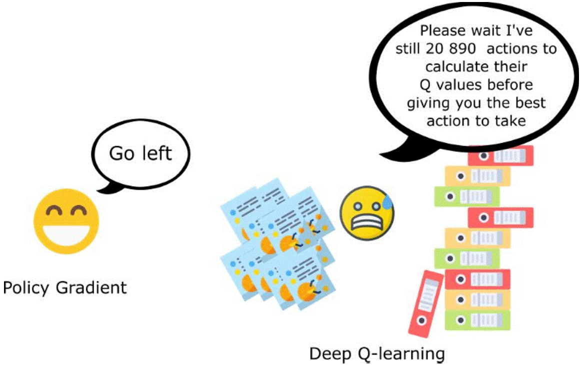
Policy gradient

Wen Yu
Departamento de Control Automático
CINVESTAV-IPN

Problems of value based method



Problems of value based method

1. DQN generally only deal with discrete actions and cannot deal with continuous actions.
2. Due to the limitations of individual observations or the limitations of modeling, two states that are originally different in the real environment have the same feature description after we model them. Value-based method cannot get the optimal solution.
3. The optimal strategy corresponding to the vlue-based reinforcement learning method is usually a deterministic strategy, while the optimal strategy for some problems is a random strategy. In this case, it is also impossible Solved through value-based learning.
4. the policies may not exist with the action-value estimation

Advantages of policy approximation

- Parameterizing policies with the soft-max in action preferences can approach a deterministic policy.
- Action-value based methods have no natural way of finding stochastic optimal policies, whereas policy approximating methods can.
- The soft-max in action preferences enables the selection of actions with arbitrary probabilities.

Policy-Based Reinforcement Learning

The action-vale function can be approximated as

$$\hat{V}(s, w) \approx V^{\pi}(s)$$
$$\hat{Q}(s, a, w) \approx Q^{\pi}(s)$$

A policy was generated directly from the value function, ϵ-greedy policy

$$\pi(a, s) = \begin{cases} \arg\max_a Q^{\pi}(s, a) & \text{probability } (1 - \epsilon) \\ a & \text{probability } \frac{\epsilon}{|\mathcal{A}|} \end{cases}$$

Now we will directly parametrize the policy (not table policy)

$$\pi(a, s, w) = P(a \mid s, w)$$

The goal is to find $\pi(a, s, w)$ such that

$$\max V^{\pi}(s), \quad \max_{\theta} \mathbb{E}[R \mid \pi_{\theta}]$$

Policy-Based vs. Value-Based

- Value based: learn value function, implicit policy (e.g, ϵ-greedy)
- Policy base: no value function, learn policy
- Actor-Critic: learn value function, learn policy

Policy search methods

- Policy search methods: model-free and model-based
- Model-free policy: random search and deterministic search
- Stochastic search: developed as the gradient method
- Policy gradient: difficulty in determining the learning rate.

Quality of a policy

The quality of a policy $\pi(\theta)$:

1. value of the policy in an episodic

$$J_1 = V^{\pi(\theta)}(s_1)$$

2. The average value

$$J_{av} = \sum_s P^{\pi(w)}(s) V^{\pi(\theta)}(s)$$

   where $P^{\pi(\theta)}$ is distribution of the Markoc chain for $\pi(w)$

3. The average reward per time step

$$J_{avR} = \sum_s P^{\pi(\theta)}(s) \sum_a \pi(\theta) R(a, s)$$

Approximate the gradient

Episodic MDP
The goal is to find $\pi\left(a, s, \theta\right)$ such that max $V^{\pi}\left(s\right)$ by gradient of the policy, w.r.t, parameter w

$$\Delta\theta = \alpha \bigtriangledown_{\theta} V\left(\theta\right)$$

where α is a step.size parameter, $\bigtriangledown_{w} V\left(w\right)$ is the policy gradient,

$$\bigtriangledown_{\theta} V\left(\theta\right) = \left[\frac{\partial V\left(\theta\right)}{\partial\theta_1} \cdots \frac{\partial V\left(\theta+\epsilon\right)}{\partial\theta_n}\right]$$

we can use

$$\frac{\partial V\left(\theta\right)}{\partial\theta_i} \approx \frac{V\left(\theta_i+\epsilon\right) - V\left(\theta_i\right)}{\epsilon}$$

to approximate the grdient $\bigtriangledown_{\theta} V\left(\theta\right)$

Compute the policy gradient analytically

Policy value of π_θ

$$V\left(\theta\right) = \mathbb{E}_{\pi_\theta}\left(\sum R_{\pi_\theta}\right) = \sum_i P\left(i,\theta\right) R\left(i\right)$$

our goald is

$$\arg\max_\theta V\left(\theta\right) = \arg\max_\theta \sum_i P\left(i,\theta\right) R\left(i\right)$$

