Learning to behave via Imitation
ESSAI 2024 Course
Lecture 3/5

George Vouros

University of Piraeus, Greece

July 24, 2024

Outline

- Day 1: Motivation & Introduction to Deep Reinforcement Learning
- Day 2: Inverse Reinforcement Learning and Connections to Probabilistic Inference
- **Day 3: Imitation Learning**
- Day 4: Non-Markovian, Multimodal Imitation Learning
- Day 5: Imitating in Constrained Settings, Multiagent Imitation Learning.

Imitation Learning

Problem (ambiguous) statement

Given a set of demonstrated trajectories D generated by an unknown expert policy π_ϵ, learn a policy π that generates trajectories that are "as close as possible" to the expert trajectories.

Imitation learning

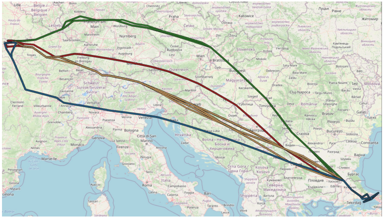
What can go wrong?

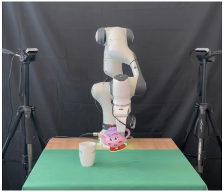
- Lack of training data
- Noisy or erroneous training data
- Distribution mismatch
- compounding errors
- Discrimination ability (different actions in very similar settings)
- Collapsing multi-modal behaviour in executing tasks in a single policy
- Being unaware of other agents' policies in multi-agent settings (collaborative or not)
- ... and others that will be revealed during the course

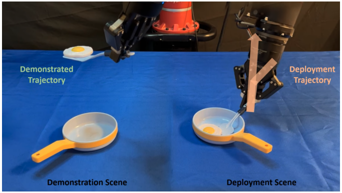
Introduction to (Deep) Reinforcement Learning

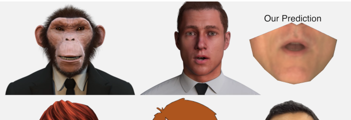
Reinforcement Learning provides a formalism for behaviour

Examples



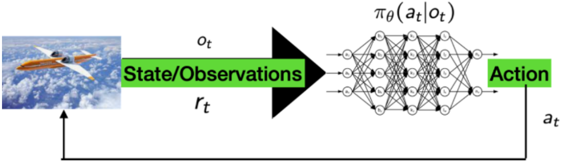


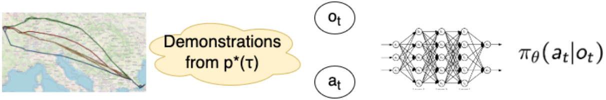




Imitation Learning

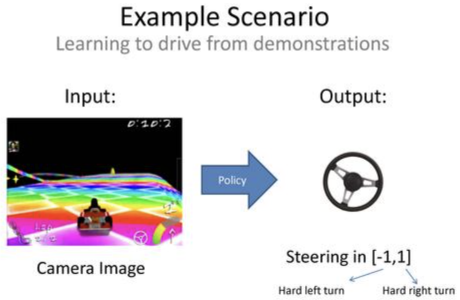
Basic Setting (Behavioural Cloning)





Imitation Learning

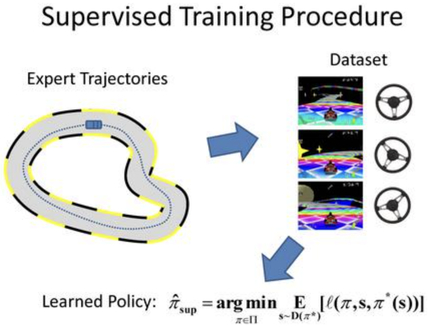
Basic Setting Example



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

Imitation Learning

Basic Setting Example



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

Imitation Learning

- Does Behavioural Cloning work well?
- Under which circumstances?
- How to mitigate limitations?
- Better algorithms?

Imitation Learning

What is important to imitation learning?

What is the major difference to supervised learning?

Imitation Learning

What is important to imitation learning?

