强化学习及其应用
Reinforcement Learning and Its Applications

第六章 策略梯度
Policy Gradient

授课人：周晓飞

zhouxiaofei@iie.ac.cn

2023-6-14

第六章 策略梯度

6.1 策略梯度定理

6.2 策略梯度强化学习

6.3 行动批评的强化机制

6.4 算法总结

第六章 策略梯度

6.1 策略梯度定理

6.2 策略梯度强化学习

6.3 行动批评的强化机制

6.4 算法总结

策略梯度定理

问题描述

- **策略不收敛的问题**

A policy was generated directly from the value function

- e.g. using ϵ-greedy

- **解决办法：策略函数逼近**

$$\pi_\theta(s, a) = \mathbb{P}[a \mid s, \theta]$$

策略梯度定理

问题描述

■ **求解问题**

- Goal: given policy $\pi_\theta(s, a)$ with parameters θ, find best θ
- But how do we measure the quality of a policy π_θ?
- In episodic environments we can use the start value

$$J_1(\theta) = V^{\pi_\theta}(s_1) = \mathbb{E}_{\pi_\theta}[v_1]$$

- In continuing environments we can use the average value

$$J_{avV}(\theta) = \sum_s d^{\pi_\theta}(s) V^{\pi_\theta}(s)$$

- Or the average reward per time-step

$$J_{avR}(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(s, a) \mathcal{R}_s^a$$

- where $d^{\pi_\theta}(s)$ is stationary distribution of Markov chain for π_θ

策略梯度定理

Policy based reinforcement learning is an optimisation problem

Find θ that maximises $J(\theta)$

$$J_1(\theta) = V^{\pi_\theta}(s_1) = \mathbb{E}_{\pi_\theta}[v_1]$$

▉ **策略梯度**

$$\nabla_\theta J(\theta) = \begin{pmatrix} \frac{\partial J(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_n} \end{pmatrix}$$

策略梯度定理

- ### 策略梯度定理

Theorem

For any differentiable policy $\pi_\theta(s, a)$,
for any of the policy objective functions $J = J_1$, J_{avR}, or $\frac{1}{1-\gamma} J_{avV}$,
the policy gradient is

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \ Q^{\pi_\theta}(s, a) \right]$$

策略梯度定理

策略梯度定理

策略梯度定理

Theorem

For any differentiable policy $\pi_\theta(s, a)$,
for any of the policy objective functions $J = J_1, J_{avR}$, or $\frac{1}{1-\gamma} J_{avV}$,
the policy gradient is

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \ Q^{\pi_\theta}(s, a) \right]$$

To evaluate policy gradient of $\pi_\theta(s, a)$

$$\nabla_\theta \pi_\theta(s, a) = \pi_\theta(s, a) \frac{\nabla_\theta \pi_\theta(s, a)}{\pi_\theta(s, a)}$$
$$= \pi_\theta(s, a) \nabla_\theta \log \pi_\theta(s, a)$$

策略梯度定理

策略函数

Softmax Policy

■ Weight actions using linear combination of features $\phi(s, a)^\top \theta$

■ Probability of action is proportional to exponentiated weight

$$\pi_\theta(s, a) \propto e^{\phi(s,a)^\top \theta}$$

■ The score function is

$$\nabla_\theta \log \pi_\theta(s, a) = \phi(s, a) - \mathbb{E}_{\pi_\theta} \left[\phi(s, \cdot) \right]$$

策略梯度定理

策略函数

Gaussian Policy

- In continuous action spaces, a Gaussian policy is natural
- Mean is a linear combination of state features $\mu(s) = \phi(s)^\top \theta$
- Variance may be fixed σ^2, or can also parametrised
- Policy is Gaussian, $a \sim \mathcal{N}(\mu(s), \sigma^2)$
- The score function is

$$\nabla_\theta \log \pi_\theta(s, a) = \frac{(a - \mu(s))\phi(s)}{\sigma^2}$$

第六章 策略梯度

One-Step MDPs

- Consider a simple class of one-step MDPs
  - Starting in state $s \sim d(s)$
  - Terminating after one time-step with reward $r = \mathcal{R}_{s,a}$
- Use likelihood ratios to compute the policy gradient

$$J(\theta) = \mathbb{E}_{\pi_\theta}[r]$$
$$= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s, a) \mathcal{R}_{s,a}$$
$$\nabla_\theta J(\theta) = \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s, a) \nabla_\theta \log \pi_\theta(s, a) \mathcal{R}_{s,a}$$
$$= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s, a) r]$$

Monte-Carlo Policy Gradient (REINFORCE)

