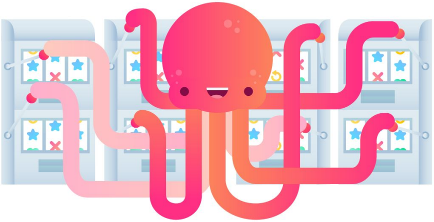
Imitation Learning

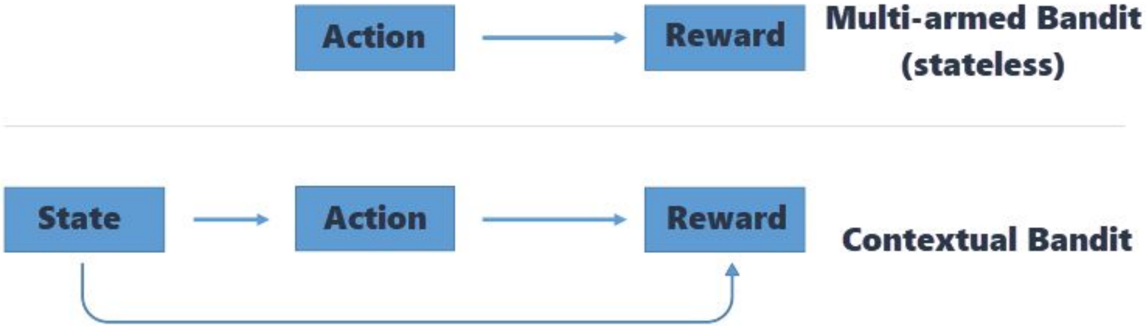
Roberto Capobianco

Sapienza Università di Roma

Recap

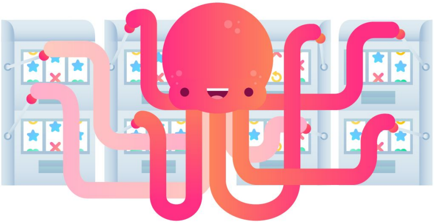
From Multi-Armed to Contextual Bandits





Contextual bandits add back some context (state)

Contextual Bandits: Interaction



The interactive process that we deal with in CB is the following:

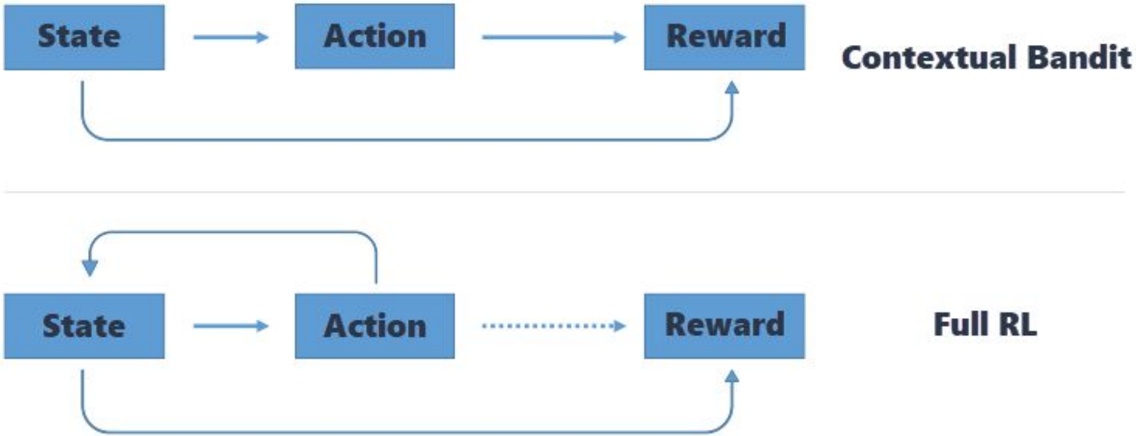
For t = 0, ..., T-1:

1. A new i.i.d. context x_t in X appears
2. Select an action a_t in A based on historical information and context
3. Observe reward $r(x_t,a_t)$ (which is context and arm dependent)

For simplicity we assume deterministic rewards, as the context is the challenge here

Contextual Bandits VS RL

In RL, conversely, states depend on previous actions: **we can say that contextual bandits are Finite-Horizon MDPs with horizon 1**



Contextual Bandits: Regret

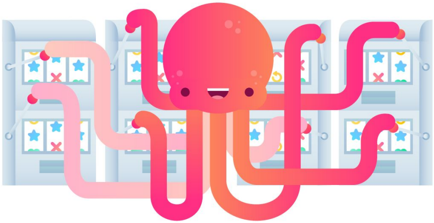
Optimal policy: $\pi^\star = \arg\max_{\pi \in \Pi} \mathbb{E}_{x \sim \mu} r(x, \pi(x))$

At every iteration $a_t = \pi_t(x_t)$ is selected and a reward $r(x_t, a_t)$ is received: the regret is the **total expected reward if we always use π^*** VS the **total expected reward if we use our learned sequence of policies**

$$\text{Regret}_T = \boxed{T\mathbb{E}_{x\sim\mu}[r(x, \pi^\star(x))]} - \boxed{\sum_{t=0}^{T-1} \mathbb{E}_{x\sim\mu}[r(x, \pi^t(x))]}$$

Note that policies are different at every iteration t

Explore & Commit Algorithm



1. For t = 0, ..., N-1: **(explore)**
   - observe state $x_t \sim \mu$
   - uniform-randomly sample $a_t \sim \text{Unif}(A)$
   - observe reward $r_t = r(x_t, a_t)$
   - build, for x_t, an unbiased estimate of $\boxed{\mathbb{E}_{a_t \sim p}\hat{\mathbf{r}}[a] = r(x_t, a), \forall a}$
2. Compute policy

$$\hat{\pi} = \arg\max_{\pi \in \Pi} \sum_{i=0}^{N-1} \hat{\mathbf{r}}_i[\pi(x_i)]$$

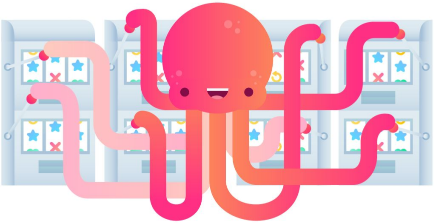
Given we are sampling from Unif(*A*)

$$\hat{\mathbf{r}}_t[a] = \begin{cases} 0 & a \neq a_t \\ \dfrac{r_t}{1/|\mathscr{A}|} & a = a_t \end{cases}$$