The difference to supervised learning: **Test data are not i.i.d and depend on the policy (current decisions affect future states and observations)**

Imitation Learning

Basic Setting (Behavioural Cloning)

Dean Pomerleau
@deanpomerleau · **Follow**

Replying to @GTARobotics

GPU? Gez, ALVINN ran on 100 MFLOP CPU, ~10x slower than iWatch; Refrigerator-size & needed 5000 watt generator. @olivercameron

What's Hidden in the Hidden Layers?

The contents can be easy to find with a geometrical problem, but the hidden layers have yet to give up all their secrets

David S. Touretzky and Dean A. Pomerleau

AUGUST 1989 • BYTE 231

tions, we fed the network road images taken under a wide variety of viewing angles and lighting conditions. It would be impractical to try to collect thousands of real road images for such a data set. Instead, we developed a synthetic road-image generator that can create as many training examples as we need.

To train the network, 1200 simulated road images are presented 40 times each, while the weights are adjusted using the back-propagation learning algorithm. This takes about 30 minutes on Carnegie Mellon's Warp systolic-array supercomputer. (This machine was designed at Carnegie Mellon and is built by General Electric. It has a peak rate of 100 million floating-point operations per second can compute weight adjustments back-propagation networks at a rate of 20 million connections per second.)

Once it is trained, ALVINN can accurately drive the NAVLAB vehicle at about 3½ miles per hour along a path through a wooded area adjoining the Carnegie Mellon campus, under a variety of weather and lighting conditions. This speed is nearly twice as fast as that achieved by non-neural-network algorithms running on the same vehicle. Part of the reason for this is that the forward

milliseconds on the Sun-3/160 workstation installed on the NAVLAB.

The hidden-layer representations ALVINN develops are interesting. When trained on roads of a fixed width, the network chooses a representation in which hidden units act as detectors for complete roads at various positions and orientations. When trained on roads of variable

continued

Photo 1: *The NAVLAB autonomous navigation test-bed vehicle and the road used*

Imitation Learning

Autonomous Land Vehicle In a Neural Network (ALVINN)

The video

Imitation Learning

Basic Setting (Behavioural Cloning)

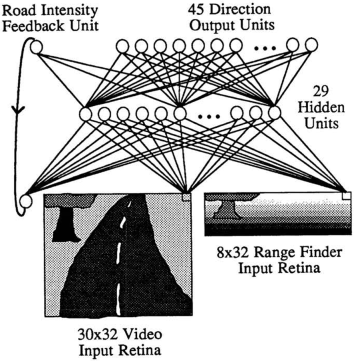
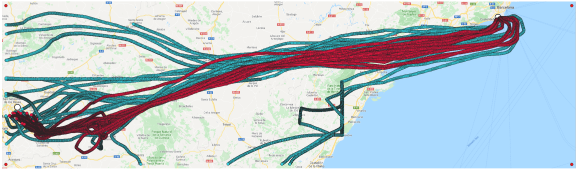


Figure 1: ALVINN Architecture

Imitation Learning

Why does not work in general?



Imitation Learning

Potential factors making it work well

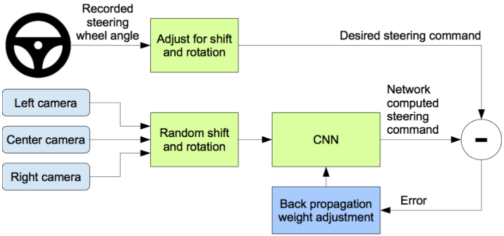
- Not much stochasticity in demonstrated actions
- Similar situations require same action
- Situations not from the training data set are unlikely

Imitation Learning

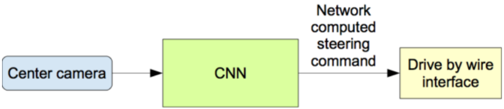
Can we make it work?