- Update parameters by stochastic gradient ascent
- Using policy gradient theorem
- Using return v_t as an unbiased sample of $Q^{\pi_\theta}(s_t, a_t)$

$$\Delta\theta_t = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$$

function REINFORCE
    Initialise θ arbitrarily
    for each episode $\{s_1, a_1, r_2, ..., s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**
        for $t = 1$ to $T - 1$ **do**
            $\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$
        end for
    end for
    return θ
end function

第六章 策略梯度

6.1 策略梯度定理

6.2 策略梯度强化学习

6.3 行动批评的强化机制

6.4 算法总结

行动批评的强化机制

Actor-critic algorithms

Actor-critic algorithms maintain *two* sets of parameters

Critic   Updates action-value function parameters w → **引入值函数逼近**

Actor   Updates policy parameters θ, in direction suggested by critic

We use a critic to estimate the action-value function,

$$Q_w(s, a) \approx Q^{\pi_\theta}(s, a)$$

Actor-critic algorithms follow an *approximate* policy gradient

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s, a)\ Q_w(s, a)\right]$$
$$\Delta\theta = \alpha\nabla_\theta \log \pi_\theta(s, a)\ Q_w(s, a)$$

Action-Value Actor-Critic (QAC)

■ Simple actor-critic algorithm based on action-value critic

■ Using linear value fn approx. $Q_w(s, a) = \phi(s, a)^\top w$

      Critic Updates w by linear TD(0)

      Actor Updates θ by policy gradient

function QAC

    Initialise s, θ

    Sample $a \sim \pi_\theta$

    for each step **do**

        Sample reward $r = \mathcal{R}_s^a$; sample transition $s' \sim \mathcal{P}_{s,\cdot}^a$

        Sample action $a' \sim \pi_\theta(s', a')$

        $\delta = r + \gamma Q_w(s', a') - Q_w(s, a)$

        $\theta = \theta + \alpha \nabla_\theta \log \pi_\theta(s, a) Q_w(s, a)$

        $w \leftarrow w + \beta \delta \phi(s, a)$

        $a \leftarrow a', s \leftarrow s'$

    end for

end function

Reducing Variance Using a Baseline

$$\nabla J(\boldsymbol{\theta}) = \sum_s \mu_\pi(s) \sum_a \Big(q_\pi(s,a) - b(s) \Big) \nabla_{\boldsymbol{\theta}} \pi(a|s, \boldsymbol{\theta})$$

The baseline can be any function, even a random variable, as long as it does not vary with a; the equation remains true, because the the subtracted quantity is zero:

$$\sum_a b(s) \nabla_{\boldsymbol{\theta}} \pi(a|s, \boldsymbol{\theta}) = b(s) \nabla_{\boldsymbol{\theta}} \sum_a \pi(a|s, \boldsymbol{\theta}) = b(s) \nabla_{\boldsymbol{\theta}} 1 = 0 \qquad \forall s \in \mathcal{S}.$$

Reducing Variance Using a Baseline

$$\nabla J(\boldsymbol{\theta}) = \sum_s \mu_\pi(s) \sum_a \Big(q_\pi(s,a) - b(s) \Big) \nabla_{\boldsymbol{\theta}} \pi(a|s,\boldsymbol{\theta})$$

The baseline can be any function, even a random variable, as long as it does not vary with a; the equation remains true, because the the subtracted quantity is zero:

$$\sum_a b(s) \nabla_{\boldsymbol{\theta}} \pi(a|s,\boldsymbol{\theta}) = b(s) \nabla_{\boldsymbol{\theta}} \sum_a \pi(a|s,\boldsymbol{\theta}) = b(s) \nabla_{\boldsymbol{\theta}} 1 = 0 \qquad \forall s \in \mathcal{S}.$$

- A good baseline is the state value function $B(s) = V^{\pi_\theta}(s)$
- So we can rewrite the policy gradient using the advantage function $A^{\pi_\theta}(s,a)$

$$A^{\pi_\theta}(s,a) = Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$$
$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s,a) \, A^{\pi_\theta}(s,a) \right]$$

Estimating the Advantage Function

■ **实际上，两套参数并不好**

- So the critic should really estimate the advantage function
- For example, by estimating *both* $V^{\pi_\theta}(s)$ *and* $Q^{\pi_\theta}(s, a)$
- Using two function approximators and two parameter vectors,

$$V_v(s) \approx V^{\pi_\theta}(s)$$
$$Q_w(s, a) \approx Q^{\pi_\theta}(s, a)$$
$$A(s, a) = Q_w(s, a) - V_v(s)$$