Take gradient with respect to θ, using $\frac{\partial}{\partial x}\log x = \frac{1}{x}$

$$\begin{aligned}&\nabla_\theta V\left(\theta\right) = \nabla_\theta \textstyle\sum_i P\left(i,\theta\right) R\left(i\right) = \sum_i R\left(i\right) \nabla_\theta P\left(i,\theta\right)\\ &= \textstyle\sum_i R\left(i\right) \frac{P\left(i,\theta\right)}{P\left(i,\theta\right)} \nabla_\theta P\left(i,\theta\right)\\ &= \textstyle\sum_k P\left(i,\theta\right) R\left(i\right) \nabla_\theta \log P\left(i,\theta\right)\end{aligned}$$

Approximate

$\nabla_\theta V(\theta)$
The policy gradient becomes the expectation and we can use the empirical average to estimate it. We use the current policy to sample m trajectories, the empirical average of these m trajectories can be used to approximate the policy gradient:
Approximate $\nabla_\theta V(\theta)$ with m sample path under the policy π_θ (same $P(i,\theta)$)

$$\frac{\partial V(\theta)}{\partial \theta_i} \approx \frac{1}{m}\sum_k R(i)\nabla_\theta \log P(i,\theta) = f(x_i)\nabla \log P(x_i \mid \theta)$$

so

$$\Delta\theta = \alpha f(x_i)\nabla \log P(x_i \mid \theta)$$

moveing in the direction to yjr logprob of the sample
The item $\nabla \log P(x_i \mid \theta)$ is the steepest direction in which the probability of the trajectory changes with the parameter θ.
$\frac{1}{m}\sum_k R(i)$ controls the direction and step length of parameter update.

Calculate the steepest direction

Calculate $\nabla \log P\left(x_i \mid \theta\right)$, for i-th episode,

$$P\left(x_i \mid \theta\right)=\prod P\left(s_{t+1}^i \mid s_t^i, a_t^i\right) \pi_\theta\left(s_t^i \mid a_t^i\right)$$

and

$$\log P\left(x_i \mid \theta\right)=\sum \log P\left(s_{t+1}^i \mid s_t^i, a_t^i\right)+\sum \log \pi_\theta\left(s_t^i \mid a_t^i\right)$$

So

$$\begin{aligned}&\nabla_\theta \log P\left(x_i \mid \theta\right)=\sum \nabla_\theta \log P\left(s_{t+1}^i \mid s_t^i, a_t^i\right)+\sum \nabla_\theta \log \pi_\theta\left(s_t^i \mid a_t^i\right)\\&=\sum \nabla_\theta \log \pi_\theta\left(s_t^i \mid a_t^i\right)\end{aligned}$$

The likelihood gradient $\nabla \log P\left(x_i \mid \theta\right)$ is transformed into the gradient of the action $\nabla_\theta \log \pi_\theta\left(s_t^i \mid a_t^i\right)$,
But it has nothing to do with dynamic model

$$P\left(s_{t+1}^i \mid s_t^i, a_t^i\right)$$

because

$$J_{avR} = \sum_s P^{\pi(w)}(s) \sum_a \pi(w) R(a, s)$$
$$\nabla J_{avR} = \nabla_\theta V(\theta) = \frac{\partial V(\theta)}{\partial \theta_i}$$
$$\approx \frac{1}{m} \sum_{k=1}^{m} R\left(T^i\right) \sum_{i=1}^{T} \nabla_\theta \log \pi_\theta\left(s_t^i \mid a_t^i\right)$$

so

$$\nabla_\theta V(\theta) \approx \frac{1}{m} \sum_k R(i) \sum_k \nabla_\theta \log \pi_\theta(s, a)$$

We need to calculate $\nabla_\theta \log \pi_\theta(s, a)$So

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

The term $\frac{1}{m} \sum_{k=1}^{m}$ likes MC, is unbiased, but the variance is large (noisy).

