2024.09.09 → (1) Behavioral Cloning
(2) Inverse RL

- **Imitation Learning**
  - → Given : Expert or expert's demonstration
  - → Goal : Train a policy (OR abstraction) to mimic demonstrations
    - function maps state → action
    - "Learn human intents, preferance, underlying reward functions"

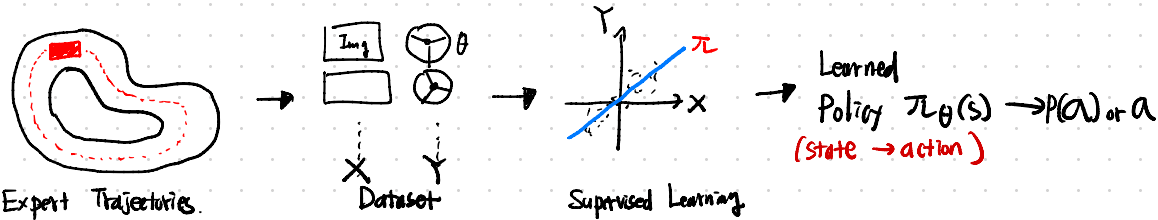
→ We need (1) demonstration or demonstrator (2) Environment / Simulator (3) Policy Class (which function family?)
(4) Loss Function (5) Learning Algorithm

- **Problem Setup : Markov Decision Process without Reward.**
  - State space S [can be partially observable], $s \in S$
  - Action space A, $a \in A$
  - An expert policy $\pi^*: S \rightarrow P(a)$ or a
    - State
    - distribution over actions.
    - or trace of π^* is Demonstration Trajectory
      $\tau = (s_0, a_0, s_1, a_1, s_2, a_2, \ldots)$
      ↳ "Roll Out"
  - Transition model $P_{sa}(s_{t+1} \mid s_t, a_t)$: characteristic / nature of simulator or environment

Goal :
↳ Inverse RL : Can we recover reward from π^*? (assume experts are optimal)
↳ Behavioral Cloning : Can we train π_θ using supervised learning?

Ex) Learning to Drive by Imitation.
- Input : RGB Image, Output : Steering Angle in $[-1, 1]$

- **Behavior Cloning : Offline & Supervised Imitation Learning.**



Expert Trajectories. → Dataset → Supervised Learning "Fixed Policy Class" (i.e. SVM, NN, ...) → Learned Policy $\pi_\theta(s) \rightarrow P(a)$ or a (state → action)

Formular of Behavior Cloning Algorithm

- Algorithm's Input : A restricted policy class Π

- Learning Objective : Train $\pi_{\theta^*}(s)$ where $\theta^* = \arg\min_\theta \mathbb{E}_{(s,a^*)\sim p^*} \mathcal{L}(a^*, \pi_\theta(s))$

  $p^* = P(s|\pi^*)$: distribution of states visited by π^*

  $\mathcal{L}$: Loss function (can vary)

  i.e.) Negative Log-likelihood (NLL) $\mathcal{L}(a^*, \pi_\theta(s)) = -\log \pi(a^*|s)$

  i.e.) Square loss $\mathcal{L}(a^*, \pi_\theta(s)) = \|\pi_\theta(s) - a^*\|_2^2$

  OR) KL-Divergence....

  Depends on the form of Policy.

  OR $\hat{\pi} = \arg\min_{\pi\in\Pi} \sum_{(s,a^*)\in\tau} \mathcal{L}(a^*, \pi(s))$ (τ : Traj.)

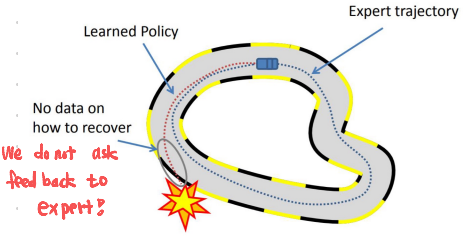
  OR $\hat{\pi} = \arg\min_{\pi\in\Pi} \sum_{\tau} \sum_{(s,a^*)\in\tau} \mathcal{L}(a^*, \pi(s))$ ($\pi^*(s)$)

★ General IL: minimize $\mathbb{E}_{s\sim p(s|\theta)} L(a^*, \pi_\theta(s))$

"Assume perfect imitation so far,
Learn to continue this perfect imitation,
Minimize 1-step deviation error along the expert traj."

