CSCI S-89C Deep Reinforcement Learning

Harvard Summer School

Dmitry Kurochkin

Spring 2020
Lecture 4

Contents

Contents

Policy Iteration: Solving for $v_*(s)$

Bellman equation for v_π:

$$v_\pi(s) = \sum_a \pi(a|s) \underbrace{\sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right]}_{q_\pi(s,a)}$$

Bellman equation for v_*:

$$v_*(s) = \max_{a \in \mathcal{A}(s)} \underbrace{\sum_{s',r} p(s',r|s,a)\left[r + \gamma v_*(s')\right]}_{q_*(s,a)}$$

Fixed point iteration to solve $x = f(x)$:

1. initialize x_0
2. for $k \geq 0$ compute $x_{k+1} \doteq f(x_k)$

Policy Iteration: Solving for $v_*(s)$

Let $v_k(s)$, $k = 0, 1, 2, \ldots$ denote an estimate of $v_*(s)$. The fixed-point iteration can be written as follows:

$$v_{k+1}(s) \doteq \max_{a \in \mathcal{A}(s)} \sum_{s', r} p(s', r|s, a) \left[r + \gamma v_k(s')\right]$$

Policy Iteration: Solving for $v_*(s)$

Let $v_k(s)$, $k = 0, 1, 2, \ldots$ denote an estimate of $v_*(s)$. The fixed-point iteration can be written as follows:

$$v_{k+1}(s) \doteq \max_{a \in \mathcal{A}(s)} \sum_{s', r} p(s', r | s, a) \left[r + \gamma v_k(s') \right]$$

Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $v \leftarrow V(s)$
| $V(s) \leftarrow \max_a \sum_{s',r} p(s', r | s, a)\left[r + \gamma V(s')\right]$
| $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
$\pi(s) = \operatorname{argmax}_a \sum_{s',r} p(s', r | s, a)\left[r + \gamma V(s')\right]$

Source: *Reinforcement Learning: An Introduction* by R. Sutton and A. Barto

Contents

$$\pi_0 \xrightarrow[\text{Evaluation}]{\text{Policy}} v_{\pi_0} \xrightarrow[\text{Improvement}]{\text{Policy}} \pi_1 \xrightarrow[\text{Evaluation}]{\text{Policy}} \dots \xrightarrow[\text{Improvement}]{\text{Policy}} \pi_*$$

$$\pi_0 \xrightarrow[\text{Evaluation}]{\text{Policy}} v_{\pi_0} \xrightarrow[\text{Improvement}]{\text{Policy}} \pi_1 \xrightarrow[\text{Evaluation}]{\text{Policy}} \dots \xrightarrow[\text{Improvement}]{\text{Policy}} \pi_*$$

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
      $\Delta \leftarrow 0$
      Loop for each $s \in \mathcal{S}$:
         $v \leftarrow V(s)$
         $V(s) \leftarrow \sum_{s',r} p(s', r \,|\, s, \pi(s)) \big[r + \gamma V(s') \big]$
         $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

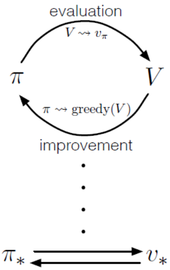
3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
      old-action $\leftarrow \pi(s)$
      $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r \,|\, s, a) \big[r + \gamma V(s') \big]$
      If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

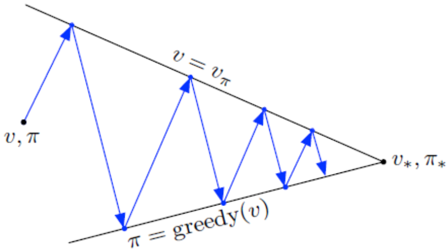
Source: *Reinforcement Learning: An Introduction* by R. Sutton and A. Barto

Contents

GPI

Policy Iteration is an example of Generalized Policy Iteration:





Source: *Reinforcement Learning: An Introduction* by R. Sutton and A. Barto

Contents

Estimating $v_\pi(s)$ via MC Simulation

We notice that $E_\pi\left[G_t | S_t\right] = v_\pi(s)$, then all we need is to generate G_t under policy π and use them to estimate $v_\pi(s)$:

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated

Initialize:

  $V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$

  $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):

  Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$

  $G \leftarrow 0$

  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

    $G \leftarrow \gamma G + R_{t+1}$

    Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:

      Append G to $Returns(S_t)$

      $V(S_t) \leftarrow \text{average}(Returns(S_t))$

Source: *Reinforcement Learning: An Introduction* by R. Sutton and A. Barto

Contents

Estimating $\pi_*(s)$: MC Control with Exploring Starts

Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$

Initialize:

$\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Loop forever (for each episode):

Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}(S_0)$ randomly such that all pairs have probability > 0

Generate an episode from S_0, A_0, following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

Source: *Reinforcement Learning: An Introduction* by R. Sutton and A. Barto

Contents

Estimating $\pi_*(s)$: MC Control without Exploring Starts

Define ε-soft policy as a policy with $\pi(a|s) \geq \frac{\varepsilon}{\mathcal{A}(s)}$.

