50.021 – AI

Alex

Week 06: Markov Decision Processes II

[The following notes are compiled from various sources such as textbooks, lecture materials, Web resources and are shared for academic purposes only, intended for use by students registered for a specific course. In the interest of brevity, every source is not cited. The compiler of these notes gratefully acknowledges all such sources.]

Key takeaways:

Be able to explain the main ideas behind:

- explain the difference between equations for optimal policy (uses $\max_a T(a)$) and for a given policy (uses $\sum_a T(a)\pi(a|s)$)
- explain the difference between fixpoint computations for optimal policy (uses $\max_a T(a)$) and for a given policy (uses $\sum_a T(a)\pi(a|s)$)
- Q-function $Q^{\pi}(s,a)$
- Q-iteration to get the Q-function $Q^{\pi^*}(s,a)$ for the optimal policy π^*

1 Recap

MDP recap

an MDP is a 4-tuple: S, A, P, R

- have world state space S, action space A
- $P(s'|a,s)$ – transition probability when taking in state s action a to arrive in state s'
- $R(s,a,s')$ reward function (can be a probability e.g. $R(s,a,s') = N(\hat{R}(s,a,s'),\sigma^2)$, where $\hat{R}(s,a,s')$ is a function)

policy

Policy $\pi(a|s)$ – what action to choose given that one is in state s. Can be a probability over all allowed actions.

Policy is just a rule (deterministic as $a = \pi(s)$ or stochastic with $\sum_a \pi(a|s) = 1$) what action a to choose in state s.

What is the performance measure of an agent? **Expected (future) reward**

for unbounded time horizons and in general with a discount factor $\gamma \in (0, 1]$

$$\bar{r}(s) = E_{(\ldots,a_t \sim \pi(a|s_t=s_t),\ldots)}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s]$$

the expectation runs over drawing an action for every time step – future reward is a function of the policy π!

the Value function

The value $V^\pi(s)$ of a policy π is the gamma-discounted expected future reward when starting in state s and continuing according to the policy:

$$V^\pi(s) = E_{(a_0 \sim \pi(a|s_0=s), a_1 \sim \pi(a|s_1=s_1), \ldots, a_t \sim \pi(a|s_t=s_t), \ldots)}[\sum_{t=0}^{T_e} \gamma^t r_t | s_0 = s]$$

A policy π^* is optimal if

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

the goal in MDPs

goal: find a policy π^* which maximizes V^π, the expected reward, averaged over all uncertainties in state transitions, actions and rewards.

The Bellman equation

If rewards and policy are deterministic, that is when the policy π returns one state $a = \pi(s)$ as a function of the current state, then

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

In the general case, replacing any terms depending on an action $T(a = \pi(s)) \rightsquigarrow \sum_a \pi(a|s) T(a)$ by the expectation under probability of an action (, we have:

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

This applies for $T(a) = R(s, a, s')$ and for $T(a) = P(s'|a, s))$

1.1 The Bellman optimality criterion

The Bellman optimality criterion

In an MDP the optimal Value and the optimal policy satisfy the following equations:

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

$$\pi^*(s) = \operatorname{argmax} \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a)V^{\pi^*}(s')$$

1.2 Estimating $V^{\pi^*}(s)$ and $\pi^*(s)$

Ok, we know what condition they should satisfy, ...
By fixpoint iteration! Start with $V_0(s) = 0$. Iterate: Compute $V_{k+1}(s)$ from $V_k(s)$

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

$$V_{k+1}(s) = \max_a \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a)V_k(s')$$

until an iteration k such that: $\max_s |V_{k+1}(s) - V_k(s)| < \delta$

2 Comparison to avoid confusion: want Value of the optimal policy π^* or of a given policy π?

If you want to estimate the value V^{π^*} under **the optimal policy** π^*, then you use

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

$$V_{k+1}(s) = \max_a \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a)V_k(s')$$

If you want the Value V^π under a fixed policy π, then the following Bellman equations hold

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

for a deterministic policy $\pi(s)$. Or for a stochastic policy:

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

As a consequence of that: If you want the Value under a fixed policy π, then this can also be computed by a fixpoint problemm but not using $\max_a$:

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

for a stochastic policy:

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

This converges analogously, as the proof for the Value function under the optimal policy.

