Lecture 7: Imitation Learning in Large State Spaces[2]

Emma Brunskill

CS234 Reinforcement Learning.

Winter 2019

[2]With slides from Katerina Fragkiadaki and Pieter Abbeel

Table of Contents

Recap: DQN (Mnih et al. Nature 2015)

$r_t + \gamma \max_a Q(s_{t+1}, a) - Q$ (with w)

- DQN uses experience replay and fixed Q-targets
- Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in replay memory $\mathcal{D}$
- Sample random mini-batch of transitions (s, a, r, s') from $\mathcal{D}$
- Compute Q-learning targets w.r.t. old, fixed parameters $\boldsymbol{w}^-$
- Optimizes MSE between Q-network and Q-learning targets
- Uses stochastic gradient descent
- Achieved human-level performance on a number of Atari games

Recap: Deep Model-free RL, 3 of the Big Ideas

- Double DQN: (Deep Reinforcement Learning with Double Q-Learning, Van Hasselt et al, AAAI 2016)
- Prioritized Replay (Prioritized Experience Replay, Schaul et al, ICLR 2016)
- Dueling DQN (best paper ICML 2016) (Dueling Network Architectures for Deep Reinforcement Learning, Wang et al, ICML 2016)

Summary of Model Free Value Function Approximation with DNN

- DNN are very expressive function approximators
- Can use to represent the Q function and do MC or TD style methods
- Should be able to implement DQN (assignment 2)
- Be able to list a few extensions that help performance beyond DQN

We want RL Algorithms that Perform

- Optimization
- Delayed consequences
- Exploration
- Generalization
- And do it all statistically and computationally efficiently

Generalization and Efficiency

- We will discuss efficient exploration in more depth later in the class
- But exist hardness results that if learning in a generic MDP, can require large number of samples to learn a good policy
- This number is generally infeasible
- Alternate idea: use structure and additional knowledge to help constrain and speed reinforcement learning
- Today: Imitation learning
- Later:
  - Policy search (can encode domain knowledge in the form of the policy class used)
  - Strategic exploration
  - Incorporating human help (in the form of teaching, reward specification, action specification, . . .)

Class Structure

- Last time: CNNs and Deep Reinforcement learning
- **This time: Imitation Learning with Large State Spaces**
- Next time: Policy Search

Consider Montezuma's revenge

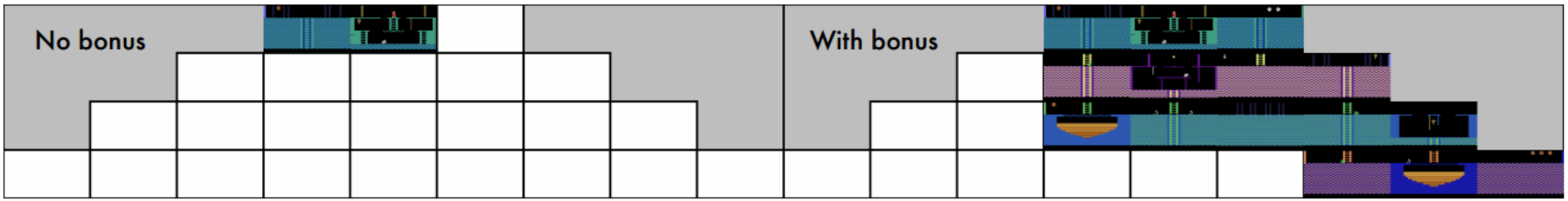


Figure 3: "Known world" of a DQN agent trained for 50 million frames with (**right**) and without (**left**) count-based exploration bonuses, in MONTEZUMA'S REVENGE.

- Bellemare et al. "Unifying Count-Based Exploration and Intrinsic Motivation"
- Vs: https://www.youtube.com/watch?v=JR6wmLaYuu4

So Far in this Course

Reinforcement Learning: Learning policies guided by (often sparse) rewards (e.g. win the game or not)

- Good: simple, cheap form of supervision
- Bad: High sample complexity

Where is it successful? where RL might work well ?

- In simulation where data is cheap and parallelization is easy
- Not when:
  - Execution of actions is slow
  - Very expensive or not tolerable to fail helicopter
  - Want to be safe

Reward Shaping

Rewards that are **dense in time** closely guide the agent
How can we supply these rewards?

