



上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Imitation Learning

Overview

- Introduction to imitation learning
- Core methods of imitation learning
- Advanced works on imitation learning
- Connection between imitation learning and GANs

Where does the Reward Function Come from?

Computer Games

reward



Mnih et al. '15

Real World Scenarios

robotics



dialog



autonomous driving



what is the **reward**?
often use a proxy

- Frequently easier to provide expert data than reasonable reward function
- Inverse reinforcement learning: infer reward function from demonstrations (rollouts) of expert policy

Imitation Learning for Auto-driving

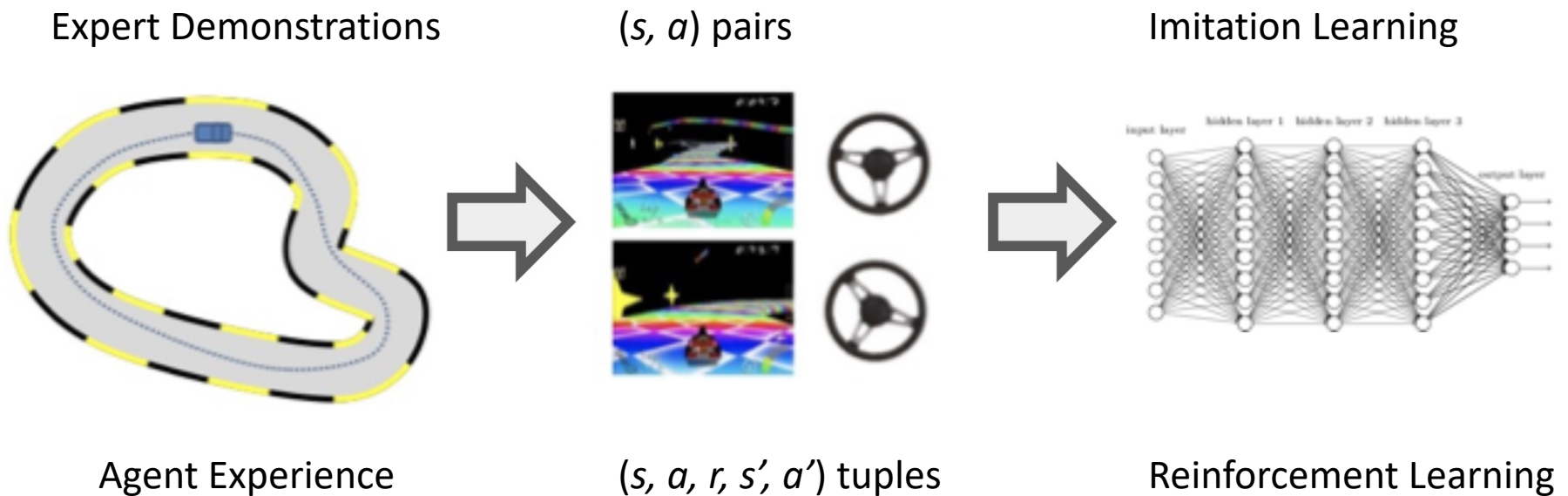


Waymo has made simulation one of the pillars of its autonomous vehicle development program. But **Latent Logic** could help Waymo make its simulation more realistic by using a form of machine learning called **imitation learning**.

Imitation learning models human behavior of motorists, cyclists and pedestrians. The idea is that by modeling the mistakes and imperfect driving of humans, the simulation will become more realistic and theoretically improve Waymo's behavior prediction and planning.

Imitation Learning in a Nutshell

- Given: demonstrations or demonstrator
 - Normally without any reward signals
- Goal: train a policy to mimic demonstrations
 - And achieve good policy performance



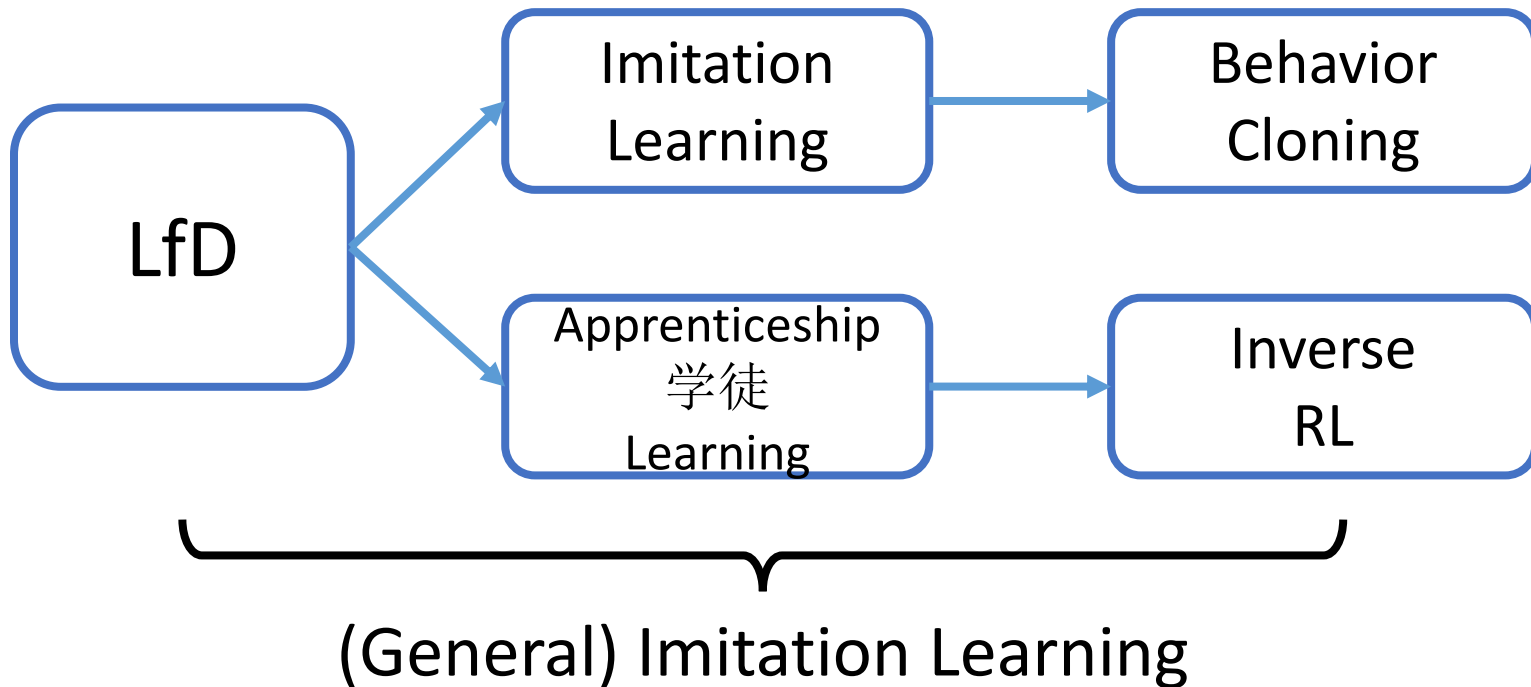
Imitation Learning Approaches on Super Tux Kart



<https://www.youtube.com/watch?v=V00npNnWzSU>

What is Imitation Learning

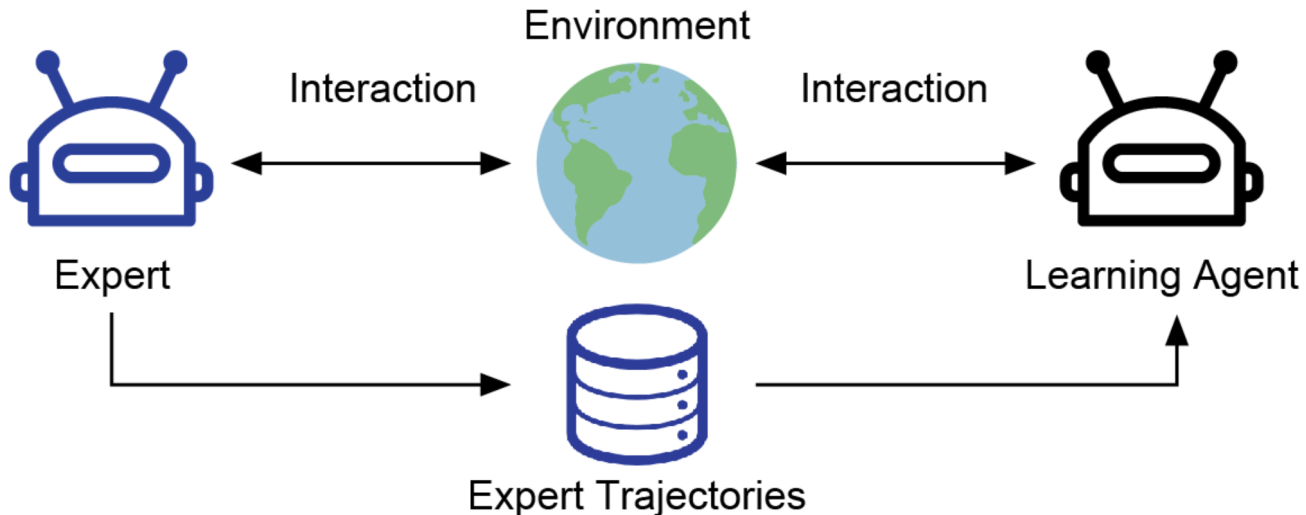
- Where from?
 - Learning from expert demonstration (LfD)
 - Try to imitate from the expert demonstrations



What is Imitation Learning

General setting (in this lecture):

- The learning agent
 1. can obtain pre-collected trajectories ((s,a) pairs) from uninteractive expert
 2. can interact with the environments (with simulators)
 3. cannot access reward signals



What is Imitation Learning

Other optional settings

- No actions and only state / observations -> **Imitation Learning From Observations (ILFO)**
- With reward signals -> **Imitation Learning with Rewards**
- Interactive expert for correctness and data aggregation -> **On-policy Imitation Learning (begin as Dagger, Dataset Aggregation)**
- Cannot interact with Environments -> **A special case of Batch RL (data in Batch RL can contain more than expert demos)**

