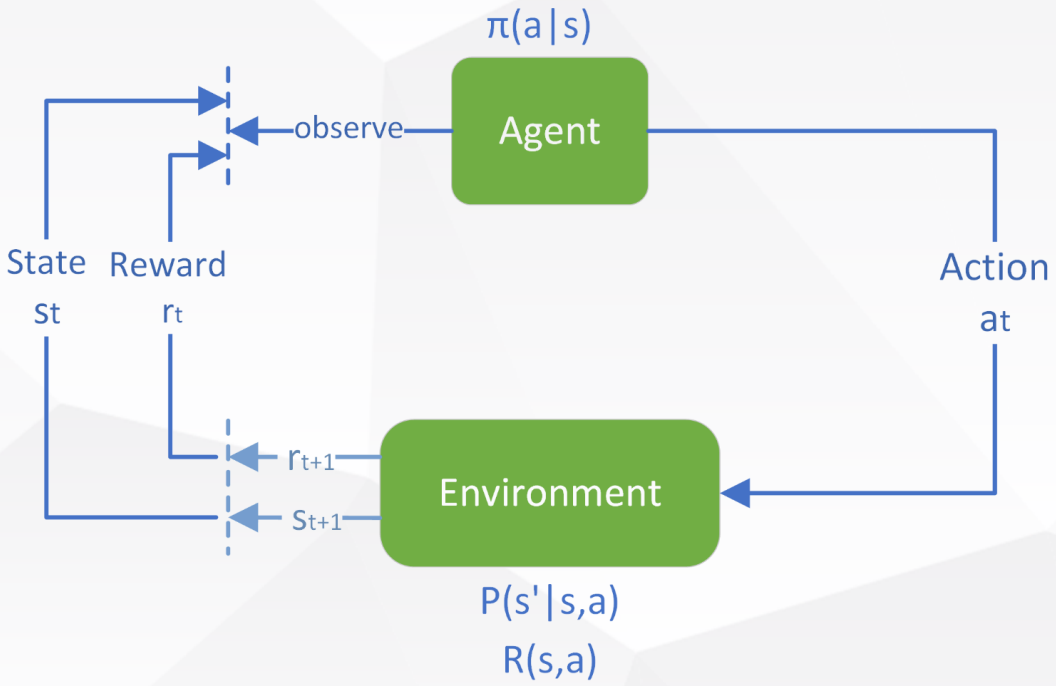
A brief introduction to imitation learning

Reinforcement Learning

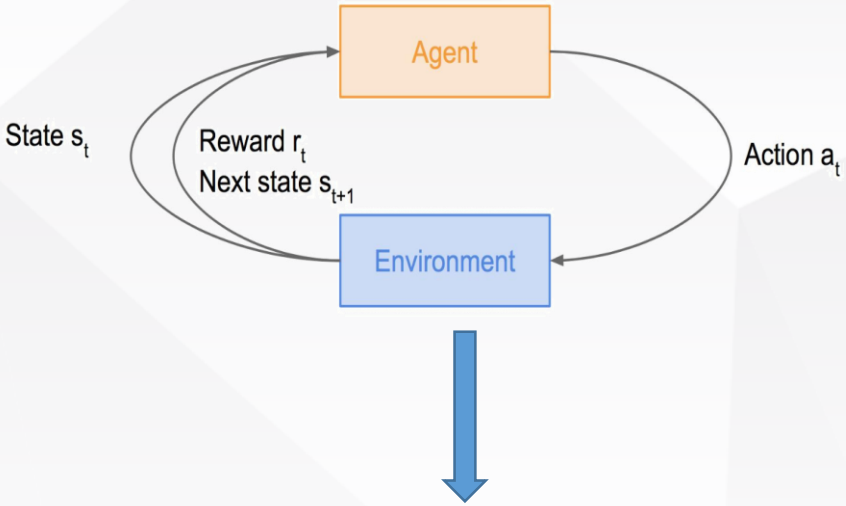
Reinforcement learning (RL):

Aim to finding optimal strategies through interaction with the environment



The interaction process can be modeled as MDP $< \mathcal{S}, \mathcal{A}, R, P, \gamma, (D) >$

Reinforcement Learning



episode/trajectory

$$s_0, a_0, r_1, s_1, a_1, r_2, s_2, \ldots s_n, a_n, r_{n+1}, \ldots$$

$$s_0, a_0, \square, s_1, a_1, \square, s_2, \ldots s_n, a_n, \square, \ldots$$

$$s_0, \square, s_1, \square\ s_2, \ldots s_n, \square, \ldots$$

horizon

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

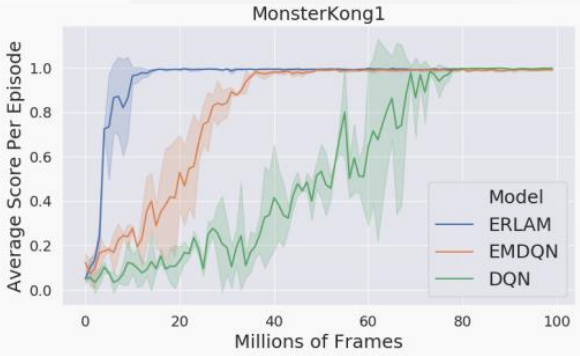
$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

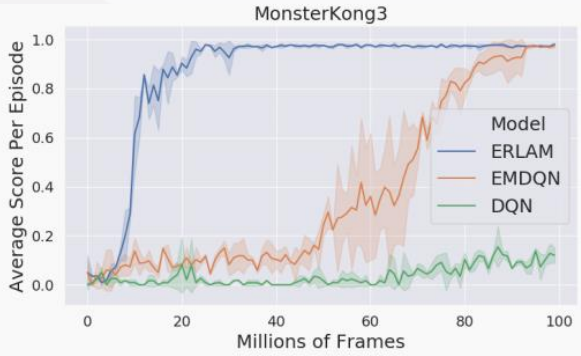
$$V_\pi = \mathbb{E}_\pi[G_t] = \mathbb{E}_\pi[\textstyle\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$$

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

$$V_\pi = \sum_{s,a} \rho(s,a) R(s,a)$$

Cons of RL

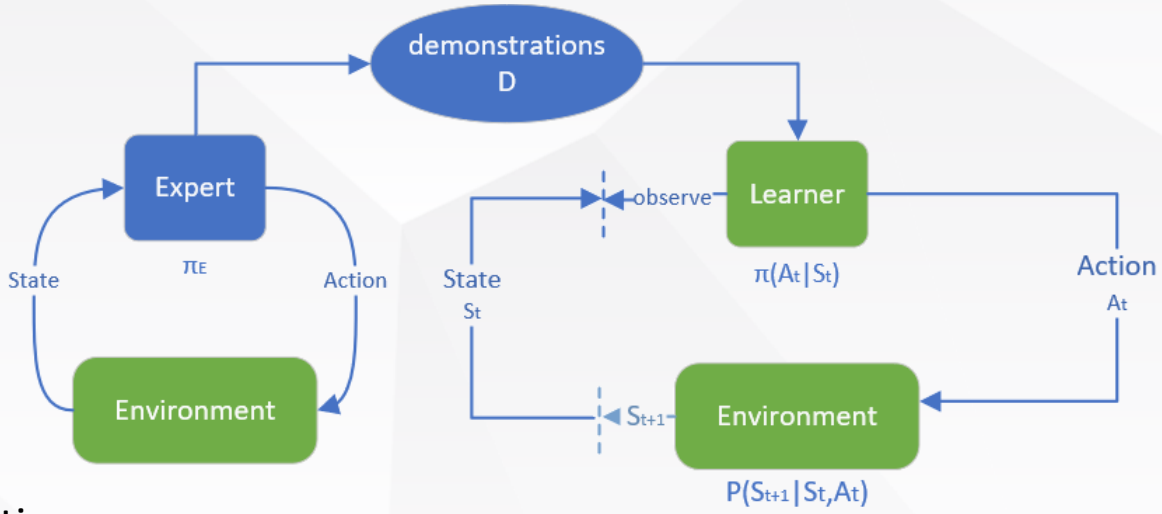




- RL requires a large amount of samples. (data inefficient)
- It's hard to design proper reward function for each particular task.

- However, it is usually esay to obtain expert-level demonstrations

Imitation Learning

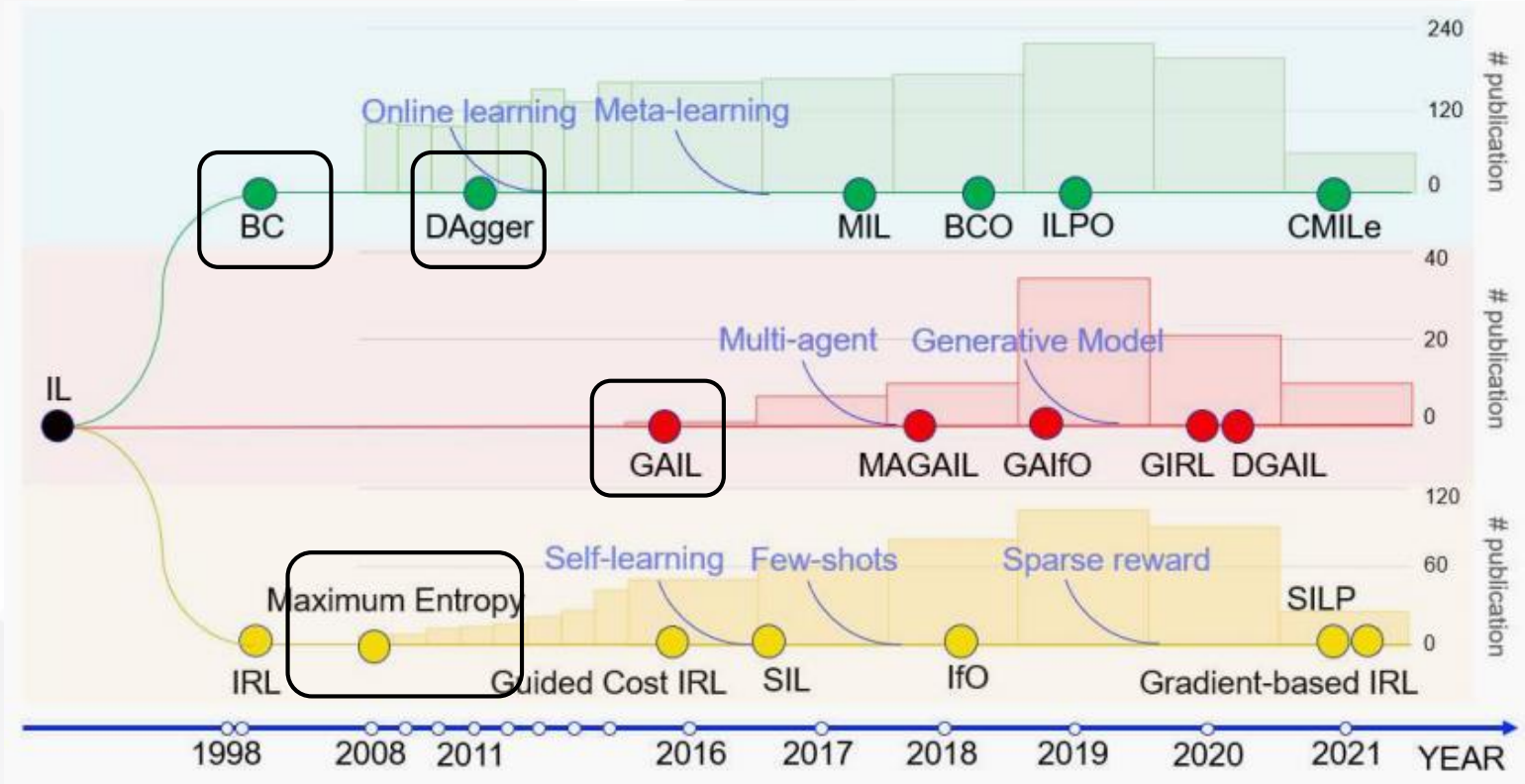


- problem setting
  - Learner is given samples of trajectories from the expert
  - Learner can interact with envrioment, but can't get reinforcement signal of any kind,which can be modeled as MDP/R
  - objective: $\max_{\pi} V^{\pi} \Leftrightarrow \min_{\pi}[V^{\pi_E} - V^{\pi}]$