3. For t = N, ..., T-1: **(commit)**
   - observe state $x_t \sim \mu$
   - play arm

$$\text{Regret}_T = T\mathbb{E}_{x \sim \mu}[r(x, \pi^\star(x))] - \sum_{t=0}^{T-1} \mathbb{E}_{x \sim \mu}[r(x, \pi^t(x))] = O\left(T^{2/3} K^{1/3} \cdot \ln(|\Pi|)^{1/3}\right)$$

ε-Greedy



Instead of setting a threshold for exploring and then committing, we can try to interleave exploration and exploitation

1. For t = 0, ..., T: **(interleave exploration & exploitation)**
   - observe state $x_t \sim \mu$
   - $a_t \sim p_t = \boxed{(1-\varepsilon)\delta(\pi^t(x_t))} + \boxed{\varepsilon \mathrm{Unif}(A)}$
   - observe reward $r_t = r(x_t, a_t)$
   - build, for x_t, an unbiased estimate of $\mathbb{E}_{a_t \sim p}\hat{\mathbf{r}}[a] = r(x_t, a), \forall a$
2. Update policy

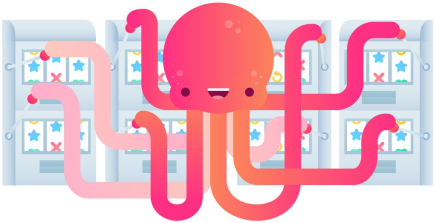
$$\pi^{t+1} = \arg\max_{\pi \in \Pi} \sum_{i=0}^{t} \hat{\mathbf{r}}_i[\pi(x_i)]$$

ε = 0 -> exploit

ε = 1 -> uniformly explore

SAPIENZA UNIVERSITÀ DI ROMA

Bayesian Bandits



So far we have made no assumptions about the reward distribution ν_i, we only derived bounds on rewards

In Bayesian Bandits, however:

- We exploit *prior* knowledge of rewards
- Update a *posterior distribution* of rewards based on historical information
- Use posterior to guide exploration using:
    - upper confidence bounds (Bayesian UCB)
    - probability matching (Thompson Sampling)

Gaussian Bayesian Bandits: UCB

Now we are modelling a distribution, so we already have confidence

What is confidence for Gaussians? **standard deviation**

Let's do UCB by selecting the action with highest standard deviation

$$a_t = \text{argmax}_{i \text{ in } K}\, \mu_t(i) + c\sigma_t(i)/\sqrt{N_t(i)}$$

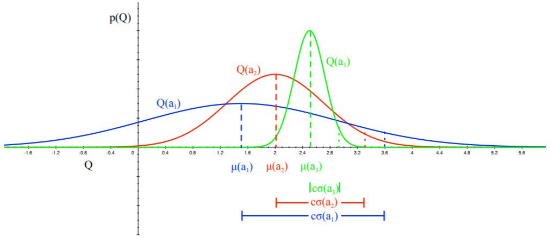
Gaussian Bayesian Bandits: Thompson Sampling

This is an estimation of the reward, in more generic MDPs this can be replaced with the Q function: we estimate a distribution of Q

For t = 0, ..., T:

1. for each arm i = 1, ..., K:
   - sample $\boxed{\hat{\mathbf{r}}_i}$ independently from $N(\mu_{t-1}(i),\ \sigma^2_{t-1}(i))$
2. pull arm

$$I_t = \arg\max_{i \in [K]} \hat{\mathbf{r}}_i$$



3. observe reward r_t
4. update posterior distribution $p(\mu_t(i), \sigma^2_t(i) | r_t)$

This can be done with different distributions as well

SAPIENZA
UNIVERSITÀ DI ROMA

End Recap

Motivation

Designing rewards is hard!

$$\hat{x}_t - 0.1||a_t||_2^2 - 3.0 \times (z_t - 1.3)^2$$

Hard tasks to define a reward (e.g. cooking)

Reward exploitation
https://openai.com/blog/faulty-reward-functions

Credits: Pieter Abbeel & Stuart Russell

Imitation Learning

Given an MDP, what happens if we cannot get access to a reward?

Imitation Learning

Given an MDP, what happens if we cannot get access to a reward?

We can learn by imitation of an expert!

Imitation Learning

Given an MDP, what happens if we cannot get access to a reward?

We can learn by imitation of an expert!

1. Collect expert demonstrations

$$D = \{s_i^*, a_i^*\}_{i=1}^{M} \sim d^{\pi^*}$$

For simplicity, let's assume expert is a (nearly) optimal policy π^*

Imitation Learning

Given an MDP, what happens if we cannot get access to a reward?

We can learn by imitation of an expert!

1. Collect expert demonstrations
2. Use a machine learning algorithm to learn to map states to actions

i.e., do regression or classification

$$\hat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{M} \ell\left(\pi, s^\star, a^\star\right)$$

loss can be negative likelihood

$$-\ln \pi(a^\star \mid s^\star)$$

or square error

$$\|\pi(s) - a^\star\|_2^2$$

Imitation Learning

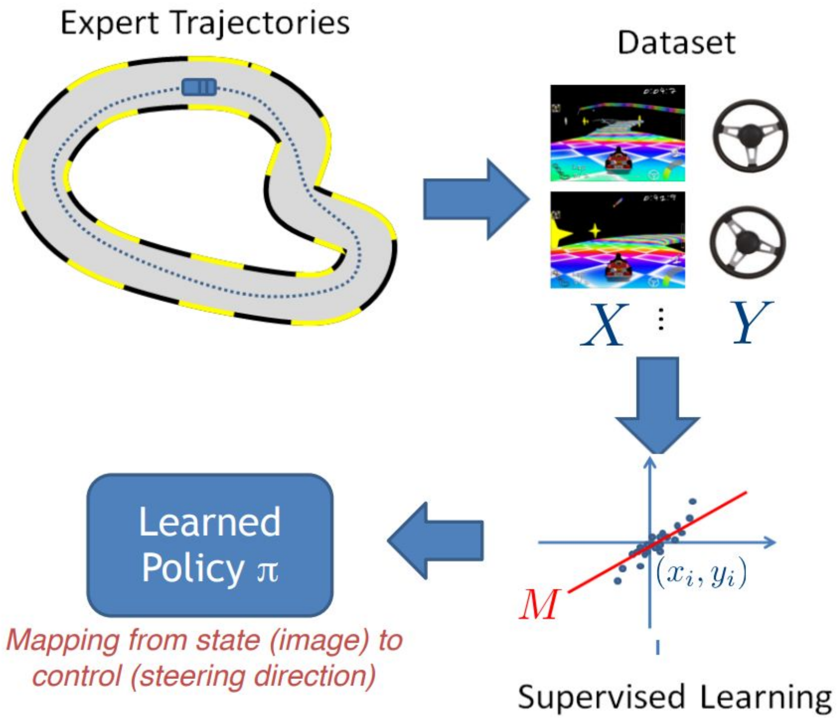
Given an MDP, what happens if we cannot get access to a reward?

