Temporal Difference Learning

Recap: Bellman equation (consistency)

Bellman Eq

$$V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}V^\pi(s')\right], \forall s$$

An iterative approach for estimating V^π

$$V^{t+1} \Leftarrow R + \gamma P V^t$$

1. Need to know the transition
2. Only works for discrete small MDPs

Today

Given MDP $\mathscr{M} = (S, A, r, P, \gamma)$ & a $\pi : S \mapsto \Delta(A)$,

how to estimate $V^\pi(s), \forall s$ WITHOUT knowing P

(i.e., how to learn V^π from experience)

Motivation

our opponent's strategy is unknown

hard to model the transitions of the other drivers/ pedestrians/cyclists)

Hard to model users' reactions to recommendations

Outline:

1. Simple Monte Carlo methods

2. Temporal Difference Learning

MC estimation

Setup: we have MDP $\mathscr{M} = (S, A, P, \gamma, r)$, and ***stochastic*** π, i.e., $a \sim \pi(\cdot \mid s)$

$$V^\pi(s) = \mathbb{E}\left[\sum_{h=0}^{\infty} \gamma^h r_h \middle| \pi, s_0 = s\right]$$

MC methods replace Expectation by Sample Average

MC estimation

For all state $s \in \mathcal{S}$:

Generate N i.i.d trajectories from $s_0 = s$ using π

T needs to be large enough (i.e., $\gamma^T \approx 0$)

$$\hat{V}^\pi(s) = \underbrace{\frac{1}{N} \sum_{i=1}^{N}}_{\text{Averaging}} \underbrace{\left[\sum_{h=0}^{T} \gamma^h r_{i,h} \right]}_{\text{Total discounted reward of the i-th trajectory}}$$

Reward at time h at the i-th trajectory

Averaging

Total discounted reward of the i-th trajectory

MC estimation

$$\hat{V}^{\pi}(s) = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{h=0}^{T}\gamma^{h} r_{i,h}\right]$$

When $T \to \infty$ and $N \to \infty$, we have $\hat{V}^{\pi}(s) \to V^{\pi}(s)$

The future after T-step has near zero contribution

Law of large number

MC estimation with incremental update

Instead of waiting for all N trajs, we can also update the estimator incrementally every traj

Initialize $\hat{V}^\pi(s) = 0, \forall s$

For all state $s \in \mathcal{S}$:

Generate one traj from $s_0 = s$ using π;

$$\hat{V}^\pi(s) \Leftarrow \hat{V}^\pi(s) + \eta \left[\underbrace{\left(\sum_{h=0}^{\infty} \gamma^h r_h \right)}_{\text{total reward of one traj}} - \hat{V}^\pi(s) \right]$$

η: learning rate

MC estimation with incremental update

The incremental update is performing
Stochastic gradient descent (SGD)

$$\hat{V}^\pi(s) \Leftarrow \hat{V}^\pi(s) + \eta \left[\left(\sum_{h=0}^{\infty} \gamma^h r_h \right) - \hat{V}^\pi(s) \right]$$

$$\ell(\hat{V}^\pi(s)) := \left(\hat{V}^\pi(s) - V^\pi(s) \right)^2$$

$$\nabla \ell(x) \,\big|_{x=\hat{V}^\pi(s)} = 2 \left(\hat{V}^\pi(s) - V^\pi(s) \right)$$

To get a stochastic gradient, replace $V^\pi(s)$ by its unbiased estimate $\sum_{h=0}^{\infty} \gamma^h r_h$

Summary

$$\hat{V}^{\pi}(s) = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{h=0}^{T}\gamma^{h} r_{i,h}\right]$$

MC estimation is simple, and is based on well-established statistical guarantee

However

It has **high variance**, i.e., total reward on an extremely long trajectory can have a lot of randomness;

Need to **wait** for completing full trajectories to update/estimate $V^{\pi}(s)$

Outline:

1. Simple Monte Carlo methods

2. Temporal Difference Learning

TD Learning

History: developed by Rich Sutton back in 1988

Learning to Predict by the Methods of Temporal Differences

RICHARD S. SUTTON (RICH@GTE.COM)
GTE Laboratories Incorporated, 40 Sylvan Road, Waltham, MA 02254, U.S.A.

