REINFORCEMENT LEARNING: THEORY AND PRACTICE

Inverse Reinforcement Learning

Profs. Amy Zhang and Peter Stone

TEXAS
The University of Texas at Austin

Introduction: Why learn from demonstration?

Introduction: Why learn from demonstration?

General purpose robot

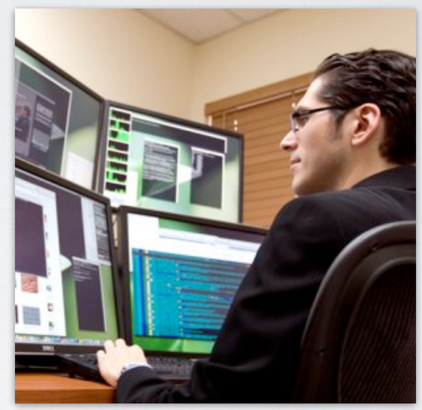
Introduction: Why learn from demonstration?

General purpose robot → Specific task

Introduction: Why learn from demonstration?

General purpose robot

Specific task



Expert engineer

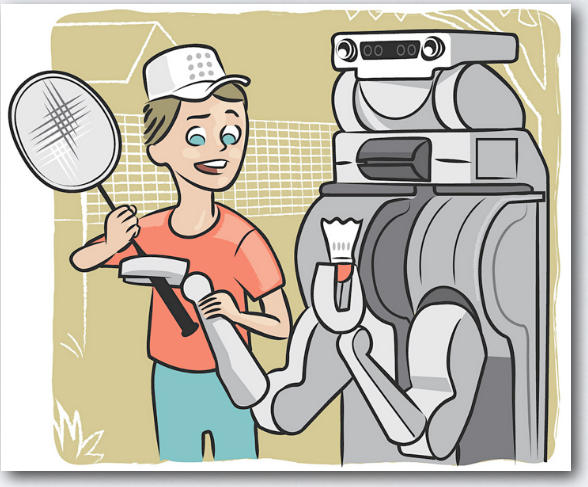
Introduction: Why learn from demonstration?

Programming robots is hard!

- Huge number of possible tasks
- Unique environmental demands
- Tasks difficult to describe formally
- Expert engineering impractical

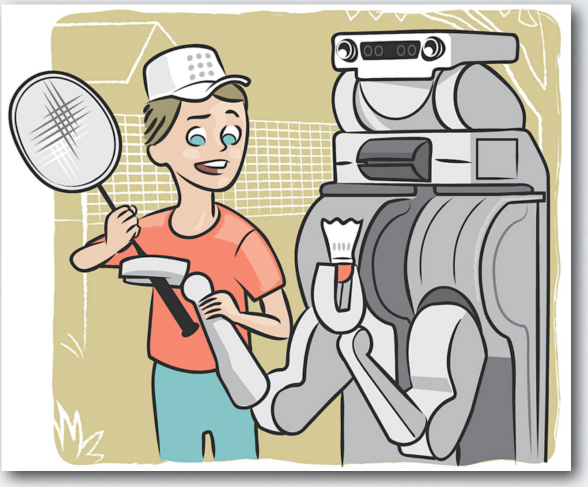
Introduction: Why learn from demonstration?

- Natural, expressive way to program
- No expert knowledge required
- Valuable human intuition
- Program new tasks as-needed