What is Imitation Learning

More Considerations

- Imitation loss
- Suboptimal demonstrations
- Partial demonstrations (e.g., weak feedback)
- Domain transfer (e.g., few-shot learning)
- Structured domains (e.g., multi-agent systems, structured prediction)

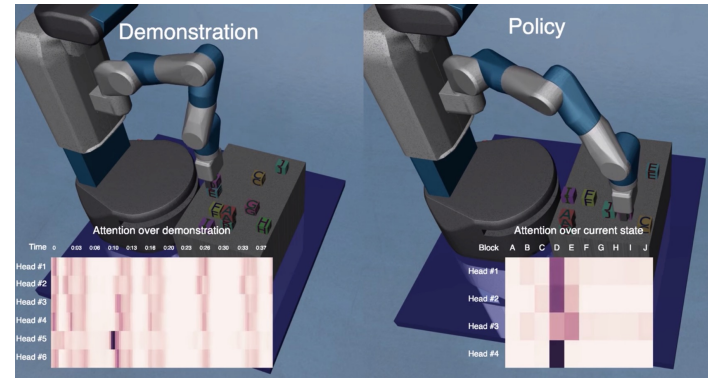
What is Imitation Learning

- When we necessarily want IL?
 - Hard to define the reward in some tasks
 - Hand-crafted rewards can lead to unwanted behavior
- What we want from IL?
 - Less interact with the **real-world** environments with expert demonstrations to improve sample efficiency and learn good policies
 - A fast and not bad policy initialization
 - A good solution that is robust to environment's slight changes (compared to RL normally overfitting the env)

Applications



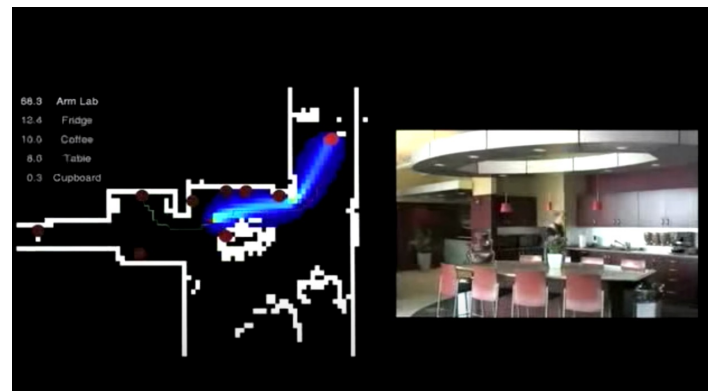
Helicopter Acrobatics [1,2]



Robotics Arm [3]



Sports Analysis [4]



Robotics Movement [5]

Formulation

- Notation & setup 1
 - State: s
 - State may only be partially observed, i.e., o
 - Action: a
 - Policy: π
 - Policy maps states to actions: $\pi(s) \rightarrow a$
 - or distributions over actions: $\pi(s) \rightarrow P(a)$
 - State transition dynamics: $P(s'|s, a)$
 - Typically not known to policy
 - Essentially the simulator/environment

Formulation

- Notation & setup 2

- Rollout: sequentially execute $\pi(s_0)$ on an initial state
 - Produce trajectory $\tau = (s_0, a_0, s_1, a_1, \dots)$

- $P(\tau|\pi)$: distribution of trajectories induced by a policy
 - 1. Sample s_0 from ρ_0 (distribution over initial states), initialize $t = 0$
 - 2. Sample action a_t from $\pi(s_t)$
 - 3. Sample next state s_{t+1} from applying a_t to s_t (requires access to environment)
 - 4. Repeat from Step 2 with $t = t + 1$

Formulation

- Notation & setup 3

- $P(s|\pi) \rightarrow \rho_\pi(s)$: distribution of states induced by a policy

$$\rho_\pi(s) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t P(s_t = s | \pi)$$

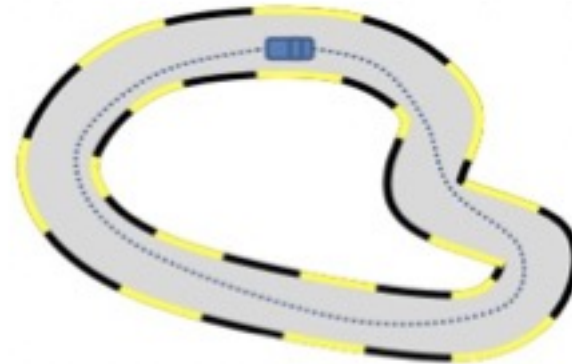
- $P(s, a|\pi) \rightarrow \rho_\pi(s, a)$: distribution of state-action pairs induced by a policy (known as **occupancy measure**)

$$\rho_\pi(s, a) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t P(s_t = s, a_t = a | \pi)$$

$$\rho_\pi(s, a) = \pi(a|s)\rho_\pi(s)$$

Formulation & Example 1

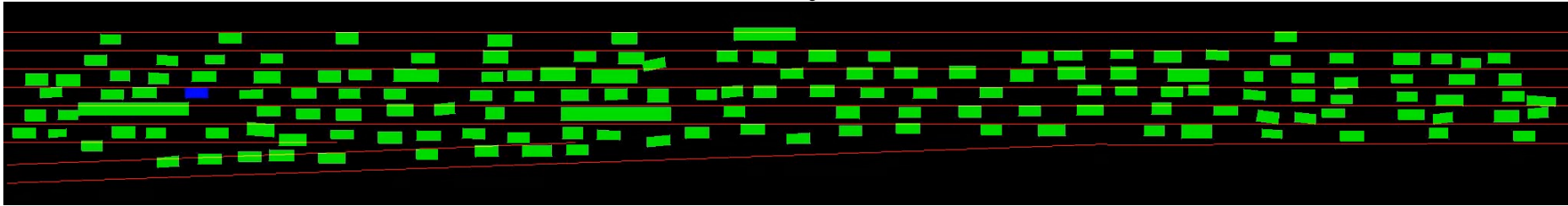
- Observation s = game screen
- Action a = turning angle
- Training set $D = \{\tau = [(s, a)]\}$ from an expert policy π^*
- **Goal:** learn a good policy $\pi(s) \rightarrow a$ that achieves high value