(Received: April 22, 1987)
(Revised: February 4, 1988)

My first research project in grad school

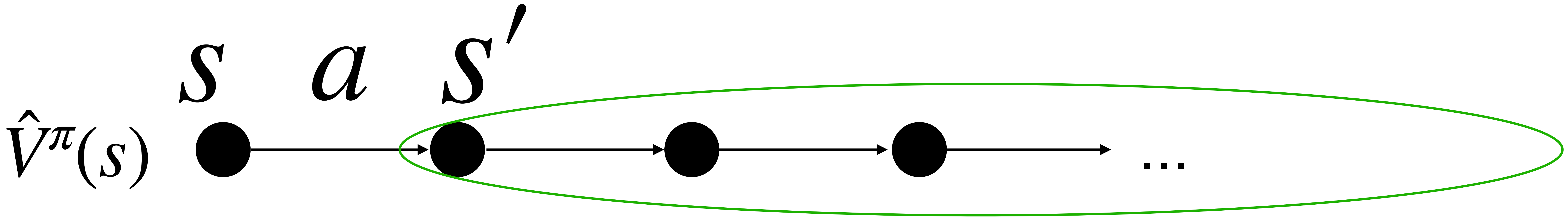
Online Bellman Residual Algorithms with Predictive Error Guarantees

Wen Sun
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
wensun@cs.cmu.edu

J. Andrew Bagnell
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
dbagnell@ri.cmu.edu

TD Learning

TD relies on the idea of **Bootstrapping** — *using predictions from our estimator to update the estimator itself*



$$r(s,a) + \gamma\, \hat{V}^{\pi}(s')$$

New target value (bootstrapping)

TD update: $\hat{V}^{\pi}(s) \Leftarrow \hat{V}^{\pi}(s) + \eta \left(r(s,a) + \gamma \hat{V}^{\pi}(s') - \hat{V}^{\pi}(s) \right)$

TD Learning

Initialize $V^\pi(s) = 0, \forall s$. Set initial state $s \in \mathcal{S}$

While True:

Take action $a \sim \pi(\cdot \mid s)$, get reward r and next state $s' \sim P(\cdot \mid s, a)$

Form TD target $r + \gamma \hat{V}^\pi(s')$

Update for s: $\hat{V}^\pi(s) \Leftarrow \hat{V}^\pi(s) + \eta \left(r + \gamma \hat{V}^\pi(s') - \hat{V}^\pi(s) \right)$

Set $s \Leftarrow s'$

Remark: (1) we perform online update while interacting w/ MDP;
(2) no need to wait to the end for an update (one update per step)

Interpret TD as "SGD" on TD loss

TD is not the usual SGD, i.e., it is **not** running SGD to minimize a fixed loss function

TD may be interpreted as running SGD on an **evolving** loss function (TD loss)

$$\ell_{td}(\hat{V}^\pi(s)) := \left(\hat{V}^\pi(s) - y\right)^2, \text{ where } y = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\hat{V}^\pi(s')\right]$$

This keeps changing as we learning

In-class exercise:

derive one-step SGD update for $\ell_{td}(\hat{V}^\pi(s))$

(Hint: how to get an unbiased estimate for y using one transition)

TD Learning

[Informal] Assume π has non-trivial probability of visiting every state.
Setting learning rate η properly, we will have:

