theta} J = E_{s,a \sim \pi} \{ (Q^{\pi}(s, a) - b(s)) \frac{\bigtriangledown_{\theta} \pi(s, a)}{\pi(s, a)} \} \tag{12}$$

$$\triangle \theta_t = \alpha (Q^{\pi}(s, a) - b(s)) \frac{\bigtriangledown_{\theta} \pi(s, a)}{\pi(s, a)} \tag{13}$$

Because $\sum_s \frac{\bigtriangledown_{\theta} \pi(s,a)}{\pi(s,a)} = 0$, thus by taking expectation of (13), one should notice that the expected update is not changed and thus this base function doesn't introduce more bias. However, from the experiments result we seen on the class, the variance is reduced, thus leading to a better solution.

Another view of this improvement(But I'm not quite sure)
If we look at the gradient update rule in equation (6), we can see that with this setting, it is moving toward the direction of $\hat{Q}_t^\pi$, i.e. if $\hat{Q}_t^\pi$ is positive, then it will assume the current state-action pair is great, thus putting more weights on the current setting in the policy function. However, this might be too aggressive, in term of the direction and the step size. So now, by adding the base functions, the algorithm tends to reduce the step size and set up a threshold that indicating how big the action value is worth going along with that direction. Intuitively, setting the base function to value function will then be better, as value function essentially is the expectation of the action value function over all the actions. However, Sutton pointed out that setting base function as value function does not lead to minimum variance, yet the proof is not shown in the paper. But in that case, what base function should we choose in order to obtain the smallest variance.

4 Reference

[1] Sutton R S, Singh S P, McAllester D A. Comparing policy-gradient algorithms[J]. 2000.
[2] Marbach P, Tsitsiklis J N. Simulation-based optimization of Markov reward processes[J]. IEEE Transactions on Automatic Control, 2001, 46(2): 191-209.
[3] Singh S P, Jaakkola T S, Jordan M I. Learning Without State-Estimation in Partially Observable Markovian Decision Processes[C]//ICML. 1994: 284-292.
[4] Jaakkola T, Singh S P, Jordan M I. Reinforcement learning algorithm for partially observable Markov decision problems[J]. Advances in neural information processing systems, 1995: 345-352.
[5] Konda V R, Tsitsiklis J N. Actor-Critic Algorithms[C]//NIPS. 1999, 13: 1008-1014.
[6] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]//NIPS. 1999, 99: 1057-1063.