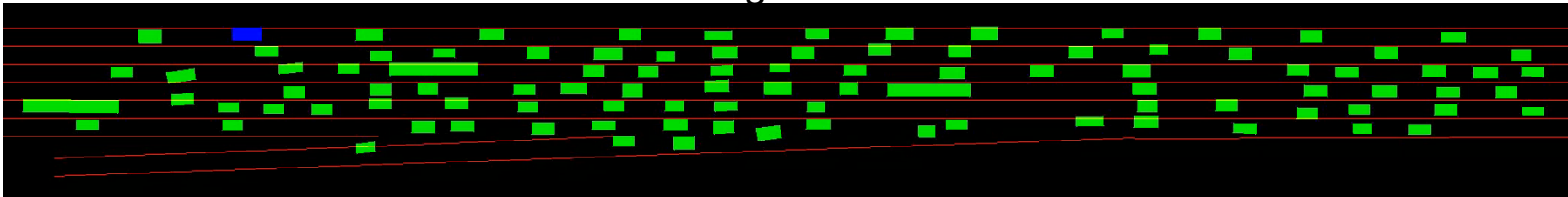
Formulation & Example 1

- NGSim dataset ■ : History Data Replay ■ : Policy Controlled Agent

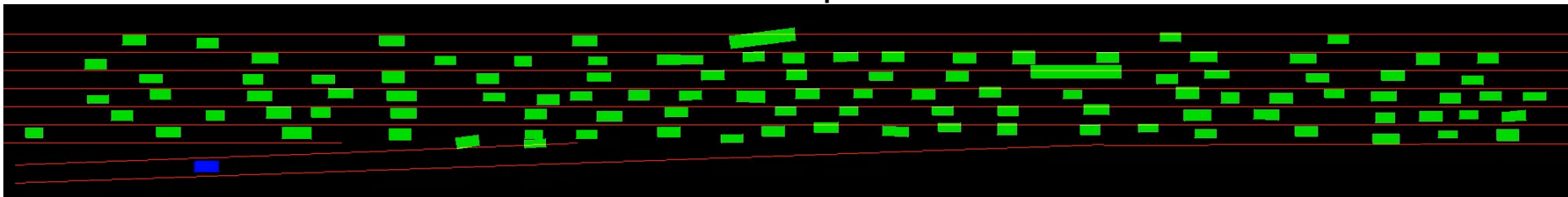
Heavy Traffic



Light Traffic

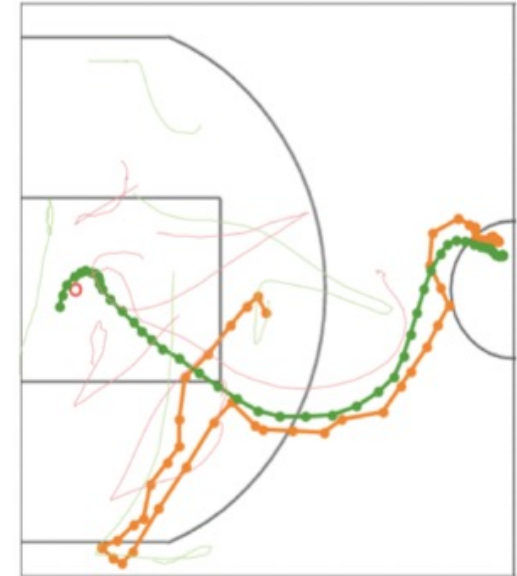


Ramp In

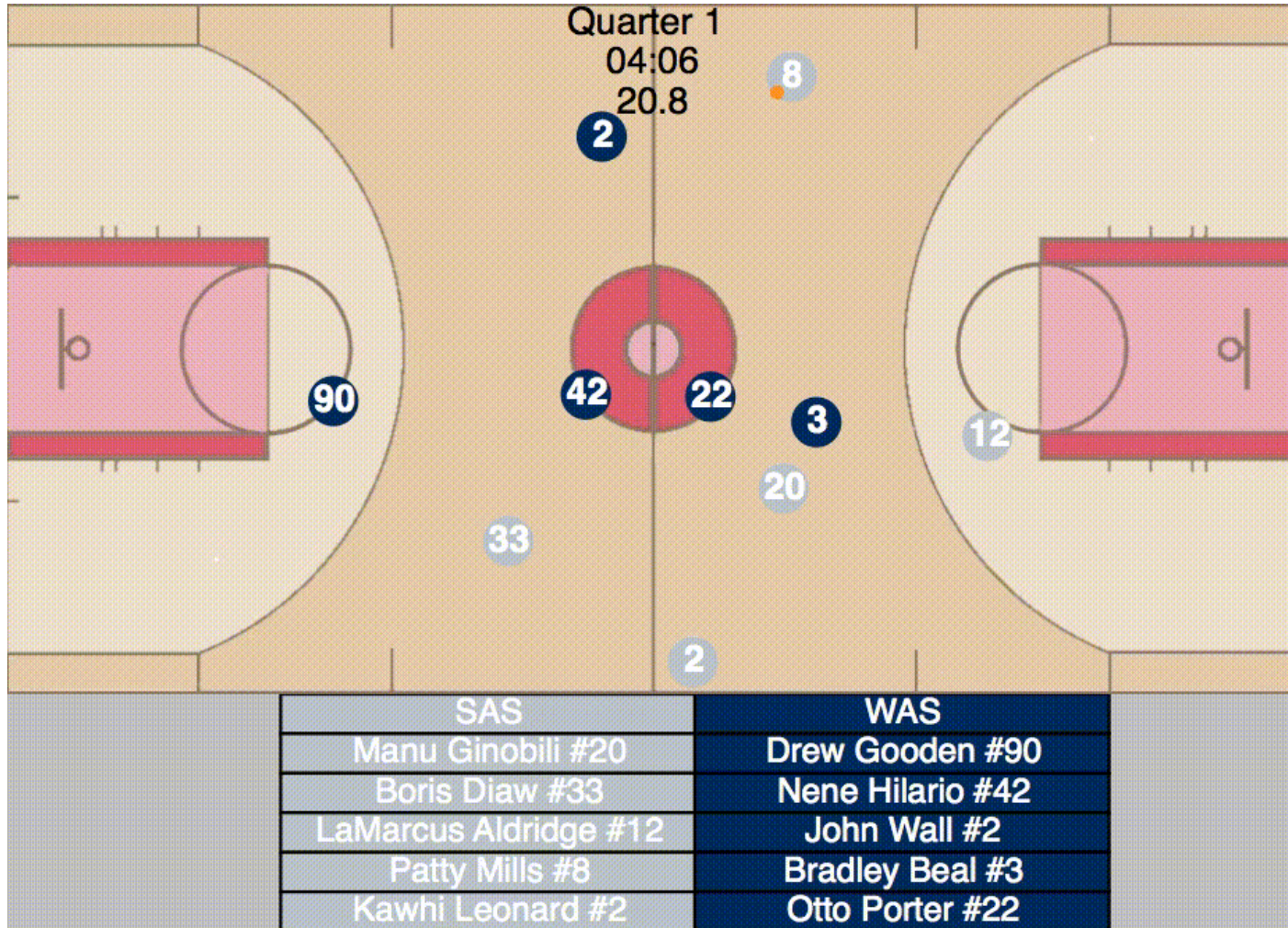


Formulation & Example 2

- Observation s = location of players & ball
- Action a = next location of player
- Training set $D = \{\tau = [(s, a)]\}$ from an expert policy π^*
- **Goal:** learn a good policy $\pi(s) \rightarrow a$ that achieves high value

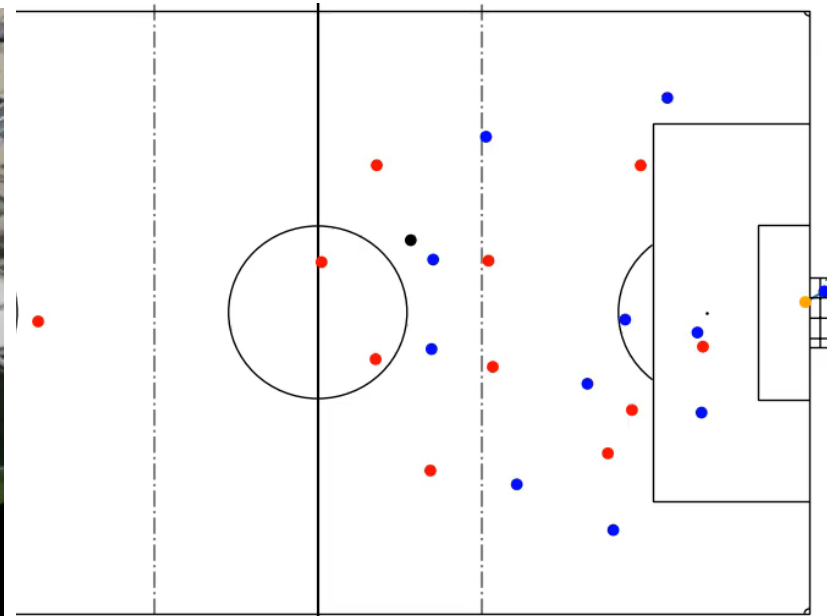
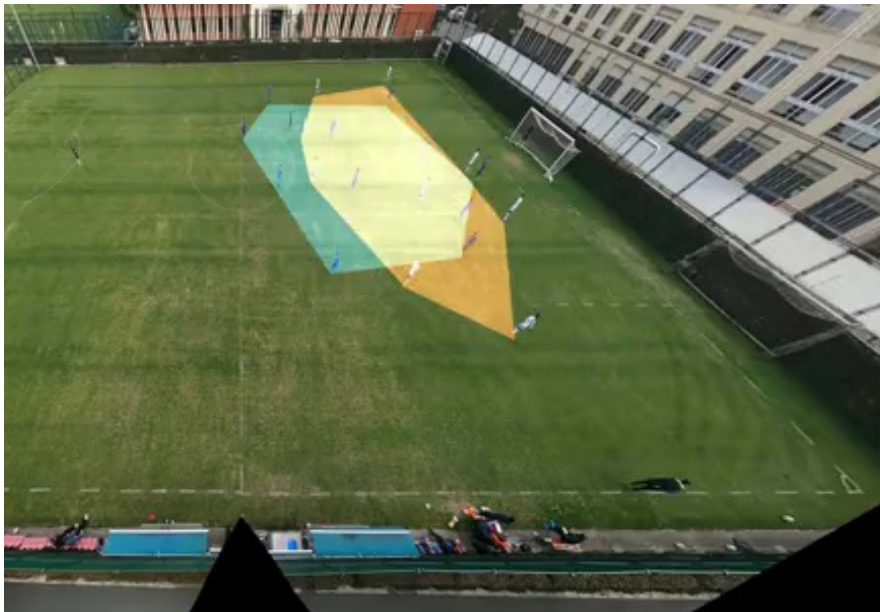


Formulation & Example 2



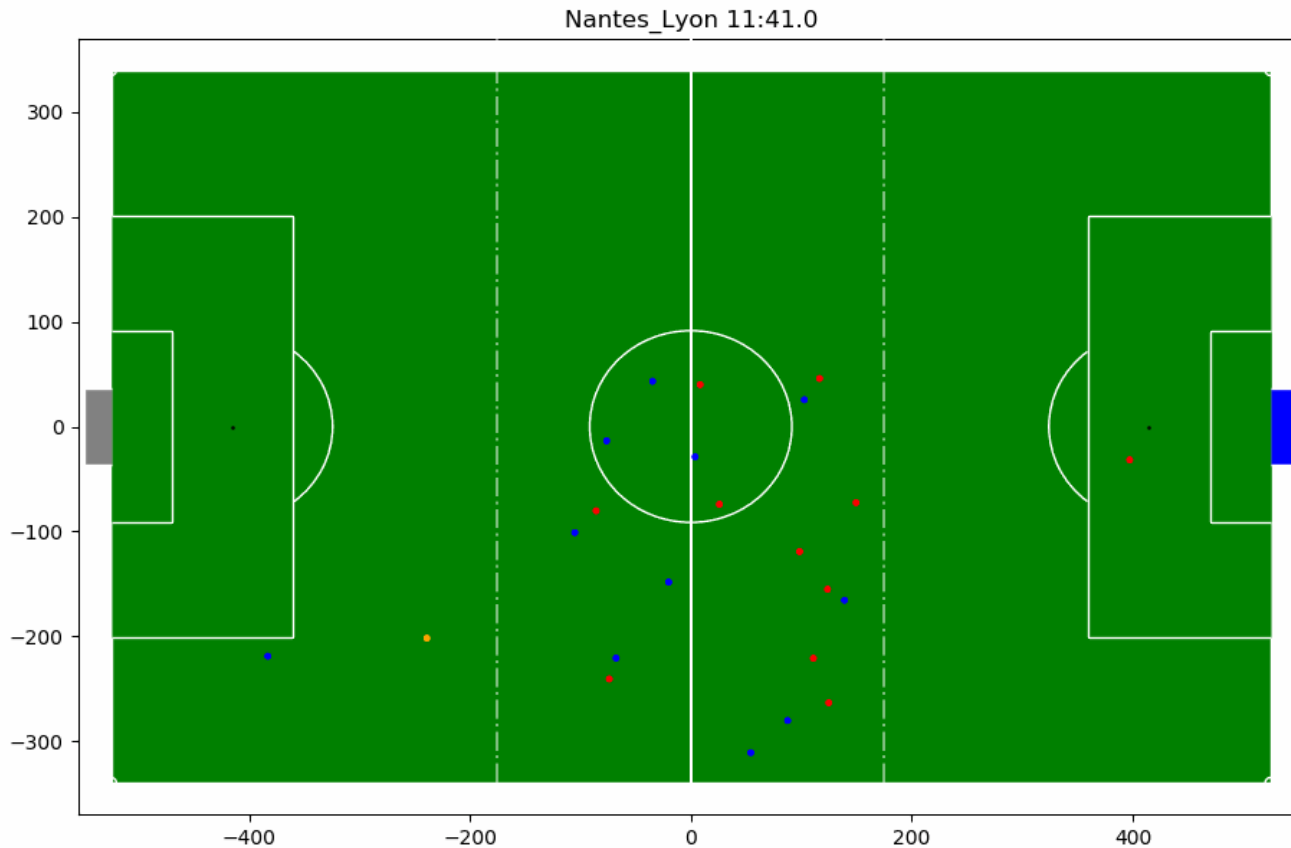
Formulation & Example 2

- Observation s = location of players & ball
- Action a = next location of player
- Training set $D = \{\tau = [(s, a)]\}$ from an expert policy π^*
- **Goal:** learn a good policy $\pi(s) \rightarrow a$ that achieves high value