Ans: YES! : The DAVE autonomous car Video

During training:



During inference:



Bojarski et al (2016) INVIDIA

Imitation Learning

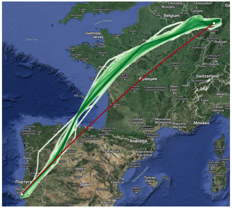
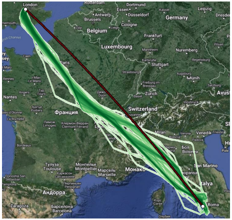
Can we make it work?

Ans: YES! : The Quadcopter Video

Imitation Learning

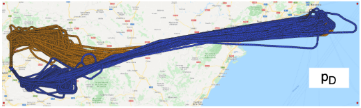
Potential factors making it work well

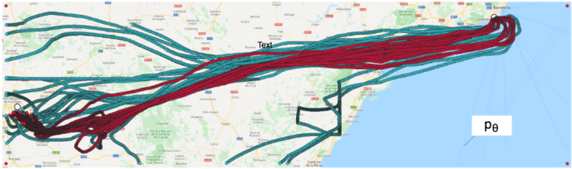
- Collect data covering the states and actions space or be smart about it.
- Train robust and/or powerful models
- Transfer knowledge from different tasks

[LIS-FRA]  [LHR-FCO] 

Imitation Learning

A subtle issue



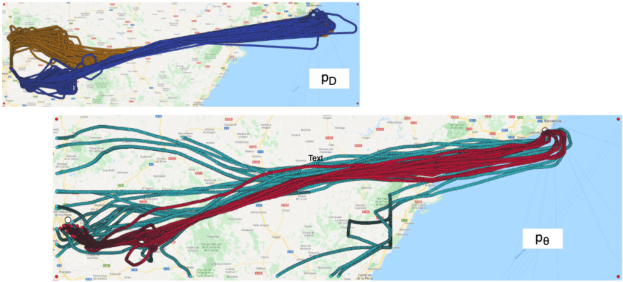


We train π_θ using samples from p_D, and this results into generating output under p_θ, different from p_D.
Under the perspective of "compounding errors", there is a subtle difference between supervised and imitation learning:

- **Supervised learning:** $\max_\theta \mathbb{E}_{s_t \sim p_D(s_t)}[log \pi_\theta(a_t|s_t))]$
- **Imitation learning:** $\min \mathbb{E}_{s_t \sim p_\theta(s_t)}[c(s_t, a_t))]$

Imitation Learning

A subtle issue



- **Imitation learning:** $\min \mathbb{E}_{s_t \sim p_\theta(s_t)}[c(s_t, a_t))]$
  The cost measures the mistaken decisions made by π_θ

Imitation Learning

A subtle issue

- **Imitation learning:** $\min \mathbb{E}_{s_t \sim p_\theta(s_t)}[c(s_t, a_t))]$
  The cost measures the mistaken decisions made by π_θ.
  A simple one:

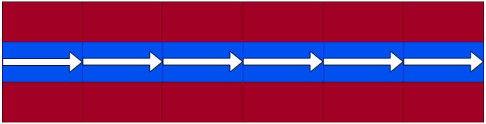
$$c(s_t, a_t) = \begin{cases} 0 & \textit{if } a_t = \pi^*(s_t) \\ 1 & \textit{otherwise} \end{cases}$$

Imitation Learning

What is the worst case?

$$c(s_t, a_t) = \begin{cases} 0 & \textit{if } a_t = \pi^*(s_t) \\ 1 & \textit{otherwise} \end{cases}$$

assume that $\pi_\theta(a \neq \pi^*(s)|s) \leq \epsilon$, for all $s \in D_{train}$



Cost to fail at step k: $(1\text{-e})^{k\text{-}1}$e (T-k-1),
k=1, eT
k=2, (1-e)e(T-1)
k=3, $(1\text{-e})^2$e(T-2)
etc.

Therefore,

$$J(\pi_\theta) = \mathbb{E}[\sum_t c(s_t, a_t)] = O(\epsilon T^2)$$

Because it does not know how to recover from errors.