We can learn by imitation of an expert!

1. Collect expert demonstrations
2. Use a machine learning algorithm to learn to map states to actions
3. Generate a policy

For simplicity, let's assume expert is a (nearly) optimal policy π^*

Behavior Cloning



SAPIENZA
UNIVERSITÀ DI ROMA

Credits: Wen Sun

Behavior Cloning

Behavior Cloning: Issues

Behavior cloning, with probability 1-δ, returns a policy such that

$$V^{\pi^\star} - V^{\hat{\pi}} \leq \frac{2}{(1-\gamma)^2}\epsilon$$

(you can prove it using performance difference lemma)

Behavior Cloning: Issues

Behavior cloning, with probability 1-δ, returns a policy such that

$$V^{\pi^\star} - V^{\hat{\pi}} \leq \frac{2}{\boxed{(1-\gamma)^2}}\epsilon$$

Quadratic!

(you can prove it using performance difference lemma)

Credits: Wen Sun

Behavior Cloning: Issues

Behavior cloning, with probability 1-δ, returns a policy such that

$$V^{\pi^\star} - V^{\hat{\pi}} \leq \frac{2}{\boxed{(1-\gamma)^2}}\epsilon$$

Quadratic!

(you can prove it using performance difference lemma)

Why?

SAPIENZA UNIVERSITÀ DI ROMA

Credits: Wen Sun

Behavior Cloning: Issues

Behavior cloning, with probability 1-δ, returns a policy such that

$$V^{\pi^\star} - V^{\hat{\pi}} \leq \frac{2}{\boxed{(1-\gamma)^2}}\epsilon$$

Quadratic!

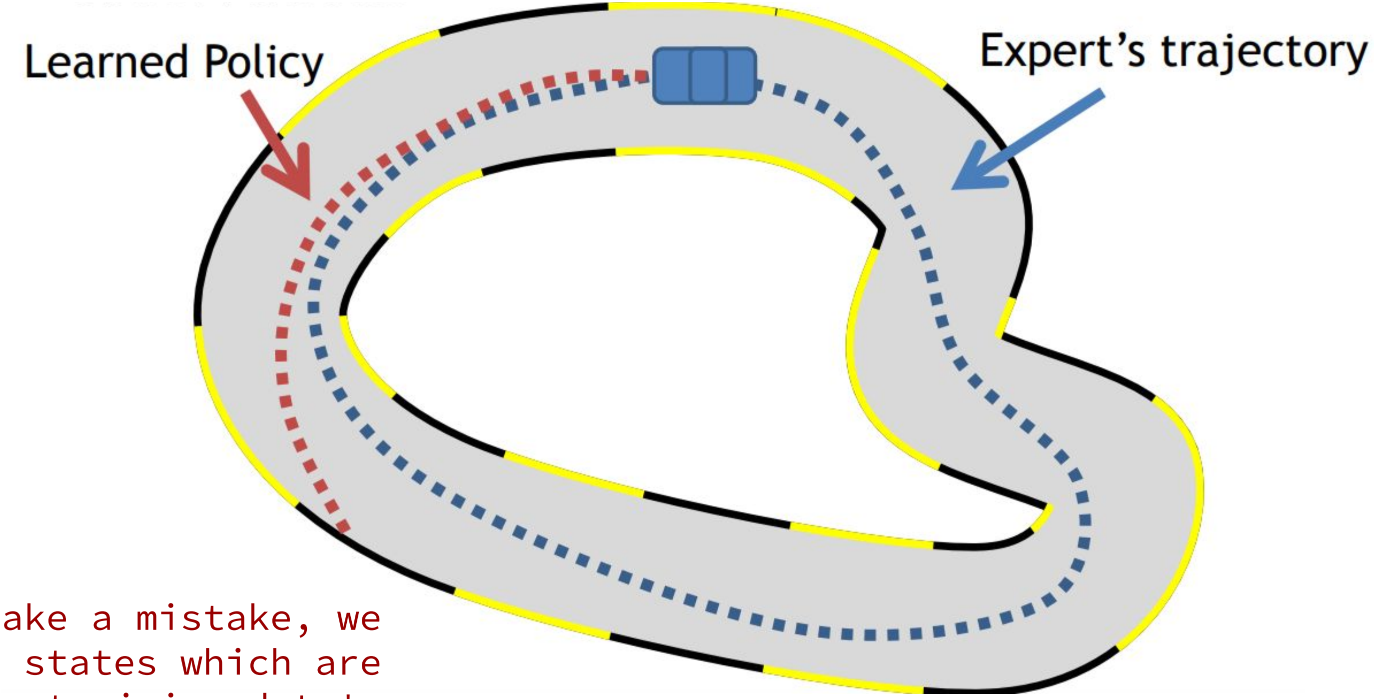
(you can prove it using performance difference lemma)

Why? Predictions affect future inputs/observations, inducing a distribution shift