Formulation & Example 2

- Nantes vs. Lyon. Red/Blue are real trajectory. Yellow is the generated



Overview

- Introduction to imitation learning
- Core methods of imitation learning
- Advanced works on imitation learning
- Connection between imitation learning and GANs

Core Methods

- Behavior Cloning (BC)
 - Learn a direct mapping from states/contexts to trajectories/actions without recovering the reward function
- Inverse Reinforcement Learning (IRL)
 - Finds a reward function which makes expert trajectories better than others
- Generative Adversarial Imitation Learning (GAIL)
 - Apply GAN under the structure of IRL to make close the two **occupancy measures** between the expert and the agent

General Imitation Learning

- Objective

$$\pi^* = \arg \min_{\pi} \mathbb{E}_{s \sim \rho_{\pi}^s} [\ell(\pi(\cdot|s), \pi_E(\cdot|s))]$$

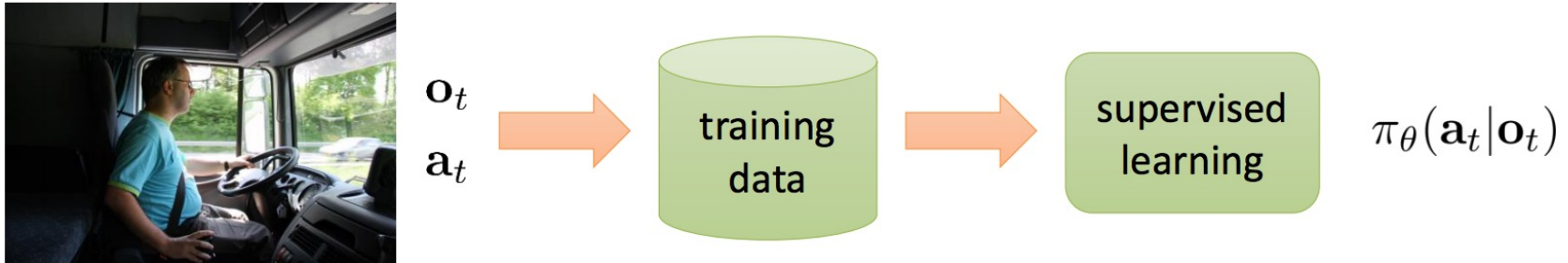
- l denotes some loss function or some distance metric
- Distribution of s depends on rollout from π
 - $P(s|\pi) \rightarrow \rho_{\pi}(s)$: distribution of states sampled by a policy

$$\rho_{\pi}(s) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)$$

- Problem

- Cannot get access to the expert during sampling!

Behavioral Cloning



- Learning objective of BC

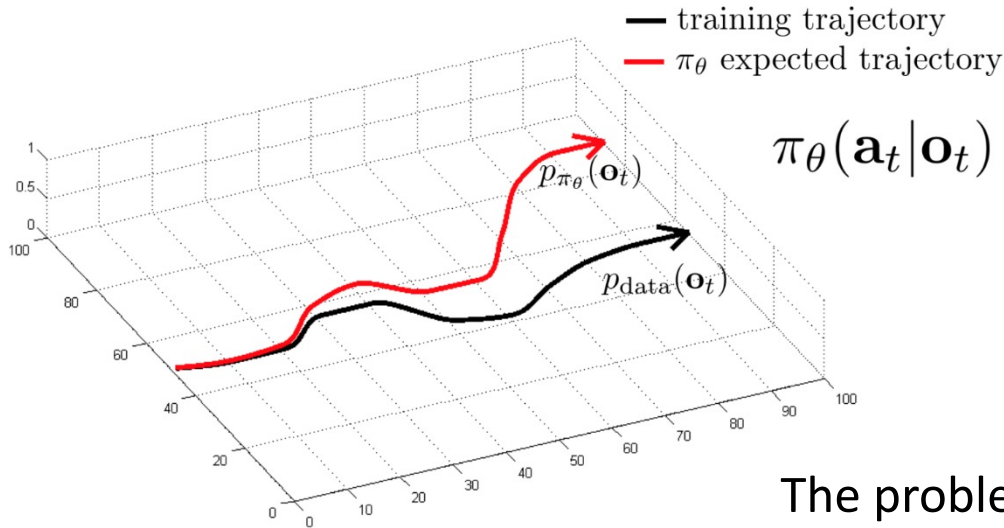
$$\hat{\pi}^* = \arg \min_{\pi} \mathbb{E}_{s \sim \rho_{\pi_E}^s} [\ell(\pi_E(\cdot | s), \pi(\cdot | s))]$$

- Compared with the original objective

$$\pi^* = \arg \min_{\pi} \mathbb{E}_{s \sim \rho_{\pi}^s} [\ell(\pi(\cdot | s), \pi_E(\cdot | s))]$$

- Distribution provided exogenously
- Essentially a Maximum Likelihood Estimation (MLE) on single step

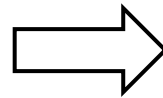
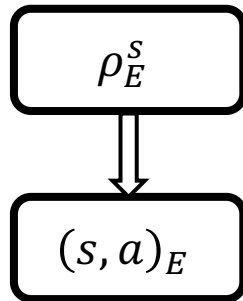
Limitations of Behavioral Cloning



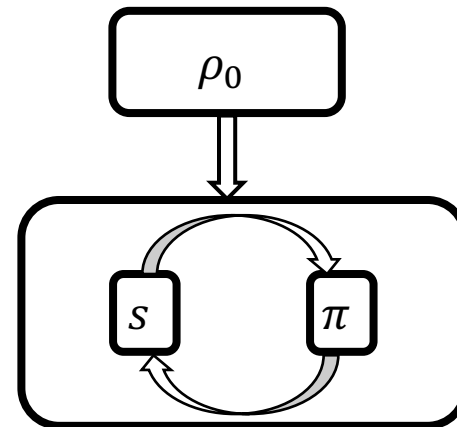
Distributional Shift

The problem is like a common problem in supervised learning, but more serious

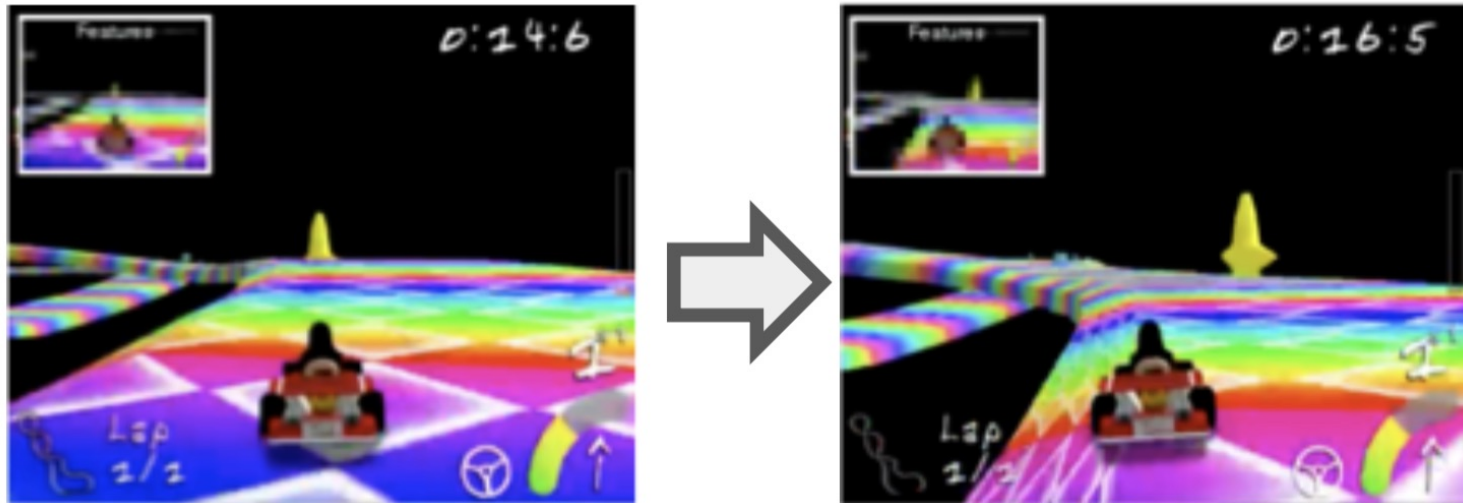
IID Assumption
(Supervised Learning)



Reality



Limitations of Behavioral Cloning



When π_θ makes a mistake, i.e., starts to diverge from the expert

- New state sampled not from ρ_E^S !
- Worst case is catastrophic!

Limitations of Behavioral Cloning



Expert Trajectories
(Training Distribution)



Behavioral Cloning
Makes mistakes, enters new states
Cannot recover from new states

Imitation Learning vs. Supervised Learning

- The solution may have **important structural properties** including constraints (for example, robot joint limits), dynamic smoothness and stability, or leading to a coherent, multi-step plan
- The interaction between the learner's decisions and its own input distribution (an **on-policy versus off-policy** distinction)
- Along side of the policy similarity, IL further cares about the **policy performance**

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

When to use BC?