- **Manually design them**: often brittle
- **Implicitly specify them through demonstrations**

Learning from Demonstration for Autonomous Navigation in Complex Unstructured Terrain, Silver et al. 2010

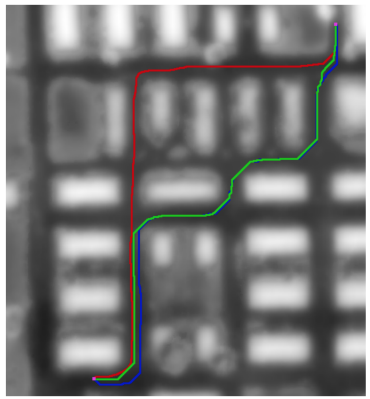
Examples

Simulated highway driving

- Abbeel and Ng, ICML 2004
- Syed and Schapire, NIPS 2007
- Majumdar et al., RSS 2017

Aerial imagery-based navigation

- Ratliff, Bagnell, and Zinkevich, ICML 2006



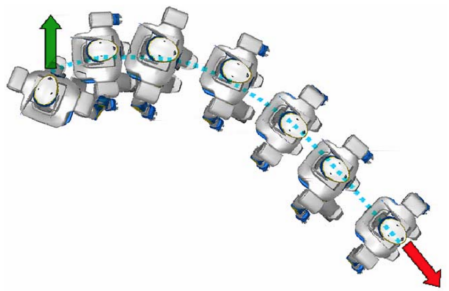
Parking lot navigation

- Abbeel, Dolgov, Ng, and Thrun, IROS 2008

Examples

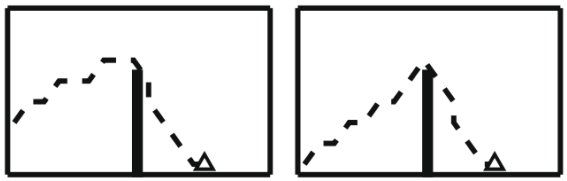
Human path planning

- Mombaur, Truong, and Laumond, AURO 2009



Human goal inference

- Baker, Saxe, and Tenenbaum, Cognition 2009



Quadruped locomotion

- Ratliff, Bradley, Bagnell, and Chestnutt, NIPS 2007
- Kolter, Abbeel, and Ng, NIPS 2008

Learning from Demonstrations

Inverse RL
Imitation learning

- Expert provides a set of **demonstration trajectories**: sequences of states and actions
- Imitation learning is useful when it is easier for the expert to demonstrate the desired behavior rather than:
  - Specifying a reward that would generate such behavior,
  - Specifying the desired policy directly

Problem Setup

- Input:
  - State space, action space
  - Transition model $P(s' \mid s, a)$
  - No reward function R
  - Set of one or more teacher's demonstrations $(s_0, a_0, s_1, s_0, \ldots)$ (actions drawn from teacher's policy π^*)
- Behavioral Cloning: $\hat{\pi} \approx \pi^*$
  - Can we directly learn the teacher's policy using supervised learning?
- Inverse RL:
  - Can we recover R?

once we have the reward function, then we can us it to compute a policy.

- Apprenticeship learning via Inverse RL:
  - Can we use R to generate a good policy?

