REINFORCEMENT LEARNING

Lecture 2 : Dynamic Programming

Ibrahim Sammour

December | 2023

EPISODIC and CONTINUING TASKS

Episodic tasks:
- Finite amount of time.
- Has a terminal state.
- Examples:
  - Car racing from a start to a finish line.
  - Playing a chess game.
  - Navigating through a maze.

Continuing tasks:
- Does not have a terminal state.
- Examples:
  - Stock trading agent.
  - Autonomous car with no clear destination.

OPTIMAL POLICY – Episodic Tasks

$Episode\ 0\text{: } \pi_0$

π_0	a_1	a_2
s_1	0.5	0.5
s_2	0.5	0.5
s_3	0.5	0.5

$Episode\ 1\text{: } \pi_1$

π_1	a_1	a_2
s_1	0.6	0.4
s_2	0.7	0.3
s_3	0.1	0.9

$Episode\ 2\text{: } \pi_2$

π_2	a_1	a_2
s_1	0.3	0.7
s_2	0.8	0.2
s_3	0.6	0.4

Bellman Equation

Cumulative Reward

$$\bar{r} = r_t + \gamma * r_{t+1} + \gamma^2 * r_{t+2} + \cdots + \gamma^T * r_T$$

Value Function in a single run (Bellman)

$$V(s_t) = r_t + \gamma V(s_{t+1})$$

Bellman Optimality

$$V(s_t) = \max_a [r_t + \gamma V(s_{t+1}) \mid s_t = s]$$

Richard Bellman

Bellman Equation

Bellman Expectation Equation

$$V_\pi(s_t) = \mathbb{E}_\pi[r_t + \gamma V_\pi(s_{t+1}) \mid s_t = s]$$

We calculate this equation at the end of the episode

It is an expectation because it includes a probability

Richard Bellman

Bellman Equation

Another representation of Bellman equation

$$V_\pi(s_t) = \sum_a \pi(a \mid s_t) \sum_{s' \cdot r} p(s', r \mid s_t, a)[r_t + \gamma V_\pi(s_{t+1})]$$

$\sum_a \pi(a \mid s_t)$: *Sum of probabilities of all possible actions at state* s_t

$\sum_{s' \cdot r} p(s', r \mid s_t, a)$: *Sum of probabilities of transisioning to state* s'

OPTIMAL POLICY – Episodic Tasks

We calculate a new value function at the end of each episode

$$\boldsymbol{End\ of\ Episode\ 0}: \boldsymbol{V_{\pi_0}}$$

$$\boldsymbol{End\ of\ Episode\ 1}: \boldsymbol{V_{\pi_1}}$$

$$\boldsymbol{End\ of\ Episode\ 2}: \boldsymbol{V_{\pi_2}}$$

The new value function is used to update the policy

$$\boldsymbol{\pi_0 \Rightarrow Episode\ 0 \Rightarrow V_{\pi_0} \Rightarrow \pi_1 \Rightarrow Episode\ 1 \Rightarrow V_{\pi_1} \Rightarrow}\ \ldots$$

Bellman Equation

Another representation of Bellman equation

$$V_\pi(s_t) = \sum_a \pi(a \mid s_t) \sum_{s' \cdot r} p(s', r \mid s_t, a)[r_t + \gamma V_\pi(s_{t+1})]$$

State	1	2	3
1	-1	-4	Goal (-1)
2	-2	-1	-1
3	-1	-2	-1
4	Start	-1	-3

π_1	a_1	a_2
s_1	0.6	0.4
s_2	0.7	0.3
s_3	0.1	0.9

Action Value function

$Q(s_t, a) = ??$

$Q(s_t, a) = [r_t + \gamma \boldsymbol{Q}(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) \mid s_t = s, a_t = a]$

$Q_\pi(s_t, a) = ??$

$Q_\pi(s_t, a) = \mathbb{E}_\pi[r_t + \gamma \boldsymbol{Q}_\pi(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) \mid s_t = s, a_t = a]$

Action Value Function

Another representation of action value function

$$Q_\pi(s_t, a_t) = \sum_{s' \cdot r} p(s', r \mid s_t, a)[r_t + \gamma Q_\pi(s_{t+1}, a\ ) \mid s_t = s, a_t = \ a]$$

$\sum_{s' \cdot r} p(s', r \mid s_t, a)$: *Sum of probabilities of transisioning to state s'*

Policy Iteration

Two steps:

1- Policy Evaluation: Asses the value of states under the current policy π.

- Apply Bellman equation

$$V_\pi(s_t) = \sum_a \pi(a \mid s_t) \sum_{s' \cdot r} p(s', r \mid s_t, a)[r_t + \gamma V_\pi(s_{t+1})]$$

Policy Iteration

2- Policy Improvement: Update the policy based on the current value function estimate.

$$\pi'(s) = argmax_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V_\pi(s')]$$

Greedily select actions that maximizes the expected reward

Continue until $\pi'(s)$ stabilizes

$$\pi^* = \pi'$$

Value Iteration

Single Step:

Update using Bellman optimality equation:

$$V(s) = \max_{a} \sum_{s',r} p(s', r \mid s, a)[r + \gamma V(s')]$$

Repeat until it converges to V^*

Then:

$$\pi^*(s) = argmax_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V^*(s')]$$