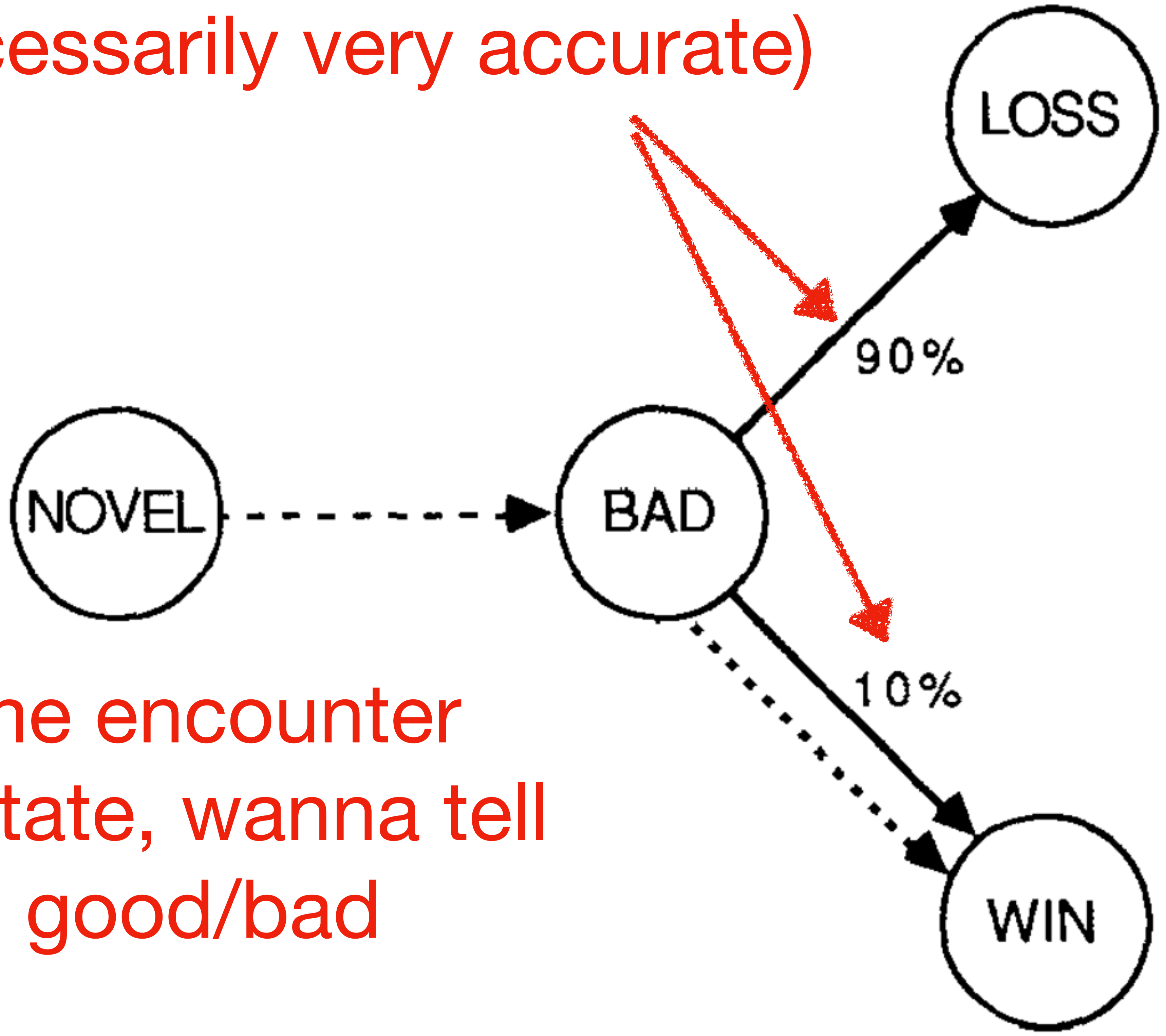
$$\hat{V}^{\pi}(s) \rightarrow V^{\pi}(s), \forall s,$$

when # of interactions approaches to ∞

(concrete convergence rates are known as well)

Example of faster learning with TD

From past game experiences
(not necessarily very accurate)



first time encounter
“novel” state, wanna tell
if it’s good/bad

[Sutton, 1988]