Like TD learning or Q learning

$$\nabla_\theta V(\theta, r) = \mathbb{E}\left[\left(\sum_{i=0}^{T-1} r_i\right)\left(\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(s,a)\right)\right]$$
$$\nabla_\theta V(\theta, r') = \mathbb{E}\left[\left(\sum_{i=0}^{T-1} r'_i\right)\left(\sum_{t=0}^{t'} \nabla_\theta \log \pi_\theta(s,a)\right)\right]$$

But $\sum_{i=0}^{T-1} r_i = G_t$,

$$\begin{aligned}\nabla_\theta V(\theta) &\approx \frac{1}{m}\sum_{k=1}^{m} R\left(T^i\right)\sum_{i=1}^{T} \nabla_\theta \log \pi_\theta\left(s_t^i \mid a_t^i\right) \\ &= \frac{1}{m}\sum_{k=1}^{m}\sum_{i=1}^{T} \nabla_\theta \log \pi_\theta\left(s_t^i \mid a_t^i\right) G_t\end{aligned}$$

For t=1 to T-1

$$\theta(t+1) = \theta(t) + \alpha \nabla_\theta \log \pi_\theta(s,a)\, G_t$$

Calculate logPi: Gussian

The policy can be parameterized in any way, as long as π_θ is differentiable. Generally, a random policyy can be written as a deterministic part plus a random part,

$$\pi_\theta = h_\theta + \varepsilon, \quad \varepsilon \sim N\left(0, \sigma^2\right)$$

For the Gaussian stribution ε with a mean value of zero and a standard deviation. The deterministic part is commonly expressed as linear in features

$$h_\theta\left(s, a\right) = \theta^T z\left(s, a\right)$$

Total is

$$a \sim N\left(\theta^T z\left(s, a\right), \sigma^2\right)$$

Calculate logPi: example-Soft-max

The action sapce distribute is soft-max distribution, for each action

$$\pi_\theta\left(a \mid s\right) = \frac{e^{z(s,a)}}{\sum_a e^{z(s,a)}}$$

$z\left(s, a\right)$ can be parameterized, such as neural network

$$z\left(s, a\right) = \phi^T\left(s, a\right)\theta$$

then

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

$\phi^T\left(s, a\right)$ is current feature, $\mathbb{E}\left[\phi^T\left(s, a\right)\right]$ is average feature over all actions under the policy

Calculate logPi: example-Gussian

We fix the variance σ^2,

$$
\begin{aligned}
&a \sim \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{[a-\mu(s)]^2}{2\sigma^2}} \\
&\pi_\theta\left(a \mid s\right) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{[a-\mu(s)]^2}{2\sigma^2}} \\
&\mu\left(s\right) = \phi^T\left(s\right)\theta \\
&\nabla_\theta \log \pi_\theta\left(s, a\right) = \frac{[a-\mu(s)]\phi(s)}{\sigma^2}
\end{aligned}
$$

Reiforce with baseline

Since

$$\nabla_\theta V(\theta) \approx \frac{1}{m} \sum_k R(i) \sum_k \nabla_\theta \log \pi_\theta(s,a)$$

then

$$\Delta\theta_t = \alpha G_t \nabla_\theta \log \pi_\theta, \quad SGD$$

The policy gradient is unbiased, but the variance is large. To fixe it, we introduce temporal structure or baseline to reduce variance, since $V(\theta) = \mathbb{E}_{\pi_\theta}\left(\sum r_{\pi_\theta}\right)$

$$\nabla_\theta V(\theta) = \nabla_\theta \mathbb{E}_{\pi_\theta}\left(\sum \nabla_\theta \log \pi_\theta(s,a)\left(\sum (r-b)\right)\right)$$

where $b = \mathbb{E}_{\pi_\theta} \sum r$, the update law is

$$\Delta\theta_t = \alpha \left[G_t - b(s)\right] \nabla_\theta \log \pi_\theta, \quad SGD$$

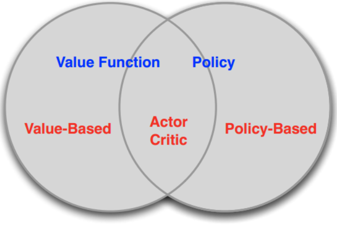
Actor-critic

- Q-learning uses the value of $Q(x, a)$ to take a certain action. It acts as a critic, who evaluates the decision and the evaluation result using $Q_k(x_k, a_k)$
- Q-learning algorithm needs to discretize the action space, which makes the Q-learning algorithm difficult to find the optimal value and the calculation speed is relatively slow.
- Policy gradient calculats the **next action**. The output is the action or distribution of actions. It is an actor.

