Policy Evaluation

Temporal Difference Learning

Marius Lindauer

Leibniz Universität Hannover

Winter Term 2021

Temporal Difference Learning

- "If one had to identify one idea as central and novel to reinforcement learning, it would undoubtedly be temporal-difference (TD) learning." – Sutton and Barto 2017
- Combination of Monte Carlo & dynamic programming methods
- Model-free
- bootstraps and samples
- Can be used in episodic or infinite-horizon non-episodic settings
- Immediately updates estimate of V after each (s, a, r, s') tuple

Temporal Difference Learning for Estimating V

- Aim: estimate $V^\pi(s)$ given episodes generated under policy π
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$ in MDP M under policy π
- $V^\pi(s) = \mathbb{E}[G_t \mid s_t = s]$

Temporal Difference Learning for Estimating V

- Aim: estimate $V^\pi(s)$ given episodes generated under policy π
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$ in MDP M under policy π
- $V^\pi(s) = \mathbb{E}[G_t \mid s_t = s]$
- Recall Bellman operator (if known MDP models)

$$B^\pi V(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s' \mid s, \pi(s)) V(s')$$

Temporal Difference Learning for Estimating V

- Aim: estimate $V^\pi(s)$ given episodes generated under policy π
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots$ in MDP M under policy π
- $V^\pi(s) = \mathbb{E}[G_t \mid s_t = s]$
- Recall Bellman operator (if known MDP models)

$$B^\pi V(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s' \mid s, \pi(s)) V(s')$$

- Insight: have an estimate of V^π, used to estimate expected return

$$V^\pi(s) = V^\pi(s) + \alpha([r_t + \gamma V^\pi(s_{t+1})] - V^\pi(s))$$

Temporal Difference [TD(0)] Learning

- Aim: estimate $V^\pi(s)$ given episodes generated under policy π
- Simplest TD learning: update value towards estimated value

$$V^\pi(s) = V^\pi(s) + \alpha(\underbrace{[r_t + \gamma V^\pi(s_{t+1})]}_{\text{TD target}} - V^\pi(s))$$

Temporal Difference [TD(0)] Learning

- Aim: estimate $V^\pi(s)$ given episodes generated under policy π
- Simplest TD learning: update value towards estimated value

$$V^\pi(s) = V^\pi(s) + \alpha(\underbrace{[r_t + \gamma V^\pi(s_{t+1})]}_{\text{TD target}} - V^\pi(s))$$

- TD error:

$$\delta_t = r_t + \gamma V^\pi(s_{t+1}) - V^\pi(s)$$

Temporal Difference [TD(0)] Learning

- Aim: estimate $V^\pi(s)$ given episodes generated under policy π
- Simplest TD learning: update value towards estimated value

$$V^\pi(s) = V^\pi(s) + \alpha(\underbrace{[r_t + \gamma V^\pi(s_{t+1})]}_{\text{TD target}} - V^\pi(s))$$

- TD error:

$$\delta_t = r_t + \gamma V^\pi(s_{t+1}) - V^\pi(s)$$

⇝ Can immediately update value estimate after (s, a, r, s') tuple

⇝ Don't need episodic setting

Temporal Difference [TD(0)] Learning Algorithm

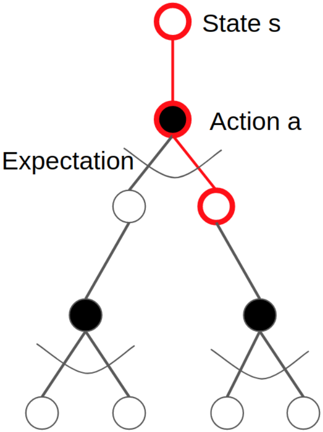
Input: α

Initialize $V^\pi(s) = 0. \forall s \in S$

Loop

- Sample tuple (s_t, a_t, r_t, s_{t+1})
- $V^\pi(s) = V^\pi(s) + \alpha(\underbrace{[r_t + \gamma V^\pi(s_{t+1})]}_{\text{TD target}} - V^\pi(s))$

Temporal Difference [TD(0)] Learning Algorithm



- TD updates the value estimate using a sample of s_{t+1} to approximate an expectation
- TD updates the value estimate by bootstrapping, uses estimates of $V(s_{t+1})$