Imitation Learning

How worse can it be for $s \sim p_{train}$?

$$c(s_t, a_t) = \begin{cases} 0 & \textit{if } a_t = \pi^*(s_t) \\ 1 & \textit{otherwise} \end{cases}$$

assume that $\pi_\theta(a \neq \pi^*(s)|s) \leq \epsilon$, for all $s \sim p_{train}$
and $\mathbb{E}_{p_{train}(s)}[\pi_\theta(a \neq \pi^*(s)|s)] \leq \epsilon$

The bound

It can be shown that

$$J(\pi_\theta) = \mathbb{E}[\sum_t c(s_t, a_t)] = O(\epsilon T^2)$$

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

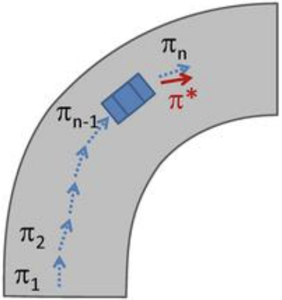
Imitation Learning

Early reduction-based approaches

Previous Work: Forward Training

[Ross 2010]

- Sequentially learn one policy/step
- # mistakes grows linearly:
  - $J(\pi_{1:T}) \leq T\varepsilon$
- **Impractical if T large**



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

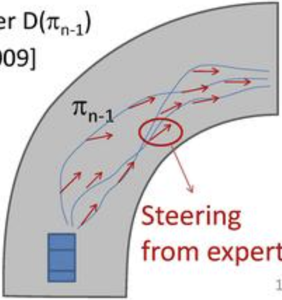
Imitation Learning

Early approaches

Previous Work: SMILe

[Ross 2010]

- Learn stochastic policy, changing policy slowly
  - $\pi_n = \pi_{n-1} + \alpha_n(\pi'_n - \pi^*)$
  - π'_n trained to mimic π^* under $D(\pi_{n-1})$
  - Similar to SEARN [Daume 2009]
- Near-linear bound:
  - $J(\pi) \leq O(T\log(T)\varepsilon + 1)$
- **Stochasticity undesirable**



10

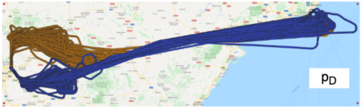
Ross et al., (2011) "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning"

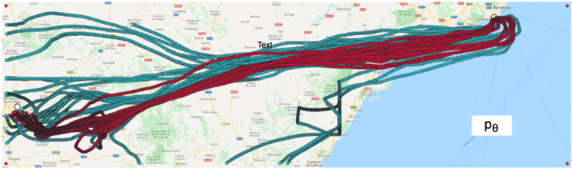
Imitation Learning

Main idea:

Intentionally add (some) mistakes and actions of recovery:
Essentially, shift $p_{train} = p_D$ towards p_θ, or incorporate agents' experience given $s \sim p_\theta$ into $p'_{train} = p'_D$
Also called "data augmentation".

Imitation Learning





Making $p_D(s_t) = p_\theta(s_t)$

So, augment $p_D(s_t)$ by states (and expert labels) sampled from $p_\theta(s_t)$.

DAgger: Dataset Aggregation

Collect data from $p_\theta(s_t)$, by running the policy $\pi_\theta(a_t|s_t)$ and getting labels a_t for unseen states s_t.

Imitation Learning

DAgger: Dataset Aggregation

Collect data from $p_\theta(s_t)$, by running the policy $\pi_\theta(a_t|s_t)$ and getting labels a_t for unseen states s_t.

DAgger: The algorithm

1. train π_θ from expert demonstrations $D = \{s_1, a_1, s_2, a_2, ..., s_T, a_T\}$
2. run $\pi_\theta(a_t|s_t)$ to get dataset $D_\theta = \{s'_1, s'_2, ..., s'_H\}$
3. ask the human to label D_θ with actions a_t
4. Aggregate $D \leftarrow D \cup D_\theta$
5. GoTo 1

Imitation Learning

DAgger: The algorithm

1. train p_θ from expert demonstrations
   $D = \{s_1, a_1, s_2, a_2, ..., s_T, a_T\}$
2. run $p_\theta(a_t|s_t)$ to get dataset $D_\theta = \{s'_1, s'_2, ..., s'_H\}$
3. ask the human to label D_θ with actions a_t
4. Aggregate $D \leftarrow D \cup D_\theta$
5. GoTo 1

DAgger: The problem

The problem is step 3: Humans cannot easily label a huge amount of data or even a small amount of data with detailed, multi-dimensional, continuous actions.