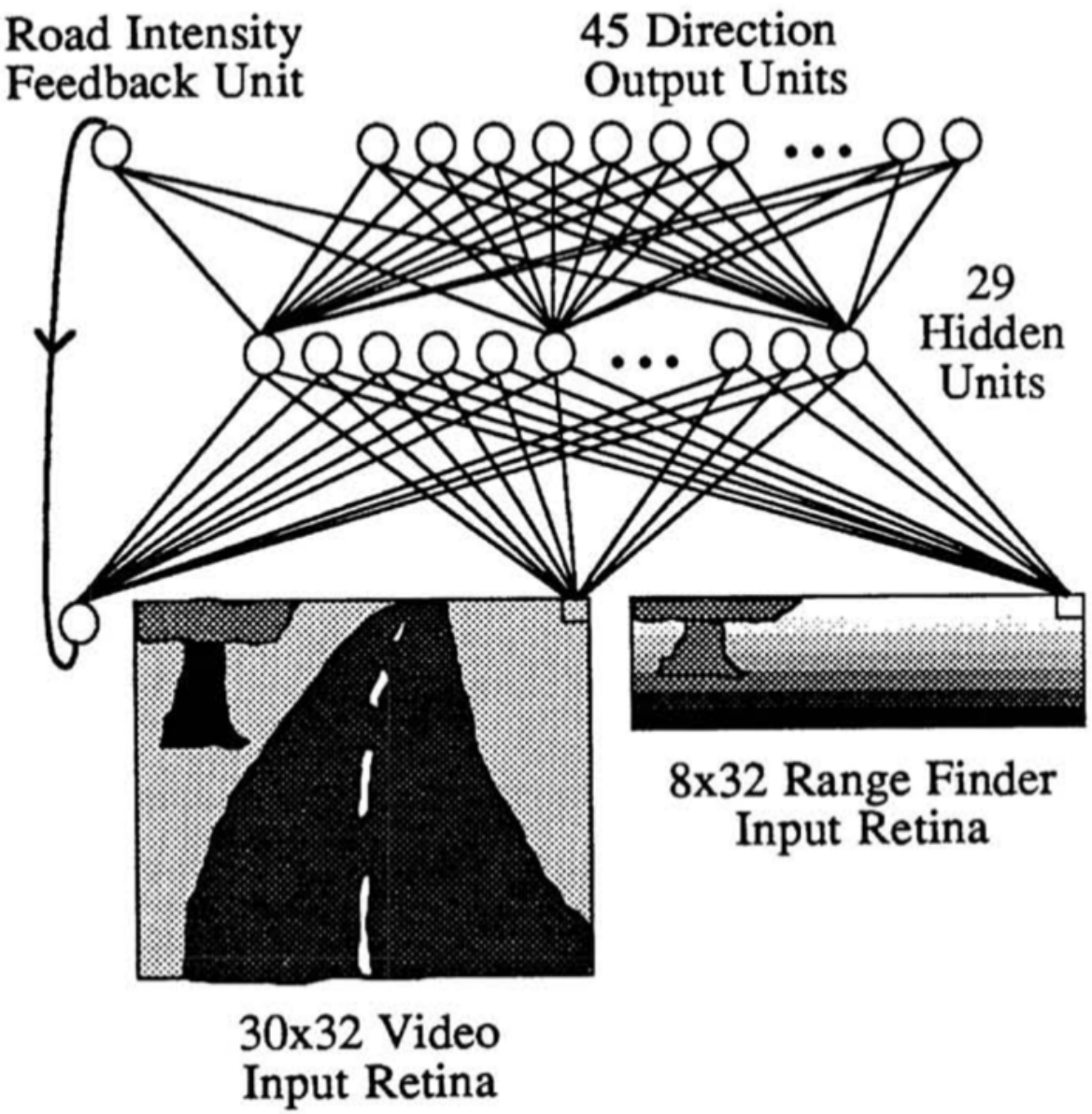
Inverse RL + RL

Table of Contents

Behavioral Cloning

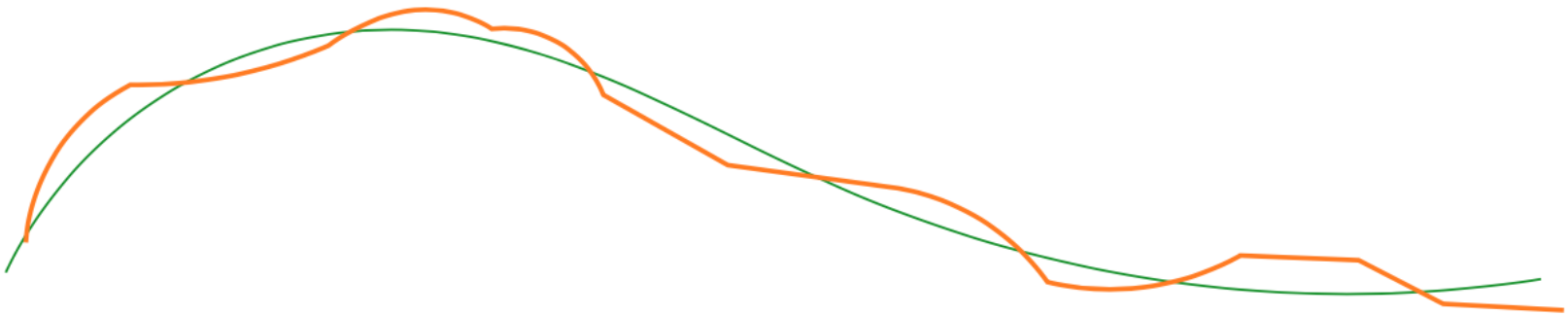
- Formulate problem as a standard machine learning problem:
  - Fix a policy class (e.g. neural network, decision tree, etc.)
  - Estimate a policy from training examples $(s_0, a_0), (s_1, a_1), (s_2, a_2), \ldots$
- Two notable success stories:
  - Pomerleau, NIPS 1989: ALVINN
  - Summut et al., ICML 1992: Learning to fly in flight simulator