SAPIENZA
UNIVERSITÀ DI ROMA

Credits: Wen Sun

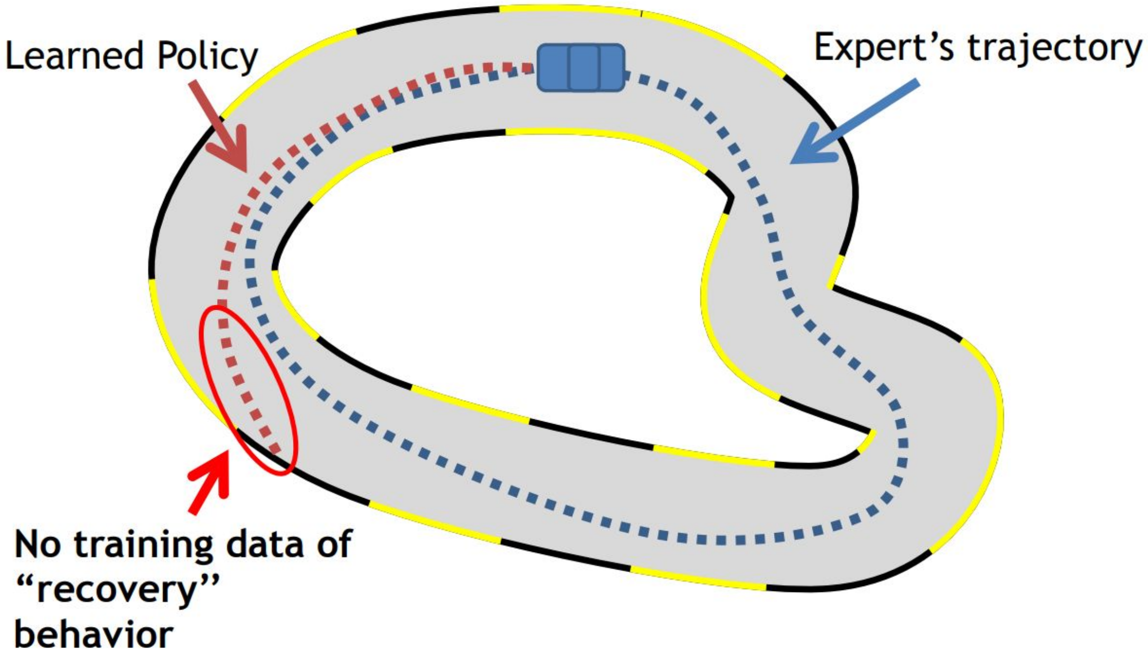
Behavior Cloning: Distribution Shift



Once we make a mistake, we end up in states which are not in the training data!

Credits: Wen Sun

Behavior Cloning: Distribution Shift

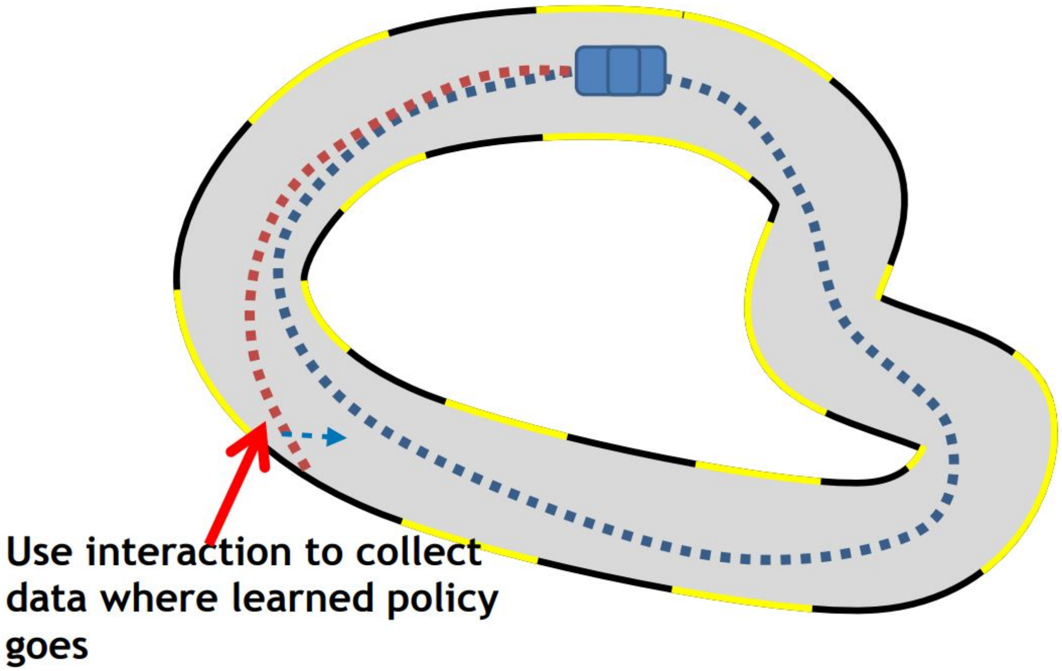


Credits: Wen Sun

Interactive Imitation Learning

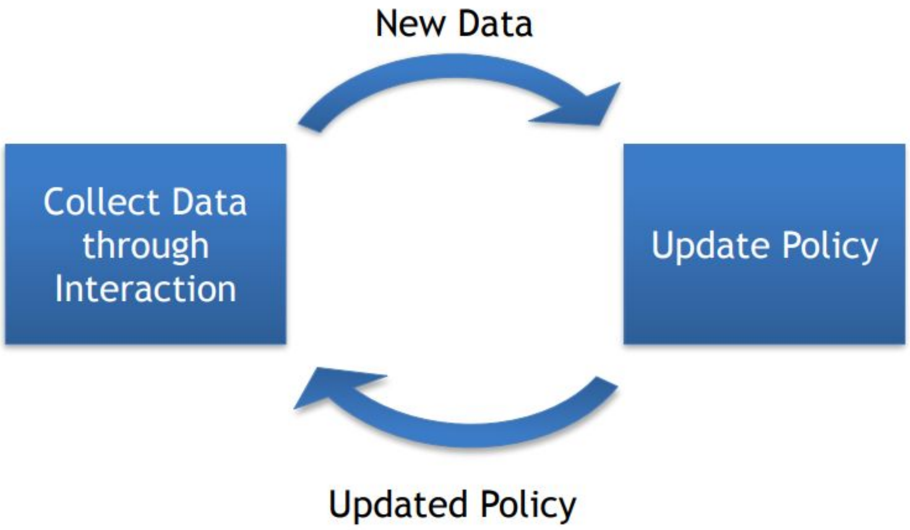
Can we alleviate such problem? Yes, by setting up an interactive process where we continuously query the expert