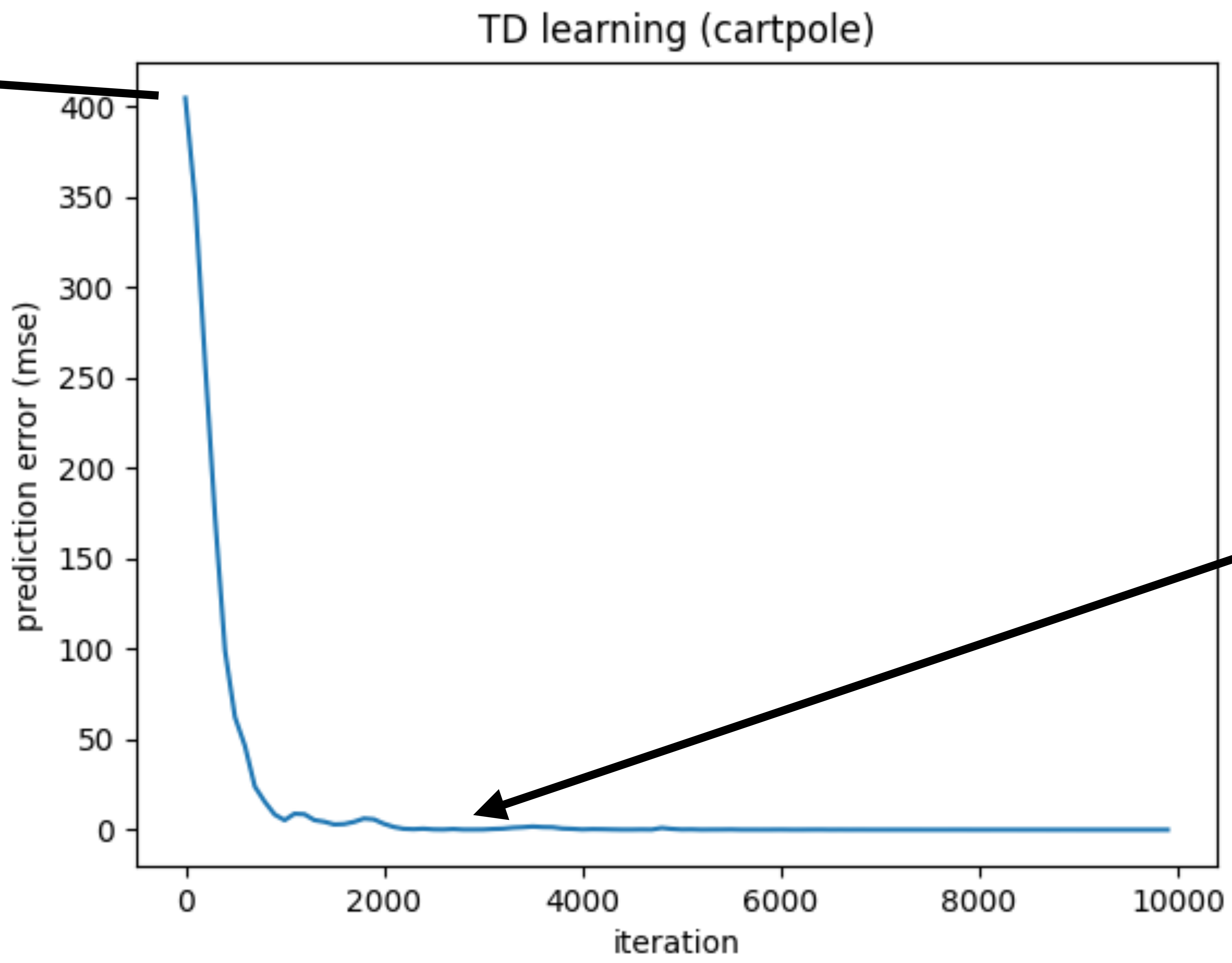
MC: have to rollout from NOVEL many times to get a reasonable estimate of future

TD (bootstrapping): NOVEL leading to BAD immediately implies NOVEL is likely bad

TD learning on CartPole

Note: Cartpole's state is continuous, so we will need TD w/ function approximation, e.g., neural network (we will get there very soon)

A randomly initialized NN as $\hat{V}^{\pi}(s)$ (very bad estimator)



Almost perfectly estimate the true target $V^{\pi}(s)$

TD vs MC: bias-variance tradeoff

MC is unbiased, but has higher variance (i.e., randomness over an entire traj)

TD can have high bias, but has lower variance

Large bias when $\hat{V}^{\pi}(s')$ is
far from $V^{\pi}(s')$

$$\hat{V}^{\pi}(s) \Leftarrow \hat{V}^{\pi}(s) + \eta \left(\underline{r + \gamma \hat{V}^{\pi}(s')} - \hat{V}^{\pi}(s) \right)$$

Randomness in target is
from **one-step transition**

Summary

TD Learning: an online algorithm for estimating V^π on the fly from agent's own experience

TD Learning uses idea of <u>bootstrapping</u>: using estimator $\hat{V}^\pi$'s prediction to improve the estimator $\hat{V}^\pi$ itself

Key update step in TD:

$$\hat{V}^\pi(s) \Leftarrow \hat{V}^\pi(s) + \eta \left(r + \gamma \hat{V}^\pi(s') - \hat{V}^\pi(s) \right)$$

TD target
(bootstrapping)

TD error