Comparison

Value $V^{\pi^*}(s)$ of optimal policy π^*

$$V_{k+1}(s) = \max_a \sum_{s'} P(s'|s, a)R(s, a, s') + \gamma \sum_{s'} P(s'|s, a)V_k(s')$$

Value $V^\pi(s)$ of any given policy π

$$\text{determ: } V_{k+1}(s) = \sum_{s'} P(s'|s, a = \pi(s))R(s, a = \pi(s), s') + \gamma \sum_{s'} P(s'|s, a = \pi(s))V_k(s')$$
$$\text{stoch: } V_{k+1}(s) = \sum_{s'} \sum_{a} P(s'|s, a)R(s, a, s')\pi(a|s) + \gamma \sum_{s'} \sum_{a} P(s'|s, a = \pi(s))\pi(a|s)V_k(s')$$

The difference: for the optimal policy you use $\max_a T(a)$, for a given policy use $T(a = \pi(s))$ or $\sum_a T(a)\pi(a|s)$

3 Q-function

idea of Q-function

Idea: $Q^\pi(s, a)$ is the expected future reward when using action a in state s (not following any policy!), and after that continuing with policy π. This means as a definition:

$$Q^\pi(s, a) = \underbrace{\sum_{s'} P(s'|s, a)R(s, a, s')}_{\text{Reward for doing } a \text{ in state } s} + \underbrace{\gamma \sum_{s'} P(s'|s, a)V^\pi(s')}_{\text{future } E[\cdot] \text{ reward under } \pi \text{ after doing } a \text{ in state } s}$$

Since Value V^π means the expected reward when choosing the action according to the current policy π, the following must hold:

plugging in the policy in Q must result in V:

Q^π-function and Value-function V^π:

$$V^\pi(s) = Q^\pi(s, a = \pi(s))$$
$$V^\pi(s) = \sum_a \pi(a|s) Q^\pi(s, a)$$

This can be plugged in into above definition to yield a definition of Q^π as a fixpoint same as we had for V^π

The Bellman equations for the Q-function:

deterministic policy:

$$\begin{aligned} Q^\pi(s,a) &= \sum_{s'} P(s'|s,a) R(s,a,s') + \gamma \sum_{s'} P(s'|s,a) V^\pi(s') \\ &= \sum_{s'} P(s'|s,a) R(s,a,s') + \gamma \sum_{s'} P(s'|s,a) Q^\pi(s', a' = \pi(s')) \end{aligned}$$

stochastic policy:

$$\begin{aligned} Q^\pi(s,a) &= \sum_{s'} P(s'|s,a) R(s,a,s') + \gamma \sum_{s'} P(s'|s,a) V^\pi(s') \\ &= \sum_{s'} P(s'|s,a) R(s,a,s') + \gamma \sum_{s'} P(s'|s,a) \sum_{a'} \pi(a'|s') Q^\pi(s', a') \end{aligned}$$

Optimal policy means that we choose the optimal action...

Q^{π^*}-function and Value-function V^{π^*} for the optimal policy π^*:

By Bellman-optimality criterion (and the definition of Q^{π^*}) we have **only for the optimal policy** π^*:

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

This one can plug in:

The Bellman optimality for the Q-function and the optimal policy π^*:

$$\begin{aligned} Q^{\pi^*}(s,a) &= \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a)V^{\pi^*}(s') \\ &= \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q^{\pi^*}(s',a') \\ V^{\pi^*}(s) &= \max_a Q^{\pi^*}(s,a) \end{aligned}$$

The difference is the same as for values: for the optimal policy you use $\max_a T(a)$, for a given policy use $T(a=\pi(s))$ or $\sum_a T(a)\pi(a|s)$

Q^π is often easier to analyze than V^π

Q-iteration

Computes the Q-function under the optimal policy π^*.

$$\begin{aligned} Q_{k+1}(s,a) &= \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q_k(s',a') \\ V^{\pi^*}(s) &= \max_a Q^{\pi^*}(s,a) \\ \pi^*(s) &= \text{argmax}_a Q^{\pi^*}(s,a) \end{aligned}$$

Compare:

Bellman-optimality criterion for Value V^{π^*} under optimal policy π^*

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

Bellman-optimality criterion for Q-function Q^{π^*} under optimal policy π^*

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

and here the iteration algorithms:

Value-iteration

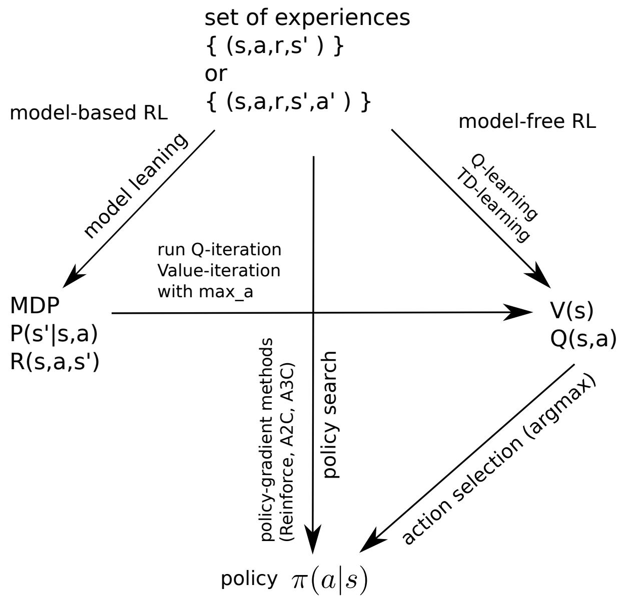
$$V_{k+1}(s) = \max_a \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a)V_k(s')$$

Q-iteration

$$\begin{aligned} Q_{k+1}(s,a) &= \sum_{s'} P(s'|s,a)R(s,a,s') + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q_k(s',a') \\ V^{\pi^*}(s) &= \max_a Q^{\pi^*}(s,a) \end{aligned}$$

Not the same.

4 Overview over some types of reinforcement learning



model-free or model-based RL?

Historically two alternative views were developed: cognitive psychologists like Wolfgang Koehler, Edward Tolman (`https://en.wikipedia.org/wiki/Wolfgang_K%C3%B6hler` `https://en.wikipedia.org/wiki/Edward_C._Tolman`) on the one side vs the behaviorism of e.g. Edward Thorndike, Clark Hull, perfectioned by Burrhus Skinner `https://en.wikipedia.org/wiki/Edward_Thorndike` `https://en.wikipedia.org/wiki/Clark_L._Hull` `https://en.wikipedia.org/wiki/B._F._Skinner`

Is learning in animals based on them learning an insight of cause and effect, which can be seen as a model of the process
`http://wkprc.eva.mpg.de/english/files/wolfgang_koehler.htm`
or did they learn by adapting to stimulus-response scenarios (reinforcement – positive reinforcement by giving positive events, negative reinforcements by removing negative events, e.g. Bart vs Cupcakes) ?

RL carries a lot of ideology from behaviourism.

Draw-backs of a model-free learning approach:

- no generalizable / reusable knowledge, only values for the current goal
- no human-like expectation about future outcomes (what states), only valye

- goal changes – learning starts from scratch

Both schools explain some aspects of learning. Touching a hot plate ...

5 the start of Q-learning

Goal: Learn Q-function $Q^{\pi^*}(s,a)$ for optimal policy π^*

Remember:

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

This translates to:

Q = expected reward + $\gamma\times$ Q for new states s' averaged with probability to land in s'.

What is when we have observed an experience (s,a,r,s') and want to learn from it?
We can replace expected reward by r. We can replace *Q for new states s' averaged ...* by Q in our observed state s', so

$$Q_{\text{current}}(s,a) \approx r + \gamma \max_{a'} Q_{\text{current}}(s',a')$$

in an iterative fashion:

$$\begin{gathered} s,a \rightsquigarrow r,s' \\ Q_{\text{new}}(s,a) \approx r + \gamma \max_{a'} Q_{\text{old}}(s',a') \end{gathered}$$

However we want a slow update, so weight it with α

$$\begin{aligned} Q_{\text{new}}(s,a) &= (1-\alpha)Q_{\text{old}}(s,a) + \alpha\left(r + \gamma \max_{a'} Q_{\text{old}}(s',a')\right) \\ &= Q_{\text{old}}(s,a) + \alpha\left(r + \gamma \max_{a'} Q_{\text{old}}(s',a') - Q_{\text{old}}(s,a)\right) \end{aligned}$$