Actor-critic

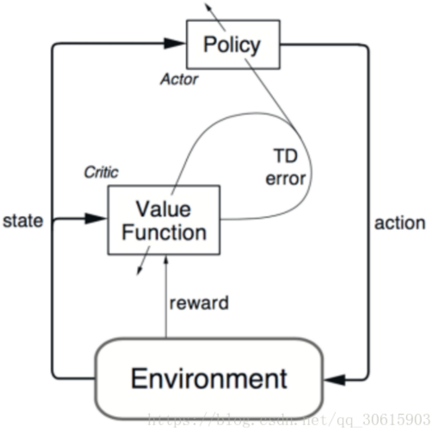
Asynchronous advantage actor-critic
Deep Deterministic Policy Gradient (DDPG): NN->DQN

- Value Based
  - Learnt Value Function
  - Implicit policy (e.g. ϵ-greedy)
- Policy Based
  - No Value Function
  - Learnt Policy
- Actor-Critic
  - Learnt Value Function
  - Learnt Policy



Actor-critic

A3C



Actor-critic

If the baseline is a critic, then the baseline method becomes actor–critic method.

$$\Delta\theta_t = \alpha \left[G_t - b\left(s\right)\right] \nabla_\theta \log \pi_\theta, \quad SGD$$

The update law is

$$\begin{aligned}
&\Delta\theta_t = \alpha \left[G_t - b\left(s_t\right)\right] \nabla \log \pi_\theta \\
&= \alpha \left[q_t - \hat{V}\left(s_t, w\right)\right] \nabla \log \pi_\theta \\
&= \alpha \left[r_{t+1} + \gamma \hat{V}\left(s_{t+1}, w\right) - \hat{V}\left(s_t, w\right)\right] \nabla \log \pi_\theta
\end{aligned}$$

or

$$\Delta\theta_t = \alpha e_t \nabla \ln \left[\pi\left(A_t \mid s_t, w_t\right)\right]$$

where the TD error is

$$e_t = r_{t+1} + \gamma \hat{V}\left(s_{t+1}, w\right) - \hat{V}\left(s_t, w\right)$$

Actor-critic

AC can be reagrded as

$$J(w) = \sum_k \ln \pi_w(x_k, u_k)\left[r + \gamma R_k(x_{k+1}) - R_k(x_k)\right]$$

Here $\left[r + \gamma R_k(x_{k+1}) - R_k(x_k)\right]$ can be regarded as the TD error and it is estimated by the Q-learning algorithm.

Actor-critic: Policy Based+Q-learning

- Learn an approximate value function based.
- Learn approximations to both policy and value functions, actor–critic methods.
  1. The 'actor' is a reference to be learned for the policy: Policy Gradients

$$\Delta\theta_t = \alpha\left[q_t - \hat{V}\left(s_t, w\right)\right]\nabla \log \pi_\theta, \quad \hat{\pi}_w = W_1\phi\left(x\right) \longrightarrow NN_1$$

  2. The 'critic' refers to the learned value function, such as state-value function: Q-learning

$$\hat{V}\left(s_t, w\right) = W_2\phi\left(x\right) \longrightarrow NN_2$$

TD error based

$$\begin{aligned} e_t &= r + \gamma R_k\left(x_{k+1}\right) - R_k\left(x_k\right) \\ e_t &= R_{t+1} + \gamma \hat{V}\left(s_{t+1}, w\right) - \hat{V}\left(s_t, w_t\right) \\ e_t &= R_{t+1} + \gamma \hat{Q}\left(s_{t+1}, a_{t+1}, w_t\right) - \hat{Q}\left(s_t, w_t\right) \end{aligned}$$

Theorem

The gradient of state-value function can be trnasform into action-value function

$$\Delta\theta_t = \alpha \nabla J(\theta_t)$$
$$= \nabla \sum_k P(i,\theta) R(i) \nabla_\theta \log P(i,\theta)$$
$$\approx \frac{1}{m} \sum_k R(i) \sum_k \nabla_\theta \log \pi_\theta(s,a)$$

