B(E)4M36SMU

Reinforcement learning 1 - MDP

Monday 20th February, 2023

Why reinforcement learning?

- https://www.youtube.com/watch?v=l8zKZLqkfII
- https://www.youtube.com/watch?v=W_gxLKSsSIE&list=PL5nBAYUyJTrM48dViibyi68urttMlUv7e
- AlphaGo, AlphaZero, . . .
- autonomous helicopter flying
- solving Rubik's cube (2019): https://openai.com/blog/solving-rubiks-cube/

Why reinforcement learning?

- https://www.youtube.com/watch?v=l8zKZLqkfII
- https://www.youtube.com/watch?v=W_gxLKSsSIE&list=PL5nBAYUyJTrM48dViibyi68urttMlUv7e
- AlphaGo, AlphaZero, . . .
- autonomous helicopter flying
- solving Rubik's cube (2019):
  https://openai.com/blog/solving-rubiks-cube/

What makes it successful?

- agent learns from reward and punishment
- inspired by nature (pain/hunger = negative, pleasure/food = positive)

Simplest setting

- sequential decision problem
- fully observable environment
- stochastic environment with Markovian transition model
- additive rewards/discounted rewards

= Markov decision process

Simplest setting

- sequential decision problem
- fully observable environment
- stochastic environment with Markovian transition model
- additive rewards/discounted rewards

= Markov decision process

Simplest setting

- sequential decision problem
- fully observable environment
- stochastic environment with Markovian transition model
- additive rewards/discounted rewards

= Markov decision process

Simplest setting

- sequential decision problem
- fully observable environment
- stochastic environment with Markovian transition model
- additive rewards/discounted rewards

= Markov decision process

Markov decision process

5-tuple $(S, A, P(s' \mid s, a), r(s), \gamma)$

- S is a finite set of states
- A is a finite set of actions
- $P(s' \mid s, a)$ is the probability that action a in state s will lead to state s'
- $r(s)$ is the immediate reward received in state s
- $\gamma \in [0, 1]$ is the discount factor
- What if $\gamma = 0$?

Markov decision process

5-tuple $(S, A, P(s' \mid s, a), r(s), \gamma)$

- S is a finite set of states
- A is a finite set of actions
- $P(s' \mid s, a)$ is the probability that action a in state s will lead to state s'
- $r(s)$ is the immediate reward received in state s
- $\gamma \in [0, 1]$ is the discount factor
- What if $\gamma = 0$?

What is a solution?

- A fixed sequence of states does not solve the problem . . . why?
- A solution must specify what to do in *any* state.
- This solution is called policy π.
- $\pi(s)$ recommends action is state s.

What is a solution?

- A fixed sequence of states does not solve the problem . . . why?
- A solution must specify what to do in *any* state.
- This solution is called policy π.
- $\pi(s)$ recommends action is state s.

What is a solution?

- A fixed sequence of states does not solve the problem . . . why?
- A solution must specify what to do in *any* state.
- This solution is called policy π.
- $\pi(s)$ recommends action is state s.

Criteria - sum of discounted rewards

- Expected utility obtained by executing π starting in s_0 is the sum of discounted rewards
- How to calculate it?

Criteria - sum of discounted rewards

- Expected utility obtained by executing π starting in s_0 is the sum of discounted rewards
- $$U^{\pi}(s_0) = \mathrm{E}\left(\sum_{k=0}^{\infty} \gamma^k r(s_k)\right)$$

Criteria - sum of discounted rewards

- Expected utility obtained by executing π starting in s_0 is the sum of discounted rewards
- $$U^{\pi}(s_0) = \mathrm{E}\left(\sum_{k=0}^{\infty} \gamma^{k} r(s_k)\right)$$
- What is policy if we know utility U^{π}?