Estimating $\pi_*(s)$: MC Control without Exploring Starts

Define ε-soft policy as a policy with $\pi(a|s) \geq \frac{\varepsilon}{\mathcal{A}(s)}$.

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

- $\pi \leftarrow$ an arbitrary ε-soft policy
- $Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
- $Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

- Generate an episode following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
- $G \leftarrow 0$
- Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
  - $G \leftarrow \gamma G + R_{t+1}$
  - Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:
    - Append G to $Returns(S_t, A_t)$
    - $Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
    - $A^* \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken arbitrarily)
    - For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

Source: *Reinforcement Learning: An Introduction* by R. Sutton and A. Barto

Estimating $\pi_*(s)$: MC Control without Exploring Starts

Is the ε-greedy policy an improvement of an ε-soft policy π?

$$
\begin{aligned}
q_\pi(s, \pi'(s)) &= \sum_a \pi'(a|s) q_\pi(s, a) \\
&= \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s, a) + (1 - \varepsilon) \max_a q(s, a) \\
&\geq \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s, a) + (1 - \varepsilon) \sum_a \frac{\pi(a|s) - \frac{\varepsilon}{\mathcal{A}(s)}}{1 - \varepsilon} q(s, a) \\
&= \sum_a \pi(a|s) q(s, a) \\
&= v_\pi(s)
\end{aligned}
$$

The policy improvement theorem applies!

Contents

Target Policy v.s. Behavior Policy

1. The goal is to learn *target* policy $\pi(a|s)$, usually deterministic greedy policy
2. Off-policy methods estimate *target* policy using data generated according to *behavior* policy $b(a|s)$
3. Off-policy methods usually have larger variance and slower convergence
4. On-policy learning, i.e. whenever $b(a|s) = \pi(a|s)$, is a special case of off-policy
5. Off-policy methods can be used to learn from available data generated by a non-learning controller
6. Assumption of coverage: $\pi(a|s) > 0 \;\Rightarrow\; b(a|s) > 0$

Contents

Importance Sampling

If we start in state S_t and follow policy π, then

$$
\begin{aligned}
& P_\pi\{A_t, S_{t+1}, A_{t+1}, \dots, S_T | S_t\} \\
& \quad = P_\pi\{A_t | S_t\} P_\pi\{S_{t+1}, A_{t+1}, \dots, S_T | S_t, A_t\} \\
& \quad = P_\pi\{A_t | S_t\} P_\pi\{S_{t+1} | S_t, A_t\} P_\pi\{A_{t+1}, \dots, S_T | S_t, A_t, S_{t+1}\} \\
& \quad = \pi(A_t | S_t) p(S_{t+1} | S_t, A_t) P_\pi\{A_{t+1}, \dots, S_T | S_{t+1}\} \\
& \qquad \vdots \\
& \quad = \pi(A_t | S_t) p(S_{t+1} | S_t, A_t) \pi(A_{t+1} | S_{t+1}) \dots p(S_T | S_{T-1}, A_{T-1}) \\
& \quad = \prod_{k=t}^{T-1} \pi(A_k | S_k) p(S_{k+1} | S_k, A_k)
\end{aligned}
$$

Importance Sampling

For any policy $\pi(a|s)$:

$$P_\pi\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T | S_t\} = \prod_{k=t}^{T-1} \pi(A_k|S_k) p(S_{k+1}|S_k, A_k)$$

Importance Sampling

For any policy $\pi(a|s)$:

$$P_\pi\{A_t, S_{t+1}, A_{t+1}, \dots, S_T | S_t\} = \prod_{k=t}^{T-1} \pi(A_k|S_k) p(S_{k+1}|S_k, A_k)$$

Define the importance-sampling ratio:

$$\begin{aligned}
\rho_{t:(T-1)} &\doteq \frac{P_\pi\{A_t, S_{t+1}, A_{t+1}, \dots, S_T | S_t\}}{P_b\{A_t, S_{t+1}, A_{t+1}, \dots, S_T | S_t\}} \\
&= \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k) p(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k) p(S_{k+1}|S_k, A_k)} \\
&= \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}
\end{aligned}$$

Given trajectories, $\rho_{t:(T-1)}$ depends on the policies only!

Importance Sampling

We notice that under policy b,

$$E_b\left[G_t|S_t=s\right]=v_b(s),$$

but the expected transformed cumulative discounted return is

$$E_b\left[\rho_{t:(T-1)}G_t|S_t=s\right]=E_\pi\left[G_t|S_t=s\right]=v_\pi(s),$$

where data were generated under b.