- Application
  - Warmup for RL (e.g. AlphaGo)
  - When it's hard to define reward function (e.g. Self-driving cars)

Methods of IL

- Behavior Cloning (BC)
- Inverse Reinforcement Learning (IRL)
- Adversarial Structured Imitation Leaning

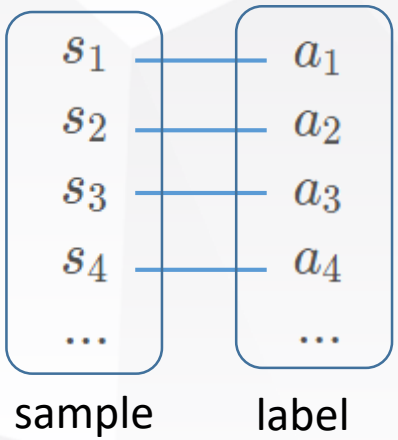


Boyuan Zheng, Sunny Verma. Imitation Learning: Progress, Taxonomies and Opportunities. arXiv preprint arXiv:2106.12177

Behavior Cloning

- mimic by matching actions distribution with supervised learning method

$$\mathcal{D} = \{\tau_1, \tau_2, ...\}, \tau = \{(s_1, a_1), (s_2, a_2), ...\}$$



treat demonstrations as i.i.d (s,a) pairs

Algorithm 1 Abstract of behavioral cloning

Collect a set of trajectories demonstrated by the expert $\mathcal{D}$
Select a policy representation $\pi_{\boldsymbol{\theta}}$
Select an objective function $\mathcal{L}$
Optimize $\mathcal{L}$ w.r.t. the policy parameter $\boldsymbol{\theta}$ using $\mathcal{D}$
return optimized policy parameters $\boldsymbol{\theta}$

Osa, Takayuki et al. (2018). “An Algorithmic Perspective on Imitation Learning”. In: Foundations and Trends in Robotics 7.1-2, pp. 1–179.

Behavior Cloning

- In the simplest case, BC learn a policy to minimize the KL divergence

$$\min_{\pi} \mathbb{E}_{s\sim d_{\pi_E}} \left[D_{KL}(\pi_E(\cdot|s), \pi(\cdot|s))\right] := \mathbb{E}_{(s,a)\sim \rho_{\pi_E}} \left[\log\left(\frac{\pi_E(a|s)}{\pi(a|s)}\right)\right]$$

- In practice, we optimize the objective with finite samples

$$\max_{\pi\in\Pi} \sum_{(s,a)\in\tau_E} \log(\pi(a|s)).$$

  - For discrete action spaces, it reduces to learn a classifier
  - For continuous action spaces, it reduces to learn a regressor

Xu, Tian & Li, Ziniu & Yu, Yang. (2020). Error Bounds of Imitating Policies and Environments. NIPS, 2020

Behavior Cloning

- some papers
  - **Alvinn**: *An autonomous land vehicle in a neural network*. 1989
  - **DAVE-2**: Bojarski M , Testa D D , Dworakowski D , et al. *End to End Learning for Self-Driving Cars*. 2016.
  - **DAgger**: Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. *A reduction of imitation learning and structured prediction to no-regret online learning*. In Proceedings of the 14th InternationalConference on Artificial Intelligence and Statistics (AISTATS'11), pages 627–635, 2011.

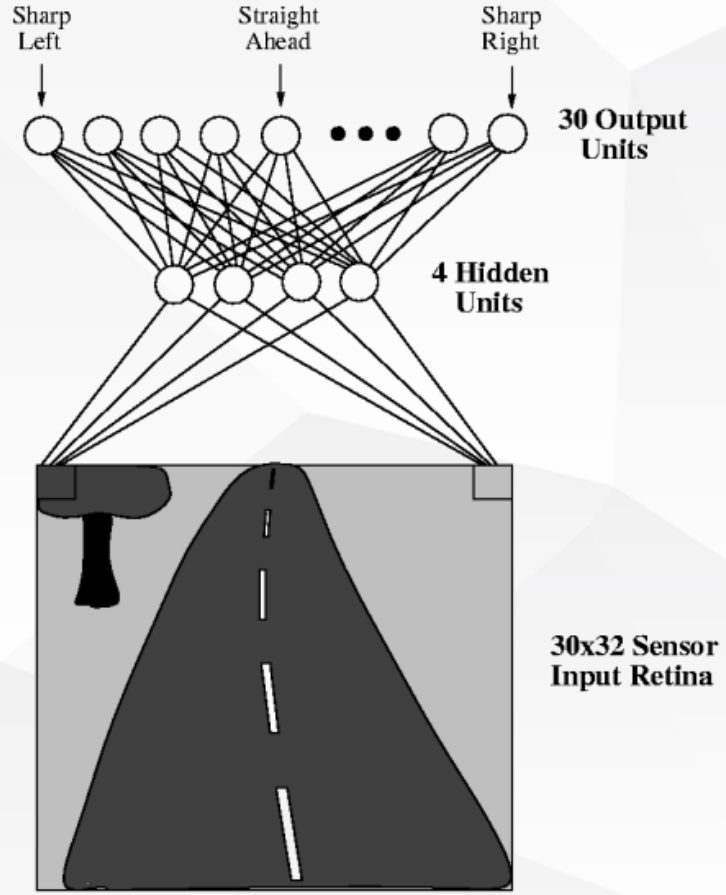
  - **DAgger by coaching：** He He, Hal Daumé, III, and Jason Eisner. *Imitation learning by coaching*. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2, NIPS?12, pages 3149?3157, USA, 2012. Curran Associates Inc.
  - **AggreVaT：** Stéphane Ross and J. Andrew Bagnell. *Reinforcement and imitation learning via interactive no-regret learning*. CoRR, abs/1406.5979, 2014.

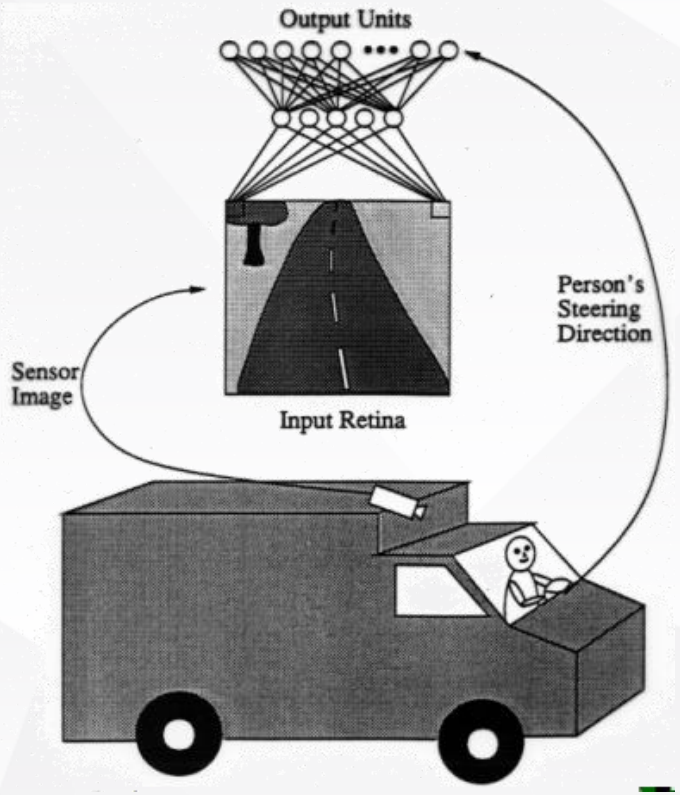
ALVINN:
An Autonomous Land Vehicle In a Neural Network

Dean A. Pomerleau
January 1989
CMU-CS-89-107

Alvinn

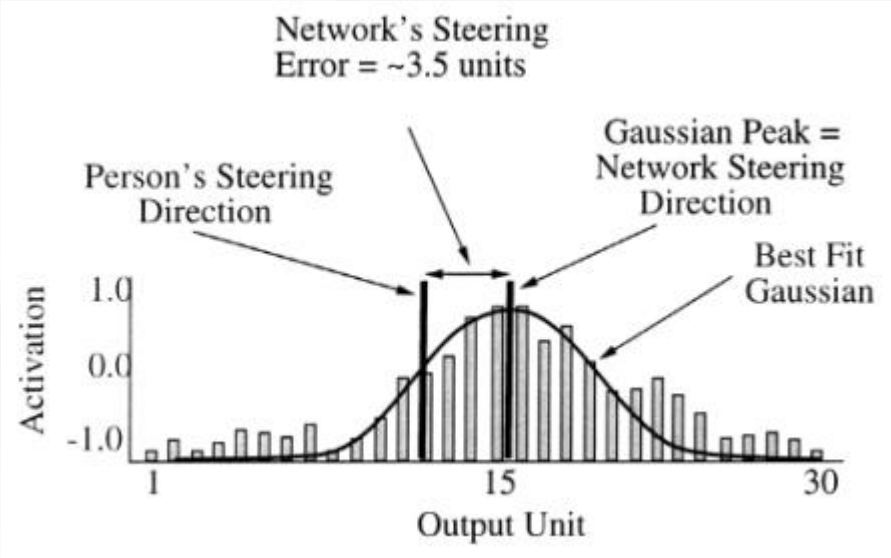
- use a 3-layer back-propagation network
- Train the network by having it observe live sensor data as a human drives the vehicle





Alvinn

- Loss
  - fit a Gaussian to the network's output vector.
  - Measure distance between Gaussian's peak and human steering direction.