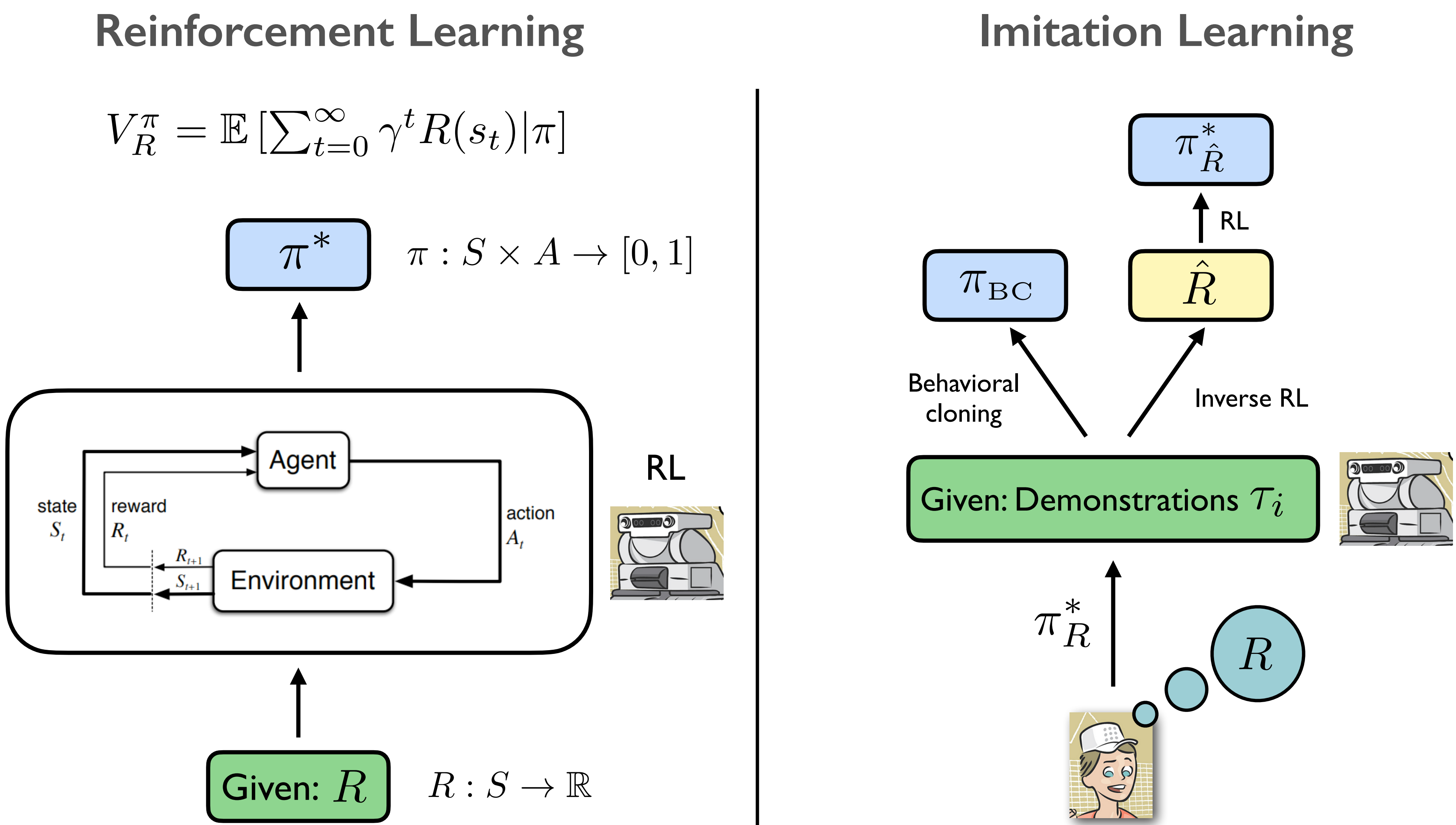


Introduction: Why learn from demonstration?

- Natural, expressive way to program
- No expert knowledge required
- Valuable human intuition
- Program new tasks as-needed



How can robots be shown how to perform tasks?



Reinforcement Learning
Imitation Learning
$V_R^\pi = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) | \pi\right]$
π^*
$\pi : S \times A \to [0, 1]$
Agent
state S_t
reward R_t
action A_t
R_{t+1}
S_{t+1}
Environment
RL
Given: R
$R : S \to \mathbb{R}$
$\pi^*_{\hat{R}}$
RL
π_{BC}
$\hat{R}$
Behavioral cloning
Inverse RL
Given: Demonstrations τ_i
π^*_R
R

Behavioral cloning

Supervised learning problem:

Demos $\longrightarrow$ Policy

i.e. from example (s,a) pairs, learn pi(s,a)

Behavioral cloning

Supervised learning problem:

Demos $\longrightarrow$ Policy

i.e. from example (s,a) pairs, learn pi(s,a)

What if we want to learn from experience via RL?

Inverse reinforcement learning:

Demos $\longrightarrow$ Inferred intent (reward function) $\longrightarrow$ Policy

Learning task objectives: Inverse reinforcement learning

Reinforcement learning basics:

states, actions, transition dynamics

MDP: (S, A, T, γ, D, R)

discount rate, start state distribution, reward function

Policy: $\pi(s, a) \rightarrow [0, 1]$

Value function: $V^{\pi}(s_0) = \sum_{t=0}^{\infty} \gamma^t R(s_t)$

What if we have an **MDP/R**?