Interactive Imitation Learning



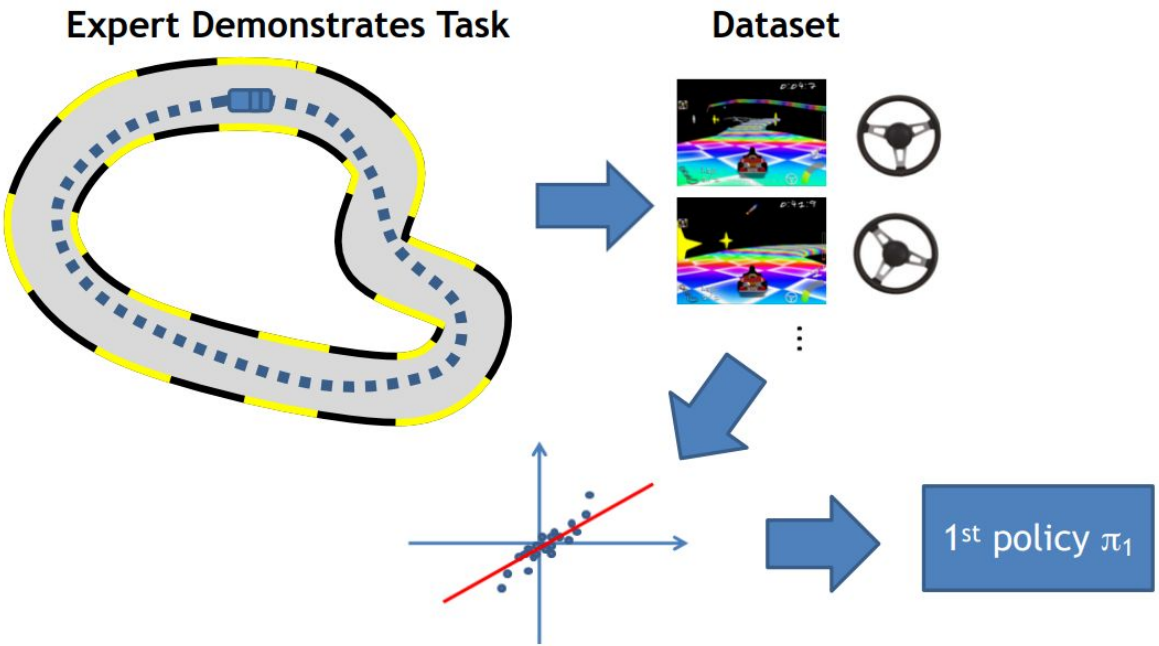
Credits: Wen Sun

DAgger



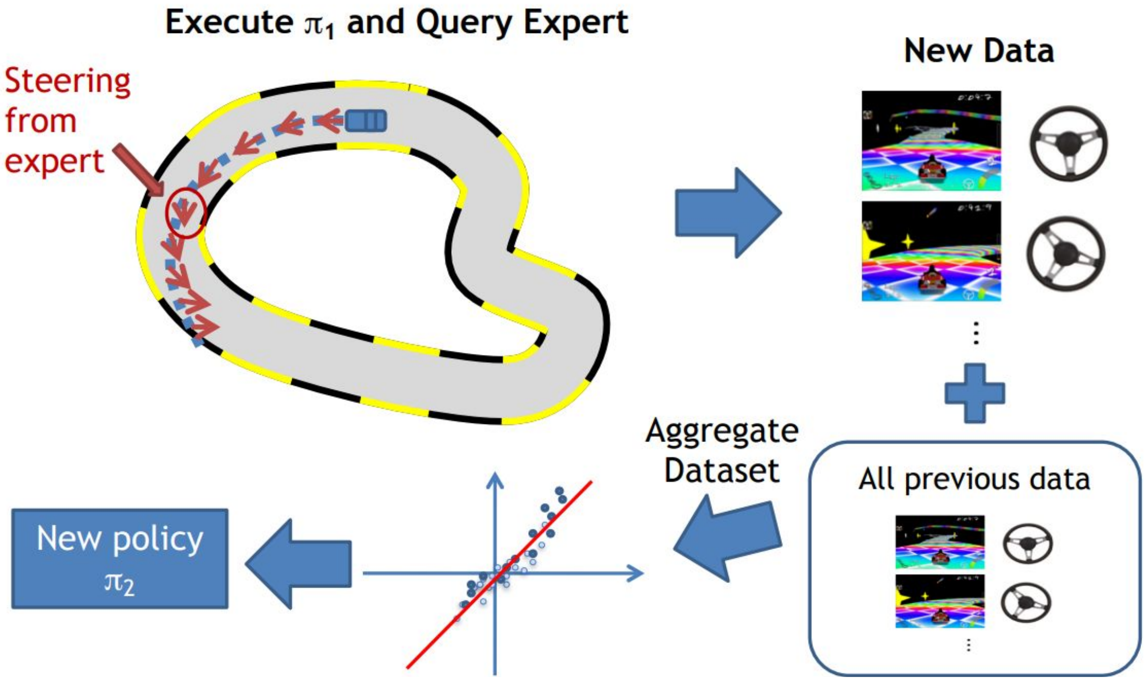
Credits: Wen Sun, Drew Bagnell, Stephane Ross, Arun Venktraman

DAgger: Iterations (0th)



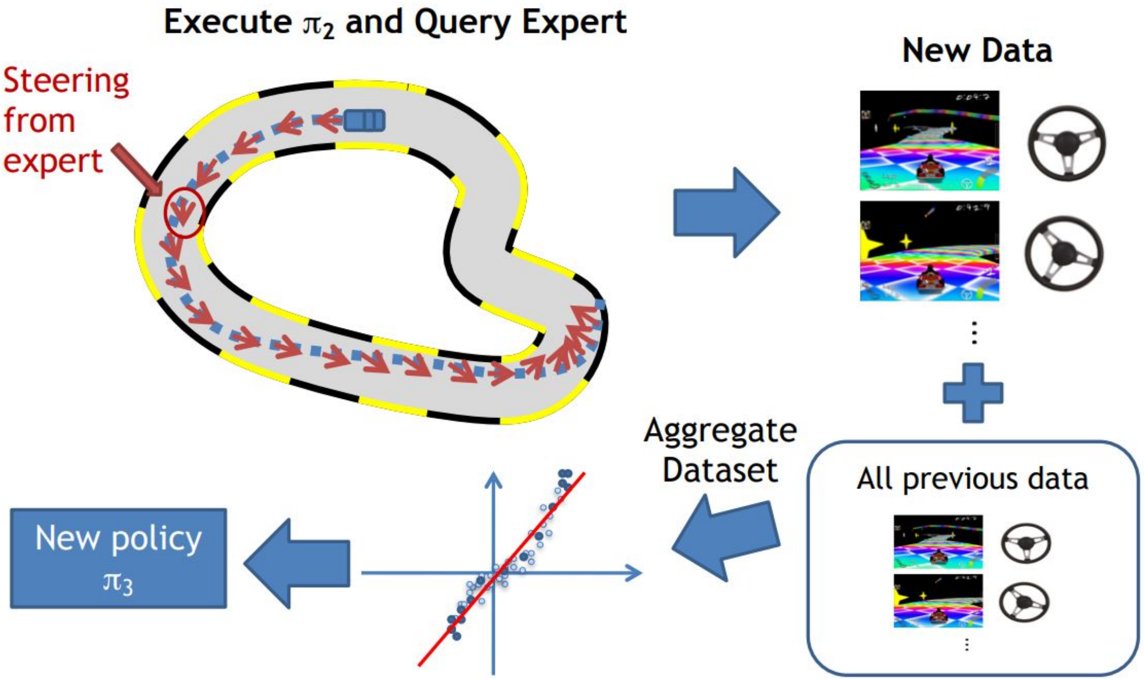
Credits: Wen Sun, Drew Bagnell, Stephane Ross, Arun Venktraman

