Model-Free Q-learning with MC and TD [2]

DOROZHKO Anton

Novosibirsk State University

May 20, 2020

[2] David Silver's Lecture 4

Outline

1. Markov decision processes
2. PI vs VI
3. MC and TD

Markov Decision Process

Definition

A Markov Decision Process is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{A}$ is a finite set of actions
- $\mathcal{P}$ is a transition probability matix

$$\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$$

- $\mathcal{R}$ is a reward function :

$$\mathcal{R}_s^a = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$$

- γ is a discount factor, $\gamma \in [0, 1]$

Recall Policy Iteration

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

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a) \big[r + \gamma V(s') \big]$
   $\quad$ 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

Novosibirsk State University
THE REAL SCIENCE

Recall : Value Iteration

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

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

Novosibirsk State University
*THE REAL SCIENCE

Comparison [8]

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

2. Policy Evaluation
   Repeat
   $\quad \Delta \leftarrow 0$
   $\quad$ For each $s \in \mathcal{S}$:
   $\quad\quad temp \leftarrow v(s)$
   $\quad\quad v(s) \leftarrow \sum_{s'} p(s'|s,\pi(s))\big[r(s,\pi(s),s') + \gamma v(s')\big]$
   $\quad\quad \Delta \leftarrow \max(\Delta, |temp - v(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   $policy\text{-}stable \leftarrow true$
   For each $s \in \mathcal{S}$:
   $\quad temp \leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \arg\max_a \sum_{s'} p(s'|s,a)\big[r(s,a,s') + \gamma v(s')\big]$
   $\quad$ If $temp \neq \pi(s)$, then $policy\text{-}stable \leftarrow false$
   If $policy\text{-}stable$, then stop and return v and π; else go to 2

Figure 4.3: Policy iteration (using iterative policy evaluation) for v_*. This algorithm has a subtle bug, in that it may never terminate if the policy continually switches between two or more policies that are equally good. The bug can be fixed by adding additional flags, but it makes the pseudocode so ugly that it is not worth it. :-)

finding optimal value function

Initialize array v arbitrarily (e.g., $v(s) = 0$ for all $s \in \mathcal{S}^+$)

Repeat
$\quad \Delta \leftarrow 0$
$\quad$ For each $s \in \mathcal{S}$:
$\quad\quad temp \leftarrow v(s)$
$\quad\quad v(s) \leftarrow \max_a \sum_{s'} p(s'|s,a)[r(s,a,s') + \gamma v(s')]$
$\quad\quad \Delta \leftarrow \max(\Delta, |temp - v(s)|)$
until $\Delta < \theta$ (a small positive number)

Output a deterministic policy, π, such that
$\pi(s) = \arg\max_a \sum_{s'} p(s'|s,a)\big[r(s,a,s') + \gamma v(s')\big]$

Figure 4.5: Value iteration.

one policy update (extract policy from the optimal value function

Novosibirsk State University
*THE REAL SCIENCE

[8] Stackoverflow

Monte-Carlo RL

- MC learns directly from episodes of experience
- Model-free: no knowledge of

$$\mathcal{P}^a_{ss'} \quad or \quad \mathcal{R}$$

- complete episodes
- idea : value = mean return
- Caveat: only episodic MDPS episodes must terminate

Novosibirsk State University
*THE REAL SCIENCE

MC Evaluation

- Goal: learn V_π from episodes under policy π

$$S_1, A_1, R_2, ..., S_k \sim \pi$$

- Total discounted reward

$$G_t = R_{t+1} + \gamma R_{t+2} + ... + \gamma^{T-1} R_T$$

- Value function

$$V_{pi}(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

MC Evaluation

- Goal: learn V_π from episodes under policy π

$$S_1, A_1, R_2, ..., S_k \sim \pi$$

- Total discounted reward

$$G_t = R_{t+1} + \gamma R_{t+2} + ... + \gamma^{T-1} R_T$$

- Value function

$$V_{pi}(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

- Monte-Carlo policy evaluation uses **empirical mean** return instead of expected return