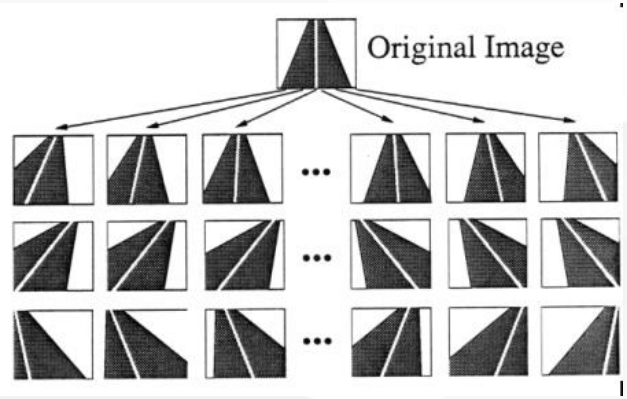


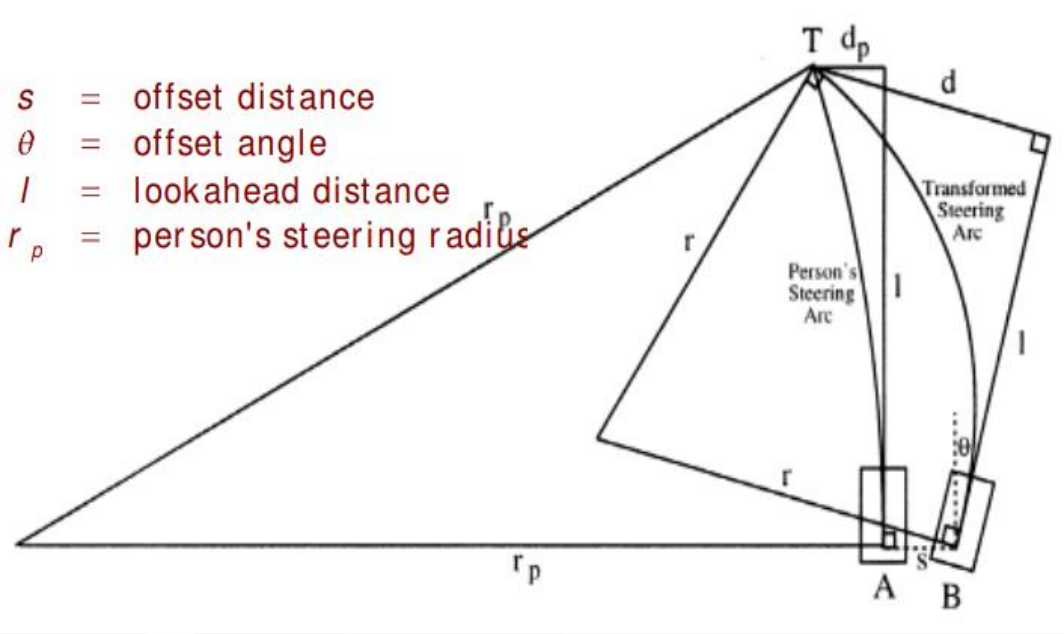
Alvinn

- Learning to Correct Steering Errors
  - If the human drives perfectly, the network never learns to make corrections when it drifts off the desired track
  - Crude solution
    - Turn learning off temporarily, and drive off course.
    - Turn learning back on, and let the network observe the human making the necessary corrections.
    - Repeat

Alvinn

- Simulating the Steering Errors
  - Let humans drive as best they can.
  - Increase training set variety by artificially shifting and rotating the video images, so that the vehicle appears at different orientations relative to the road.
  - Generate 14 random shift/rotations for each image.
  - A simple steering model is used to predict how a human driver would react to each transformation





Alvinn

- After a long right turn, the network will be biased toward turning right, since recent training data focused on right turns

- Balanced Training Images： Keep a buffer of 200 training images. Replace 15 old exemplars with new ones derived from the current camera image. Replacement strategies:
  - Replace the image with the lowest error
  - Replace the image with the closest steering direction

- Online training details
  1. Take current camera image plus 14 shifted/rotated variants, each with computed steering direction.
  2. Replace 15 old exemplars in the 200 element training exemplar buffer with these 15 new ones.
  3. Perform one epoch of backpropagation learning on the training exemplar buffer.
  4. Repeat steps 1-3 until the network's predicted steering direction reliably matches the person's steering direction.

End to End Learning for Self-Driving Cars

Mariusz Bojarski
NVIDIA Corporation
Holmdel, NJ 07735

Davide Del Testa
NVIDIA Corporation
Holmdel, NJ 07735

Daniel Dworakowski
NVIDIA Corporation
Holmdel, NJ 07735

Bernhard Firner
NVIDIA Corporation
Holmdel, NJ 07735

Beat Flepp
NVIDIA Corporation
Holmdel, NJ 07735

Prasoon Goyal
NVIDIA Corporation
Holmdel, NJ 07735

Lawrence D. Jackel
NVIDIA Corporation
Holmdel, NJ 07735

Mathew Monfort
NVIDIA Corporation
Holmdel, NJ 07735

Urs Muller
NVIDIA Corporation
Holmdel, NJ 07735

Jiakai Zhang
NVIDIA Corporation
Holmdel, NJ 07735

Xin Zhang
NVIDIA Corporation
Holmdel, NJ 07735

Jake Zhao
NVIDIA Corporation
Holmdel, NJ 07735

Karol Zieba
NVIDIA Corporation
Holmdel, NJ 07735

DAVE-2

- Overview of the DAVE-2 System

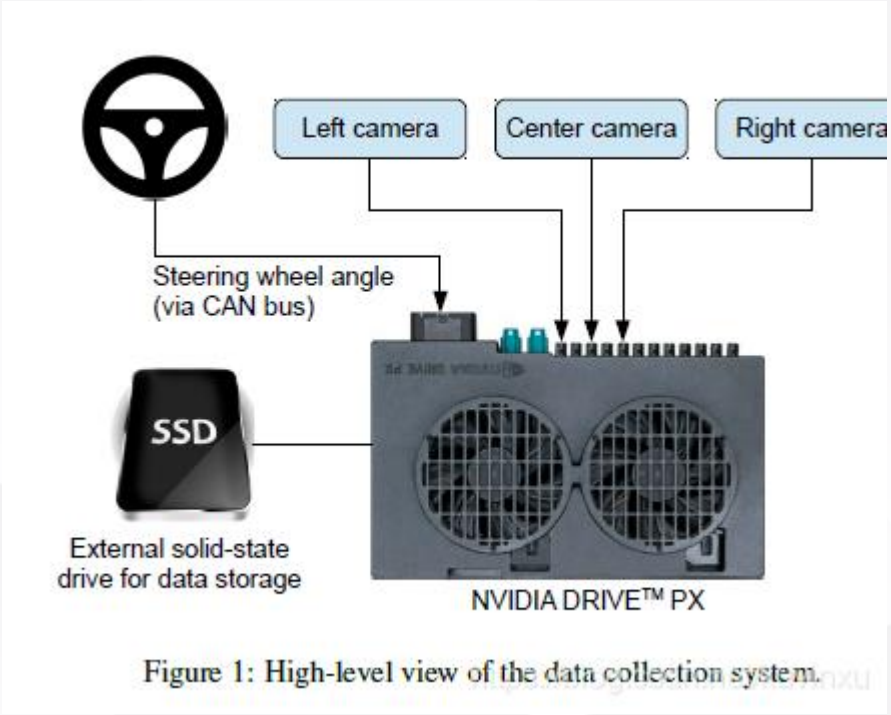
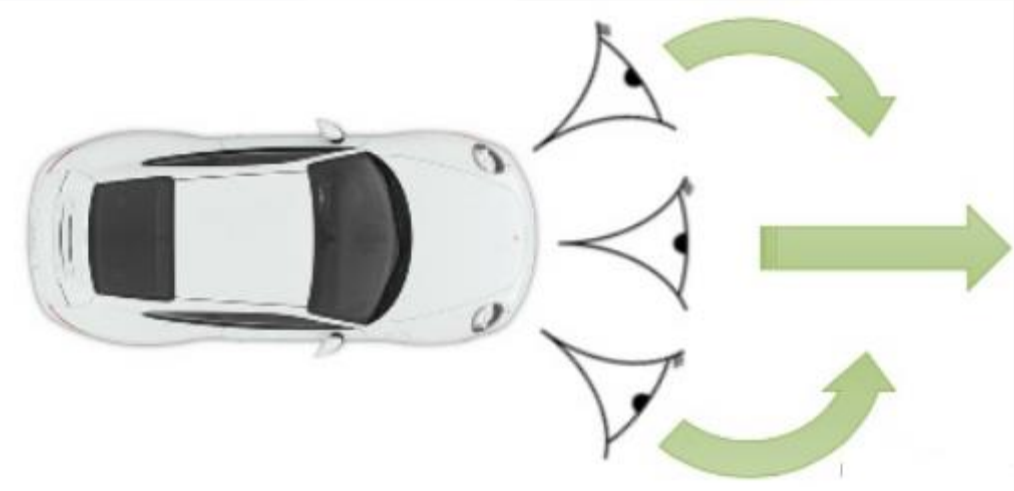


Figure 1: High-level view of the data collection system.

- We represent the steering command as 1/r to make our system independent of the car geometry, where r is the turning radius in meters.
- We use 1/r instead of r to prevent a singularity when driving straight (the turning radius for driving straight is infinity).

DAVE-2

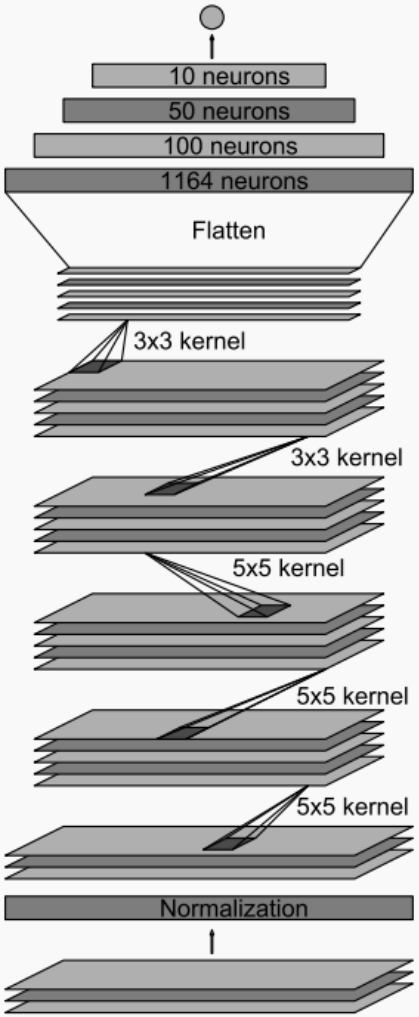
- Training data was collected by driving on a wide variety of roads and in a diverse set of lighting and weather conditions



- Training with data from only the human driver is not sufficient. The network must learn how to recover from mistakes.
  - The left and the right camera —> Images for two specific off-center shifts
  - viewpoint transformation of the image from the nearest camera —> Additional shifts between the cameras
  - The steering label for transformed images is adjusted to one that would steer the vehicle back to the desired location and orientation in two seconds.