- Advantages

- Simple
- Efficient

- When to use

- 1-step deviations not too bad
- Learning reactive behaviors
- Expert trajectories “cover” state space

- Disadvantages

- State distribution mismatch between training and test
- No long-term planning

- When not to use

- 1-step deviations can lead to catastrophic error
- Optimizing long-term objective

BC with Dataset Aggregation


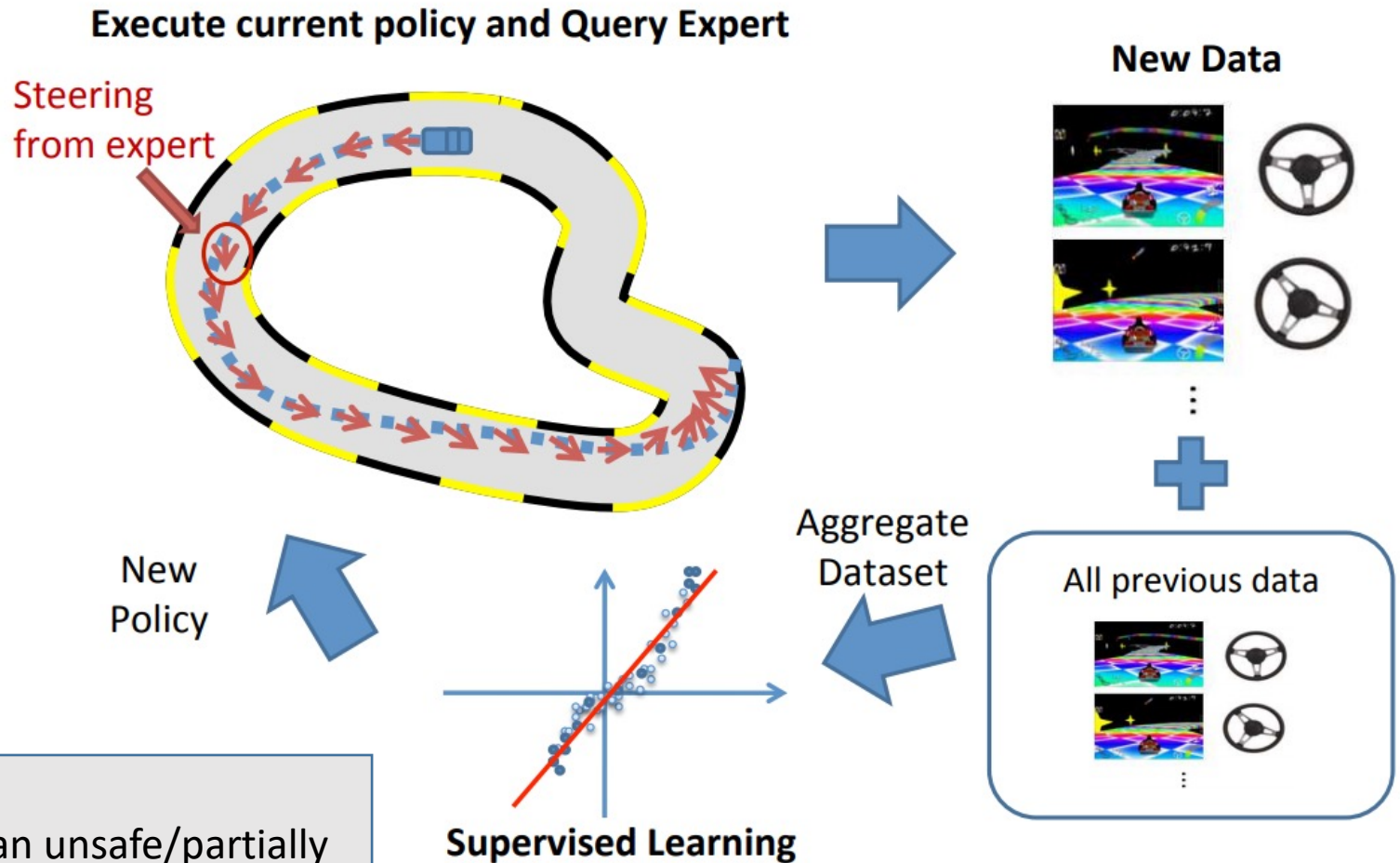
- Samples from a stable trajectory distribution
 - Learning from a stabilizing controller (with noise)
 - Add more **on-policy** data
 - e.g. DAgger
 - DAgger: Dataset Aggregation
 - train $\pi_\theta(a_t, o_t)$ from a human data
 $\mathcal{D} = \{o_1, a_1, \dots, o_N, a_N\}$.
 - run $\pi_\theta(a_t|o_t)$ to get dataset $\mathcal{D}_\pi = \{o_1, a_1, \dots, o_N, a_N\}$.
 - Ask human to label states in \mathcal{D}_π with action a_t .
 - Aggregate $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$ Human-in-the-loop IL
- 

Illustration of DAgger



Problems:

- execute an unsafe/partially trained policy
- repeatedly query the expert

Inverse Reinforcement Learning

“Forward” RL

- Given:
 - State $s \in \mathcal{S}, a \in \mathcal{A}$
 - (sometimes) transitions $p(s'|s, a)$
 - Reward function $r(s, a)$
- Learn $\pi^*(a|s)$

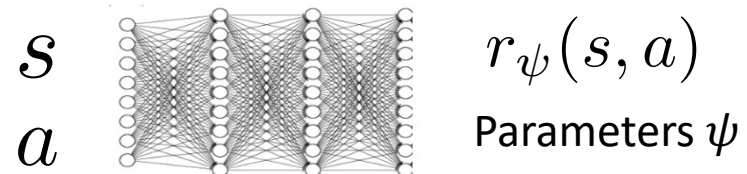
Inverse RL

- Given:
 - State $s \in \mathcal{S}, a \in \mathcal{A}$
 - (sometimes) transitions $p(s'|s, a)$
 - Samples $\{\tau_i\}$ sampled from $\pi^*(\tau_i)$
- Learn $r_\psi(s, a)$
- Then use it to learn $\pi^*(a|s)$

Linear reward function

$$r_\psi(s, a) = \sum_i \psi_i f_i = \psi^T f(s, a)$$

Neural net reward function



Inverse Reinforcement Learning

- Objective

$$\pi^* = \arg \max_{\pi} \mathbb{E}_{(s,a) \sim \rho_{\pi}} [r^*(s, a)]$$

- looks for a reward function r^* under which the expert policy is the optimal solution.

- Need to recover r^*

- Principle: **expert is optimal**, i.e., find r^* such that

$$r^* = \arg \max_r \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t r(s, a) | \pi^* \right] - \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t r(s, a) | \pi \right]$$

- Usually with a bi-level optimization

- Outer loop: find r
 - Inner loop: train policy π with r
 - Check whether $V(\pi^*) - V(\pi)$ is minimized

This is ambiguous and the solution of r may not be unique

Maximum Entropy Inverse RL

(Ziebart et al. '08)


handle ambiguity using probabilistic model of behavior

Notation:

$$\begin{array}{lll} \tau = \{s_1, a_1, \dots, s_t, a_t, \dots, s_T\} & R_\psi(\tau) = \sum_t r_\psi(s_t, a_t) & \mathcal{D} : \{\tau_i\} \sim \pi^* \\ \text{trajectory} & \text{learned reward} & \text{expert demonstrations} \end{array}$$

MaxEnt formulation:

$$p(\tau) = \frac{1}{Z} \exp(R_\psi(\tau))$$

$$Z = \int \exp(R_\psi(\tau)) d\tau$$


$$\max_{\psi} \sum_{\tau \in \mathcal{D}} \log p_{r_\psi}(\tau)$$

(energy-based model for behavior)

Maximum Entropy IRL Optimization

$$\begin{aligned}\max_{\psi} \mathcal{L}(\psi) &= \sum_{\tau \in \mathcal{D}} \log p_{r_{\psi}}(\tau) \\ &= \sum_{\tau \in \mathcal{D}} \log \frac{1}{Z} \exp(R_{\psi}(\tau)) \\ &= \sum_{\tau \in \mathcal{D}} R_{\psi}(\tau) - M \log Z \\ &= \sum_{\tau \in \mathcal{D}} R_{\psi}(\tau) - M \log \sum_{\tau} \exp(R_{\psi}(\tau))\end{aligned}$$

$$\nabla_{\psi} \mathcal{L}(\psi) = \sum_{\tau \in \mathcal{D}} \frac{dR_{\psi}(\tau)}{d\psi} - M \frac{1}{\sum_{\tau} \exp(R_{\psi}(\tau))} \sum_{\tau} \exp(R_{\psi}(\tau)) \frac{dR_{\psi}(\tau)}{d\psi}$$


Maximum Entropy IRL Optimization

$$\nabla_{\psi} \mathcal{L}(\psi) = \sum_{\tau \in \mathcal{D}} \frac{dR_{\psi}(\tau)}{d\psi} - M \underbrace{\frac{1}{\sum_{\tau} \exp(R_{\psi}(\tau))} \sum_{\tau} \exp(R_{\psi}(\tau)) \frac{dR_{\psi}(\tau)}{d\psi}}_{\sum_{\tau} p(\tau | \psi) \frac{dR_{\psi}(\tau)}{d\psi}}$$
$$\sum_{\mathbf{s}} p(\mathbf{s} | \psi) \frac{dr_{\psi}(\mathbf{s})}{d\psi}$$

Maximum Entropy Inverse RL

(Ziebart et al. '08)

handle ambiguity using probabilistic model of behavior


0. Initialize ψ , gather demonstrations \mathcal{D}
 1. Solve for optimal policy $\pi(\mathbf{a}|\mathbf{s})$ w.r.t. reward r_ψ
 2. Solve for state visitation frequencies $p(\mathbf{s}|\psi)$
 3. Compute gradient $\nabla_\psi \mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{\tau_d \in \mathcal{D}} \frac{dr_\psi}{d\psi}(\tau_d) - \sum_s p(s|\psi) \frac{dr_\psi}{d\psi}(s)$
 4. Update ψ with one gradient step using $\nabla_\psi \mathcal{L}$
- 

How can we:

(1) handle unknown dynamics? (2) avoid solving the MDP in the inner loop

$$\max_{\psi} \sum_{\tau \in \mathcal{D}} \log p_{r_{\psi}}(\tau)$$

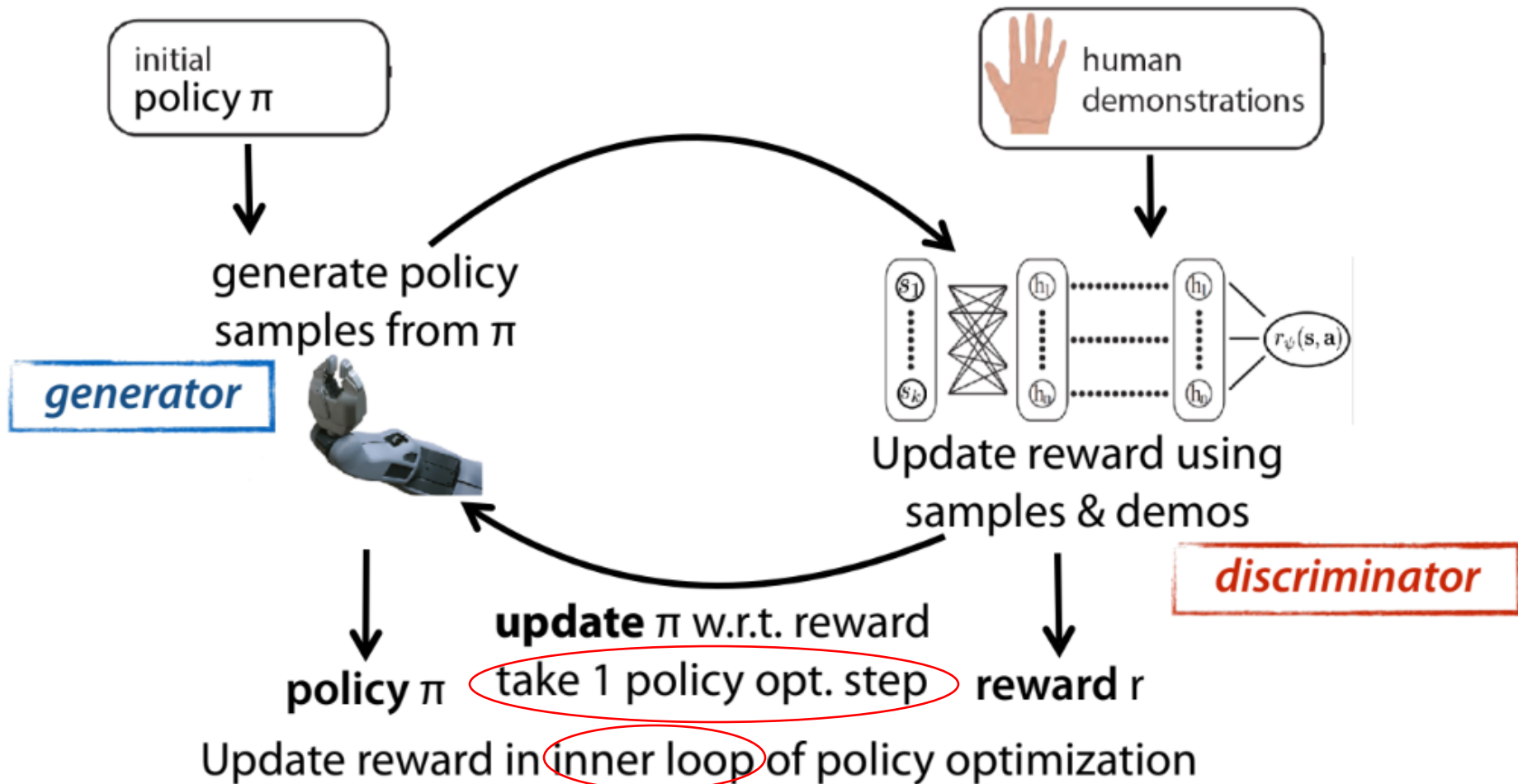
$$p(\tau) = \frac{1}{Z} \exp(R_{\psi}(\tau))$$


$$Z = \int \exp(R_{\psi}(\tau)) d\tau$$

Sample *adaptively* to estimate Z
[by constructing a policy]

guided cost learning algorithm

(Finn et al. ICML '16)



Update reward in inner loop of policy optimization

Ho & Ermon, NIPS '16

$$\mathcal{L}_{\text{Ioc}}(\theta) = \frac{1}{N} \sum_{\tau_i \in \mathcal{D}_{\text{demo}}} c_\theta(\tau_i) + \log Z$$

$$\approx \frac{1}{N} \sum_{\tau_i \in \mathcal{D}_{\text{demo}}} c_\theta(\tau_i) + \log \frac{1}{M} \sum_{\tau_j \in \mathcal{D}_{\text{samp}}} \frac{\exp(-c_\theta(\tau_j))}{q(\tau_j)}$$

Generative Adversarial Imitation Learning



- Review the **occupancy measure** of each policy interacting with the environment

$$\rho^\pi(s, a) = (1 - \gamma) \mathbb{E}_{a \sim \pi(s), s' \sim p(s, a)} \left[\sum_{t=0}^T \gamma^t p(s_t = s, a_t = a) \right]$$

- Theorem 1: for two policies π_1, π_2 and their occupancy measures $\rho^{\pi_1}, \rho^{\pi_2}$, it has $\rho^{\pi_1} = \rho^{\pi_2}$ iff $\pi_1 = \pi_2$
- Theorem 2: given an occupancy measure ρ , the only policy generating ρ is $\pi_\rho = \rho(s, a) / \sum_{a'} \rho(s, a')$

Generative Adversarial Imitation Learning

- GAIL: match the occupancy measures with GAN

$$\min_{\pi} \max_D \mathbb{E}_{\pi_E} [\log D(s, a)] + \mathbb{E}_{\pi} [\log(1 - D(s, a))] - \lambda H(\pi)$$

- GAN

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{x \sim G} [\log(1 - D(x))]$$

- Occupancy measure is analogous to the data distribution
- Discriminator D distinguishes between the distribution of data generated by G (π in GAIL) and the true data distribution (π_E in GAIL)

GAIL Algorithm

Algorithm 1 Generative adversarial imitation learning

- 1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
- 2: **for** $i = 0, 1, 2, \dots$ **do**
- 3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
- 4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

Objective of D:
$$\hat{\mathbb{E}}_{\tau_i} [\nabla_w \log(D_w(s, a))] + \hat{\mathbb{E}}_{\tau_E} [\nabla_w \log(1 - D_w(s, a))] \quad (17)$$

- 5: Take a policy step from θ_i to θ_{i+1} , using the TRPO rule with cost function $\log(D_{w_{i+1}}(s, a))$. Specifically, take a KL-constrained natural gradient step with

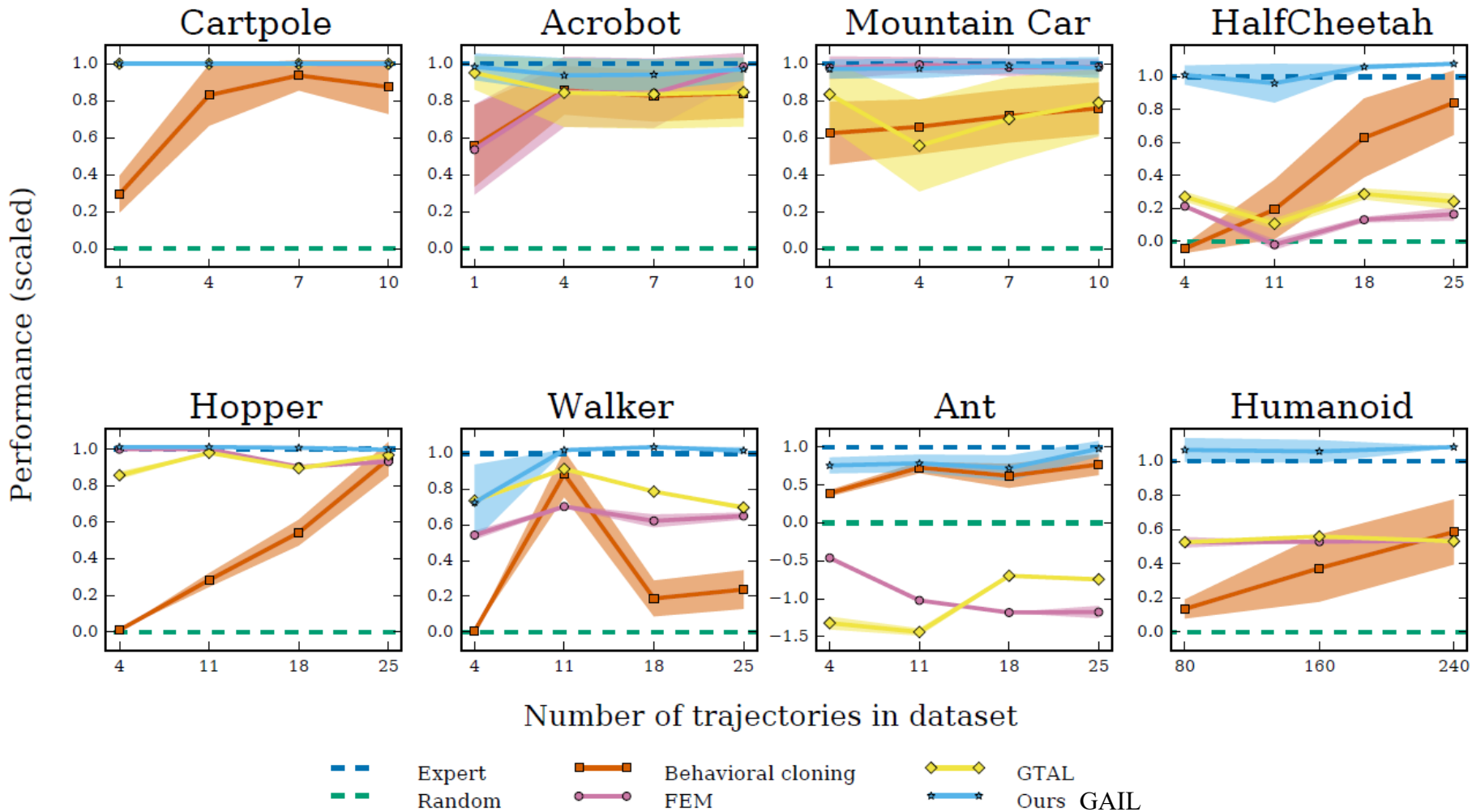
$$\hat{\mathbb{E}}_{\tau_i} [\nabla_{\theta} \log \pi_{\theta}(a|s) Q(s, a)] - \lambda \nabla_{\theta} H(\pi_{\theta}), \quad (18)$$

where $Q(\bar{s}, \bar{a}) = \hat{\mathbb{E}}_{\tau_i} [\log(D_{w_{i+1}}(s, a)) | s_0 = \bar{s}, a_0 = \bar{a}]$