[11] David Silver's Lecture 4

First-Visit Monte-Carlo Policy Evaluation

- To evaluate state s
- The first time-step t that state s is visited in each episode
- $N(s) = N(s) + 1$
- $S(s) = S(s) + G_t$
- $V(s) = S(s)/N(s)$
- By law of large numbers $V(s) \rightarrow V^{\pi}(s)$ as $N(s) \rightarrow \infty$

Novosibirsk State University
THE REAL SCIENCE

Incremental Mean

$$
\begin{aligned}
\mu_k &= \frac{1}{k}\sum_{j=1}^{k} x_j \\
&= \frac{1}{k}\left(x_k + \sum_{j=1}^{k-1} x_j\right) \\
&= \frac{1}{k}(x_k + (k-1)\mu_{k-1}) \\
&= \mu_{k-1} + \frac{1}{k}(x_k - \mu_{k-1})
\end{aligned}
$$

Novosibirsk State University
*THE REAL SCIENCE

- Update $V(s)$ incrementally after episode $S_1, A_1, R_2, ..., S_T$
- For each state S_t with return G_t

$$N(S_t) = N(S_t) + 1$$

$$V(S_t) = V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$$

- In non-stationary problems: running mean

$$V(S_t) = V(S_t) + \alpha(G_t - V(S_t))$$

Novosibirsk State University
*THE REAL SCIENCE

Temporal-Difference Learning

- TD learns from episodes
- model-free
- incomplete episodes, bootstrappping
- Update a guess towards a guess

MC and TD

- Goal: learn V_π online from experience under policy π
- Incremental every-visit MC
  - $V(S_t) = V(S_t) + \alpha(G_t - V(S_t))$
- Simplest TD : TD(0)
  - Update $V(S_t)$ toward estimated return $R_{t+1} + \gamma V(S_{t+1})$

$$V(S_t) = V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$

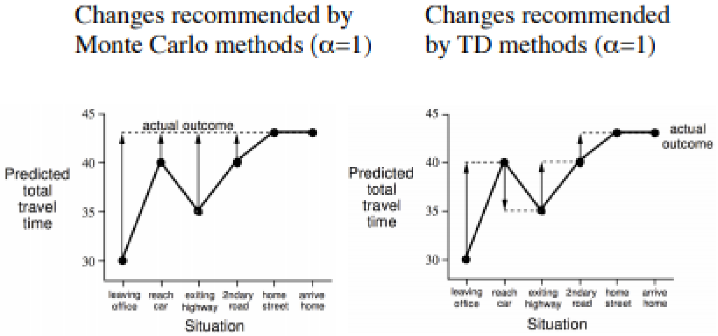
  - $R_{t+1} + \gamma V(S_{t+1})$ - TD target
  - $\delta_t = R_{t+1} + \gamma V(S_{t+1} - V(S_t)$ - TD error

Driving Home Example

State	Elapsed Time (minutes)	Predicted Time to Go	Predicted Total Time
leaving office	0	30	30
reach car, raining	5	35	40
exit highway	20	15	35
behind truck	30	10	40
home street	40	3	43
arrive home	43	0	43

Novosibirsk State University
*THE REAL SCIENCE

MC vs TD


Changes recommended by
Monte Carlo methods (α=1)
Changes recommended
by TD methods (α=1)
actual outcome
Predicted
total
travel
time
45
40
35
30
leaving
office
reach
car
exiting
highway
2ndary
road
home
street
arrive
home
Situation
actual
outcome

Advantages and Disadvantages

- TD can learn before knowing the final outcome
  - TD learn after each step
  - MC must end the episode
- TD can learn without the final outcome
  - TD - incomplete sequences
  - TD - continuing envs
  - MC - complete sequences
  - MC - only episodic envs

Advantages and Disadvantages (2)

MC has high variance, zero bias

- Good convergence
- Not sensitive to initial value
- Simple

TD has low variance, some bias

- More efficient than MC
- TD(0) converges to V_π
- (not always for function approximation)
- Sensitive to initial value

