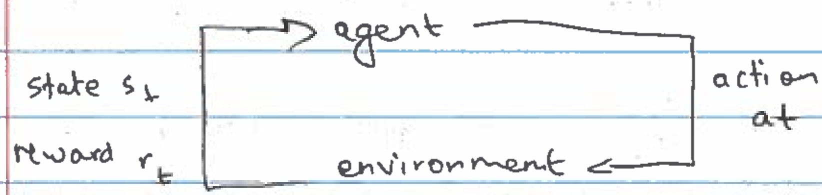
Reinforcement learning

07/21



* Challenges
  - uncertainty
  - explore or exploit
  - delayed v/s immediate feedback
  - evaluative vs instructive feedback
  - complex worlds, computational guarantees

Markov decision processes (MDPs)

* Definition
  - state space $\mathcal{S}$ with states $s \in \mathcal{S}$
  - action space $\mathcal{A}$ with actions $a \in A$
  - transition probabilities

    ↓

    for all state action pairs (s, a),

    $$P(s'|s,a) = P(s_{t+1} = s' \mid s_t = s, a_t = a)$$

    prob moving from state s to state s' after taking action a.

* Assumptions:

- time-independent

$$P(s_{t+1}=s' \mid s_t=s, a_t=a)$$
$$= P(s_{t+\Delta+1}=s' \mid s_{t+\Delta}=s, a_{t+\Delta}=a)$$

- Markov condition

$$P(s_{t+1} \mid s_t, a_t) = P(s_{t+1} \mid s_t, a_t, \overbrace{s_{t-1}, a_{t-1}, s_{t-2} \ldots}^{\text{additional history}})$$

conditional independence.

* Defn (contd)

- reward function.

$R(s, s', a)$ = real-valued reward signal after taking action a in state s and moving to state s'.

- Simplifications for CSE 150

- discrete, finite state space $\mathcal{S}$
- discrete, finite action space A

} vs continuous, infinite

- reward function $R(s, s', a) = R(s) = R_s$ only depends on current state
- bounded, deterministic rewards, $\max_s |R_s| < \infty$

EX: board game with dice (eg. backgammon, monopoly, ..)

$\mathcal{S}$ = board position and roll of dice (right before agent decides to move)

$P(s' \mid s, a)$ = how state changes due to agent's move, opponent rolls dice, opponent's move, agent rolls dice

rewards : $R(s) = \begin{cases} +1 & \text{win} \\ -1 & \text{lose} \\ 0 & \text{otherwise (at any other move)} \end{cases}$

$MDP = \{ \mathcal{S}, A, P(s'|s,a), R(s) \}$

* Decision-making

policy : deterministic mapping from state to actions

$\pi : \mathcal{S} \to A$

policies $= |A|^{|\mathcal{S}|}$ exponential in # states

- dynamics under policy π

$P(s' \mid s, \underbrace{\pi(s)}_{\text{action taken by agent in state } s})$

- experience under policy π

state $s_0 \xrightarrow[\text{action}]{a_0 = \pi(s_0)} s_1 \xrightarrow{a_1 = \pi(s_1)} s_2 \longrightarrow \cdots$

reward r_0 $\quad r_1 \quad r_2$

* How to measure accumulated rewards over time?

long-term discounted return

discount factor $0 \leq \gamma < 1$

$$\text{return} = \sum_{t=0}^{\infty} \gamma^t r_t$$

Possibilities

$\gamma = 0 \rightarrow$ only immediate reward at $t=0$ matters

$\gamma << 1 \rightarrow$ near-sighted agents

$\gamma \simeq 1 \rightarrow$ far-sighted agents

Intuitively : $\gamma < 1$

- near future weighted more heavily than distant future
- confidence in MDP as model of real world may diminish with far off predictions

Also, mathematically convenient : leads to recursive algorithms whose convergence is calculated by γ.

$\boxed{\text{State value function}}$

$V^{\pi}(s)$ = expected discounted return following policy π from initial state s.

$$V^{\pi}(s) = E^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid s_0 = s\right]$$

expectation operator
(Computing mean)

Relating value function in different states:

$$V^{\pi}(s) = E^{\pi}\left[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots \mid s_0 = s\right]$$

$$= R(s) + \gamma E^{\pi}\left[R(s_1) + \gamma R(s_2) + \gamma^2 R(s_3) + \dots \mid s_0 = s\right]$$

$$= R(s) + \gamma \sum_{s'=1}^{n} P(s' \mid s, \pi(s))\, E^{\pi}\left[R(s_1) + \gamma R(s_2) + \dots \mid s_1 = s'\right]$$

$$\boxed{V^{\pi}(s) = R(s) + \gamma \sum_{s'=1}^{n} P(s' \mid s, \pi(s))\, V^{\pi}(s')}$$

Bellman equation

Action-value function

$Q^{\pi}(s,a)$ = expected return from initial state s, taking action a, then following policy π.

$$= E^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid s_0 = s,\ a_0 = a\right]$$

↳ a may equal $\pi(s)$, or not.

$$\boxed{Q^{\pi}(s,a) = R(s) + \gamma \sum_{s'} P(s' \mid s, a)\, V^{\pi}(s')}$$

Optimality in MDPs

Thm: there is always (at least) one optimal policy π^* for which $V^{\pi^*}(s) \geq V^{\pi}(s)$ for all policies π and states $s \in \mathcal{S}$.

