Principles of Robot Autonomy II

Imitation Learning
Today’s itinerary

• Intro to Imitation Learning

• Behavioral Cloning, DAgger, COIL

• Inverse RL (Apprenticeship Learning, MMP, Max Ent IRL)

• Learning from other sources of data (preferences, physical feedback)

• Planning for robots based on human models
Today’s itinerary

• Intro to Imitation Learning

• Behavioral Cloning, DAgger, COIL

• Inverse RL (Apprenticeship Learning, MMP, Max Ent IRL)

• Learning from other sources of data (preferences, physical feedback)

• Planning for robots based on human models
Why Imitation Learning?

• It is difficult to learn with sparse rewards
  (unless data is cheap and you don’t care about seeing lots of failures)
• Hand-designing reward functions is hard
Just design the right reward function

\[ a^*_R = \arg\max_{a_R} R_H(s) \]
Why Imitation Learning?

• It is difficult to learn with sparse rewards (unless data is cheap and you don’t care about seeing lots of failures)
• Hand-designing reward functions is hard
• Just want to imitate for the sake of imitating!
Today’s itinerary

• Intro to Imitation Learning

• Behavioral Cloning, DAgger, COIL

• Inverse RL (Apprenticeship Learning, MMP, Max Ent IRL)

• Learning from other sources of data (preferences, physical feedback)

• Planning for robots based on human models
Problem Set up

MDP with no reward functions:
- State space, $S$
- Actions space, $A$
- Transition model $P(s_{t+1}|s_t, a_t)$
- set of expert demonstrations: $\xi = ((s_0, a_0), (s_1, a_1), ...)$ drawn from the expert policy $\pi^*$.

How can you learn a policy $\pi$ that learns from demonstrations or imitates the expert?
How to solve this?

1. Direct estimation of the expert policy from expert data (*behavioral cloning*).
2. Estimate the reward function (*inverse RL*) and then learn a policy from that (*apprenticeship learning*).
Behavioral Cloning

Can we learn the expert’s policy through supervised learning?

$$\pi^* = \underset{\pi}{\arg \min} \sum_{s_t \in D} L (\pi(s_t), \pi^*(s_t))$$
Aside: Similarity metrics

\[ L_p(X, Y) = \left( \sum_{i=1}^{N} |x_i - y_i|^p \right)^{1/p} \]

1. **Hamming distance** \((p = 0)\): Number of places the vectors differ
2. **Manhattan distance** \((p = 1)\): Sum of length differences on each dimension
3. **Euclidean distance** \((p = 2)\): Length of a straight line between the two vectors
4. **Chebyshev distance** \((p = \infty)\): maximum difference on any dimension

Minkowski distance is only useful when we have pairings of points. There is an assumption that the points are defined over the same support.
Aside: Similarity metrics

**f-Divergence:** What if we have expert and estimate distributions that we’d need to compare?

\[ D_f(P, Q) = \int f \left( \frac{p(x)}{q(x)} \right) q(x) \]

1. **KL Divergence:**
   \[ f(x) = x \log(x) \]

2. **Total Variation Distance:**
   \[ f(x) = \frac{|x - 1|}{2} \]

3. **Jensen-Shannon Divergence:**
   \[ f(x) = -(x + 1) \log \left( \frac{x + 1}{2} \right) + x \log(x) \]

4. **Hellinger Distance:**
   \[ f(x) = (\sqrt{x} - 1)^2 \]
Behavioral Cloning

Can we learn the expert’s policy through supervised learning?

$$\pi^* = \arg \min_{\pi} \sum_{s_t \in D} L(\pi(s_t), \pi^*(s_t))$$

$$= \arg \min_{\pi} \sum_{s_t \in D} KL(\pi(s_t), \pi^*(s_t))$$

$$= \arg \min_{\pi} \sum_{s_t \in D} \sum_{\pi(s_t) \in A} \pi(s_t) \log \left( \frac{\pi(s_t)}{\pi^*(s_t)} \right)$$

What can go wrong?

**Errors in supervised learning:**
- Assume iid state, action pairs, then if we have error at time $t$ with probability $\epsilon$, then over a time period the error would be bounded by $\epsilon T$ in expectation.

In imitation learning, the state distribution of our data depends on the choice of actions.

End up in states that you have not seen before...

... compounding errors

During training:

$$s_t \sim D_{\pi^*}$$

In test time:

$$s_t \sim D_{\pi_\theta}$$
Compounding Errors
How to fix this?

Dagger (Dataset Aggregation)

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in $\Pi$.

for $i = 1$ to $N$ do

Let $\pi_i = \beta_i \pi^* + (1 - \beta_i) \hat{\pi}_i$.
Sample $T$-step trajectories using $\pi_i$.
Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by $\pi_i$ and actions given by expert.
Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$.
Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.

done

Return best $\hat{\pi}_i$ on validation.

Ask people for more data!

Ross, et al. 2011
DAgger
What can go wrong with behavioral cloning?

