Part 8: Actor-Critic algorithms

EL 2805 – Reinforcement Learning

Alexandre Proutiere

KTH, The Royal Institute of Technology

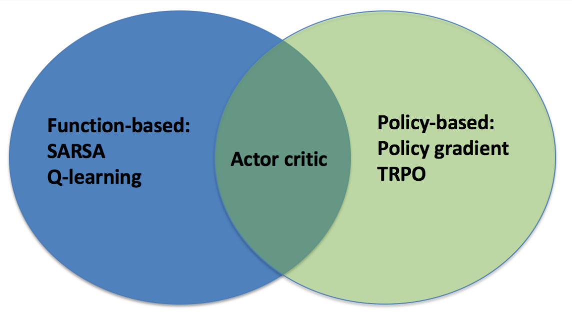
Objectives of this part

- Going back to policy gradient methods
- Solve infinite horizon problem using a *critic*

References

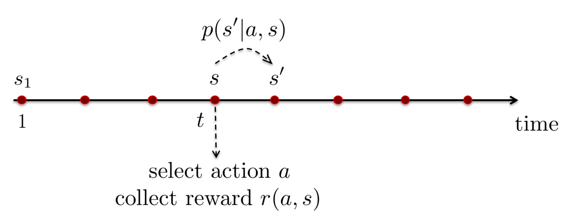
- Barto-Sutton's book chapter 13
- David Silver's lecture 7
- Sutton, McAllester, Singh, Mansour (1999). Policy gradient methods for reinforcement learning with function approximation: actor-critic algorithms

Actor-critic methods



- Function-based methods: evaluate the Q-function or the (state, action) value function of a policy to be improved
- Policy-based methods: a direct gradient on the policy
- Actor-critic methods: a policy-gradient method where function evaluation is needed

Infinite horizon discounted RL problems



Discounted RL problems:

- **Unknown** stationary transition probabilities $p(s'|s, a)$ and rewards $r(s, a)$, uniformly bounded: $\forall a, s,\ |r(s, a)| \le 1$
- Objective: for a given discount factor $\lambda \in [0, 1)$, from the data, find a policy $\pi^\star \in MD$ maximizing (over all possible policies)

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=1}^{T} \lambda^{t-1} r(s_t^\pi, a_t^\pi,)|s_1^\pi = s\right]$$

Objective function and its gradient

Objective: maximize $J(\theta) = \mathbb{E}_{s_1 \sim p}[V^{\pi_\theta}(s_1)]$
Discounted stationary distribution ρ_θ under π_θ:

$$\forall s \in \mathcal{S}, \quad \rho_\theta(s) = (1-\lambda) \sum_{s'} p(s') \sum_{k=0}^{\infty} \lambda^k \mathbb{P}_{\pi_\theta}[s_k = s | s_1 = s']$$

Theorem. *The gradient of the objective function w.r.t. the policy is:*

$$\nabla J(\theta) = \frac{1}{1-\lambda} \mathbb{E}_{s \sim \rho_\theta, a \sim \pi_\theta(s,\cdot)} \left[\nabla \log \pi_\theta(s,a) Q^{\pi_\theta}(s,a) \right]$$

Policy evaluation

How can we evaluate Q^{π_θ}?

TD learning and function approximation $Q^{\pi_\theta} \approx Q_\phi$:
when the experience (s, a, r, s', a') is observed, update ϕ following the semi-gradient descent algorithm

$$\phi \leftarrow \phi + \beta(r + \lambda Q_\phi(s', a') - Q_\phi(s, a))\nabla_\phi Q_\phi(s, a)$$

Q Actor-critic algorithm

QAC Algorithm:

1. **Initialization:** θ, ϕ, state $s = s_1$
2. **Iterations:** Loop
   Take action $a \sim \pi_\theta(s, \cdot)$
   Observe r, s' (reward, next state)
   Sample the next action $a' \sim \pi_\theta(s', \cdot)$
   Update the parameters

$$\phi \leftarrow \phi + \beta(r + \lambda Q_\phi(s', a') - Q_\phi(s, a))\nabla_\phi Q_\phi(s, a)$$
$$\theta \leftarrow \theta + \alpha\left(\nabla_\theta \log \pi_\theta(s, a) Q_\phi(s, a)\right)$$

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

Reducing the variance: baseline

Instead of $Q^{\pi_\theta}(s,a)$, use $Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$.

Advantage: $A^{\pi_\theta}(s,a) = Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$

$$\nabla_\theta J(\theta) = \frac{1}{1-\lambda}\mathbb{E}_{s\sim\rho_\theta, a\sim\pi_\theta(s,\cdot)}\left[\nabla \log \pi_\theta(s,a) A^{\pi_\theta}(s,a)\right]$$

Now when (s,a,r,s',a') is observed under π_θ, we get:

$$A^{\pi_\theta}(s,a) = r + \lambda\mathbb{E}[V^{\pi_\theta}(s')] - V^{\pi_\theta}(s)$$

Hence we can use and fit $V^{\pi_\theta} \approx V_\phi$ only! Using TD learning we get the following update:

$$\phi \leftarrow \phi + \beta(r + \lambda V_\phi(s') - V_\phi(s))\nabla_\phi V_\phi(s)$$

Actor-critic algorithm with baseline

AC Algorithm:

1. **Initialization:** θ, ϕ, state $s = s_1$
2. **Iterations:** Loop

   Take action $a \sim \pi_\theta(s, \cdot)$

   Observe r, s' (reward, next state)

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

   Update the parameters

$$\phi \leftarrow \phi + \beta(r + \lambda V_\phi(s') - V_\phi(s))\nabla_\phi V_\phi(s)$$
$$\theta \leftarrow \theta + \alpha \left(\nabla_\theta \log \pi_\theta(s, a)(r + \lambda V_\phi(s') - V_\phi(s))\right)$$

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

Examples

A3C: Asynchronous Advantage Actor Critic algorithm (better than DQN?)
https://arxiv.org/pdf/1602.01783.pdf

https://youtu.be/xXP77QiHFTs
https://www.youtube.com/watch?v=gMpK7I0vHUc&t=76s