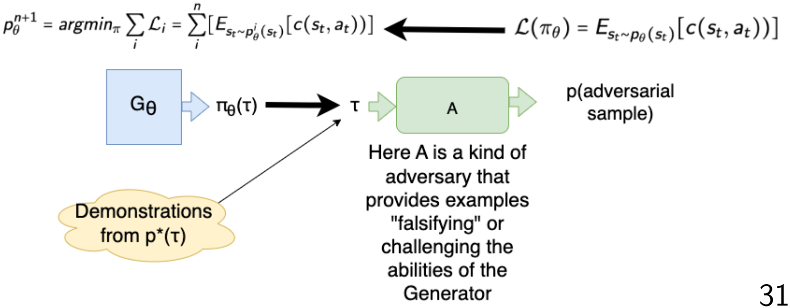
DAgger in the context of adversarial and online learning

Cost of π_θ:

$$\mathcal{L}(\pi_\theta) = \mathbb{E}_{s_t \sim p_\theta(s_t)}[c(s_t, a_t))]$$

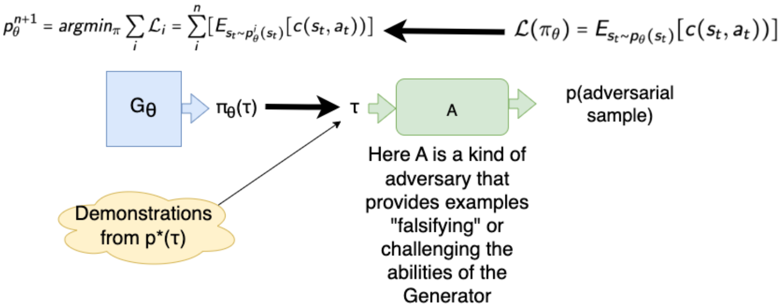
Learning a new policy:

$$\pi_\theta^{n+1} = argmin_\pi \sum_{i=1}^{n} \mathcal{L}_i = argmin_\pi \sum_{i=1}^{n} \mathbb{E}_{s_t \sim p_\theta^i(s_t)}[c(s_t, a_t)]$$



Imitation Learning

DAgger in the context of adversarial and online learning

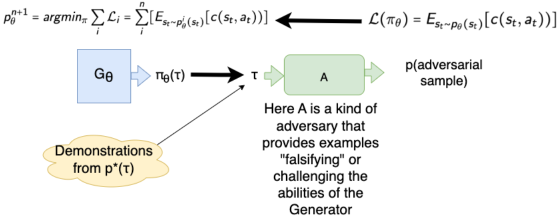


Avg.Regret

$$\gamma_n = \frac{1}{n}[\sum_{i=1}^{n} \mathcal{L}_i(\pi_\theta^i) - min_{\pi_\theta \in \Pi} \sum_{i=1}^{n} \mathcal{L}_i(\pi_\theta)]$$

Imitation Learning

Theoretical guarantees of DAgger



The best policy in the sequence of policies $\pi^{1:N}$ guarantees:

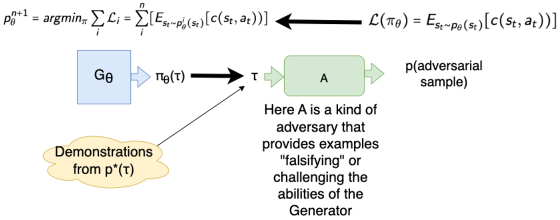
$$J(\pi_\theta) \leq T(\epsilon_N + \gamma_N) + O(T/N)$$

where,

- ϵ_N: Average loss on aggregated dataset
- γ_N: Average regret of $\pi^{1:N}$
- N: Iterations of DAgger