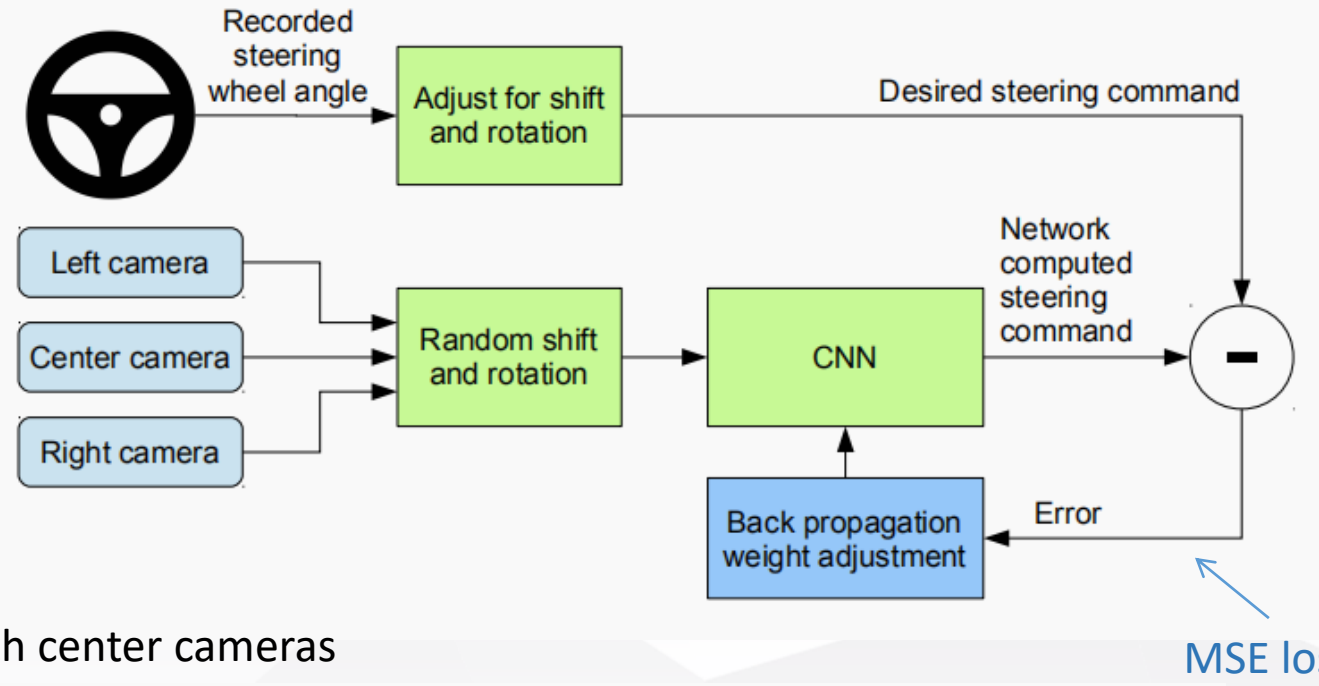
DAVE-2

- Network Architecture

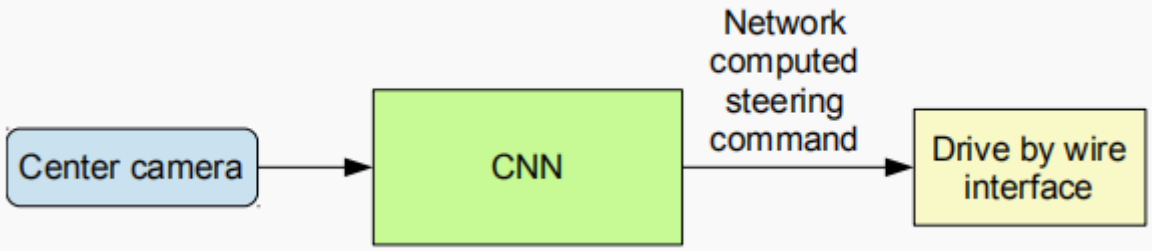


DAVE-2

- Training with three cameras

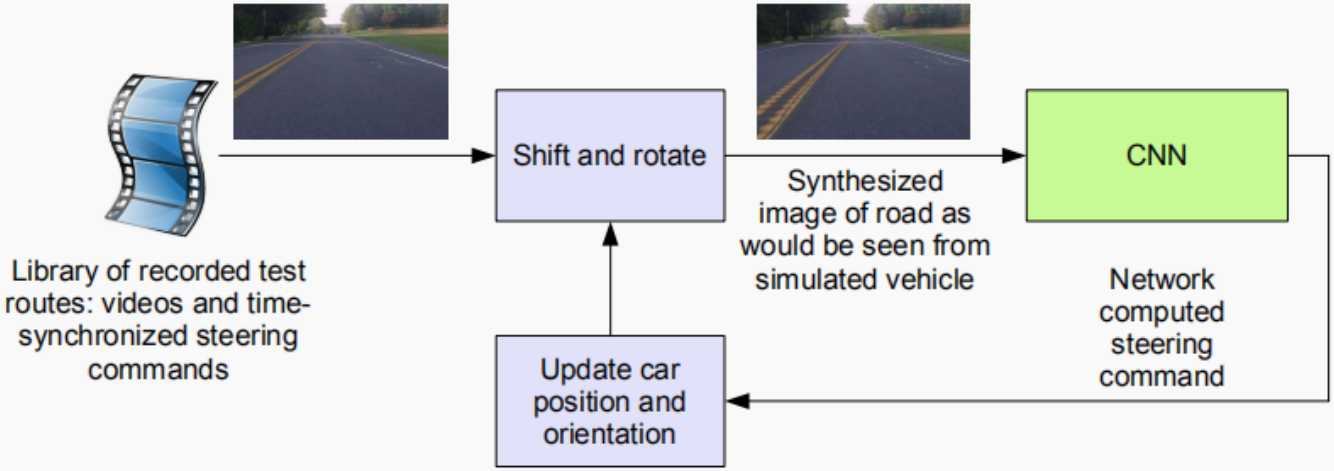


- Testing with center cameras



DAVE-2

- Simulation & on-road test



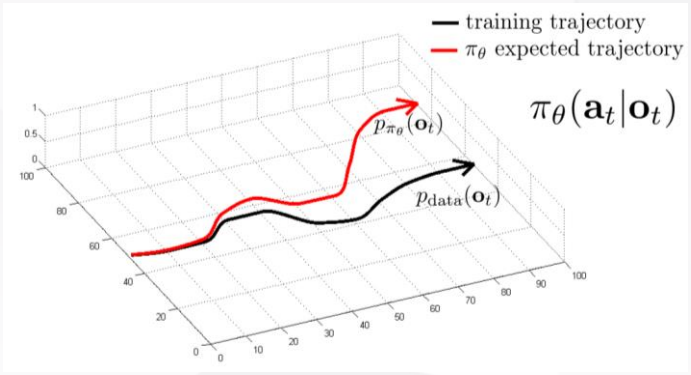
- When the off-center distance exceeds one meter, a virtual human intervention is triggered to do interventions. We estimate what percentage of the time the network could drive the car (autonomy)

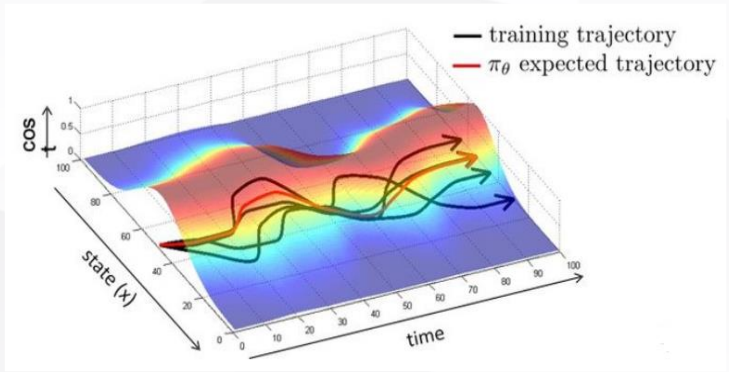
$$\text{autonomy} = (1 - \frac{(\text{number of interventions}) \cdot 6 \text{ seconds}}{\text{elapsed time [seconds]}}) \cdot 100$$

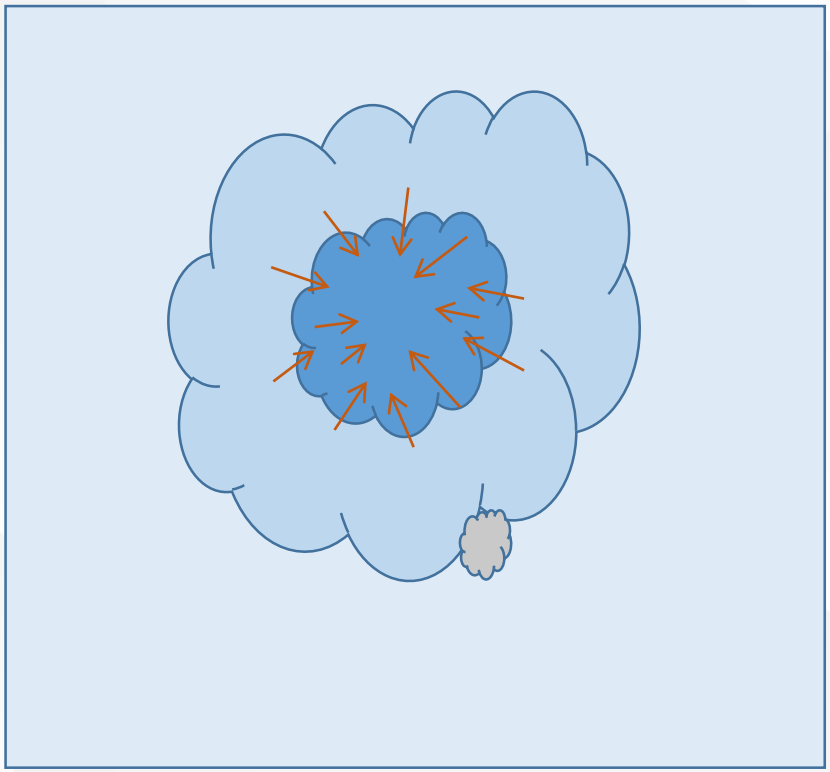
- After a trained network has demonstrated good performance in the simulator, the network is loaded on the test car and taken out for a road test. For a typical drive, we are autonomous approximately 98% of the time.