Behavioral cloning: mimics the expert directly
- No reasoning about outcomes or dynamics
- No notion of intentions
- Expert can be suboptimal
- Expert might have different degrees of freedom
- Safety and Robustness
Today’s itinerary

• Intro to Imitation Learning

• Behavioral Cloning, DAgger, COIL

• Inverse RL (Apprenticeship Learning, MMP, Max Ent IRL)

• Learning from other sources of data (preferences, physical feedback)

• Planning for robots based on human models
History of Inverse Reinforcement Learning

- **1964**: Kalman posed the inverse optimal control problem and solved it in 1D
- **1994**: Boyd et al. A linear matrix inequality (LMI) characterization for the linear quadratic setting
- **2000**: Ng, Russell. Proposed the first MDP formulation and issues around reward function ambiguity
- **2004**: Abbeel, Ng. Inverse RL with feature matching for apprenticeship learning
- **2006**: Ratliff et al. Max Margin Planning (MMP) Formulation
- **2008**: Zeibart et al. Max Entropy Formulation
- Since then... Active Inverse RL, Integration with other types of data, Iterative approaches to update Reward and Policy (GAIL, etc.), images as inputs, etc.
Problem Set up

MDP with no reward functions:
- State space, $S$
- Actions space, $A$
- Transition model $P(s_{t+1}|s_t, a_t)$
- set of expert demonstrations: $\xi = ((s_0, a_0), (s_1, a_1), \ldots)$ drawn from the expert policy $\pi^*$.

How can you learn a policy $\pi$ that learns from demonstrations or imitates the expert?
Problem Set up

MDP with no reward functions:
  - State space, $S$
  - Actions space, $A$
  - Transition model $P(s_{t+1}|s_t, a_t)$
  - set of expert demonstrations: $\xi = ((s_0, a_0), (s_1, a_1), \ldots)$ drawn from the expert policy $\pi^*$.  

How can you learn a policy $\pi$ that learns from demonstrations or imitates the expert?

How can you learn a reward function (assuming experts were optimal)?
Inverse Reinforcement Learning

Assume the reward function is a linear combination of features:

\[ R(s) = w^T \varphi(s) \]

where \( w \in \mathbb{R}^n \) and \( \varphi: S \rightarrow \mathbb{R}^n \)

(a) Features for the boundaries of the road
(b) Feature for staying inside the lanes.
(c) Features for avoiding other vehicles.
Inverse Reinforcement Learning

Assume the reward function is a linear combination of features:

\[ R(s) = w^T \varphi(s) \quad w \in \mathbb{R}^n \quad \varphi: S \to \mathbb{R}^n \]

The goal is to recover the weights: \( w \)

\[
V^\pi(s) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \right] \\
= \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t w^T \varphi(s_t) \right] = w^T \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t \varphi(s_t) \right] = w^T \mu(\pi)
\]
Feature Matching

By definition, the value of optimal policy with respect to true reward is greater than the value of any other policy:

\[
V^{\pi^*}(s) > V^{\pi}(s) \quad \forall \pi
\]

\[
\mathbb{E}_{\pi^*} \left[ \sum_{t=0}^{\infty} \gamma^t R^*(s_t) \right] > \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R^*(s_t) \right]
\]

\[
w^{*\top} \mu(\pi^*) > w^{*\top} \mu(\pi)
\]
Apprenticeship Learning

For a policy to be guaranteed to work as good as the expert policy, it suffices to show that feature expectations match:

If $||\mu(\pi) - \mu(\pi^*)||_1 \leq \epsilon$ and $||w||_\infty \leq 1$ then:

$$|w^T \mu(\pi) - w^T \mu(\pi^*)| \leq ||w||_\infty ||\mu(\pi) - \mu(\pi^*)||_1 \leq 1 \cdot \epsilon = \epsilon$$
1: Initialize policy $\pi_0$
2: for $i = 1, 2, \ldots$ do
3: Find reward function weights $w$ such that the teacher maximally outperforms all previous controllers:

$$\arg \max_w \max_{\gamma} \gamma$$

s.t. $w^T \mu(\pi^* | s_0 = s) \geq w^T \mu(\pi | s_0 = s) + \gamma$, $\forall \pi \in \{\pi_0, \pi_1, \ldots, \pi_{i-1}\}$, $\forall s$

$$\|w\|_2 \leq 1$$

4: Find optimal policy $\pi_i$ for current $w$
5: if $\gamma \leq \epsilon/2$ then return $\pi_i$

Need to be able to compute optimal policy, which is not always easy
Apprenticeship Learning

Abbeel, Ng, 2004
How to deal with reward ambiguity?

Reward ambiguity: There are many reward functions under which the expert demonstrations are optimal!!

Which reward function should we pick?

- Maximum Margin Planning: Looks for the one that separates the optimal policy best.