- 6: **end for**
-

- GAIL alternates between
 - An Adam gradient step of w to increase the objective of D
 - A TRPO step of θ to decrease the objective of D

GAIL Experiments



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

Overview

- Introduction to imitation learning
- Core methods of imitation learning
- **Advanced works on imitation learning**
- Connection between imitation learning and GANs

Recent Works

- One-pass IL methods with fixed reward function
 - First estimate the reward then apply a forward RL procedure
 - Set the reward with specific intention
- Soft Q imitation Learning (SQL)
- Random Expert Distillation (RED)
- Disagreement-Regularized Imitation Learning (DRIL)
- Energy-Based Imitation Learning (EBIL)

Soft Q Imitation Learning (SQIL)

- Reward definition

- Expert data $r(s^*, a^*) = 1$
- New interaction data $r(s, a) = 0$

where Q is the soft Q function. The soft Q values are a function of the rewards and dynamics, given by the soft Bellman equation,

$$Q(s, a) \triangleq R(s, a) + \gamma \mathbb{E}_{s'} \left[\log \left(\sum_{a' \in \mathcal{A}} \exp(Q(s', a')) \right) \right]. \quad (3)$$

- Off-policy Learning

- A replay buffer initialized with expert data
- Then add new interaction data (50% each)
- RL algorithm: Soft Q-Learning (or Soft Actor-Critic)

$$\delta^2(\mathcal{D}, r) \triangleq \frac{1}{|\mathcal{D}|} \sum_{(s, a, s') \in \mathcal{D}} \left(Q_{\theta}(s, a) - \left(r + \gamma \log \left(\sum_{a' \in \mathcal{A}} \exp(Q_{\theta}(s', a')) \right) \right) \right)^2$$

Random Expert Distillation (RED)

- Random Network Distillation (RND) for exploration
 - Fit a randomly initialized neural network $f_\theta(s, a)$
 - Use the MSE prediction error as the intrinsic reward

$$\|f_{\hat{\theta}}(s, a) - f_\theta(s, a)\|^2$$

- Similar idea for imitation learning

$$\hat{\theta} = \arg \min_{\theta'} \|f_{\theta'}(s, a) - f_\theta(s, a)\|^2 \mid \mathcal{D}$$

$$r(\cdot) = \exp\left(-\sigma \|f_{\hat{\theta}}(s, a) - f_\theta(s, a)\|^2\right)$$

$$\hat{\theta}_k = \min_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^N (f_\theta(x_i) - f_{\theta_k}(x_i))^2,$$

for any $x \in \mathcal{X}$, we can test whether it

$$\frac{1}{K} \sum_{k=1}^K (f_{\hat{\theta}_k}(x) - f_{\theta_k}(x))^2,$$

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N (f_\theta(x_i) - I_k(x_i))^2$$

Reward is high on *familiar* state-actions of expert

- Then run an RL algorithm with the reward function

Burda, Yuri, et al. Exploration by random network distillation. 2018.

Wang R, et al. Random expert distillation: Imitation learning via expert policy support estimation. ICML 2019.

Disagreement-Regularized IL (DRIL)

- Motivation
 - Policy should move towards the expert data distribution if it is away from it
- How to train the policy?
 - Variance (uncertainty) minimization $C_U(s, a) = \text{Var}_{\pi \sim p(\cdot|\mathcal{D})}[\pi(a|s)]$
 - Minimizing variance encourages the policy to **return to regions of dense coverage by the expert, where the variance is low**
- How to estimate the variance?
 - Ensemble (bagging) of policies
 - The **disagreement** in imitation serves as the prediction **variance**
- Clipped reward definition
$$C_U^{\text{clip}}(s, a) = \begin{cases} -1 & \text{if } C_U(s, a) \leq q \\ +1 & \text{otherwise} \end{cases}$$

Energy-Based imitation Learning (EBIL)

- Motivation

- Estimate the energy (can be regarded as an **unnormalized density**) of expert's occupancy measure and use it as the surrogate reward to run an RL algorithm

$$\rho_{\pi}(s, a) = \frac{1}{Z} \exp(-E(s, a))$$

$$\pi^* = \arg \max_{\pi} \mathbb{E}_{\pi} [-E_{\pi_E}(s, a)] + H(\pi)$$

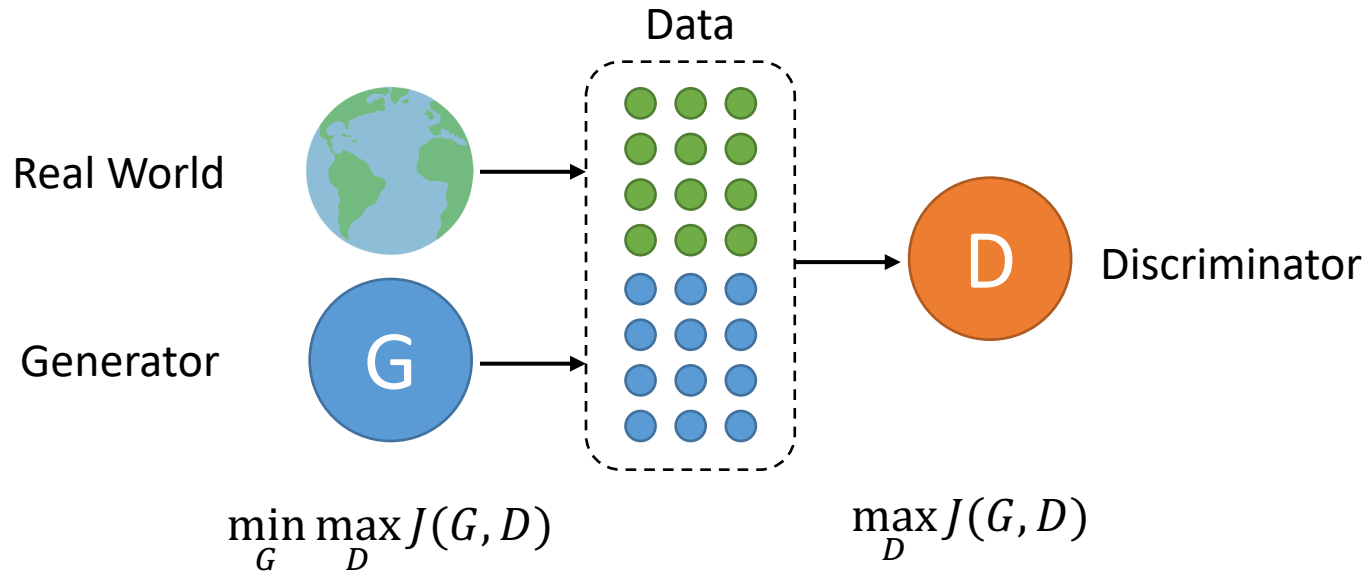
- This can be regarded as minimizing the KL-divergence:

$$\pi^* = \arg \min_{\pi} \mathbf{D}_{\text{KL}}(\rho_{\pi} \parallel \rho_{\pi_E})$$

Overview

- Introduction to imitation learning
- Core methods of imitation learning
- Advanced works on imitation learning
- Connection between imitation learning and GANs