Problem of BC

- There are three common problems with supervised imitation learning
  1. copies unnecessay action (SL take all errors equally)
  2. compounding error because of fit single-timestep decisions
  3. mismatch/covriate shift (not satisfy the i.i.d assumption)







$$P(s,a) = P(a|s)P(s)$$

A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning

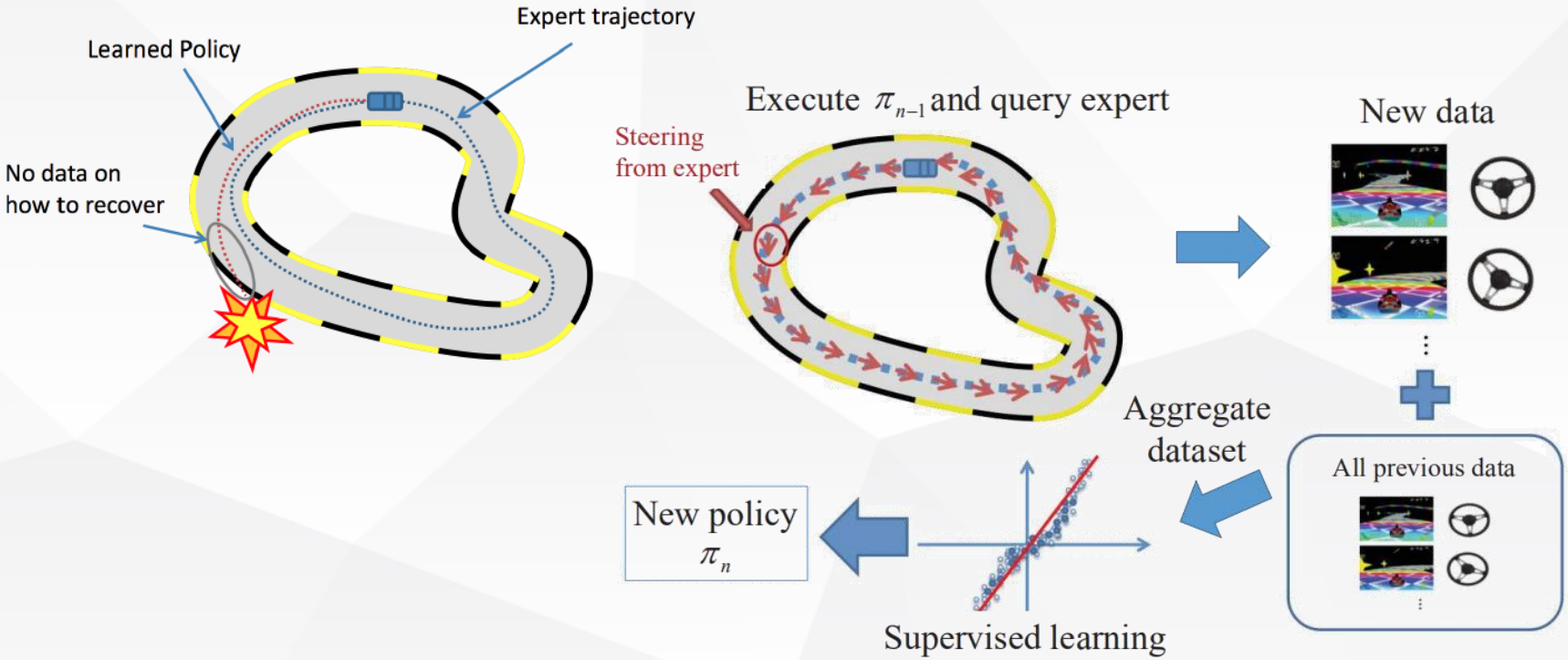
Stéphane Ross
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
stephaneross@cmu.edu

Geoffrey J. Gordon
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
ggordon@cs.cmu.edu

J. Andrew Bagnell
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
dbagnell@ri.cmu.edu

DAgger

- DAgger attempts to collect expert demonstrations under the state distribution induced by the learned policy.
- It can be seen most naturally as an on-policy to imitation learning: the expert provides the correct actions to take, but the input distribution of examples comes from the learner's own behavior.



Osa, Takayuki et al. (2018). "An Algorithmic Perspective on Imitation Learning". In: Foundations and Trends in Robotics 7.1-2, pp. 1–179.

DAgger

- The general DAGGER algorithm.

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.

offline

human or other algorithm

- We typically use $\beta_1 = 1, \beta_i = p^{i-1}(i >= 2)$ or $\beta_i = I(i = 1)$, optionally allowing the algorithm queries the expert to choose controls for a fraction of the time to better leverage the presence of the expert
- in general， $\{\beta_i\}$ be a sequence such that $\bar{\beta}_N = \frac{1}{N}\sum_{i=1}^{N}\beta_i \rightarrow 0 \ as \ N \rightarrow 0$

DAgger

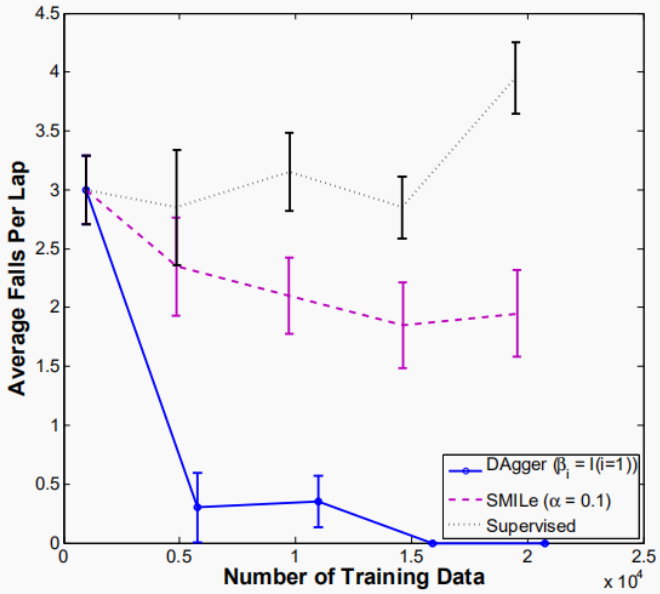
- This algorithm can be interpreted as a *Follow-The-Leader* algorithm that at each iteration we pick the best policy under all trajectories seen so far over the iterations.
- As a special case of FTL, DAgger enjoys the property of being no-regret

$$\frac{1}{N}\sum_{i=1}^{N}\ell_i(\pi_i) - \min_{\pi\in\Pi}\frac{1}{N}\sum_{i=1}^{N}\ell_i(\pi) \leq \gamma_N$$

$$\lim_{N\to\infty}\gamma_N = 0$$

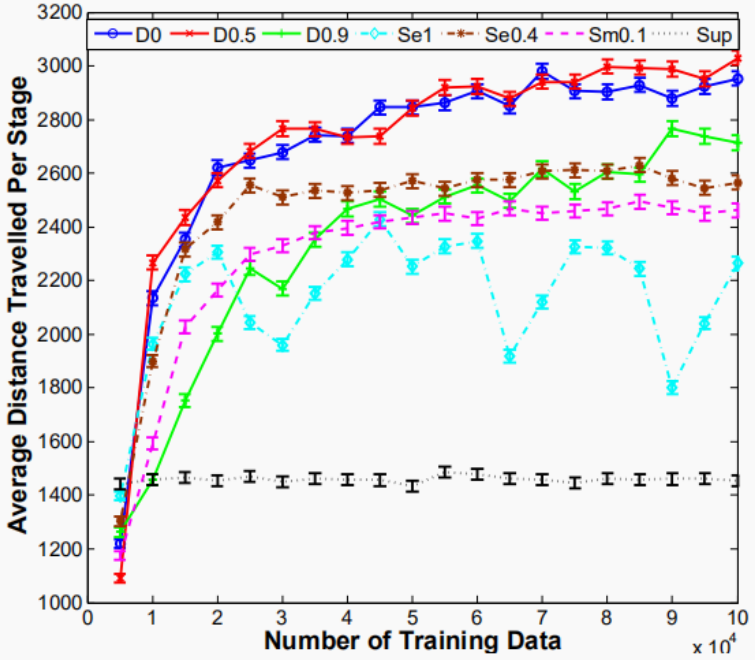
DAgger

Super Tux Kart



human expert
training laps are all very similar

Super Mario Bros



expert is a planning algorithm which hacked the game

DAgger

- $J(\pi)$ the expected total reward of trajectories starting with the initial state I
- $d_\pi = \frac{1}{T}\sum_{t=1}^{T} d_\pi^t$ the empirical mean of state distribution induced over each time step
- $C_\pi(s) = \mathbb{E}_{a\sim\pi(s)}[R(s,a)]$ the total reward in a T-step trajectory
- $l(s,\pi)$ the observed surrogate loss

- Assuming l(s, π) is the 0-1 loss (or upper bound on the 0-1 loss) implies the following performance guarantee with respect to any task reward function C bounded in [0, 1]:

Theorem 1 *Let denote* $\epsilon = \mathbb{E}_{s\sim d_{\pi^*}}[l(s,\pi,\pi^*)]$, *then there exists* $\pi \in \pi_{1:N}$ *such that* $J(\pi) \leq J(\pi^*) + T^2\epsilon$

Theorem 5 *Let* $\epsilon_N = \min_{\pi\in\Pi} \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{s\sim d_{\pi_i}}[l(s,\pi)]$ *be the true loss of the best policy, then if* $N = O(\frac{T}{\epsilon})$ *there exists* $\pi \in \pi_{1:N}$ *and* u *such as* $J(\pi) \leq J(\pi^*) + uT\epsilon_N + O(1)$

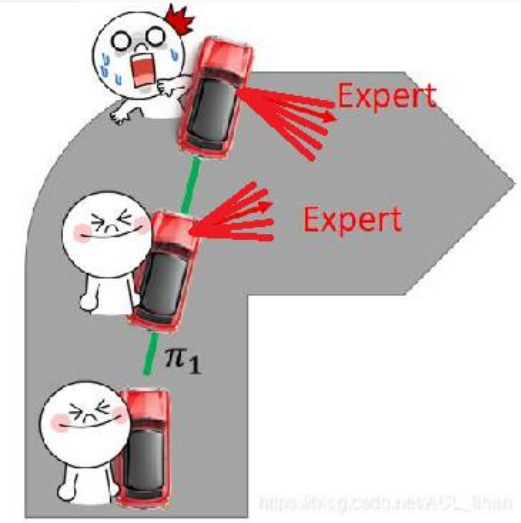
Attia A , Dayan S . Global overview of Imitation Learning. 2018. arXiv preprint arXiv:arXiv:1801.06503