ALVINN


Road Intensity
Feedback Unit
45 Direction
Output Units
29
Hidden
Units
8x32 Range Finder
Input Retina
30x32 Video
Input Retina

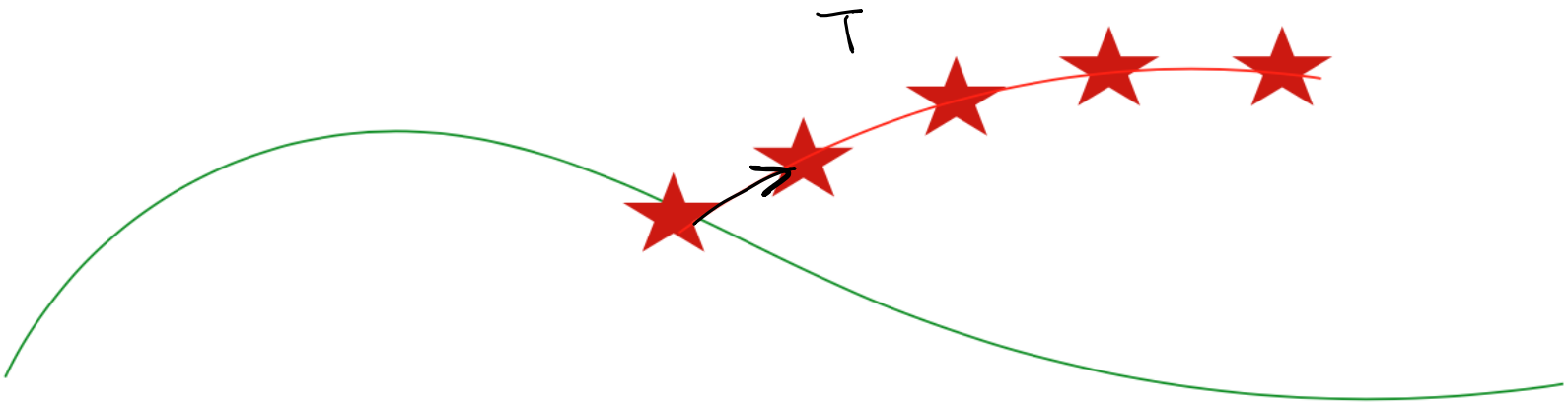
Problem: Compounding Errors

Supervised learning assumes iid. (s, a) pairs and ignores temporal structure
Independent in time errors:



Error at time t with probability ϵ
$\mathbb{E}[\text{Total errors}] \leq \epsilon T$

