Lecture 21 MDPs + RL (first look) CS181

Markov Decision Process

"Model" (S , A , r , p) Policy $\pi(s) \in A$

states actions reward transition

$\boxed{\text{Planning}}$ $\boxed{\text{Reinforcement Learning}}$

Given model, output a policy.

☑ Finite horizon

☑ Infinite horizon

☑ Planning - Infinite Horizon

$\gamma \in [0,1)$ "Discount factor"

MDP value function

$$V^{\pi}(s) = \mathbb{E}_{s \sim p}\left[\sum_{t=0}^{\infty} \gamma^{t} r(s_t, \pi(s_t)) \,\middle|\, s_0 = s \right]$$

Policy π is better than or equal to policy π' if $V^{\pi}(s) \geq V^{\pi'}(s)$ in all states s.

Optimal policy π^* better than or equal to all policies π.

Optimal value function

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

Bellman equation / Principle of Optimality

$$(\square) \quad V^*(s) = \max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V^*(s') \right]$$

Value state = Value of the optimal action + Value of continuing optimally

② Algorithm 1: Value Iteration

Solve $(\square)$ iteratively

Initialize $V(s) = 0$, all states s

Repeat $\quad V'(s) \leftarrow \max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V(s') \right]$

all s

$V(s) \leftarrow V'(s)$, all s

Theorem VI converges to the optimal value function V^*

Note Policy extraction

$$\pi^*(s) \in \arg\max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V^*(s') \right]$$

Idea Show that $\underline{V}' \leftarrow B(\underline{V})$, for "Bellman operator" B, is a contra 'on.

|(Aside)

Consider $f: \mathbb{R}^D \to \mathbb{R}^D$, update $x' \leftarrow f(x)$, fixpoint $f(x^*) = x^*$.

Function f is a contraction when

$$\| f(x) - f(y) \| < \| x - y \| \quad , \quad x \neq y$$

eg., $f(x) = \frac{x}{2}$. $(8, 2)$ $(4, 1)$ $(2, \frac{1}{2})$ ...

fixpoint $x^* = 0$

Theorem Given contraction property, then $f(x)$ has a unique fixpoint and $x' \leftarrow f(x)$ converges.

Proof ▫ f has unique fixpoint, else

$$\| f(x^*) - f(y^*) \| = \| x^* - y^* \| \text{, violate contraction}$$

▫ f ~~has~~ converge to fixpoint; consider $x \neq x^*$

$$\| f(x) - x^* \| = \| f(x) - f(x^*) \| < \| x - x^* \|$$

Fact Bellman operator $\mathcal{B}$ is a contraction for norm $\|\underline{V}\| = \max_s |V(s)|$

Comments

1. VI converges to V^* asymptotically
2. Optimal Poliy extracted from V in a finite # of steps
3. Extraction

$$\pi(s) \in \arg\max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V(s) \right]$$

③ Algorithm 2 : Policy Iteration

$$\pi^{(0)} \xrightarrow{E} V^{(0)} \xrightarrow{I} \pi^{(1)} \xrightarrow{E} V^{(1)} \xrightarrow{I} \pi^{(2)} \longrightarrow$$

Initialize $\pi^{(0)}$ (arbitrary)

Repeat ① Evaluate V^{π} (π is current policy)

② improve:

$$\pi'(s) \leftarrow \arg\max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V^{\pi}(s') \right]$$

for all s

$$\pi \leftarrow \pi'$$

Theorem PI converges to optimal policy in a finite # of steps ($\Rightarrow$ also get optimal value function)

Note Suppose for state s, exists an a

$$r(s,a) + \gamma \sum_{s'} p(s'|s,a) V^{\pi}(s') > V^{\pi}(s)$$

(taking action once, then following π is better than following π)

Can show that $V^{(k+1)} > V^{(k)}$ if the policy changes.

Notes

① Policy evaluation

Given π, can solve a system of equations to get V^π.

$$V^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s'} p(s'|s, \pi(s))\, V^\pi(s') \text{; all } s$$

$|S|$ unknowns, $|S|$ equations

In vector form:

$$\underline{V}^\pi = \underline{R}^\pi + \gamma\, \underline{P}^\pi\, \underline{V}^\pi, \quad \underline{P}^\pi \text{ is } |S| \times |S| \text{ transition matrix}$$

$$\Leftrightarrow (\underline{I} - \gamma \underline{P}^\pi)\underline{V}^\pi = \underline{R}^\pi$$

$$\Leftrightarrow \underline{V}^\pi = \underbrace{(I - \gamma \underline{P}^\pi)}_{\text{full rank}}{}^{-1}\, \underline{R}^\pi$$

② PI requires fewer iterations than VI to solve for optimal π^*

Comparison:

$\boxed{\text{VI}}$ $$V'(s) \leftarrow \max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V(s') \right]$$

$O(|S||A| L)$ per iteration

L is the max # reachable next states

$\boxed{\text{PI}}$ $$\pi'(s) \leftarrow \arg\max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V^{\pi}(s') \right]$$

$$O(|S||A| L + \underbrace{|S|^3}_{\text{policy evaluation}}) \text{ per iteration}$$

Reinforcement learning

Learn from environment, no knowledge of r or p ("the model").

New challenge: explore (learning new things)
vs. exploit (leverage what you know to do well)

Two main approaches

Model-Based

- → Learn model (predict next state, reward)
- → Use planning, to decide how to act

✓ Can accomodate changes in rewards, transitions

✗ Costly (lots of computation)

Model-free

- → Don't learn model
- → Directly learn a policy, or an action-value function (Q-function)

✓ Simple, inexpensive computation

✗ If things change, have to "act a lot" to learn a new behavior

(5) Model-free RL: Value-based methods

Learn Q-function

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) V^{\pi}(s')$$

value of taking a, followed by policy π

$$Q^{*}(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) V^{*}(s')$$

$$\pi^{*}(s) = \arg\max_{a} Q^{*}(s,a)$$

<u>Bellman Equation</u>

$$Q^{*}(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \overbrace{\max_{a'} Q^{*}(s',a')}^{V^{*}(s')}$$

$\boxed{\text{Reinforcement Learner}}$

Initialize $Q(s,a)$ values (tabular)

Repeat:

1. Act based on Q values
2. Use s, a, r, s', a' to update Q values

[Better approximate Q^{*}]

① Act: ε-greedy agent

$$\pi(s) \leftarrow \begin{cases} \arg\max_a Q(s,a) & \text{w. prob } 1-\varepsilon \\ \text{random} & \text{w. prob } \varepsilon \end{cases}$$

② Learn Q^* through "Temporal difference" updates

☒ SARSA (on policy) Each time get new experience s, a, r, s', a'

$$Q(s,a) \leftarrow Q(s,a) + \underbrace{\alpha_t}_{\substack{\text{learning rate}\\ \text{(period } t)}} \underbrace{\Big[\underbrace{r + \gamma Q(s',a')}_{\text{1-step estimate of } Q^\pi(s,a)} - Q(s,a)\Big]}_{\text{TD-error}}$$

☒ Q-LEARNING (off-policy) Each time get new experience (s, a, r, s')

$$Q(s,a) \leftarrow Q(s,a) + \alpha_t \underbrace{\Big[\underbrace{r + \gamma \max_{a'} Q(s',a')}_{\text{1 step estimate of } Q^*(s,a)} - Q(s,a)\Big]}_{\text{TD-error}}$$

☑ SARSA is "on policy" because it estimates Q^π for policy followed (including ε-exploration)

☑ Q-LEARING is "off policy" because it estimates Q^* while following π
(converge to Q^* as long as (s,a) visited often enough)