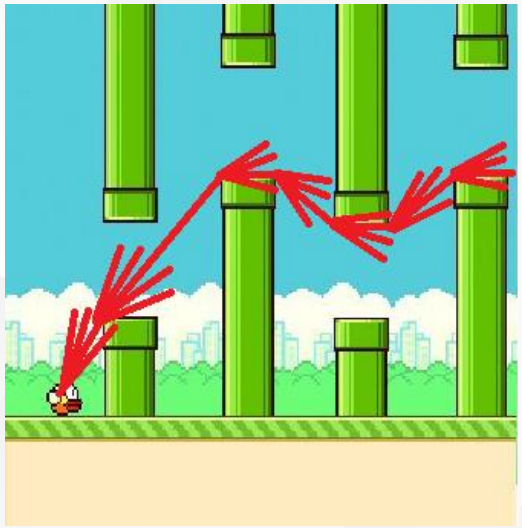
Summary of BC

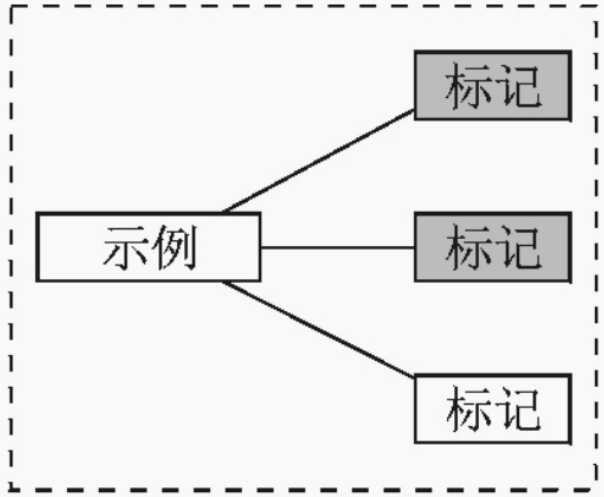
- Advantages
  1. easy to implement
  2. efficient and stable to train
- Disadvantage
  1. distribution mistmatch
  2. compounding error
- When to use
  1. 1-step deviation not too bad
  2. expert episodes cover state space
- When not to use
  1. 1-step deviation lead to catastrophic error
  2. optimze long-term objective

Summary of BC

- other problem setting ?
  - Using ambiguous demonstration data to imitate learning, thus reducing the cost of collecting demonstration
  - ...



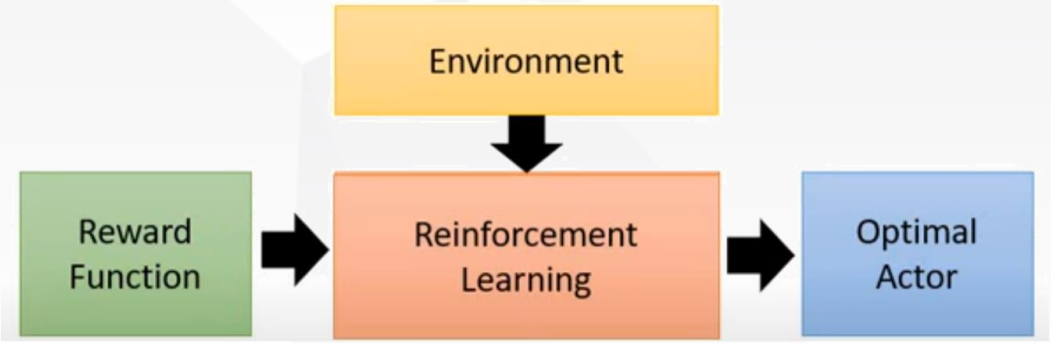




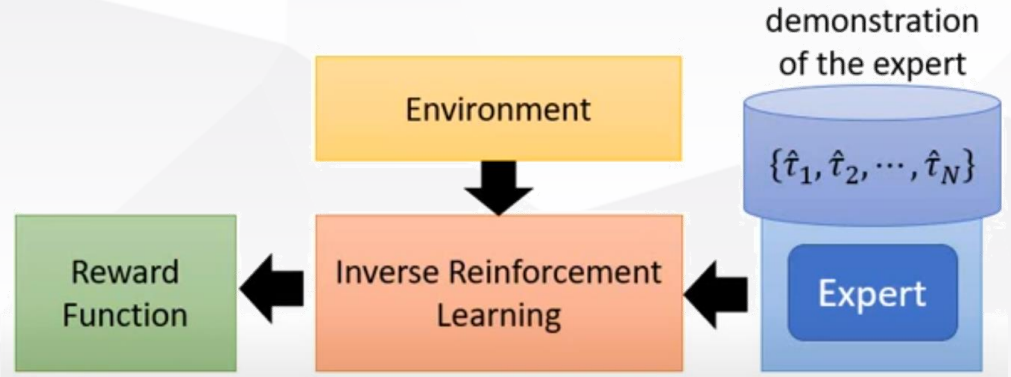
Inverse RL

- The idea is to learn the optimal reward function that can explain the expert's behavior most appropriately

RL



IRL



Inverse RL

- However, since a policy can be optimal for multiple reward functions, the problem of determining the reward function is "ill-posed"

- To obtain the unique solution, many studies have proposed additional objective functions to be optimized
  - maximize the margin
  - maximize the entropy
  - ....

Inverse RL

- some papers
  - **ALVIL:** P. Abbeel and A. Y. Ng. *Apprenticeship learning via inverse reinforcement learning*. In Proceedings of the international conference on Machine learning (ICML), 2004.
  - **MaxEnt**: B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey. *Maximum entropy inverse reinforcement learning*. In Proceedings of the Twenty-Second Con_x0002_ference on Artificial Intelligence (AAAI), pages 1433–1438, 2008.
  - **MaxCausalEnt**: B. D. Ziebart, J. A. Bagnell, and A. K. Dey. *Modeling interaction via the principle of maximum causal entropy*. in Proc. of International Conference on Machine Learning, Haifa, Israel, 2010.
  - N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich. *Maximum margin planning*. In Proceedings of the international conference on Machine learning (ICML), pages 729–736, 2006b.
  - **Guided cost learning:** Chelsea Finn, Sergey Levine, and Pieter Abbeel. 2016. *Guided cost learning: deep inverse optimal control via policy optimization*. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16). JMLR.org, 49–58.
  - Boularias, A., Kober, J., and Peters, J. *Relative entropy inverse reinforcement learning*. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2011.

Apprenticeship Learning via Inverse Reinforcement Learning

Pieter Abbeel PABBEEL@CS.STANFORD.EDU
Andrew Y. Ng ANG@CS.STANFORD.EDU
Computer Science Department, Stanford University, Stanford, CA 94305, USA

ALVIL

- finite state MDP $(\mathcal{S}, \mathcal{A}, P, \gamma, D, R)$, the reward function assume to be bounded in absolute value by 1
- features mapping function: $\phi : \mathcal{S} \to [0,1]^k$
  - feature vector: $\mathbf{f}_s = \phi(s)$
  - feature expectations: $\mu(\pi) = \mathbb{E}_{\pi, s_0 \sim D}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)] \in \mathbb{R}^k$
- assume there is some "true" reward function: $R^*(S) = \boldsymbol{w}^{*\top} \phi(S), \ \boldsymbol{w}^* \in \mathbb{R}^k$, In order to ensure that the rewards are bounded by 1, we also assume $\|w^*\|_1 \leq 1$
  - $R(s) = \boldsymbol{w}^\top \phi(s)$
  - $G_\tau = \sum_{t=0}^{\infty} \gamma^t R(s_t) = \sum_{t=0}^{\infty} \gamma^t \boldsymbol{w}^\top \phi(s_t)$
  - $V^\pi(s) = \mathbb{E}_{\pi, s_0 = s}[G_\tau]$

$$
\begin{aligned}
V^\pi &= \mathbb{E}_{\pi, s_0 \sim D}[V^\pi(s_0)] \\
&= \mathbb{E}_{\pi, s_0 \sim D}[\sum_{t=0}^{\infty} \gamma^t R(s_t)] \\
&= \boldsymbol{w}^\top \mathbb{E}_{\pi, s_0 \sim D}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)] \\
&= \boldsymbol{w}^\top \mu(\pi)
\end{aligned}
$$

- if we have found some set of policies $\pi_1, \pi_2, \ldots, \pi_n$, we can mix them to get a new policy
  - At the start of a trajectory, choose one policy with probability γ_i, then we always acts according to the selected policy until trajectory end
  - the feature expectations of the mixed policy is

$$\sum_{i=1}^{n} \lambda_i \mu(\pi_i) \ (\lambda_i \geq 0, \sum_i \lambda_i = 1)$$

- Use demonstration data $\{s_0^{(i)}, s_1^{(i)}, \ldots\}_{i=1}^{m}$ to estimate the expert's feature expectations $\mu_E = \mu(\pi_E)$

$$\hat{\mu}_E = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{\infty} \gamma^t \phi(s_t^{(i)})$$

- to find a policy whose performance is close to expert's on the unknown reward function, just need to find a policy $\tilde{\pi}$ such that $\|\mu(\tilde{\pi}) - \mu_E\|_2 \leq \epsilon$

$$
\begin{aligned}
&|E[\textstyle\sum_{t=0}^{\infty} \gamma^t R(s_t)|\pi_E] - E[\sum_{t=0}^{\infty} \gamma^t R(s_t)|\tilde{\pi}]| \\
&= |w^T \mu(\tilde{\pi}) - w^T \mu_E| \\
&\leq \|w\|_2 \|\mu(\tilde{\pi}) - \mu_E\|_2 \\
&\leq 1 \cdot \epsilon = \epsilon
\end{aligned}
$$

gap of policy value

gap of feature expection

$$|\boldsymbol{x}^\top \boldsymbol{y}| = ||\boldsymbol{x}||_2 ||\boldsymbol{y}||_2 cos(\theta) \leq ||\boldsymbol{x}||_2 ||\boldsymbol{y}||_2$$

$$||\boldsymbol{w}||_2 \leq ||\boldsymbol{w}||_1 \leq 1$$

ALVIL

1. Randomly pick some policy $\pi^{(0)}$, compute (or approximate via Monte Carlo) $\mu^{(0)} = \mu(\pi^{(0)})$, and set $i = 1$.
2. Compute $t^{(i)} = \max_{w:\|w\|_2 \leq 1} \min_{j \in \{0..(i-1)\}} w^T(\mu_E - \mu^{(j)})$, and let $w^{(i)}$ be the value of w that attains this maximum.
3. If $t^{(i)} \leq \epsilon$, then terminate.
4. Using the RL algorithm, compute the optimal policy $\pi^{(i)}$ for the MDP using rewards $R = (w^{(i)})^T \phi$.
5. Compute (or estimate) $\mu^{(i)} = \mu(\pi^{(i)})$.
6. Set $i = i + 1$, and go back to step 2.

$$t^{(i)} = \max_{w:||w||_2 \leq 1} \min_{j \in \{0,1,2,\ldots,(i-1)\}} \boldsymbol{w}^\top (\mu_E - \mu^{(i)})$$

- min: fix reward function as $\boldsymbol{w}^\top \phi(s)$, find a policy (by RL) to minimize the gap of feature expection
- max: fix policy, find parameter $\boldsymbol{w}^\top$ (i.e. reward function) to maximize te gap of policy value

$$\begin{aligned} \max_{t,w} \quad & t \quad \leftarrow \text{margin of policy value} \\ \text{s.t.} \quad & w^T \mu_E \geq w^T \mu^{(j)} + t, \; j = 0, \ldots, i-1 \\ & ||w||_2 \leq 1 \end{aligned}$$