GAN: A Minimax Game

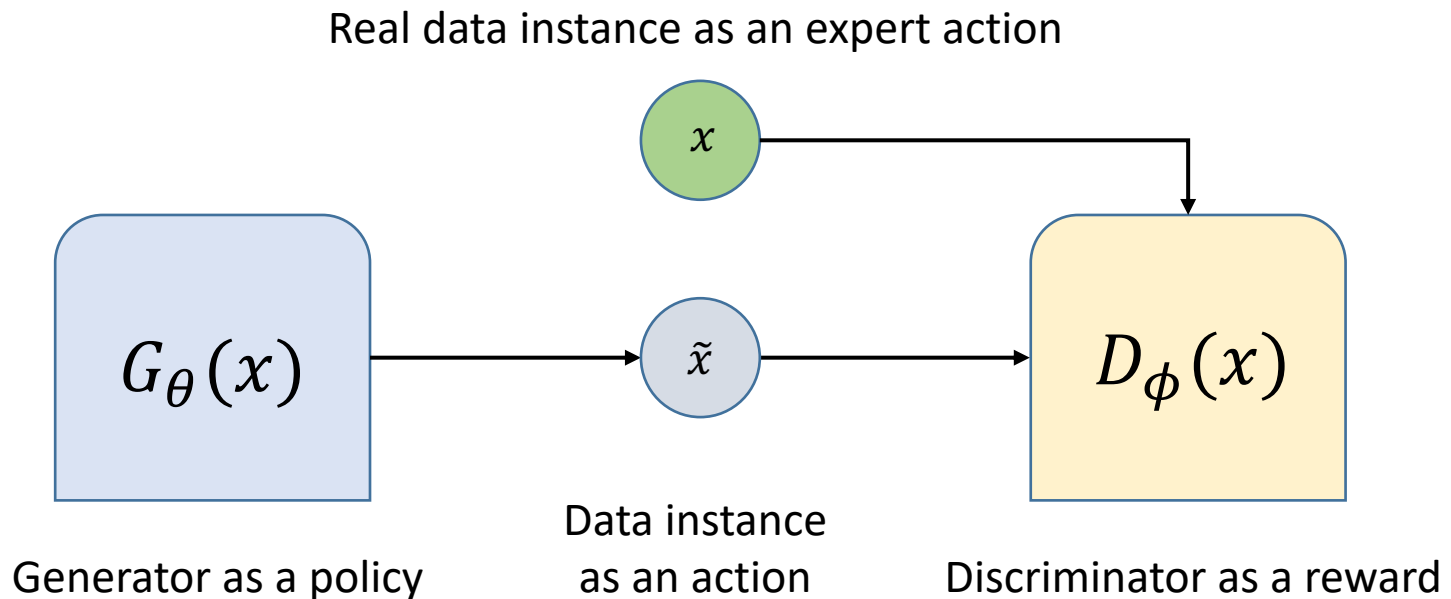


The joint objective function

$$J(G, D) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))]$$

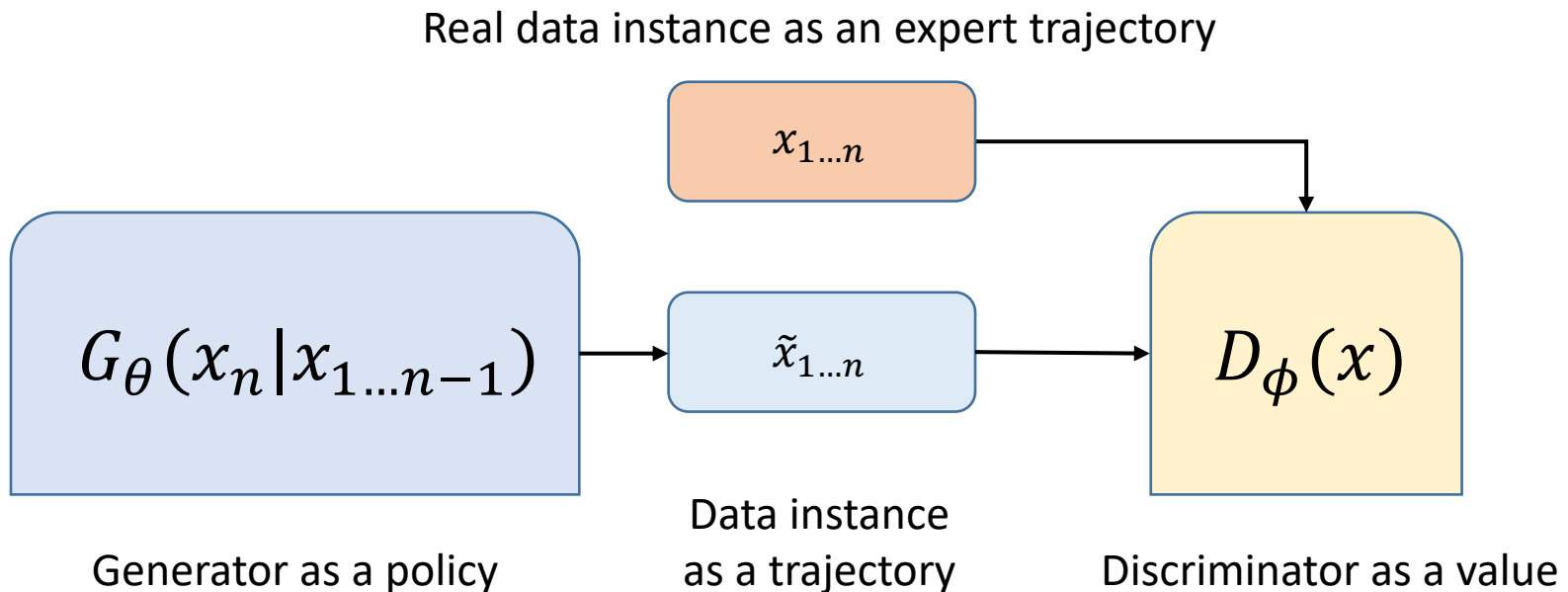
Connection between GAN and IL

- Analogy to **Imitation learning**
 - In imitation learning, a value function is learned from expert data to guide the policy optimization
 - In GAN, a discriminator is trained with real (positive) data and generated (negative) data to guide the generator optimization
- One step generation: stateless or one-step MDP



Connection between GAN and IL

- Analogy to **Imitation learning**
 - In imitation learning, a value function is learned from expert data to guide the policy optimization
 - In GAN, a discriminator is trained with real (positive) data and generated (negative) data to guide the generator optimization
- Multi-step generation: MDP



GAN and RL on Learning Rules

For continuous data/action

- Deterministic policy gradient (DPG) $\frac{\partial J(\pi_\theta)}{\partial \theta} = \mathbb{E}_{x \sim \rho^\pi} \left[\frac{\partial Q^\pi(s, a)}{\partial a} \frac{\partial \pi_\theta(s)}{\partial \theta} \Big|_{a=\pi_\theta(s)} \right]$
- GAN for continuous data $\frac{\partial J(G_\theta, D)}{\partial \theta} = \mathbb{E}_{x \sim p(z)} \left[\frac{\partial J(G_\theta, D(x))}{\partial x} \frac{\partial G_\theta(z)}{\partial \theta} \Big|_{x=G_\theta(z)} \right]$

For discrete data/action

- Stochastic policy gradient (PG) $\frac{\partial J(\theta)}{\partial \theta} = \mathbb{E}_{\pi_\theta} \left[\frac{\partial \log \pi_\theta(a|s)}{\partial \theta} Q^{\pi_\theta}(s, a) \right]$
- GAN for discrete data $\frac{\partial J(G_\theta, D)}{\partial \theta} = \mathbb{E}_{x \sim G_\theta} \left[\frac{\partial \log G_\theta(x)}{\partial \theta} D(x) \right]$

IRL & GAN & Energy-based Models

- Guided cost learning optimizes the MaxEnt IRL objective:

$$p_{\theta}(\tau) = \frac{1}{Z} \exp(-c_{\theta}(\tau)) \quad L_{cost}(\theta) = \mathbb{E}_{\tau \sim p}[-\log p_{\theta}(\tau)]$$

- GAN with this discriminator:

$$D_{\theta}(\tau) = \frac{\frac{1}{Z} \exp(-c_{\theta}(\tau))}{\frac{1}{Z} \exp(-c_{\theta}(\tau)) + \pi(\tau)}$$

$$L_D(\theta) = \mathbb{E}_{\tau \sim p}[-\log D_{\theta}(\tau)] + \mathbb{E}_{\tau \sim q}[-\log(1 - D_{\theta}(\tau))]$$

- It can be proved that $\partial_{\theta} L_{cost}(\theta) = \partial_{\theta} L_D(\theta)$

Finn C, Christiano P, Abbeel P, et al. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. arXiv 2016.

EBIL & MaxEnt IRL

- MaxEnt IRL essentially aims to recover the expert's energy with maximizing the likelihood on the whole trajs -> need to estimate the partition function Z

$$p_{\hat{r}^*}(\tau) = \frac{1}{Z} \exp(\hat{r}^*(\tau))$$
$$\mathcal{L}_{\text{IOC}}(\theta) = \frac{1}{N} \sum_{\tau_i \in \mathcal{D}_{\text{demo}}} c_{\theta}(\tau_i) + \log Z$$
$$\min_{\pi} D_{\text{KL}}(p(\tau) \| p(\tau_E))$$

- EBIL proposes that one can use **any** kind of method to estimate the expert's energy first, showing much high efficiency and flexibility

$$\rho_{\pi}(s, a) = \frac{1}{Z} \exp(-E(s, a))$$
$$\pi^* = \arg \min_{\pi} D_{\text{KL}}(\rho_{\pi} \| \rho_{\pi_E})$$

Summary of Imitation Learning

- Imitation learning is important when reward function is unavailable or hard to properly define
- Behavior cloning is straightforward and easy to implement, but suffer from distribution shift or exposure bias
- Inverse RL first recover the underlying reward of the expert from the trajectory and then perform RL to obtain the policy, but suffer from high-complexity of bi-level optimization
- GAIL aims to match the occupancy measures with GAN
- IL & GAN are highly related. You can say GAN is IL for data generation tasks

Open Questions of IL

- Problems related to algorithms
 - How to generalize skills with complex conditions?
 - How to find solutions with guarantees?
 - How to scale up with respect to the number of dimensions?
 - How to find globally optimal solutions in high dimensional spaces? How to make it tractable?
 - How to perform imitation by multiple agents?
 - How to perform incremental/active learning in IRL?
- Performance evaluation
 - How to establish benchmark problems for imitation learning?
 - What metric should be used to evaluate imitation learning methods?

Reference

1. Learning for Control from Multiple Demonstrations. Adam Coates, Pieter Abbeel, Andrew Ng, ICML 2008
2. An Application of Reinforcement Learning to Aerobatic Helicopter Flight Pieter Abbeel, Adam Coates, Morgan Quigley, Andrew Y. Ng, NIPS 2006
3. Duan Y, Andrychowicz M, Stadie B, et al. One-shot imitation learning, NIPS2018
4. Data Driven Ghosting using Deep Imitation Learning Hoang M. Le et al., SAC 2017
5. Apprenticeship learning via inverse reinforcement learning . Abbeel P, Ng A Y. ICML 2004.
6. Maximum Entropy Inverse Reinforcement Learning, Ziebart B D, Maas A L, Bagnell J A, et al., AAI 2008.
7. Guided cost learning: Deep inverse optimal control via policy optimization Finn C, Levine S, Abbeel P. ICML 2016.
8. Generative adversarial imitation learning. Ho J, Ermon S. NIPS 2016.
9. Guided cost learning: Deep inverse optimal control via policy optimization. Finn C, Levine S, Abbeel P. ICML 2016.

Reference

10. Random expert distillation: Imitation learning via expert policy support estimation. Wang R, Ciliberto C, Amadori P, et al. ICML 2019.
11. SQIL: imitation learning via regularized behavioral cloning. Reddy S, Dragan A D, Levine S. ICLR 2020.
12. Disagreement-Regularized Imitation Learning. Brantley K, Sun W, Henaff M. ICLR 2020.
13. Energy-Based Imitation Learning. Liu M, He T, Xu M, et al. arXiv preprint 2020.
14. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. Finn C, Christiano P, Abbeel P, et al. arXiv preprint 2016.