What is the idea?

$$\begin{aligned} r + \gamma \max_{a'} Q_{\text{old}}(s',a') > Q_{\text{old}}(s,a) &\Rightarrow \text{ increase } Q_{\text{old}}(s,a) \\ r + \gamma \max_{a'} Q_{\text{old}}(s',a') < Q_{\text{old}}(s,a) &\Rightarrow \text{ decrease } Q_{\text{old}}(s,a) \end{aligned}$$

Compare observed reward r plus $\gamma\times$ your estimated Q from the observed new state s' (from experience (s,a,r,s')) against you your estimated $Q(s,a)$ from the observed old state s and update iteratively.

Reinforcement:

- more reward then currently estimated $\Rightarrow$ increase $Q_{\text{old}}(s,a)$
- less reward then currently estimated $\Rightarrow$ decrease $Q_{\text{old}}(s,a)$

Analogously can be executed for the value function.

Q-Learning by TD(0)-Learning for the Q-function

$$
\begin{aligned}
Q_{\text{new}}(s,a) &= (1-\alpha)Q_{\text{old}}(s,a) + \alpha\left(r + \gamma \max_{a'} Q_{\text{old}}(s',a')\right) \\
&= Q_{\text{old}}(s,a) + \alpha\left(r + \gamma \max_{a'} Q_{\text{old}}(s',a') - Q_{\text{old}}(s,a)\right)
\end{aligned}
$$

As algorithm?

- init $Q(s,a)=0$, choose start state s
- run while loop:
  - choose action $a \approx_\epsilon \text{argmax}_a Q(s,a)$
  - observe reward r and new state s' to obtain (s,a,r,s')
  - update $Q(s,a) = Q_{\text{old}}(s,a) + \alpha\,(r + \gamma \max_{a'} Q_{\text{old}}(s',a') - Q_{\text{old}}(s,a))$
  - set oldstate to new state $s = s'$

$a \approx_\epsilon \text{argmax}_a Q(s,a)$ is what ? – so called ϵ-greedy exploration

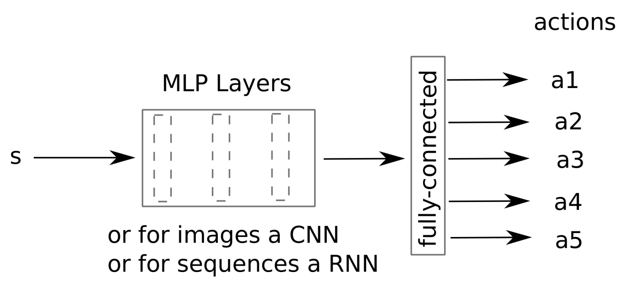
$$
a = \begin{cases} \text{random} & \text{with prob } \epsilon \\ \text{argmax}_a Q(s,a) & \text{else} \end{cases}
$$

Idea: Do not follow strictly your current estimate of best action. Allow a random action with some probability to explore new options.

Limitation: above works for discrete states s. Not if states are continuous.

5.1 Towards Q-learning for continuous states with neural nets and the like

Suppose your states are continuous. Then we can replace a tabular $Q(s,a)$ by a parametrized function $Q_w(s,a)$. Q_w can be e.g. an Multi-layer perceptron with as many outputs as $|A|$, or the like.



How to optimize in that case? We need a loss to compute a gradient.
Can start from:

$$
\begin{aligned}
s, a &\rightsquigarrow r, s' \\
Q_{\text{new}}(s, a) &\approx r + \gamma \max_{a'} Q_{\text{old}}(s', a') \\
\Rightarrow 0 &\approx r + \gamma \max_{a'} Q_{\text{old}}(s', a') - Q_{\text{new}}(s, a)
\end{aligned}
$$

This looks like a regression problem, and it can be approached (this is a simplification by)

$$L((s, a, r, s')) = (r + \gamma \max_{a'} Q_w(s', a') - Q_w(s, a))^2$$

as a loss for a single experience (s, a, r, s'). Now can do minibatch-gradient and SGD training or anything similar.