DAgger: Iterations (1st)



Credits: Wen Sun, Drew Bagnell, Stephane Ross, Arun Venktraman

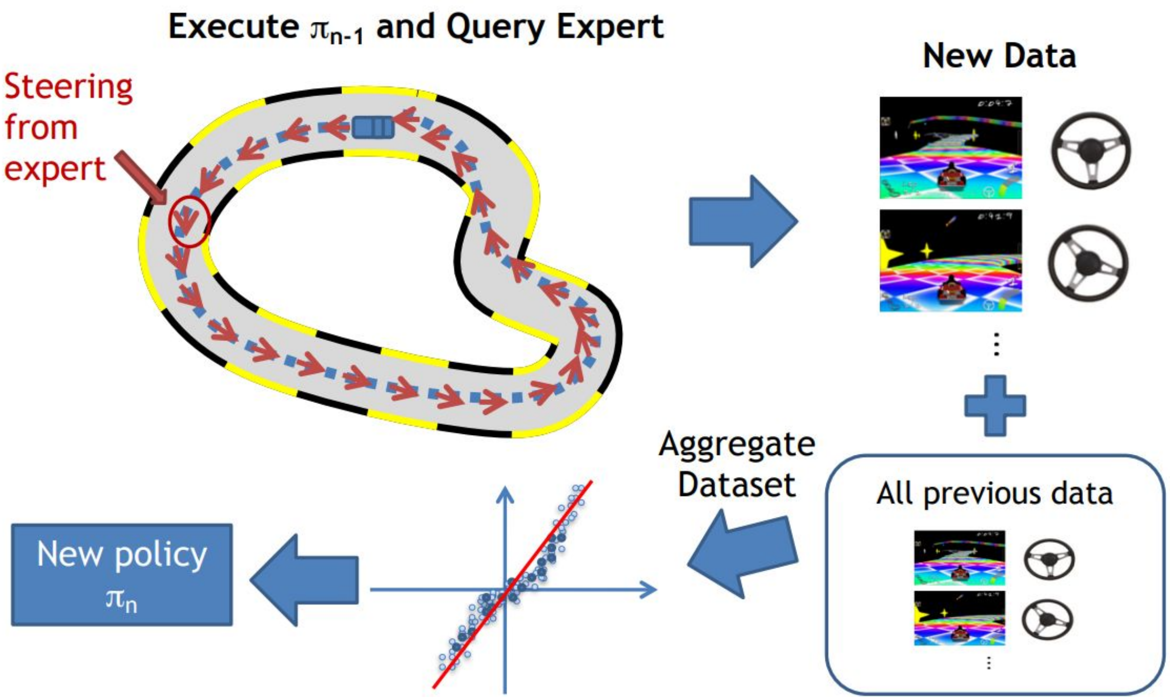
DAgger: Iterations (2nd)



SAPIENZA
UNIVERSITÀ DI ROMA

Credits: Wen Sun, Drew Bagnell, Stephane Ross, Arun Venktraman

DAgger: Iterations (n-th)



SAPIENZA
UNIVERSITÀ DI ROMA

Credits: Wen Sun, Drew Bagnell, Stephane Ross, Arun Venktraman

DAgger: Video

Inverse Reinforcement Learning

Given an MDP, what happens if we cannot get access to a reward?

We can learn through an optimal expert that minimizes the true cost!

Assume transition function is known and we have our dataset *D*

Inverse Reinforcement Learning

Given an MDP, what happens if we cannot get access to a reward?

We can learn through an optimal expert that minimizes the true cost!

Assume transition function is known and we have our dataset *D*

Also assume the true reward/cost is linear in some features □(s,a)

Entropy

Given a distribution P, the entropy is:

$$\text{Entropy}(P) = -\sum_x P(x) \cdot \ln P(x)$$

Higher entropy means higher uncertainty (i.e., a deterministic distribution has 0 entropy, uniform the highest)

Maximum Entropy

We want to find a distribution whose mean and covariance matrix equal some values, but there are infinitely many such distributions:

we choose the least committing one, with maximum entropy

Maximum Entropy IRL

We want to find a policy such that

$$\max_{\pi} \mathbb{E}_{s \sim d_\mu^\pi} \left[\text{entropy} \left(\pi(\cdot \mid s) \right) \right]$$

$$s.t, \mathbb{E}_{s,a \sim d_\mu^\pi} \phi(s,a) = \boxed{\mathbb{E}_{s,a \sim d_\mu^{\pi^\star}} \phi(s,a)}$$

From expert data:

$$\sum_{i=1}^{N} \phi(s_i^\star, a_i^\star)/N$$

Maximum Entropy IRL

We want to find a policy such that

$$\max_{\pi} \mathbb{E}_{s \sim d_\mu^\pi} \boxed{\left[\text{entropy}\left(\pi(\cdot \mid s)\right)\right]}$$

$$s.t, \mathbb{E}_{s,a \sim d_\mu^\pi} \phi(s,a) = \mathbb{E}_{s,a \sim d_\mu^{\pi^\star}} \phi(s,a)$$

$$\mathbb{E}_{s \sim d_\mu^\pi}\left[\text{entropy}(\pi(\cdot \mid s))\right] = -\mathbb{E}_{s \sim d_\mu^\pi} \mathbb{E}_{a \sim \pi(\cdot \mid s)} \ln \pi(a \mid s) = -\mathbb{E}_{s,a \sim d_\mu^\pi} \ln \pi(a \mid s)$$

Maximum Entropy IRL

We want to find a policy such that

$$\boxed{\max_{\pi} \mathbb{E}_{s \sim d_\mu^\pi} \left[\text{entropy} \left(\pi(\cdot \mid s) \right) \right]}$$

$$s.t, \mathbb{E}_{s,a \sim d_\mu^\pi} \phi(s, a) = \mathbb{E}_{s,a \sim d_\mu^{\pi^\star}} \phi(s, a)$$

$$\arg\max_{\pi} \mathbb{E}_{s \sim d_\mu^\pi} \left[\text{entropy}(\pi(\cdot \mid s)) \right] = \arg\min_{\pi} \mathbb{E}_{s,a \sim d_\mu^\pi} \ln \pi(a \mid s)$$