Learning task objectives: Inverse reinforcement learning

1. Collect user demonstration $(s_0, a_0), (s_1, a_1), \ldots, (s_n, a_n)$ and assume it is sampled from the expert's policy, π^E

2. Explain expert demos by finding R^* such that:

$$E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) | \pi^E] \quad \geq \quad E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) | \pi] \quad \forall \pi$$

$$E_{s_0 \sim D}[V^{\pi^E}(s_0)] \quad \geq \quad E_{s_0 \sim D}[V^{\pi}(s_0)] \quad \forall \pi$$

How can search be made tractable?

[Abbeel and Ng 2004]

Learning task objectives: Inverse reinforcement learning

Define R^* as a linear combination of features:

$R^*(s) = w^T\phi(s)$, where $\phi : S \to \mathbb{R}^n$

Then,

$$
\begin{aligned}
E[\textstyle\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi] &= E[\textstyle\sum_{t=0}^{\infty} \gamma^t w^T \phi(s_t)|\pi] \\
&= w^T E[\textstyle\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi] \\
&= w^T \mu(\pi)
\end{aligned}
$$

Thus, the expected value of a policy can be expressed as a weighted sum of the expected features $\mu(\pi)$

[Abbeel and Ng 2004]

Learning task objectives: Inverse reinforcement learning

Originally - Explain expert demos by finding R^* such that:

$$E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi^E] \quad \geq \quad E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi] \quad \forall\pi$$

Use expected features:

$$E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi] = w^T \mu(\pi)$$

Restated - find w^* such that:

$$w^* \mu(\pi^E) \quad \geq \quad w^* \mu(\pi) \quad \forall\pi$$

[Abbeel and Ng 2004]

Learning task objectives: Inverse reinforcement learning

Goal: Find w^* such that: $w^* \mu(\pi^E) \geq w^* \mu(\pi) \;\; \forall \pi$

1. Initialize π_0 to any policy

Iterate for i = 1, 2, … :

2. Find w^* s.t. expert maximally outperforms all previously examined policies $\pi_{0 \ldots i-1}$:

$$\max_{\epsilon, w^* : \|w^*\|_2 \leq 1} \epsilon \quad \text{s.t.} \quad w^* \mu(\pi^E) \geq w^* \mu(\pi_j) + \epsilon$$

3. Use RL to calc. optimal policy π_i associated with w^*

4. Stop if $\epsilon \leq$ threshold

[Abbeel and Ng 2004]

Learning task objectives: Inverse reinforcement learning

Goal: Find w^* such that: $w^* \mu(\pi^E) \geq w^* \mu(\pi) \;\; \forall \pi$

1. Initialize π_0 to any policy

Iterate for i = 1, 2, … :

2. Find w^* s.t. expert maximally outperforms all previously examined policies $\pi_{0 \dots i-1}$:

$$\max_{\epsilon, w^* : \|w^*\|_2 \leq 1} \epsilon \quad \text{s.t.} \quad w^* \mu(\pi^E) \geq w^* \mu(\pi_j) + \epsilon$$

SVM solver

3. Use RL to calc. optimal policy π_i associated with w^*

4. Stop if $\epsilon \leq$ threshold

[Abbeel and Ng 2004]

Reading responses

Zhili Xiong

By the definition of the algorithm, why does it say that the learned reward will make sure that any other policies are worse than the expert by a certain threshold t? Can any other policies that are better than the expert? What if the expert is not the optimal?

Lingyun Xiao

Inverse RL also assumes that the "expert" is near optimal, which plays a critical role in finding the optimal w* such that the weighted sum associated with the expert policy w*\mu(\pi^E) is at least as good as any possible policy \pi. However, what if the expert is in fact sub-optimal? Is it possible that the learned w* is also suboptimal and even has a significant deviation from the true w*?

Reading responses

Xiwen Wei
How can the proposed inverse reinforcement learning (IRL) framework be adapted to learn non-linear reward functions, considering that many real-world tasks may not be well-represented by linear combinations of features?

Victor Wang
I think there are many settings where the reward relies on more than a linear combination of the chosen features. Can the method be adapted to relax this assumption?

Reading responses

Rosemary Lach
What is the point of using reinforcement learning if there is no good way to represent a reward function? If we are merely trying to mimic some expert functionality, wouldn't traditional supervised learning also be sufficient? Why choose RL in particular?

Reading responses

Surya Murthy
One advantage of learning a reward function is transferability. What are some examples of transferring a reward model between tasks?