Theorem

Policy gradient is

$$\Delta\theta_t = \alpha \nabla J(\theta_t) = \alpha \nabla v_\pi(s,\theta)$$

gives an analytic expression for the gradient of performance with respect to the policy parameter, and it does not involve the derivative of the state distribution

$$\frac{\partial}{\partial \theta_t} J(\theta_t) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \frac{\partial}{\partial \theta_t} \pi(a \mid s, \theta_t)$$
$$\nabla V(\theta) = \mathbb{E}_{\pi_\theta}\left[q_\pi(a,s) \nabla_\theta \log \pi(a \mid s)\right]$$

Proof

The object of the proof:

$$J = \mathbb{E}\left[v \mid \pi_\theta\right] = v_\pi$$

$$\begin{aligned} \nabla v_\pi\left(s\right) &= \nabla\left(\textstyle\sum_a \pi\left(a \mid s\right) q_\pi\left(a, s\right)\right) \\ &= \textstyle\sum_a \left[\nabla \pi\left(a \mid s\right) q_\pi\left(a, s\right) + \pi\left(a \mid s\right) \nabla q_\pi\left(a, s\right)\right] \end{aligned}$$

where $\pi\left(a \mid s\right) \nabla q_\pi\left(a, s\right)$ is

$$\begin{aligned} \nabla q_\pi\left(a, s\right) &= \nabla\left[\textstyle\sum_{\bar{s}, r} p\left(\bar{s}, r \mid s, a\right)\left[r + \nabla v_\pi\left(\bar{s}\right)\right]\right] \\ &= \textstyle\sum_{\bar{s}} p\left(\bar{s} \mid s, a\right) \nabla v_\pi\left(\bar{s}\right) \end{aligned}$$

where

$$\nabla v_\pi\left(\bar{s}\right) = \sum_{\bar{a}} \nabla \pi\left(\bar{a} \mid \bar{s}\right) q_\pi\left(\bar{a} \mid \bar{s}\right) + \pi\left(\bar{a} \mid \bar{s}\right) \sum_{\bar{s}} p\left(\breve{s} \mid \bar{s}, \bar{a}\right)$$

where $\breve{s}$ is the next step of the state $\bar{s}$.

$\nabla v_\pi\left(\bar{s}\right)$ is in regression form,

$$\nabla v_\pi\left(s\right) = \textstyle\sum_a \left[\begin{array}{l} \nabla \pi\left(a \mid s\right) q_\pi\left(a, s\right) + \\ \pi\left(a \mid s\right) \textstyle\sum_{\bar{s}} p\left(\bar{s} \mid s, a\right) \left\{ \begin{array}{c} \textstyle\sum_{\bar{a}} \nabla v_\pi\left(\bar{a} \mid \bar{s}\right) q_\pi\left(\bar{a} \mid \bar{s}\right) \\ +\pi\left(\bar{a} \mid \bar{s}\right) \textstyle\sum_{\breve{s}} p\left(\breve{s} \mid \bar{s}, \bar{a}\right) \nabla v_\pi\left(\breve{s}\right) \end{array} \right. \end{array}\right.$$

Proof

With the parameter θ and a special state s_0

$$\nabla J(\theta) = \nabla v_{\pi}(s_0) = \sum_s \sum_{k\to\infty} \Pr(s_0 \to s, k, \pi) \sum_a \nabla \pi(a \mid s)\, q_{\pi}(a, s)$$
$$= \sum_s \eta(s) \sum_a \nabla \pi(a \mid s)\, q_{\pi}(a, s)$$

where $\sum_s \eta(s)$ cane rewritten as

$$\sum_s \eta(s) = \sum_{\bar{s}} \sum_s \frac{\eta(\bar{s})}{\eta(\bar{s})} \eta(s) = \sum_{\bar{s}} \eta(\bar{s}) \sum_s \frac{\eta(s)}{\sum_{\bar{s}} \eta(\bar{s})}$$
$$= \sum_{\bar{s}} \eta(\bar{s}) \sum_s \eta(s)$$

So

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a \nabla \pi(a \mid s)\, q_{\pi}(a, s)$$

It is

$$\frac{\partial}{\partial \theta_t} J(\theta_t) \propto \sum_s \mu(s) \sum_a q_{\pi}(s, a) \frac{\partial}{\partial \theta_t} \pi(a \mid s, \theta_t)$$