Maximum Entropy IRL

We want to find a policy such that

$$\boxed{\begin{aligned}&\max_{\pi} \mathbb{E}_{s\sim d_\mu^\pi}\left[\text{entropy}\left(\pi(\cdot\,|\,s)\right)\right]\\ &s\,.\,t,\ \mathbb{E}_{s,a\sim d_\mu^\pi}\phi(s,a) = \mathbb{E}_{s,a\sim d_\mu^{\pi^\star}}\phi(s,a)\end{aligned}}$$

Using Lagrange multipliers

$$\max_{w\in\mathbb{R}^d}\min_{\pi} \mathbb{E}_{s,a\sim d_\mu^\pi} \ln \pi(a\,|\,s) + w^\top\left(\mathbb{E}_{s,a\sim d_\mu^\pi}\phi(s,a) - \mathbb{E}_{s,a\sim d_\mu^{\pi^\star}}\phi(s,a)\right)$$

Maximum Entropy IRL: Algorithm

Initialize $w^0 \in \mathbb{R}^d$

This is like an RL problem w/ cost

For $t = 0 \rightarrow T-1$

$c(s,a) := (w^t)^\top \phi(s,a)$, but w/ an additional $\ln \pi(a \mid s)$

$$\pi^t = \arg\min_{\pi} \mathbb{E}_{s,a \sim d_\mu^\pi} \left[(w^t)^\top \phi(x,a) + \ln \pi(a \mid s) \right]$$

(# best response: $\pi^t = \arg\min_{\pi} \ell(\pi, w^t)$)

$$w^{t+1} = w^t + \eta \left(\mathbb{E}_{s,a \sim d_\mu^{\pi^t}} \phi(s,a) - \mathbb{E}_{s,a \sim d_\mu^{\pi^\star}} \phi(s,a) \right)$$

(# gradient update: $w^{t+1} = w^t + \eta \nabla_w \ell(\pi^t, w^t)$)

Return $\bar{\pi} = \text{Uniform}(\pi^0, \ldots, \pi^{T-1})$

Maximum Entropy IRL: Algorithm

Initialize $w^0 \in \mathbb{R}^d$

This is like an RL problem w/ cost $c(s,a) := (w^t)^\top \phi(s,a)$, but w/ an additional $\ln \pi(a \mid s)$

For $t = 0 \rightarrow T-1$

$$\pi^t = \arg\min_{\pi} \mathbb{E}_{s,a\sim d_\mu^\pi}\left[(w^t)^\top \phi(x,a) + \ln \pi(a \mid s)\right]$$

Uses soft-value iteration

(# best response: $\pi^t = \arg\min_{\pi} \ell(\pi, w^t)$)

$$w^{t+1} = w^t + \eta\left(\mathbb{E}_{s,a\sim d_\mu^{\pi^t}}\phi(s,a) - \mathbb{E}_{s,a\sim d_\mu^{\pi^\star}}\phi(s,a)\right)$$

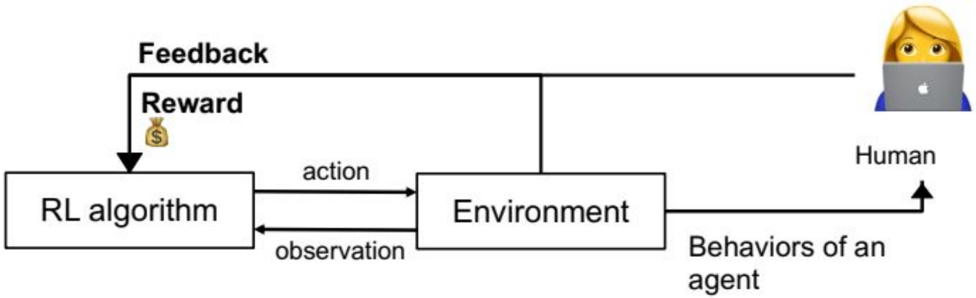
(# gradient update: $w^{t+1} = w^t + \eta \nabla_w \ell(\pi^t, w^t)$)

Return $\bar{\pi} = \text{Uniform}(\pi^0, \ldots, \pi^{T-1})$

Human-in-the-loop RL

Possible assumptions on human availability:

- Humans watches all the time
- Human watches periodically
- Human can be sent queries



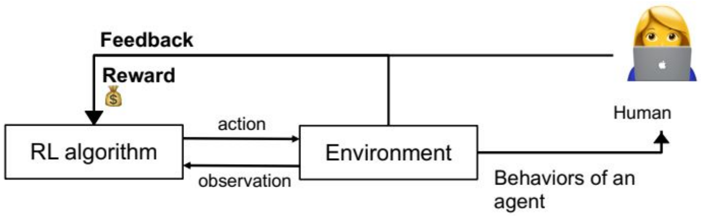
Credits: Pieter Abbeel & Stuart Russell

- Can teach harder tasks, where we can’t easily define the reward
- Can avoid reward exploitation

Human-in-the-loop RL

Possible assumptions on human availability:

- Humans watches all the time
- Human watches periodically
- Human can be sent queries

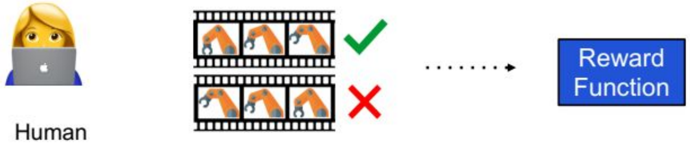


Credits: Pieter Abbeel & Stuart Russell

Approaches:

- Preference-based learning
  - Human is asked to compare 2 trajectories (or segments)
- Advantage-based learning (COACH / TAMER)

Preference-Based Learning



Fitting a reward model [3] we can formulate this problem as a binary classification: we can model a **preference predictor** as follows:

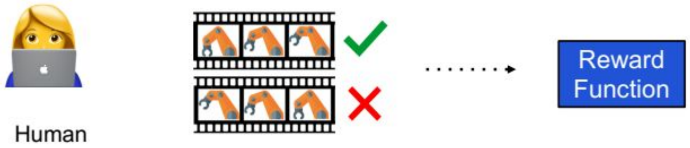
Sum of rewards over segment 1

$$P_\psi[\sigma^1 \succ \sigma^0] = \frac{\exp \sum_t \widehat{r}(\mathbf{s}_t^1, \mathbf{a}_t^1)}{\sum_{i\in\{0,1\}} \exp \sum_t \widehat{r}(\mathbf{s}_t^i, \mathbf{a}_t^i)}$$