Imitation Learning

Theoretical guarantees of DAgger



The best policy in the sequence of policies $\pi^{1:N}$ guarantees:

$$J(\pi_\theta) \leq T(\epsilon_N + \gamma_N) + O(T/N)$$

Follow-the-Leader is a no-regret algorithm.
For strongly convex loss, $N = O(T log T)$ iterations:

$$J(\pi_\theta) \leq T\epsilon_N + O(1)$$

Imitation Learning

DAgger: The algorithm

1. train p_θ from expert demonstrations $D = \{s_1, a_1, s_2, a_2, ..., s_T, a_T\}$
2. run $p_\theta(a_t|s_t)$ to get dataset $D_\theta = \{s'_1, s'_2, ..., s'_H\}$
3. ask the human to label D_θ with actions a_t
4. Aggregate $D \leftarrow D \cup D_\theta$
5. GoTo 1

DAgger: The problem

The problem is step 3: Humans cannot easily label a huge amount of data or even a small amount of data with detailed, multi-dimensional, continuous actions.
For DAgger, if the number of trajectories sampled per iteration is small, then the probability of getting a high bound on error increases.

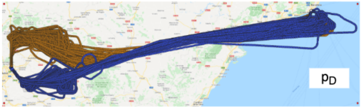
Imitation Learning

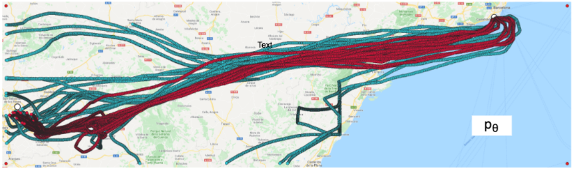
Can machines learn autonomously?

Lets revisit the objective of minimizing mistaken decisions.

Imitation Learning

A subtle issue





- **Imitation learning:** $\min \mathbb{E}_{s_t \sim p_\theta(s_t)}[c(s_t, a_t))]$
  The cost measures the mistaken decisions made by π_θ.

$$c(s_t, a_t) = \begin{cases} 0 & \textit{if } a_t = \pi^*(s_t) \\ 1 & \textit{otherwise} \end{cases}$$

Imitation Learning

A subtle issue

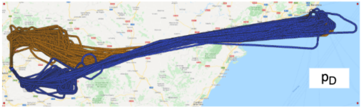
Imitation learning objective

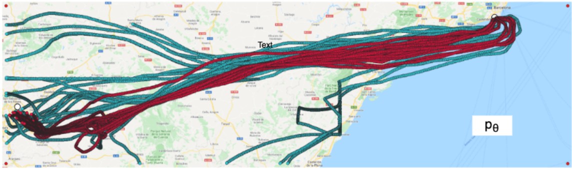
$$\min_{\theta} \mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t), s_{t+1} \sim p(s_{t+1}|s_t, a_t)}[c(s_{t+1})]$$

$$\min_{\theta} \mathbb{E}_{s_{1:T}, a_{1:T}}[\sum_t c(s_t, a_t)]$$

Imitation Learning

A subtle issue





Imitation learning objective

$$\min_{\theta} \mathbb{E}_{s_{1:T}, a_{1:T}} \Big[\sum_{t} -r_E(s_t, a_t) \Big]$$

Imitation Learning

Assuming that demonstrations are produced by THE expert, who acts with a reward r_E, unknown to us.

$$\max_{\theta} \mathbb{E}_{s_{1:T}, a_{1:T}} [\sum_t r_E(s_t, a_t)]$$

recall that given the probabilistic model

$$p(\mathcal{O}_{1:T}|\tau) = exp(\sum_t r_E(s_t, a_t))$$

thus,

$$\sum_t r_E(s_t, a_t) = logp(\mathcal{O}_{1:T}|\tau)$$

Imitation Learning

$$\max_{\theta} \mathbb{E}_{s_{1:T}, a_{1:T}} [\sum_t r_E(s_t, a_t)]$$

with

$$\sum_t r_E(s_t, a_t) = logp(\mathcal{O}_{1:T} | \tau)$$

Maximum likelihood learning:

$$\mathcal{L} = max_{\theta} \frac{1}{N} \sum_{i=1}^{N} logp(\tau_i | \mathcal{O}_{1:T}, E) = max_{\theta} \frac{1}{N} \sum_{i=1}^{N} r_E(\tau_i) - logZ$$

$$Z = \int p(\tau) exp(r_E(\tau)) \, d\tau$$

Imitation Learning

Maximum likelihood learning:

$$\mathcal{L} = max_\theta \frac{1}{N}\sum_{i=1}^{N} logp(\tau_i|\mathcal{O}_{1:T}, E) = max_\theta \frac{1}{N}\sum_{i=1}^{N} r_E(\tau_i) - logZ$$

$$Z = \int p(\tau) exp(r_E(\tau))\, d\tau$$

and it turns out that the objective is:

$$\mathcal{L} = max_\theta[\mathbb{E}_{\tau\sim\pi^*(\tau)} r_E(\tau) - \mathbb{E}_{\tau\sim\pi_\theta(\tau)} r_E(\tau)]$$

Assuming the r_E is known, in the MaxEnt RL setting

$$\mathcal{L}' = max_\theta(\mathbb{E}_{\tau\sim\pi_\theta(\tau)}[r_E(\tau)] + \mathbb{E}_{\pi_\theta}[\mathcal{H}(\pi_\theta(\tau))]) - \mathbb{E}_{\tau\sim\pi^*(\tau)} r_E(\tau)$$

Imitation Learning

However, the expert reward is unknown, so the objective in an IRL setting would be:

$$\mathcal{L}' = min_r[max_\theta \mathbb{E}_{\tau\sim\pi(\tau)}[r(\tau) + \mathcal{H}(\pi(\tau))]] - \mathbb{E}_{\tau\sim\pi^*(\tau)} r(\tau)]$$

Imitation Learning

Given this objective, can we approximate the expert policy without learning the reward?

$$\mathcal{L} = min_r[max_\theta \mathbb{E}_{\tau \sim \pi(\tau)}[r(\tau) + \mathcal{H}(\pi(\tau)))] - \mathbb{E}_{\tau \sim \pi^*(\tau)} r(\tau)]$$

In an IRL setting, we solve this problem by finding a reward function such that the expert performs better than the other policies.
An then, running RL on the output of IRL to approximate the expert policy.
Can we skip the first part, avoiding the RL part for every reward approximation?

Imitation Learning

Given this objective, can we approximate the expert policy without learning the reward?

Consider occupancy measures: states, actions distributions that an agent encounters when navigating the environment with policy π

$$\rho : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$$

Defined to be[1]

$$\rho(s, a) = \pi(a|s) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi).$$

[1] Ho & Ermon, Generative Adversarial Learning, 2016

Imitation Learning

Imitation Learning in Constrained Settings

As shown by (Puterman, 1994) the set of valid occupancy measures can be written as a feasible set of affine constraints

$$\mathbb{D} = \{\rho : \rho \geq 0 \text{ and } \\ \sum_a \rho(s,a) = \mu(s) + \gamma \sum_{s',a} P(s|s',a)\rho(s',a), \\ \forall s \in S\}$$

Very inefficient to evaluate and we need to know the transition function.

.

Puterman, M. L. "Markov Decision Processes: Discrete Stochastic Dynamic Programming". John Wiley Sons, Inc., USA, 1st edition, 1994.

Imitation Learning

Imitation Learning in Constrained Settings

Given $\mathbb{D}$ here is an one-to-one relation between occupancy measures and policies.

text given $\rho \in \mathbb{D}, \rho(s,a) \longleftrightarrow \pi_\rho(s,a) = \frac{\rho(s,a)}{\sum_{a'} \rho(s,a')}$

– Syed, U., et al., "Apprenticeship learning using linear programming", 2008.

– Ho, J. and Ermon, S. Generative adversarial imitation learning, 2016.

