

From Policy Gradient to Actor-Critic methods

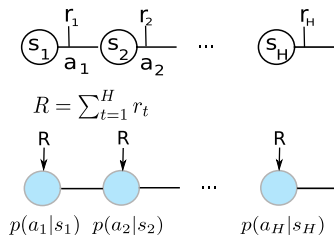
Policy Gradient and Reward Weighted Regression

Olivier Sigaud

Sorbonne Université
<http://people.isir.upmc.fr/sigaud>



Reminder: the most basic PG algorithm



- ▶ Sample a set of trajectories from π_{θ}
- ▶ Compute:

$$Loss(\theta) = \frac{1}{m} \sum_{i=1}^m \sum_{t=1}^H \log \pi_{\theta}(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)}) R(\tau^{(i)}) \quad (1)$$

- ▶ Minimize the loss
- ▶ Iterate: sample again

Behavioral cloning

- ▶ Assume we have a set of expert trajectories,
- ▶ Data is a list of pairs $(\mathbf{s}_t^{(i)}, \mathbf{a}_t^{(i)})$, t is time, H is horizon, i is the trajectory index
- ▶ If the trajectories are optimal, a good option is **behavioral cloning**
- ▶ Use regression to find a policy π_θ behaving as close as possible to data
- ▶ Use a validation set to avoid overfitting.
- ▶ If the policy π_θ is deterministic, this amounts to minimizing the loss function:

$$Loss(\theta) = \frac{1}{m} \sum_{i=1}^m \sum_{t=1}^H (\mathbf{a}_t^{(i)} - \pi_\theta(\mathbf{s}_t^{(i)}))^2$$

- ▶ If the policy π_θ is stochastic, a standard approach (among many others) consists in minimizing the log likelihood loss function:

$$Loss(\theta) = \frac{1}{m} \sum_{i=1}^m \sum_{t=1}^H \log \pi_\theta(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)})$$

Reward Weighted Regression

- ▶ Now, if the expert trajectories are not optimal
- ▶ Let $R(\tau)$ be the return of trajectory τ
- ▶ Still use regression, but weight each sample depending on the return of the corresponding trajectory.
- ▶ That is, imitate “more strongly” what is good in the batch than what is bad.
- ▶ Still use a validation set to avoid overfitting.
- ▶ If the policy π_θ is deterministic, this amounts to minimizing the loss function:

$$Loss(\theta) = \frac{1}{m} \sum_{i=1}^m \sum_{t=1}^H (\mathbf{a}_t^{(i)} - \pi_\theta(\mathbf{s}_t^{(i)}))^2 R(\tau^{(i)})$$

- ▶ If the policy π_θ is stochastic, we minimize the function:

$$Loss(\theta) = \frac{1}{m} \sum_{i=1}^m \sum_{t=1}^H \log \pi_\theta(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)}) R(\tau^{(i)})$$

- ▶ Then we can iterate: generate new data from the new policy, and so on

PG = RWR !

- ▶ Equation (2) is the same as (1)!
- ▶ But wait, the basic PG algorithm is on-policy, and RWR uses expert data in the first step! What's happening?
- ▶ My guess: An on-policy algorithm will work from behavioral samples if they are not worse than the current policy
- ▶ There also exists AWR, close to REINFORCE with $V(s)$ baseline, thus weight = $\hat{A}_{\phi_j}^{\pi}(s_t^{(i)}, a_t^{(i)})$
- ▶ See my youtube video
- ▶ And this blogpost for a wider perspective:
Data-driven Deep Reinforcement Learning
<https://bair.berkeley.edu/blog/2019/12/05/bear/>



Peng, X. B., Kumar, A., Zhang, G., and Levine, S. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019



Any question?



Send mail to: Olivier.Sigaud@upmc.fr



Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine.

Advantage-weighted regression: Simple and scalable off-policy reinforcement learning.

arXiv preprint arXiv:1910.00177, 2019.