Goal : how to compute optimal policy π^* ?

* Optimal state-value function

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

* optimal action-value function

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

Note : $V^*(s) = \max_a Q^*(s,a)$

There may be multiple optimal policies, but optimal value functions are unique.

* Relations - given MDP, how to recover π^* from V^*, Q^* ?

$$\pi^*(s) = \arg\max_a \left[Q^*(s,a) \right]$$

$$= \arg\max_a \left[R(s) + \gamma \sum_{s'} P(s'|s,a) \, V^*(s') \right]$$

$$= \arg\max_a \left[\sum_{s'} R(s'|s,a) \, V^*(s') \right]$$

Planning

Assume complete world of environment as
$MDP = \{\mathcal{S}, A, P(s'|s,a), R(s)\}$, also $\gamma < 1$
how to compute $\pi^*(s)$, or equivalently $V^*(s)$ or $Q^*(s,a)$

i) Policy evaluation

How to compute $V^\pi(s)$?

From Bellman eqn:

$$V^\pi(s) = R(s) + \gamma \sum_{s'=1}^{N} P(s'|s,\pi(s))\, V^\pi(s')$$

for $s = 1, 2, \ldots, N$ where $N = \#$ states of MDP

This is a system of linear equations: N equations, N unknowns $V^\pi(s)$, $s = 1, \ldots N$

~~Putt~~ Put all unknowns on left side:

$$V^\pi(s) - \gamma \sum_{s'=1}^{n} \{P(s'|s,\pi(s))\, V^\pi(s')\} = R(s)$$

$$\sum_{s'=1}^{n} \{[I(s,s') - \gamma P(s'|s,\pi(s))]\, V^\pi(s')\} = R(s)$$

indicator function

Can rewrite this:

$$\underbrace{(I - \gamma P)}_{} V = R$$

I: $n \times n$ identity matrix

P: $n \times n$ matrix

V: $n \times 1$ vector of unknowns

R: $n \times 1$ vector of knowns

Ex: states $s \in \{0, 1\}$

transitions $P(s' | s, \pi(s))$

rewards $R(s) = \begin{pmatrix} r_0 \\ r_1 \end{pmatrix}$

state value function $V^\pi(s) = \begin{pmatrix} v_0 \\ v_1 \end{pmatrix}$

solve:

$$\left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \gamma \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix} \right] \begin{pmatrix} v_0 \\ v_1 \end{pmatrix} = \begin{pmatrix} r_0 \\ r_1 \end{pmatrix}$$

2 eq'ns
2 unknowns

2) Policy improvement

* How to compute π' such that $V^{\pi'}(s) \geq V^\pi(s)$ for all states s?

* Recall $Q^\pi(s, a)$ = expected return from state s, follow action a, then revert to policy π.

How to compute $Q^\pi(s, a)$?

Evaluate policy (step #1) to compute $V^\pi(s)$.

Then: $Q^\pi(s, a) = R(s) + \gamma \sum_{s'} P(s' | s, a) V^\pi(s')$

* Define "greedy" ~~st~~ policy:

$$\pi'(s) = \text{greedy}[\pi]$$
$$= \arg\max_a [Q^\pi(s,a)]$$
$$= \arg\max_a \left[\sum_{s'} P(s'|s,a)\, V^\pi(s')\right]$$

Thm greedy policy π' everywhere performs better or equal to original policy π

$$V^{\pi'}(s) \geq V^\pi(s) \quad \text{for all } s$$

Intuition: if better to choose action a over $\pi(s)$ in state s before following policy π, it's always better to choose action a in state s (for all times)

Proof:

$$V^\pi(s) = Q^\pi(s, \pi(s))$$
$$\leq \max_a Q^\pi(s,a)$$
$$= Q^\pi(s, \pi'(s))$$
$$= R(s) + \gamma \sum_{s'} P(s'|s, \pi'(s))\, V^\pi(s')$$

So far: better to (take one step under π', then resort to π), than to (always ~~for~~ follow π)

"One-step" inequality.

$$V^\pi(s) \leq R(s) + \gamma \sum_{s'} P(s'|s, \pi'(s))\, V^\pi(s')$$

Apply one-step inequality to itself on the right side:

$$V^{\pi}(s) \leq R(s) + \gamma \sum_{s'} P(s' | s, \pi'(s)) \left[R(s') + \gamma \sum_{s''} P(s'' | s', \pi'(s')) \cdot V^{\pi}(s'') \right]$$

Better to take two steps under π', then revert to π, than to always follow policy π.

Apply this one-step inequality to itself t-times to get t-step inequality: better to take t steps under π' (then revert to π), than to always follow π.

Let $t \to \infty$: always better to follow π' than π. $V^{\pi}(s) \leq V^{\pi'}(s)$ for all states s since right side converges for $\gamma < 1$.