★ Distribution Shift (mismatch) in BC.

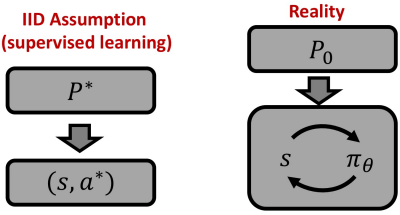
→ Data distribution mismatch



★ Supervised learning assumes (s,a) are i.i.d.
(Independent & identically distributed)
→ If error for each time is ϵ, error in time period T is expected to bounded in ϵT.

<Training> $s \sim p^*$

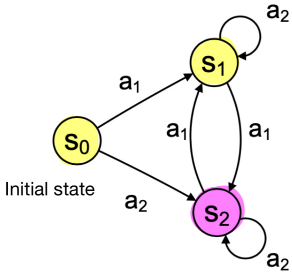
<Inference> $s \sim P(s|\pi_\theta)$: unseen states! ~ makes mistake & cannot recover.



∴ BC : Simple, efficient but No long-term planning.

USE → 1-step deviations not too bad, Learning Reactive behavior, expert traj. cover S.

DON'T → 1-step deviation lead to disaster, has long-term objective.



● : Training ● : Test.

If $\hat{\pi}(s_0) = \begin{cases} a_1 & \text{w. prob } 1-\frac{\epsilon}{1-\epsilon} \\ a_2 & \text{w. prob } \frac{\epsilon}{1-\epsilon} \end{cases}$

$\hat{\pi}(s_1) = a_2$, $\hat{\pi}(s_2) = a_2$

If mistake is made at s_0, we stay in s_2, ∞.

How to overcome distribution shift in BC.

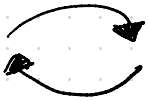
⇒ Interactive Expert.

→ Query π^* at any s, Build loss as $L(\pi^*(s), \pi_\theta(s))$

↳ provides feedback on state visited by π_θ.

(Naive Approach) : Not guaranteed to converge.

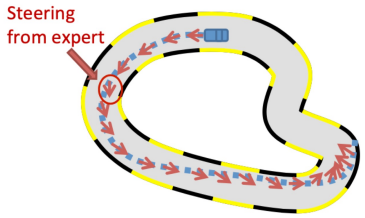
Fix P
Estimate π_θ
Solve $\operatorname{argmin}_\theta \mathbb{E}_{s\sim p} L(\pi^*(s), \pi_\theta(s))$



Fix π_θ
Estimate P_θ empirically
by rolling out π_θ

(Sequential Learning Reductions) Harder ↦ easier.

1) set initial predictor π_0
2) for $1:m$,
   → Collect τ via rolling out π_{n-1} for multiple times
   → Estimate state distribution p_n with $s \in \tau$.
   → Collect feedback $\{\pi^*(s) \mid s \in \tau\}$
   ↳ Interactive expert



→ Data Aggregation (i.e. DAgger)
→ Policy Aggregation (i.e. SEARN & SMILe)

✱ DAgger.

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.

decreasing coefficient.
"mixture" policy.

✱ SMILe & SEARN.

: at iter n,

$$\pi^n = (1-\alpha)\pi^{n-1} + \alpha \hat{\pi}^n$$

current policy | old policy | new policy trained by querying π^* under P_{n-1}.

$$\hat{\pi}^n = \operatorname*{argmin}_{\pi \in \Pi} \mathbb{E}_{s\sim P_{n-1}} \left(\mathcal{L}(a^*, \pi(s))\right)$$

after N iters, final policy is $\tilde{\pi}^N = \dfrac{\pi^N - (1-\alpha)^N \pi^*}{1-(1-\alpha)^N}$

ETC

Learning by
- Watching (Imitation) : Need to define correspondence (i.e. body matching.) → Object-based, End effector-based, Shadowing, Retargeting
- doing (Demonstration) Avoid correspondence → Tele-operating, Kinesthetic Teaching. Key frame demonstration