行动批评的强化机制

Estimating the Advantage Function

■ **T D误差可看作 Advantage Function 的无偏估计**

$$\delta^{\pi_\theta} = r + \gamma V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$$

$$\mathbb{E}_{\pi_\theta}\left[\delta^{\pi_\theta}|s,a\right] = \mathbb{E}_{\pi_\theta}\left[r + \gamma V^{\pi_\theta}(s')|s,a\right] - V^{\pi_\theta}(s)$$
$$= Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$$
$$= A^{\pi_\theta}(s,a)$$

Estimating the Advantage Function

■ T D误差可看作 Advantage Function 的无偏估计

$$\delta^{\pi_\theta} = r + \gamma V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$$

$$\mathbb{E}_{\pi_\theta}\left[\delta^{\pi_\theta}|s,a\right] = \mathbb{E}_{\pi_\theta}\left[r + \gamma V^{\pi_\theta}(s')|s,a\right] - V^{\pi_\theta}(s)$$
$$= Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$$
$$= A^{\pi_\theta}(s,a)$$

So we can use the TD error to compute the policy gradient

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\, \delta^{\pi_\theta}\right]$$

行动批评的强化机制

Estimating the Advantage Function

■ **T D误差可看作 Advantage Function 的无偏估计**

$$\delta^{\pi_\theta} = r + \gamma V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$$

$$\mathbb{E}_{\pi_\theta}[\delta^{\pi_\theta}|s,a] = \mathbb{E}_{\pi_\theta}\left[r + \gamma V^{\pi_\theta}(s')|s,a\right] - V^{\pi_\theta}(s)$$
$$= Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$$
$$= A^{\pi_\theta}(s,a)$$

So we can use the TD error to compute the policy gradient

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\, \delta^{\pi_\theta}\right]$$

In practice we can use an approximate TD error

$$\delta_v = r + \gamma V_v(s') - V_v(s)$$

This approach only requires one set of critic parameters v

Estimating the Advantage Function

Critics at Different Time-Scales

Critic can estimate value function $V_V(s)$ from many targets at different time-scales From last lecture...

- For MC, the target is the return v_t

$$\Delta v = \alpha(v_t - V_V(s))\phi(s)$$

- For TD(0), the target is the TD target $r + \gamma V(s')$

$$\Delta v = \alpha(r + \gamma V(s') - V_V(s))\phi(s)$$

- For forward-view TD(λ), the target is the λ-return v_t^λ

$$\Delta v = \alpha(v_t^\lambda - V_V(s))\phi(s)$$

- For backward-view TD(λ), we use eligibility traces

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$$
$$e_t = \gamma\lambda e_{t-1} + \phi(s_t)$$
$$\Delta v = \alpha\delta_t e_t$$

Estimating the Advantage Function

▌ Actors at Different Time-Scales

- The policy gradient can also be estimated at many time-scales

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \, A^{\pi_\theta}(s, a) \right]$$

- Monte-Carlo policy gradient uses error from complete return

$$\Delta\theta = \alpha(v_t - V_v(s_t))\nabla_\theta \log \pi_\theta(s_t, a_t)$$

- Actor-critic policy gradient uses the one-step TD error

$$\Delta\theta = \alpha(r + \gamma V_v(s_{t+1}) - V_v(s_t))\nabla_\theta \log \pi_\theta(s_t, a_t)$$

forward-view TD(λ), we can mix over time-scales

$$\Delta\theta = \alpha(v_t^\lambda - V_v(s_t))\nabla_\theta \log \pi_\theta(s_t, a_t)$$

第六章 策略梯度

算法总结

- The policy gradient has many equivalent forms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \, v_t \right] \qquad \text{REINFORCE}$$

$$= \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \, Q^w(s, a) \right] \qquad \text{Q Actor-Critic}$$

$$= \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \, A^w(s, a) \right] \qquad \text{Advantage Actor-Critic}$$

$$= \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \, \delta \right] \qquad \text{TD Actor-Critic}$$

$$= \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(s, a) \, \delta e \right] \qquad \text{TD}(\lambda) \text{ Actor-Critic}$$

$$G_\theta^{-1} \nabla_\theta J(\theta) = w \qquad \text{Natural Actor-Critic}$$

- Each leads a stochastic gradient ascent algorithm
- Critic uses policy evaluation (e.g. MC or TD learning) to estimate $Q^\pi(s, a)$, $A^\pi(s, a)$ or $V^\pi(s)$

本讲参考文献

1. Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction．（Second edition, in progress，draft）.
2. David Silver,Slides@《Reinforcement Learning:An Introduction》,2016.