- The key idea is to match the feature expectations (FEM), and then use additional objective functions (max margin here) to deal with the "ill-posed" problem

Maximum Entropy Inverse Reinforcement Learning

Brian D. Ziebart, Andrew Maas, J.Andrew Bagnell, and **Anind K. Dey**
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
bziebart@cs.cmu.edu, amaas@andrew.cmu.edu, dbagnell@ri.cmu.edu, anind@cs.cmu.edu

MaxEnt

- Objective: maximize the the entropy of trajectory distribution under the condition of feature matching

Problem Formulation

$$\begin{aligned} \arg\max_p \quad & H(p) && \text{(entropy)} \\ \text{subject to} \quad & \mathbb{E}_{\pi^E}[\phi(\tau)] = \mathbb{E}_{\pi^L}[\phi(\tau)], && \text{(feature matching)} \\ & \textstyle\sum_{\tau\in\mathcal{T}} p(\tau) = 1, \quad \forall\tau \in \mathcal{T} : p(\tau) > 0 && \text{(prob. distr.)} \end{aligned}$$

- choose the distribution that does not exhibit any additional preferences beyond matching feature expectations

MaxEnt

- for Deterministic MDP

$$P(\tau|\theta) = \frac{1}{Z(\theta)} exp(\theta^\top \mathbf{f}_\tau) = \frac{1}{Z(\theta)} exp(\theta^\top \sum_{s_j \in \tau} \mathbf{f}_{s_j})$$

- for Non-Deterministic MDP, paths are determined by the action choices of the agent and the random outcomes of the MDP

$$P(\tau|\theta, T) = \sum_{o \in O} P_T(o) \frac{exp(\theta^\top \mathbf{f}_\tau)}{Z(\theta, o)} I_{\tau \in o}$$

$$\approx \frac{1}{Z(\theta, T)} exp(\theta^\top \mathbf{f}_\tau) \prod_{s_{t+1}, a_t, s_t \in \tau} P_T(s_{t+1}|a_t, s_t)$$

transition distribution

deterministic transition

- Stochastic Policy $P(a|\theta, T) \propto \sum_{\tau: a \in \tau_{t=1}} P(\tau|\theta, T)$

MaxEnt

- Maximizing the entropy of the distribution over paths subject to the feature constraints from observed data implies that we maximize the likelihood of the observed data under the maximum entropy (exponential family) distribution derived above

$$\theta^* = arg\max_\theta L(\theta) = arg\max_\theta \sum_{examples\ \tilde{\tau}} logP(\tilde{\tau}|\theta, T)$$

$$\nabla L(\theta) = \tilde{\mathbf{f}} - \sum_\tau P(\tau|\theta, T)\mathbf{f}_\tau = \tilde{\mathbf{f}} - \sum_{s_i} D_{s_i}\mathbf{f}_{s_i}$$

state visitation frequencies

MaxCausalEnt

- **MaxCausalEnt**: B. D. Ziebart, J. A. Bagnell, and A. K. Dey. *Modeling interaction via the principle of maximum causal entropy*. in Proc. of International Conference on Machine Learning, Haifa, Israel, 2010.

$$P(\mathbf{A}^T||\mathbf{S}^T) \triangleq \prod_{t=1}^{T} P(A_t|\mathbf{S}_{1:t}, \mathbf{A}_{1:t-1}).$$

$$P(\bar{\mathbf{A}}|\mathbf{S}) = \prod_{t=1}^{T} P(A_t|\mathbf{S}_{1:T}, \mathbf{A}_{1:t-1})$$

$$\begin{aligned} H(\mathbf{A}^T||\mathbf{S}^T) &\triangleq E_{\mathbf{A},\mathbf{S}}[-\log P(\mathbf{A}^T||\mathbf{S}^T)] \\ &= \sum_{t=1}^{T} H(A_t|\mathbf{S}_{1:t}, \mathbf{A}_{1:t-1}), \end{aligned}$$

$$\underset{\{P(A_t|\mathbf{S}_{1:t},\mathbf{A}_{1:t-1})\}}{\operatorname{argmax}} H(\mathbf{A}^T||\mathbf{S}^T)$$

$$\text{such that: } E_{\mathbf{S},\mathbf{A}}[\mathcal{F}(\mathbf{S},\mathbf{A})] = \tilde{E}_{\mathbf{S},\mathbf{A}}[\mathcal{F}(\mathbf{S},\mathbf{A})]$$

$$\text{and } \forall_{\mathbf{S}_{1:t},\mathbf{A}_{1:t-1}} \sum_{A_t} P(A_t|\mathbf{S}_{1:t}, \mathbf{A}_{1:t-1}) = 1,$$

$$\text{and given: } P(\mathbf{S}^T||\mathbf{A}^{T-1}).$$

Summary of IRL

- advantages:
  1. does not need interactive expert
  2. very efficient when trained (in some cases can outperform the demonstrator)
  3. has long-term planning

- disadvantages:
  1. can be difficult to train

- use when:
  1. an interactive expert is not available
  2. it might be easier to learn the reward functions than the expert‘s policy

Generative Adversarial Imitation Learning

Jonathan Ho
Stanford University
hoj@cs.stanford.edu

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

GAIL

- IRL methods aim to recover the expert's cost function and then extract a policy from that cost function with reinforcement learning. This approach is indirect and can be slow
- We propose a new general framework for directly extracting a policy from data, as if it were obtained by reinforcement learning following inverse reinforcement learning

- IRL:

$$\underset{c \in \mathcal{C}}{\text{maximize}} \left(\min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_{\pi}[c(s,a)] \right) - \mathbb{E}_{\pi_E}[c(s,a)]$$

$$\mathrm{RL}(c) = \arg\min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_{\pi}[c(s,a)]$$

GAIL

- IRL methods aim to recover the expert's cost function and then extract a policy from that cost function with reinforcement learning. This approach is indirect and can be slow
- We propose a new general framework for directly extracting a policy from data, as if it were obtained by reinforcement learning following inverse reinforcement learning

- general form of IRL:

$$\underset{c\in\mathcal{C}}{\text{maximize}} \left(\min_{\pi\in\Pi} -H(\pi) + \mathbb{E}_{\pi}[c(s,a)] \right) - \mathbb{E}_{\pi_E}[c(s,a)]$$

$$\text{RL}(c) = \arg\min_{\pi\in\Pi} -H(\pi) + \mathbb{E}_{\pi}[c(s,a)]$$

$$\mathbb{E}_{\pi}[c(s,a)] \triangleq \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t c(s_t,a_t)\right]$$

$H(\pi) \triangleq \mathbb{E}_{\pi}[-\log \pi(a|s)]$ is the γ-discounted causal entropy

GAIL

- in case of overfit, we incorporate a (closed, proper) convex cost function regularizer $\psi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \overline{\mathbb{R}}$ into our study, it must be convex as a function defined on all of $\mathbb{R}^{\mathcal{S}\times\mathcal{A}}$

$$\mathrm{IRL}_\psi(\pi_E) = \arg\max_{c\in\mathbb{R}^{\mathcal{S}\times\mathcal{A}}} -\psi(c) + \left(\min_{\pi\in\Pi} -H(\pi) + \mathbb{E}_\pi[c(s,a)]\right) - \mathbb{E}_{\pi_E}[c(s,a)]$$

$$\tilde{c} \in \mathrm{IRL}_\psi(\pi_E)$$

- one-to-one correspondence between policy set and occupancy measures set

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

$$\mathcal{D} \triangleq \{\rho_\pi : \pi \in \Pi\}$$

$$= \left\{\rho : \rho \geq 0 \quad \text{and} \quad \sum_a \rho(s,a) = p_0(s) + \gamma\sum_{s',a} P(s|s',a)\rho(s',a) \;\; \forall\, s \in \mathcal{S}\right\}$$

distribution of starting states

Proposition 3.1 (Theorem 2 of Syed et al. [29]). *If $\rho \in \mathcal{D}$, then ρ is the occupancy measure for $\pi_\rho(a|s) \triangleq \rho(s,a)/\sum_{a'}\rho(s,a')$, and π_ρ is the only policy whose occupancy measure is ρ.*

GAIL

- for a function $f : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \overline{\mathbb{R}}$, its convex conjugate is $f^*(x) = \sup_{y\in\mathbb{R}^{\mathcal{S}\times\mathcal{A}}} x^T y - f(y)$

Proposition 3.2. $\mathrm{RL}\circ\mathrm{IRL}_\psi(\pi_E) = \arg\min_{\pi\in\Pi} -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E})$

Lemma 3.1. *Let* $\bar{H}(\rho) = -\sum_{s,a}\rho(s,a)\log(\rho(s,a)/\sum_{a'}\rho(s,a'))$. *Then,* $\bar{H}$ *is strictly concave, and for all* $\pi\in\Pi$ *and* $\rho\in\mathcal{D}$, *we have* $H(\pi) = \bar{H}(\rho_\pi)$ *and* $\bar{H}(\rho) = H(\pi_\rho)$.

Proof of Lemma 3.1. First, we show strict concavity of $\bar{H}$. Let ρ and ρ' be occupancy measures, and suppose $\lambda\in[0,1]$. For all s and a, the log-sum inequality [6] implies:

$$-(\lambda\rho(s,a)+(1-\lambda)\rho'(s,a))\log\frac{\lambda\rho(s,a)+(1-\lambda)\rho'(s,a)}{\sum_{a'}(\lambda\rho(s,a')+(1-\lambda)\rho'(s,a'))} \tag{19}$$

$$= -(\underbrace{\lambda\rho(s,a)}_{a}+\underbrace{(1-\lambda)\rho'(s,a)}_{a})\log\frac{\lambda\rho(s,a)+(1-\lambda)\rho'(s,a)}{\underbrace{\lambda\sum_{a'}\rho(s,a')}_{b_1}+\underbrace{(1-\lambda)\sum_{a'}\rho'(s,a')}_{b_2}} \tag{20}$$

$$\geq -\lambda\rho(s,a)\log\frac{\lambda\rho(s,a)}{\lambda\sum_{a'}\rho(s,a')} - (1-\lambda)\rho'(s,a)\log\frac{(1-\lambda)\rho'(s,a)}{(1-\lambda)\sum_{a'}\rho'(s,a')} \tag{21}$$

$$= \lambda\left(-\rho(s,a)\log\frac{\rho(s,a)}{\sum_{a'}\rho(s,a')}\right) + (1-\lambda)\left(-\rho'(s,a)\log\frac{\rho'(s,a)}{\sum_{a'}\rho'(s,a')}\right), \tag{22}$$

$\sum a_i \log\frac{a_i}{b_i} \geq \sum a_i \log\frac{\sum a_i}{\sum b_i}$

Summing both sides over all s and a shows that

$$\bar{H}(\lambda\rho+(1-\lambda)\rho') \geq \lambda\bar{H}(\rho)+(1-\lambda)\bar{H}(\rho')$$

