Actor-critic methods

David Krueger

Modified from slides by Milica Gašić

In this lecture...

Actor Critic Methods

Least-Squares Policy Iteration

"Soft" actor-critic (SAC) [Haarnoja et al. 2018]

Natural actor-critic

Relation to other RL methods

Value-based methods:

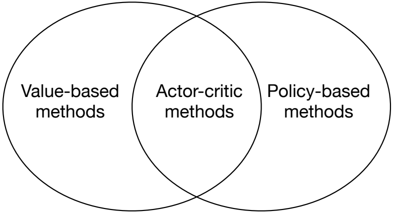
- estimate the value function
- policy is implicit (eg ϵ-greedy)

Policy-based methods

- estimate the policy
- no value function

Actor-critic methods

- estimate the policy ("actor")
- estimate the value function ("critic")



Actor-critic methods

- Actor-critic methods implement *generalised policy iteration* - alternating between a policy evaluation and a policy improvement step.

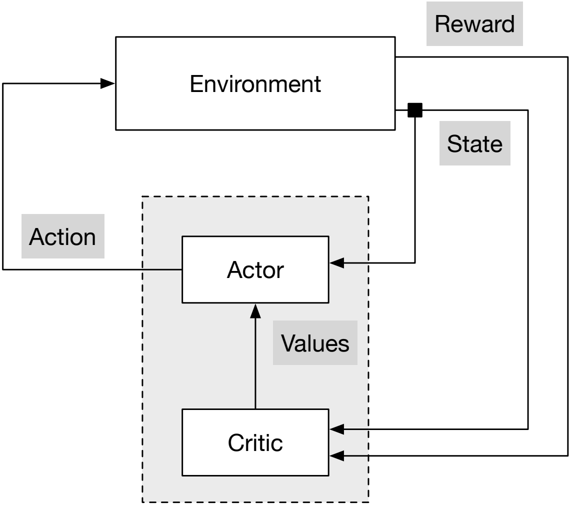
  in practice these may happen simultaneously

- There are two closely related processes of

  actor improvement which aims at improving the current policy

  critic evaluation which evaluates the current policy

Actor-critic architecture



Behaviour vs target policy for actor-critic methods

- The policy used to generate the samples (*behaviour policy*) could be different from the one which is evaluated and improved (*target policy*).
- Want behavior policy to be random (for exploration, coverage)
- ...but not too random (won't get anywhere interesting)
- Good choice of behavior policy: noisier target policy.

Implementing a critic

- The critic estimates the Q-values of the current policy
- For small state-spaces we could use tabular TD algorithms to estimate the Q-function (SARSA, Q-learning, etc)
- For large state-spaces we could use LSTD or dyna / experience replay to estimate the Q-function.

Implementing actor-critic architecture

Small state-action space The critic is a Q-function estimator and the actor is ϵ-greedy or Boltzmann policy estimated in a tabular way.

Large state-action spaces Both the critic and the actor use function approximation

Implementing an actor

Policy improvement can be implemented in two ways:

greedy improvement Moving the policy towards the greedy policy underlying the Q-function estimate obtained from the critic

policy gradient Perform policy gradient directly on the performance surface underlying the chosen parametric policy class

Greedy improvement

- For small state-action spaces the policy is greedy with respect to the obtained Q-value
- For large state-action spaces the policy is parametrised and the greedy action is computed on the fly

Least-Squares Policy Iteration

Algorithm 1 Least-Squares Policy Iteration

Input: parametrisation of $Q(\cdot, \cdot; \boldsymbol{\theta}) = \boldsymbol{\theta}^\mathsf{T} \phi(\cdot, \cdot)$
Initialise $\boldsymbol{\theta}$ arbitrarily
repeat
    $\pi(s) = \arg\max_a \boldsymbol{\theta}^\mathsf{T} \phi(s, a)$ {policy improvement}
    $\boldsymbol{\theta} = LSTD(\pi, \phi, \boldsymbol{\theta})$ {policy evaluation}
until convergence

Policy gradient

- Policy gradient methods perform stochastic gradient descent on the performance surface of the parametrised policy.
- Policy gradient theorem (last lecture) gives

$$
\begin{aligned}
\nabla J(\boldsymbol{\omega}) &= E_\pi\left[\gamma^t R_t \nabla_{\boldsymbol{\omega}} \log \pi(a|s,\boldsymbol{\omega})\right] && (1)\\
&= E_\pi\left[\gamma^t Q_\pi(s,a) \nabla_{\boldsymbol{\omega}} \log \pi(a|s,\boldsymbol{\omega})\right] && (2)\\
&= E_\pi\left[\gamma^t \left(Q_\pi(s,a) - V_\pi(s)\right) \nabla_{\boldsymbol{\omega}} \log \pi(a|s,\boldsymbol{\omega})\right] && (3)
\end{aligned}
$$

- **Advantage function** $A_\pi(s,a)$ is defined as

$$A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$$

Compatible function approximation

See page 28-31 of
`https://web.stanford.edu/class/cme241/lecture_slides/PolicyGradient.pdf`

Note differences in notation!