Criteria - sum of discounted rewards

- Expected utility obtained by executing π starting in s_0 is the sum of discounted rewards
- $$U^{\pi}(s_0) = \mathrm{E}\left(\sum_{k=0}^{\infty} \gamma^k r(s_k)\right)$$
- $$\pi(s) = \arg\max_{\pi} U^{\pi}(s)$$

Utility in Marcovian setting

- Choose action $a \in A(s)$ that maximizes the expected utility of the subsequent state:

$$\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')$$

- Bellman equation

$$U(s) = r(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s').$$

- An alternative formulation . . . what is different?

$$U(s) = \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)(r(s, a, s') + \gamma U(s'))$$

Utility in Marcovian setting

- Choose action $a \in A(s)$ that maximizes the expected utility of the subsequent state:

$$\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')$$

- Bellman equation

$$U(s) = r(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s').$$

- An alternative formulation ... what is different?

$$U(s) = \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)(r(s, a, s') + \gamma U(s'))$$

Utility in Marcovian setting

- Choose action $a \in A(s)$ that maximizes the expected utility of the subsequent state:

$$\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')$$

- Bellman equation

$$U(s) = r(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s').$$

- An alternative formulation . . . what is different?

$$U(s) = \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)(r(s, a, s') + \gamma U(s'))$$

Value iteration

- Bellman update

$$\widehat{U}_{i+1}(s) \leftarrow r(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\widehat{U}_i(s')$$

- The utility function is a fixed point of Bellman update.
- Connection to fixed point iteration.

Value iteration

```
$\widehat{U}(s)$ gets initial values
repeat
    $\Delta \leftarrow 0$
    for all $s \in S$ do
        $tmp \leftarrow \widehat{U}(s)$
        $\widehat{U}(s) \leftarrow r(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) tmp(s')$
        $\Delta \leftarrow \max(\Delta, |tmp - \widehat{U}(s)|)$
    end for
until $\Delta < \theta$
```

- Convergence guaranteed.
- A bound on relative error available.

Policy iteration

```
$\widehat{U}(s) \in \mathbb{R}$ and $\pi(s) \in A(s)$ are arbitrary
repeat
    $\Delta \leftarrow 0$
    for all $s \in S$ do                                   ▷ Policy evaluation
        $tmp \leftarrow \widehat{U}(s)$
        $\widehat{U}(s) \leftarrow r(s) + \gamma \sum_{s'} P(s' \mid s, \pi(s)) tmp(s')$
        $\Delta \leftarrow \max(\Delta, |tmp - \widehat{U}(s)|)$
    end for
until $\Delta < \theta$    ▷ Or simply solve the system of linear equations.

$\pi(s) \leftarrow \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) \widehat{U}(s')$    ▷ P. improvement
if policy did not change, stop.
Otherwise goto policy evaluation step.
```

- We do not need exact utility values.

Example

There are 4 matches lying on a table. The goal of an automated robot is to gradually remove them such that there is no match remaining on the table. The robot can remove 1 or 2 matches in one step. The problem is that the robot's arm is unreliable, it can remove more matches than the robot planned. To be precise, in half the attempts the arm removes one more match than planned. If the robot tries to remove more matches than actually available, the task becomes cyclic (-1 turns into 4, -2 turns into 3). Propose the optimal robot control strategy, the goal is to minimize the number of steps.

(a) Propose a task formalization based on Markov Decision Process (MDP).

(b) Formally derive the optimal strategy. If the derivation turns out difficult, show a few steps only and define the termination conditions.

(c) Use the derivation ad b and for each state select one out of two available actions.

(d) How many steps the robot with the optimal control strategy needs to reach zero matches?

Recommended literature

- Stuart Russell and Peter Norvig
  Artificial Intelligence: A Modern Approach, third edition.
  http://aima.cs.berkeley.edu/
  Chapter 17
- Richard S. Sutton and Andrew G. Barto
  Reinforcement Learning: An Introduction, second edition.
  http://www.incompleteideas.net/book/the-book-2nd.html
  Chapters 3,4