- Maximum Entropy IRL: Looks for the one where expert demonstrations are drawn from a high entropy distribution.
Aside: Maximum Margin Classifiers

Given a training dataset of \((x_1, y_1), \ldots, (x_n, y_n)\), where \(y_i\) is either 1 or -1 identifying the class \(x_i\) is in. We want to find the maximum margin hyperplane that divides the points so the distance between the hyperplane and the nearest point from each class is maximized.

"Minimize \(\|\mathbf{w}\|\) subject to \(y_i (\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1\), for \(i = 1, \ldots, n\)"
Maximally separate the policy induced by our learned reward functions from suboptimal policies.
Maximum Margin Planning (MMP)

Standard formulation: \[
\min_w \|w\|_2^2
\]
\[
\text{s.t. } w^T \mu(\pi^*) \geq w^T \mu(\pi) + 1 \quad \forall \pi
\]

More involved formulation:
\[
\min_w \|w\|_2^2 + C\nu
\]
\[
\text{s.t. } w^T \mu(\pi^*) \geq w^T \mu(\pi) + m(\pi^*, \pi) - \nu \quad \forall \pi
\]

Give more margin if \(\pi\) and \(\pi^*\) are very different from each other.

Add slack variables to incorporate expert suboptimality.

Ratliff et al. 2006
Max Entropy IRL

Let $\xi = \{(s_1, a_1), \ldots, (s_T, a_T)\}$ be a sequence of state and actions. We let $D = \{\xi_1, \ldots, \xi_{|D|}\}$ to be the set of expert demonstrations.

Let’s define a feature function over trajectories: $f: \Xi \to \mathbb{R}^n$

$$f_D = \frac{1}{|D|} \sum_{\xi \in D} f(\xi)$$

Empirical feature expectations

$$\mathbb{E}_{\xi \sim P(\xi)} \left[ \sum_{t=1}^{T} \gamma^t R(s_t) \right] = \mathbb{E}_{\xi \sim P(\xi)} \left[ \sum_{t=1}^{T} \gamma^t w^T \varphi(s_t) \right] = w^T \mathbb{E}_{\xi \sim P(\xi)} [f(\xi)]$$

Expected Return

Weighted Feature Expectations
Max Entropy IRL

Selects the least committed distribution (maximizing entropy)

**Goal:** Find the distribution over the observations (expert trajectories) that matches empirical feature counts in expectation, and maximizes entropy.

\[
\max_{P} \int -P(\xi) \log P(\xi) \, d\xi
\]

s.t. \[\mathbb{E}_{\xi \sim P(\xi)}[f(\xi)] = \int P(\xi)f(\xi) \, d\xi = f_D\]

\[\int P(\xi) \, d\xi = 1\]

\[P(\xi) \geq 0, \quad \forall \xi \in \Xi\]
Max Entropy IRL

By solving the optimization, we will get:  

\[ P^*(\xi; \lambda) = \frac{\exp(\lambda^T f(\xi))}{\int \exp(\lambda^T f(\xi))} \]

We look for \( \lambda \) parameters that maximize the likelihood of observing expert trajectories

\[ \lambda^* = \arg \max_{\lambda} P(\xi_D; \lambda) = \arg \max_{\lambda} \lambda^T f(\xi_D) - \log(\int \exp(\lambda^T f(\xi))d\xi) \]

\[ \nabla_{\lambda} M = f(\xi_D) - \mathbb{E}_{\xi \sim P(\xi; \lambda)}[f(\xi)] \]

\[ \lambda_{i+1} \leftarrow \lambda_i + \alpha (f(\xi_D) - \mathbb{E}_{\xi \sim P(\xi; \lambda)}[f(\xi)]) \]
Max Entropy IRL

1) Initialize $\lambda$ and collect expert demonstrations $D$.
2) Solve for the optimal policy $\pi_\lambda(a|s)$ with respect to $\lambda$.
3) Solve for state visitation frequencies $p(x|\lambda)$.
4) Compute the gradient $\nabla_\lambda M$.
5) Update $\lambda$ with one gradient step.

This assumes access to the dynamics (transition function) and having low dimensional systems to be able to solve for the policy.
End-to-end driving via conditional imitation Learning

End-to-end Driving via Conditional Imitation Learning

Felipe Codevilla, Antonio López - Computer Vision Center (CVC)
Matthias Müller - King Abdullah University of Science and Technology (KAUST)
Vladlen Koltun, Alexey Dosovitskiy - Intel Visual Computing Lab

We propose conditional imitation learning which allows an autonomous vehicle trained end-to-end to be directed by high-level commands.

Experiments in simulation and on a physical vehicle show that the method allows for goal-directed navigation guided by a topological planner or a user.
Today’s itinerary

• Intro to Imitation Learning

• Behavioral Cloning, DAgger, COIL

• Inverse RL (Apprenticeship Learning, MMP, Max Ent IRL)

• Learning from other sources of data (preferences, physical feedback)

• Planning for robots based on human models
Next time

- Deriving Max Ent IRL formulation
- Learning from other sources of data (preferences, physical feedback)
- Planning for robots based on human models