The Boltzmann Distribution

- Generalization of Softmax function to continuous inputs
- $\mathrm{Softmax}(x_1, ..., x_n) = \frac{\exp x_i}{\Sigma_i \exp x_i}$
- $\mathrm{Boltzmann}(f(x)) = \frac{\exp f(x)}{\int_{\mathbb{R}^n} \exp f(x)}$

Boltzmann Rationality

- Perfect rationality selects $\mathrm{argmax}(f(x))$
- Boltzmann rationality selects $\mathrm{Boltzmann}(f(x)/\tau)$
- τ is called *temperature* (from physics)
- $\tau \to \infty$: Uniform random behavior
- $\tau \to 0$: perfect rationality

"Soft" actor-critic (SAC) [Haarnoja et al. 2018]

- Use Boltzmann-rational target policy instead of perfectly rational target policy.
- Removes discontinuities in map $Q \to \pi$.
- Stabilizes training, state-of-the-art Deep RL method.
- Can be motivated via a modified reward function: $\tilde{\mathcal{R}} \doteq \mathcal{R} + \mathcal{H}(\pi)$ ("maximum entropy RL")

Natural actor-critic [Peters and Schaal, 2008]

- Uses compatible function approximation for actor and critic
- A modified form of gradient – *natural gradient* is used to find the optimal parameters

Natural Policy Gradient

- Advantage function is parametrised with parameters $\boldsymbol{\theta}$ such that the direction of change is the same as for the policy parameters $\boldsymbol{\omega}$

$$\gamma^t \nabla_{\boldsymbol{\theta}} A(s_t, a, \boldsymbol{\theta}) = \nabla_{\boldsymbol{\omega}} \log \pi(s_t, a, \boldsymbol{\omega})$$

- Then by replacing

$$\gamma^t A(s_t, a, \boldsymbol{\theta}) = \nabla_{\boldsymbol{\omega}} \log \pi(s_t, a, \boldsymbol{\omega})^{\mathsf{T}} \boldsymbol{\theta}$$

  in Eq 3
- It can be shown

$$\boldsymbol{\theta} = G_{\boldsymbol{\omega}}^{-1} \nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega})$$

  where $G_{\boldsymbol{\omega}}$ is the Fisher information matrix

$$G_{\boldsymbol{\omega}} = E_{\pi(\boldsymbol{\omega})} \left[\nabla \log \pi(\mathbf{b}, a, \boldsymbol{\omega}) \nabla \log \pi(\mathbf{b}, a, \boldsymbol{\omega})^{\mathsf{T}} \right]$$

- $\boldsymbol{\theta}$ is the natural gradient of $J(\boldsymbol{\omega})$

Natural gradient [Amari, 1998]

- Distance in Riemann space: $|d\omega|^2 = d\omega^\mathsf{T} G_\omega d\omega$, where G_ω is a metric tensor
- Direction of steepest descent in Riemann space for some loss function $L(\omega)$ is $G_\omega^{-1} \nabla_\omega L(\omega)$
- If ω is used to optimise the estimate of a probability distribution $p(x|\omega)$ then the optimal metric tensor is Fisher information matrix as this give distances invariant to scaling of the parameters.

$$G_\omega = E(\nabla \log p(x|\omega) \nabla \log p(x|\omega)^\mathsf{T})$$

- It can be shown that $KL(p(x|\omega)||p(x|\omega + d\omega)) \approx d\omega^\mathsf{T} G_\omega d\omega$

Episodic Natural Actor Critic

Algorithm 2 Episodic Natural Actor Critic

Input: parametrisation of $\pi(\boldsymbol{\omega})$
Input: parametrisation of $\gamma^t A(\boldsymbol{\theta}) = \boldsymbol{\theta}^\mathsf{T}\boldsymbol{\phi}$
Input: step size $\alpha > 0$
Initialise $\boldsymbol{\omega}$ and $\boldsymbol{\theta}$
repeat
    Execute the episode according to the current policy $\pi(\boldsymbol{\omega})$
    Obtain sequence of states s_t, actions a_t and return R
    Critic evaluation Choose $\boldsymbol{\theta}$ and J to minimise $(\sum_t \boldsymbol{\theta}^\mathsf{T}\phi(s_t, a_t) + J - R)^2$
    Actor update $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + \alpha\boldsymbol{\theta}$
until convergence

In practice the update is not performed after every episode but rather after a number of episodes to improve stability and efficiency.

Summary

- Actor-critic methods implement generalised policy iteration where the actor aims at improving the current policy and the critic evaluates the current policy.
- For large state-action spaces, both the actor and the critic are parametrised functions.
- The actor and the critic can be estimated using compatible function approximation, where their parameters depend on each other and are estimated using stochastic gradient descent.
- Instead of the vanilla gradient which has low convergence rates, the natural gradient can be used and this yields natural actor-critic algorithm.

Next lecture

- Deep reinforcement learning
- To prepare for the next lecture please read
  - *Mastering the game of Go with deep neural networks and tree search*, http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html
  - *Mastering the game of Go without human knowledge*, https://www.nature.com/articles/nature24270

References I

Amari, S.-I. (1998).
Natural gradient works efficiently in learning.
Neural Comput., 10(2):251–276.

Peters, J. and Schaal, S. (2008).
Natural actor-critic.
Neurocomputing, 71(7):1180–1190.