Imitation Learning

Imitation Learning in Constrained Settings

Given that the causal entropy

$$\tilde{H} = -\sum_{s,a} \rho(s,a)(log(\rho(s,a)/\sum_{a'} \rho(s,a'))$$

for occupancy measures is strictly concave, and for all $\pi \in \Pi$ and $\rho \in \mathbb{D}$, it holds that

$$\mathcal{H}(\pi) = \tilde{\mathcal{H}}(\rho_\pi) \text{ and } \mathcal{H}(\pi_\rho) = \tilde{\mathcal{H}}(\rho)$$

, allow us to switch between policies and occupancy measures when considering functions involving causal entropy and expected rewards.

Ho, J. and Ermon, S. Generative adversarial imitation learning, 2016.

Imitation Learning

So, what about "matching" occupancy measures between π_E and π_θ?

Given the objective

$$\mathcal{L} = min_r[max_\theta \mathbb{E}_{\tau \sim \pi(\tau)}[r(\tau) + \mathcal{H}(\pi(\tau)))] - \mathbb{E}_{\tau \sim \pi^*(\tau)} r(\tau)]$$

This results to the following optimization problem

$$max_{\pi_\theta \in \Pi} H(\pi_\theta) \;\; \text{subject to } \rho_{\pi_\theta}(s, a) = \rho_E(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}$$

Imitation Learning

So, what about "matching" occupancy measures between π_E and π_θ?

Given the objective

$$\mathcal{L} = min_r [max_\theta \mathbb{E}_{\tau \sim \pi(\tau)}[r(\tau) + \mathcal{H}(\pi(\tau)))] - \mathbb{E}_{\tau \sim \pi^*(\tau)} r(\tau)]$$

This results to the following optimization problem

$$max_\theta [\lambda H(\pi_\theta) \; - D_m(\rho_{\pi_\theta}(s,a), \rho_E(s,a))]$$

where, D_m is a distance metric between distributions, penalizing violations in difference between occupancy measures.

Imitation Learning

So, what about "matching" occupancy measures between π_E and π_θ?

Given the optimization problem

$$max_\theta[\lambda H(\pi_\theta) \ - D_m(\rho_{\pi_\theta}, \rho_E(s,a))]$$

and setting

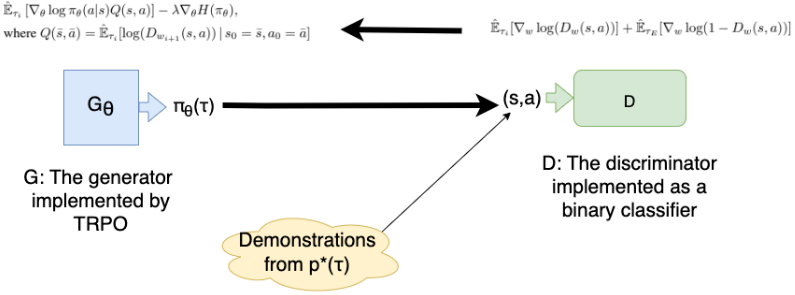
$$\begin{aligned}
&D_m(\rho_{\pi_\theta}, \rho_E(s,a)) = \\
&D_{JS}(\rho_{\pi_\theta}, \rho_E(s,a)) = \\
&D_{KL}(\rho_{\pi_\theta} \| (\rho_{\pi_\theta} + \rho_E)/2) + D_{KL}(\rho_E \| (\rho_{\pi_\theta} + \rho_E)/2)
\end{aligned}$$

it turns out the optimal loss is the optimal negative log loss of the binary classification problem of distinguishing state,action pairs of π_θ and π_E:

$$max_{D \in (0,1)^{\mathcal{S} \times \mathcal{A}}} \mathbb{E}_{\pi_\theta}[log(D(s,a))] + \mathbb{E}_{\pi_E}[log(1 - D(s,a))]$$

Inverse reinforcement learning: connection to probabilistic models

Generative Adversarial Imitation Learning (GAIL)



The generator improves itself to foul the discriminator, while the discriminator is updated to distinguish expert samples from those of the generator, as better as possible.

Inverse reinforcement learning: connection to probabilistic models

Generative Adversarial Imitation from Observations (GAIfO)