Problem: Compounding Errors

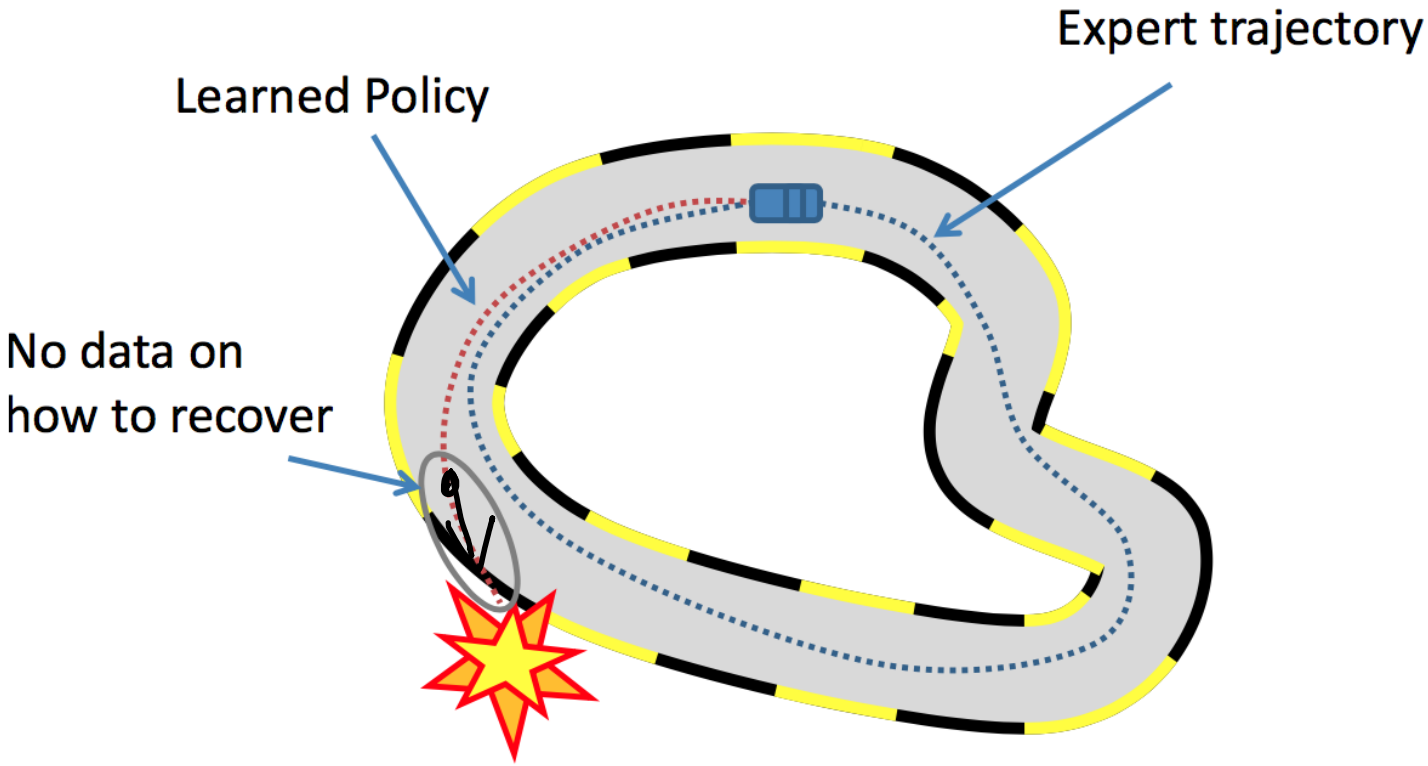


Error at time t with probability ϵ ϵT

$\mathbb{E}[\text{Total errors}] \leq \epsilon(T + (T-1) + (T-2)\ldots + 1) \propto \epsilon T^2$

A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

Problem: Compounding Errors



Data distribution mismatch!

In supervised learning, $(x, y) \sim D$ during train **and** test. In MDPs:

- Train: $s_t \sim D_{\pi^*}$
- Test: $s_t \sim D_{\pi_\theta}$

DAGGER: Dataset Aggregation

is essentially you just keep growing your data set.

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i
  and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

You start without any data.
You initialize your policy.

expert policy

an expert is taking some steps, and also your have other policy you collect trajectory.

ask the expert what would you do here.

add those tuples to your dataset, and you train your supervised learning policy on that.

- Idea: Get more labels of the expert action along the path taken by the policy computed by behavior cloning
- Obtains a stationary deterministic policy with good performance under its induced state distribution

Table of Contents

Feature Based Reward Function

- Given state space, action space, transition model $P(s' \mid s, a)$
- No reward function R the only thing we don't know.
- Set of one or more teacher's demonstrations $(s_0, a_0, s_1, s_0, \ldots)$ (actions drawn from teacher's policy π^*)
- Goal: infer the reward function R PS. not to directly learn a policy
- With no assumptions on the optimality of the teacher's policy, what can be inferred about R?

