

Reinforcement Learning of Motor Skills with Policy Gradients

Ruiming Zhang,

{amber.zhang, }@tum.de

Problem Setting

- Task: Learning complex motor skills with an anthropomorphic robot arm using reinforcement learning.
- Typical characteristics:
 - high-dimensional and continuous space and action space
 - has to be model-free
 - high degrees of freedom cannot deal with parameterised policies

Problem statement in Mathematics

- The general goal is to optimize the policy parameters $\theta \in R^K$ so that the expect return

$$J(\theta) = \frac{1}{a_\Sigma} E\left\{\sum_{k=0}^H a_k r_k\right\} \quad (1)$$

is optimized where a_k denote time-step dependent weighting factors and r_k the reward the learning system received by each instant of time.

Policy Gradient Method

- The policy parameter θ is updated at each time step by:

$$\theta_{m+1} = \theta_m + \alpha_m \nabla_\theta J|_{\theta=\theta_m} \quad (2)$$

where $\alpha_m \in R^+$ denotes a learning rate and fulfill:

$$\sum_{m=0}^{\infty} \alpha_m > 0 \quad \text{and} \quad \sum_{m=0}^{\infty} \alpha_m^2 = 0 \quad (3)$$

- The main problem is obtaining a good estimator of the policy gradient $\nabla_\theta J|_{\theta=\theta_m}$.

Motor Primitive Framework

- We use motor commands to well represent a trajectory, which forms the stochastic policy:

$$\pi(\ddot{q}_d|q_d) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\ddot{q}_d - \ddot{q}_d)^2}{2\sigma^2}\right) \quad (4)$$

The \ddot{q}_d term denotes for a perturbed target output considering the exploration controlled only through σ .

Likelihood Ratio Method

- Use τ to represent a real generated trajectory, $\tau \sim p_\theta(\tau) = p(\tau|\theta)$, with rewards $r(\tau) = \sum_{k=0}^H a_k r_k$.
- LEMMA 1 *save for use.....*
- Then the expect return of a policy can be written as an expectation over all possible trajectories:

$$J(\theta) = \int p_\theta(\tau) r(\tau) d\tau \quad (5)$$

- Subsequently, the gradient can be rewritted by:

$$\begin{aligned} \nabla_\theta J(\theta) &= \int \nabla_\theta p_\theta(\tau) r(\tau) d\tau \\ &= \int p_\theta(\tau) \nabla_\theta \log p_\theta(\tau) r(\tau) d\tau \\ &= E\{\nabla_\theta \log p_\theta(\tau) r(\tau)\} \end{aligned} \quad (6)$$

- The derivative $\nabla_\theta \log p_\theta(\tau)$ can be computed by:

$$\nabla_\theta \log p_\theta(\tau) = \sum_{k=0}^H \nabla_\theta \log \pi_\theta(u_k|x_k) \quad (7)$$

- A constant baseline can be inserted since it has no effect to the derivative, and results in the final form of the gradient estimator:

$$\nabla_\theta J(\theta) = E\{\nabla_\theta \log p_\theta(\tau) (r(\tau) - b)\} \quad (8)$$

Usually b is chosen with the goal to minimize the variance of the gradient estimator.

Natural Actor-Critic

- We first introduce second-order Taylor expansion to approximate the closeness of two distribution, i.e. the gradient descent with respect to an invariant measure of the closeness between the current policy and the updated policy based on the distribution of the paths generated by each of these:

$$d_{KL}(p_\theta(\tau) || p_{\theta+\Delta_\theta}(\tau)) \approx \frac{1}{2} \Delta_\theta^T F_\theta \Delta_\theta \quad (9)$$

where

$$F_\theta = \int p_\theta(\tau) \nabla_\theta \log p_\theta(\tau) \nabla_\theta \log p_\theta(\tau)^T d\tau = \langle \nabla_\theta \log p_\theta(\tau) \nabla_\theta \log p_\theta(\tau)^T \rangle \quad (10)$$

is known as the Fisher information matrix.