Novosibirsk State University
*THE REAL SCIENCE

Advantages and Disadvantages (3)

TD exploits Markov property

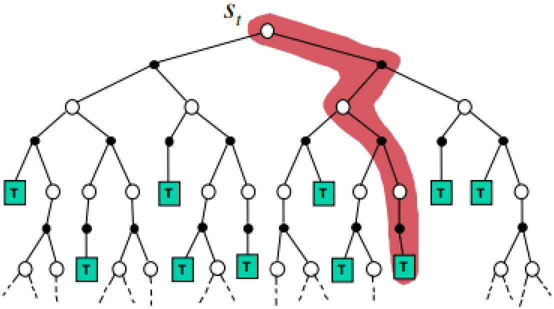
- Usually more efficient in Markov envs

MC does not exploit Markov property

- Usually more effective in non-Markov envs

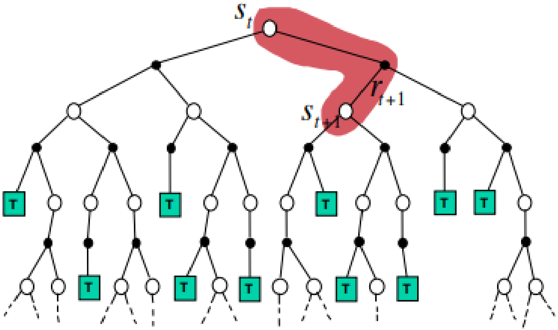
MC Backup

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$



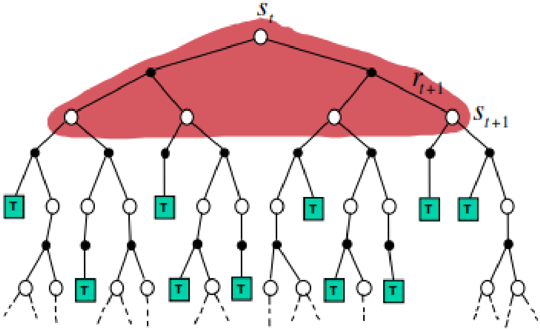
TD Backup

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$



DP Backup

$$V(S_t) \leftarrow \mathbb{E}[R_{t+1} + \gamma V(S_{t+1})]$$



Bootstrapping and Sampling

Bootstrapping update involves an estimate

- MC
- DP
- TD

Sampling update sample an expectation

- MC
- DP
- TD

Bootstrapping and Sampling

Bootstrapping update involves an estimate

- MC ✗
- DP ✓
- TD ✓

Sampling update sample an expectation

- MC
- DP
- TD

[26] David Silver's Lecture 4

Bootstrapping and Sampling

Bootstrapping update involves an estimate

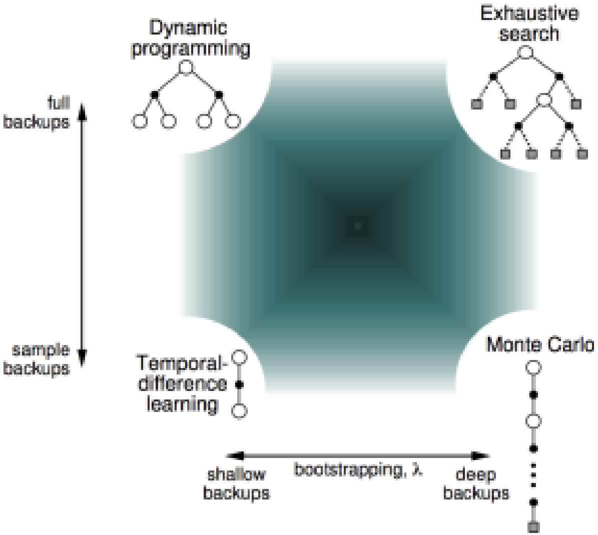
- MC ✗
- DP ✓
- TD ✓

Sampling update sample an expectation

- MC ✓
- DP ✗
- TD ✓

[26] David Silver's Lecture 4

Unified View of RL



Value and Policy Iteration Lab

https://bit.ly/2JVv6rc