- Now assume that the teacher's policy is optimal. What can be inferred about R?

not unique

Andrew Ng Stuart Russell 2000

Linear Feature Reward Inverse RL

- Recall linear value function approximation
- Similarly, here consider when reward is linear over features
  - $R(s) = \boldsymbol{w}^T x(s)$ where $w \in \mathbb{R}^n, x : S \rightarrow \mathbb{R}^n$
- Goal: identify the weight vector $\boldsymbol{w}$ given a set of demonstrations
- The resulting value function for a policy π can be expressed as

$$V^{\pi} = \mathbb{E}_{s \sim \pi}[\sum_{t=0}^{\infty} \gamma^t R(s_t)] \quad (1)$$

$R(s_t) = w^T x(s_t)$

Linear Feature Reward Inverse RL

- Recall linear value function approximation
- Similarly, here consider when reward is linear over features
  - $R(s) = \boldsymbol{w}^T x(s)$ where $w \in \mathbb{R}^n, x : S \rightarrow \mathbb{R}^n$
- Goal: identify the weight vector $\boldsymbol{w}$ given a set of demonstrations
- The resulting value function for a policy π can be expressed as

$$V^\pi = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi] \quad = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \underbrace{\boldsymbol{w}^T}\underbrace{x(s_t)} \mid \pi] \tag{2}$$

$$= w^T \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t x(s_t) \mid \pi] \tag{3}$$

$$= \boldsymbol{w}^T \mu(\pi) \tag{4}$$

- where $\mu(\pi)(s)$ is defined as the discounted weighted frequency of state features under policy π.

Table of Contents

Linear Feature Reward Inverse RL

- Recall linear value function approximation
- Similarly, here consider when reward is linear over features
  - $R(s) = \boldsymbol{w}^T x(s)$ where $w \in \mathbb{R}^n, x : S \rightarrow \mathbb{R}^n$
- Goal: identify the weight vector $\boldsymbol{w}$ given a set of demonstrations
- The resulting value function for a policy π can be expressed as

$$V^\pi = \boldsymbol{w}^T \mu(\pi) \tag{5}$$

- where $\mu(\pi)(s)$ is defined as the discounted weighted frequency of state s under policy π.
- Note
$\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) \mid \pi^*] = V^* \geq V^\pi = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) \mid \pi] \quad \forall \pi,$
- Therefore if the expert's demonstrations are from the optimal policy, to identify $\boldsymbol{w}$ it is sufficient to find w^* such that

$$w^{*T} \mu(\pi^*) \geq w^{*T} \mu(\pi), \forall \pi \neq \pi^* \tag{6}$$

Feature Matching

- Want to find a reward function such that the expert policy outperforms other policies.
- For a policy π to be guaranteed to perform as well as the expert policy π^*, it suffices that we have a policy such that its discounted summed feature expectations match the expert's policy[42].
- More precisely, if

$$\|\mu(\pi) - \mu(\pi^*)\|_1 \leq \epsilon \tag{7}$$

then for all w with $\|w\|_\infty \leq 1$:

$$|w^T \mu(\pi) - w^T \mu(\pi^*)| \leq \epsilon$$

[42] Abbeel and Ng, 2004

Apprenticeship Learning

- This observation leads to the following algorithm for learning a policy that is as good as the expert policy
- Assumption: $R(s) = w^T x(s)$
- Initialize policy π_0
- For $i = 1, 2 \ldots$
  - Find a reward function such that the teacher maximally outperforms all previous controllers:

$$\arg\max_{\boldsymbol{w}} \max_{\gamma} s.t. w^T \mu(\pi^*) \geq w^T \mu(\pi) + \gamma \quad \forall \pi \in \{\pi_0, \pi_1, \ldots, \pi_{i-1}\} \tag{8}$$

  - s.t. $\|w\|_2 \leq 1$
  - Find optimal control policy π_i for the current $\boldsymbol{w}$
  - Exit if $\gamma \leq \epsilon/2$

Feature Expectation Matching

- If expert policy is suboptimal then the resulting policy is a mixture of somewhat arbitrary policies which have expert in the convex hull
- In practice: pick the best one of this set and pick the corresponding reward function.

Ambiguity

- There is an infinite number of reward functions with the same optimal policy.
- There are infinitely many stochastic policies that can match feature counts
- Which one should be chosen?

Learning from Demonstration / Imitation Learning Pointers

- Many different approaches
- Two of the key papers are:
  - Maximumum Entropy Inverse Reinforcement Learning (Ziebart et al. AAAI 2008)
  - Generative adversarial imitation learning (Ho and Ermon, NeurIPS 2016)

DNN

GAIL

Transition models unknown

Summary

- Imitation learning can greatly reduce the amount of data need to learn a good policy
- Challenges remain and one exciting area is combining inverse RL / learning from demonstration and online reinforcement learning

Class Structure

- Last time: Deep reinforcement learning
- This time: Imitation Learning
- Next time: Policy Search