- Assume that the amount of change is fixed using step size ε . Then we have an Optimization problem:

$$\max_{\Delta_\theta} J(\theta + \Delta_\theta) \approx J(\theta) + \Delta_\theta^T \nabla_\theta J \quad (11)$$

$$s.t. \quad \varepsilon = d_{KL}(p_\theta(\tau) || p_{\theta+\Delta_\theta}(\tau)) \approx \frac{1}{2} \Delta_\theta^T F_\theta \Delta_\theta$$

and has the solution:

$$\Delta_\theta = \alpha_n F_\theta^{-1} \nabla_\theta J \quad (12)$$

the natural gradient:

$$\tilde{\nabla}_\theta J(\theta) = \Delta_\theta / \alpha_n \quad (13)$$

- The natural gradient can be simply computed as $\tilde{\nabla}_\theta J(\theta) = G_\theta^{-1} F_\theta \omega = \text{omega}$ using the function approximation concept. Therefore the resulting policy improvement step becomes $\theta_{i+1} = \theta_i + \alpha \omega$.

Episodic Natural Actor-Critic Algorithm

- For episodic tasks the natural actor-critic can be simplified as:

$$\sum_{t=0}^H a_t \nabla \log \pi(u_t, x_t)^T w + J_0 = \sum_{t=0}^H a_t r(x_t, u_t) \quad (14)$$

- This means for non-stochastic tasks we can obtain a natural gradient after $\dim \theta + 1$ roll-outs using least-squares regression:

$$\begin{bmatrix} \omega \\ J_0 \end{bmatrix} = (\Psi^T \Psi)^{-1} \Psi^T R \quad (15)$$

with

$$\Psi_i = \left[\sum_{t=0}^H a_t \nabla \log \pi(u_t, x_t)^T, 1 \right] \quad (16)$$

$$R_i = \sum_{t=0}^H a_t r(x_t, u_t) \quad (17)$$

- In order to take time-variance rewards significantly better into account, we use a time-variant average rewards $\bar{r} = [\bar{r}_1, \bar{r}_2, \dots, \bar{r}_K]$ and then we have to solve:

$$\begin{bmatrix} g_{eNACn} \\ \bar{r} \end{bmatrix} = \begin{bmatrix} F_2 & \bar{\Phi} \\ \bar{\Phi}^T & m I_H \end{bmatrix} \begin{bmatrix} g \\ \bar{r} \end{bmatrix} \quad (18)$$

and finally obtained:

$$b = Q^{-1}(\bar{r} - \bar{\Phi}^T F_2^{-1} g) \quad (19)$$

with $Q^{-1} = m^{-1}(I_n + \bar{\Phi}^T (m F_2 - \bar{\Phi} \bar{\Phi}^T)^{-1} \bar{\Phi})$ and Φ is the eligibility matrix.

- The complete solution can be transformed into the form below:

input: policy parameterization θ .	
1	repeat
2	perform M trials and obtain $\mathbf{x}_{0:H}, \mathbf{u}_{0:H}, r_{0:H}$ for each trial.
	Obtain the sufficient statistics
3	Policy derivatives $\boldsymbol{\psi}_k = \nabla_\theta \log \pi_\theta(\mathbf{u}_k \mathbf{x}_k)$.
4	Fisher matrix $\mathbf{F}_\theta = \langle \sum_{k=0}^H (\sum_{l=0}^k \boldsymbol{\psi}_l) \boldsymbol{\psi}_k^T \rangle$.
	Vanilla gradient $\mathbf{g} = \langle \sum_{k=0}^H (\sum_{l=0}^k \boldsymbol{\psi}_l) a_k r_k \rangle$.
5	Eligibility matrix $\Phi = [\phi_1, \phi_2, \dots, \phi_K]$
	with $\phi_h = \langle (\sum_{k=0}^h \boldsymbol{\psi}_k) \rangle$.
6	Average reward vector $\bar{\mathbf{r}} = [\bar{r}_1, \bar{r}_2, \dots, \bar{r}_K]$
	with $\bar{r}_h = \langle a_h r_h \rangle$.
	Obtain natural gradient by computing
7	Baseline $\mathbf{b} = \mathbf{Q}(\bar{\mathbf{r}} - \Phi^T \mathbf{F}_\theta^{-1} \mathbf{g})$
	with $\mathbf{Q} = M^{-1} \left(\mathbf{I}_K + \Phi^T (M \mathbf{F}_\theta - \Phi \Phi^T)^{-1} \Phi \right)$.
8	Natural gradient $\mathbf{g}_{\text{NG}} = \mathbf{F}_\theta^{-1} (\mathbf{g} - \Phi \mathbf{b})$.
9	until gradient estimate $\mathbf{g}_{\text{eNACn}}$ converged.
return: gradient estimate $\mathbf{g}_{\text{eNACn}}$.	