Event that segment 1 is preferable to segment 0

1. Akrour, R., Schoenauer, M., and Sebag, M. Preference-based policy learning. In ECML-PKDD, 2011.
2. Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S. and Amodei, D., Reward learning from human preferences and demonstrations in atari. In NeurIPS, 2018.
3. Christiano, P., Leike, J., Brown, T.B., Martic, M., Legg, S. and Amodei, D., Deep reinforcement learning from human preferences. NeurIPS, 2017.
4. Lee, K., Smith, L., Abbeel, P. PEBBLE: Feedback-Efficient Interactive RL via Relabeling Experience and Unsupervised Pre-Training, ICML 2021

Preference-Based Learning



Fitting a reward model [3] we can formulate this problem as a binary classification: we can model a **preference predictor** as follows:

Sum of rewards over segment 1

$$P_\psi[\sigma^1 \succ \sigma^0] = \frac{\exp \sum_t \widehat{r}(\mathbf{s}_t^1, \mathbf{a}_t^1)}{\sum_{i \in \{0,1\}} \exp \sum_t \widehat{r}(\mathbf{s}_t^i, \mathbf{a}_t^i)}$$

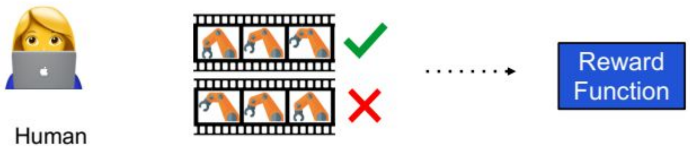
Event that segment 1 is preferable to segment 0

and learn a reward model by optimizing cross entropy:

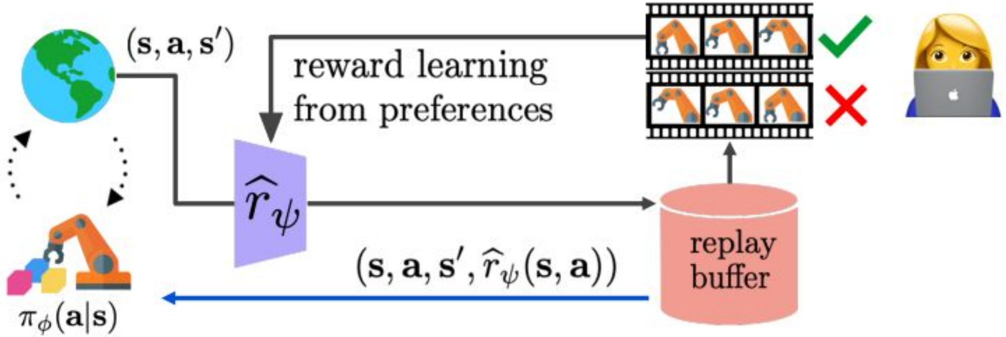
$$\mathcal{L}^{\texttt{Reward}} = -\mathbb{E}_{(\sigma^0,\sigma^1,y)}\Big[y(0)\log P_\psi[\sigma^0 \succ \sigma^1] + y(1)\log P_\psi[\sigma^1 \succ \sigma^0]\Big]$$

1. Akrour, R., Schoenauer, M., and Sebag, M. Preference-based policy learning. In ECML-PKDD, 2011.
2. Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S. and Amodei, D., Reward learning from human preferences and demonstrations in atari. In NeurIPS, 2018.
3. Christiano, P., Leike, J., Brown, T.B., Martic, M., Legg, S. and Amodei, D., Deep reinforcement learning from human preferences. NeurIPS, 2017.
4. Lee, K., Smith, L., Abbeel, P. PEBBLE: Feedback-Efficient Interactive RL via Relabeling Experience and Unsupervised Pre-Training, ICML 2021

Preference-Based Learning



1. Collect samples via interactions with environment
2. Collect human preferences
3. Optimize a reward model using cross entropy loss
4. Optimize a policy using off-policy algorithms



Credits: Pieter Abbeel & Stuart Russell

TAMER & COACH

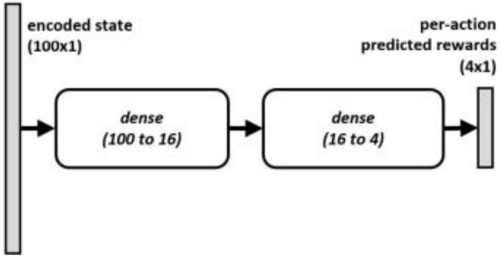
- TAMER
  - Applies human feedback uniformly to past in fixed relative window
  - Train a **reward model** with human **reward** feedback

- COACH
  - Applies human feedback exponentially decaying weight to the past
  - Train a **policy** with human **advantage** feedback

1. Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, Peter Stone. Deep TAMER: Interactive Agent Shaping in High-Dimensional State Spaces, AAAI 2018
2. Dilip Arumugam, Jun Ki Lee, Sophie Saskin, Michael L. Littman. DeepCOACH: Deep Reinforcement Learning from Policy-Dependent Human Feedback. 2019

TAMER

- Maps state representation to a vector of per action rewards
- Humans give direct real valued reward targets
- Optimize weighted mean squared error
- policy = argmax(r(*f*(state)))
    - Hard to assign numerical reward scores
    - Myopic policy
    - Experiments only done on environments when human feedback easy



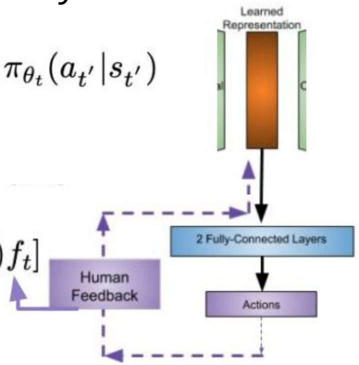
1. Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, Peter Stone. Deep TAMER: Interactive Agent Shaping in High-Dimensional State Spaces, AAAI 2018

COACH

- Policy maps state representation to an action
- Human feedback in {-1, 0, 1} acts as advantage
  - Must adjust expectations over time
  - Experiments only done on environments when human feedback easy
- Gradients actor-critic style
  - Likelihood weighting enables replay buffer use

$$\frac{\pi_{\theta_t}(a_{t'}|s_{t'})}{\pi_{\theta_{t'}}(a_{t'}|s_{t'})}\nabla_{\theta_t}\log\pi_{\theta_t}(a_{t'}|s_{t'})$$

$$\nabla_{\theta_t}J(\theta_t) = \mathbb{E}_{a\sim\pi_{\theta_t}(\cdot|s)}[\nabla_{\theta_t}\log\pi_{\theta_t}(a|s)f_t]$$



1. Dilip Arumugam, Jun Ki Lee, Sophie Saskin, Michael L. Littman. DeepCOACH: Deep Reinforcement Learning from Policy-Dependent Human Feedback. 2019