GAIL

Lemma 3.1. *Let* $\bar{H}(\rho) = -\sum_{s,a} \rho(s,a) \log(\rho(s,a)/\sum_{a'} \rho(s,a'))$. *Then,* $\bar{H}$ *is strictly concave, and for all* $\pi \in \Pi$ *and* $\rho \in \mathcal{D}$, *we have* $H(\pi) = \bar{H}(\rho_\pi)$ *and* $\bar{H}(\rho) = H(\pi_\rho)$.

$$
\begin{aligned}
H(\pi) &= \mathbb{E}_\pi[-\log \pi(a|s)] \\
&= -\sum_{s,a} \rho_\pi(s,a) \log \pi(a|s) \\
&= -\sum_{s,a} \rho_\pi(s,a) \log \frac{\rho_\pi(s,a)}{\sum_{a'} \rho_\pi(s,a')} \\
&= \bar{H}(\rho_\pi),
\end{aligned}
$$

$$
\begin{aligned}
\bar{H}(\rho) &= -\sum_{s,a} \rho(s,a) \log \frac{\rho(s,a)}{\sum_{a'} \rho(s,a')} \\
&= -\sum_{s,a} \rho_{\pi_\rho}(s,a) \log \pi_\rho(a|s) \\
&= \mathbb{E}_{\pi_\rho}[-\log \pi_\rho(a|s)] \\
&= H(\pi_\rho).
\end{aligned}
$$

GAIL

Proposition 3.2. $\mathrm{RL} \circ \mathrm{IRL}_\psi(\pi_E) = \arg\min_{\pi \in \Pi} -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E})$

$f^*(x) = \sup_{y \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}} x^T y - f(y)$

Let $\tilde{c} \in \mathrm{IRL}_\psi(\pi_E)$, $\tilde{\pi} \in \mathrm{RL}(\tilde{c}) = \mathrm{RL} \circ \mathrm{IRL}_\psi(\pi_E)$, and

$$\pi_A \in \arg\min_\pi -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}) \tag{31}$$

$$= \arg\min_\pi \max_c -H(\pi) - \psi(c) + \sum_{s,a}(\rho_\pi(s,a) - \rho_{\pi_E}(s,a))c(s,a) \tag{32}$$

We wish to show that $\pi_A = \tilde{\pi}$. To do this, let ρ_A be the occupancy measure of π_A, let $\tilde{\rho}$ be the occupancy measure of $\tilde{\pi}$, and define $\bar{L} : \mathcal{D} \times \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}$ by

$$\bar{L}(\rho, c) = -\bar{H}(\rho) - \psi(c) + \sum_{s,a} \rho(s,a)c(s,a) - \sum_{s,a} \rho_{\pi_E}(s,a)c(s,a). \tag{33}$$

凸 凹

The following relationships then hold, due to Proposition 3.1:

$$\rho_A \in \arg\min_{\rho \in \mathcal{D}} \max_c \bar{L}(\rho, c), \tag{34}$$

$$\tilde{c} \in \arg\max_c \min_{\rho \in \mathcal{D}} \bar{L}(\rho, c), \tag{35}$$

$$\tilde{\rho} \in \arg\min_{\rho \in \mathcal{D}} \bar{L}(\rho, \tilde{c}). \tag{36}$$

$$\mathrm{IRL}_\psi(\pi_E) = \arg\max_{c \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}} -\psi(c) + \left(\min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_\pi[c(s,a)]\right) - \mathbb{E}_{\pi_E}[c(s,a)]$$

$$\mathbb{E}_\pi[c(s,a)] = \sum_{s,a} \rho_\pi(s,a)c(s,a)$$

$$\min_\rho \max_c \bar{L}(\rho, c) = \max_c \min_\rho \bar{L}(\rho, c)$$

GAIL

- Proposition 3.2 tells us that ψ -regularized inverse reinforcement learning, implicitly, seeks a policy whose occupancy measure is close to the expert's, as measured by the convex function ψ^*
- conclusion
  - IRL is a dual of an occupancy measure matching problem
  - The induced optimal policy is the primal optimum

- Enticingly, this suggests that various settings of ψ lead to various imitation learning algorithms that directly solve the optimization problem given by Proposition 3.2
  - **Corollary 3.2.1.** *If ψ is a constant function, $\tilde{c} \in \mathrm{IRL}_\psi(\pi_E)$, and $\tilde{\pi} \in \mathrm{RL}(\tilde{c})$, then $\rho_{\tilde{\pi}} = \rho_{\pi_E}$.*
  - apprenticeship learning：

$$\underset{\pi}{\text{minimize}} \ \max_{c \in \mathcal{C}} \mathbb{E}_\pi[c(s,a)] - \mathbb{E}_{\pi_E}[c(s,a)]$$

$$\delta_{\mathcal{C}} : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \overline{\mathbb{R}}, \text{ defined by } \delta_{\mathcal{C}}(c) = 0 \text{ if } c \in \mathcal{C} \text{ and } +\infty \text{ otherwise,}$$

$$\max_{c \in \mathcal{C}} \mathbb{E}_\pi[c(s,a)] - \mathbb{E}_{\pi_E}[c(s,a)] = \max_{c \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}} -\delta_{\mathcal{C}}(c) + \sum_{s,a} (\rho_\pi(s,a) - \rho_{\pi_E}(s,a)) c(s,a) = \delta^*_{\mathcal{C}}(\rho_\pi - \rho_{\pi_E})$$

$$\underset{\pi}{\text{minimize}} -H(\pi) + \max_{c \in \mathcal{C}} \mathbb{E}_\pi[c(s,a)] - \mathbb{E}_{\pi_E}[c(s,a)]$$

GAIL

- Generative adversarial imitation learning

$$\psi_{\mathrm{GA}}(c) \triangleq \begin{cases} \mathbb{E}_{\pi_E}[g(c(s,a))] & \text{if } c < 0 \\ +\infty & \text{otherwise} \end{cases} \quad \text{where } g(x) = \begin{cases} -x - \log(1 - e^x) & \text{if } x < 0 \\ +\infty & \text{otherwise} \end{cases}$$

$$\psi^*_{\mathrm{GA}}(\rho_\pi - \rho_{\pi_E}) = \max_{D \in (0,1)^{\mathcal{S} \times \mathcal{A}}} \mathbb{E}_\pi[\log(D(s,a))] + \mathbb{E}_{\pi_E}[\log(1 - D(s,a))]$$

$$\underset{\pi}{\text{minimize}}\ \psi^*_{\mathrm{GA}}(\rho_\pi - \rho_{\pi_E}) - \lambda H(\pi) = D_{\mathrm{JS}}(\rho_\pi, \rho_{\pi_E}) - \lambda H(\pi)$$

GAIL

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** $i = 0, 1, 2, \ldots$ **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

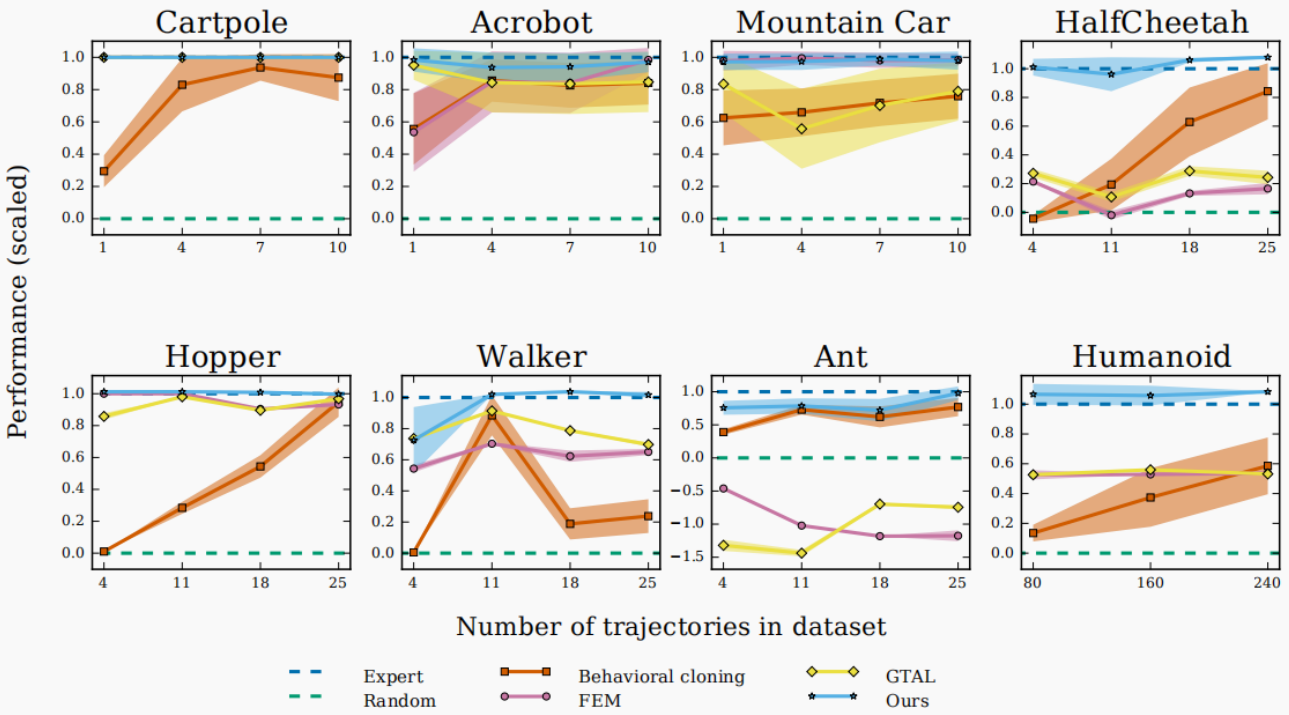
$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \tag{17}$$

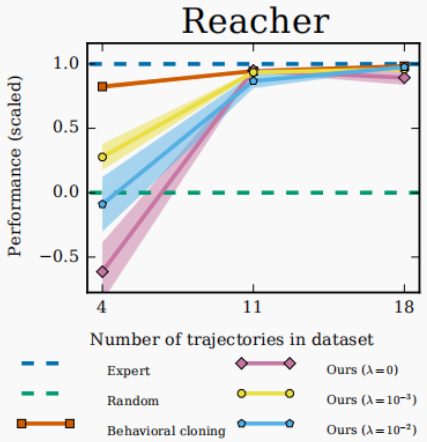
5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

$$\begin{aligned} &\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta), \\ &\text{where } Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \end{aligned} \tag{18}$$

6: **end for**

GAIL


Cartpole
Acrobot
Mountain Car
HalfCheetah
Hopper
Walker
Ant
Humanoid
Performance (scaled)
Number of trajectories in dataset
Expert
Random
Behavioral cloning
FEM
GTAL
Ours


Reacher
Performance (scaled)
Number of trajectories in dataset
Expert
Random
Behavioral cloning
Ours (λ=0)
Ours (λ=10⁻³)
Ours (λ=10⁻²)

RL platform

https://github.com/wxc971231/RL_Platform-CrossTheWall