Machine Learning Summer School 2025

Arequipa

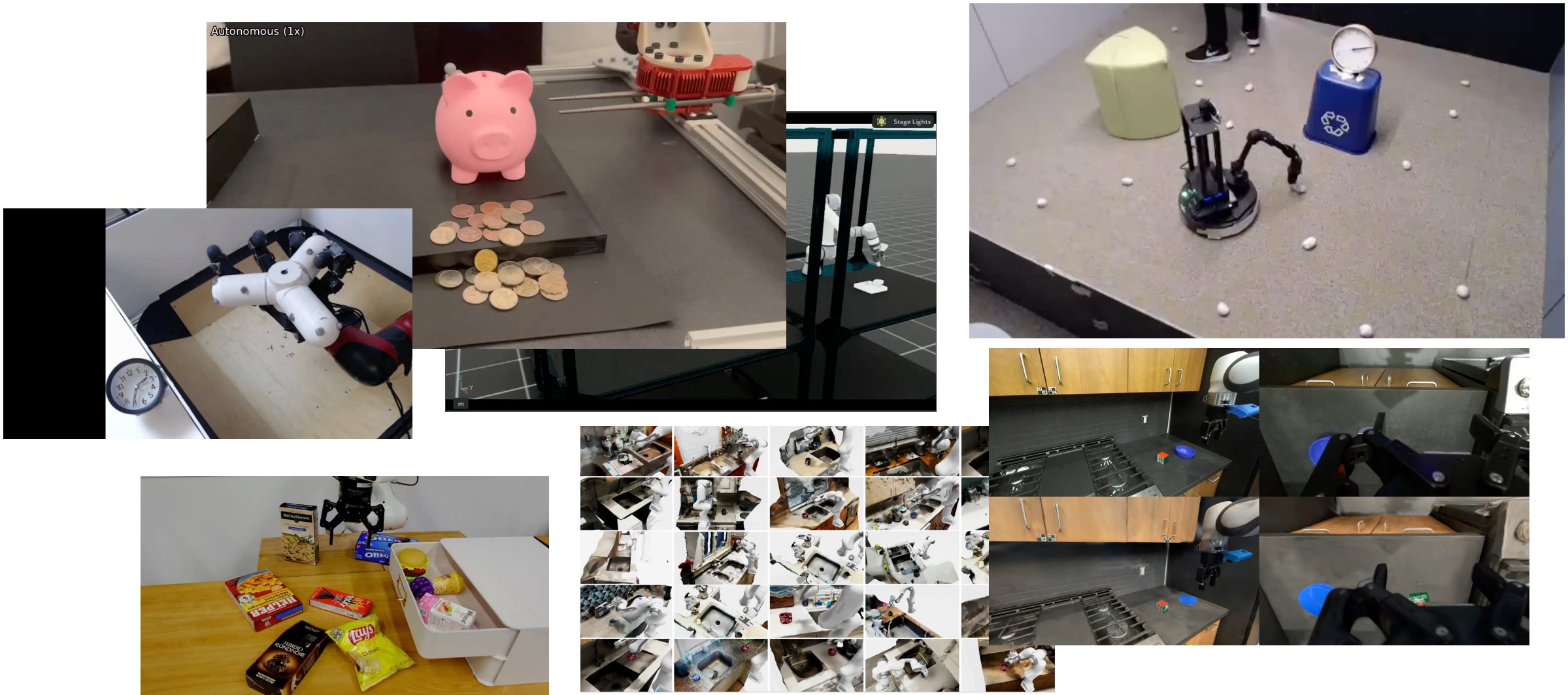
Abhishek Gupta

Who am I?

- Assistant professor in CSE at University of Washington
- Grew up in Oregon/India, spent 10 years in Berkeley
- Undergrad Berkeley, Ph.D. Berkeley, Postdoc MIT.
- Interests: RL/robotics/optimization and control/robustness and generalization
- Outside of work: Tennis/soccer/sketching/dog enthusiast

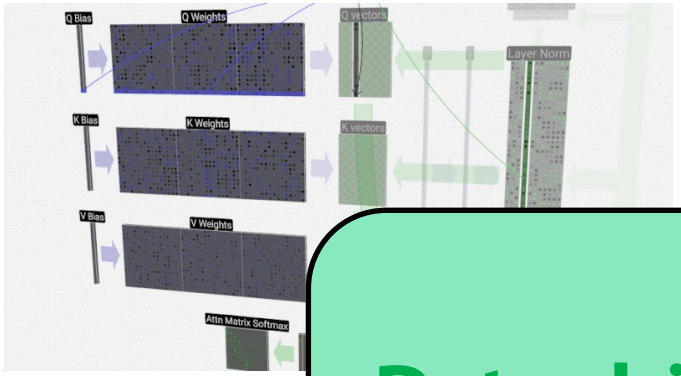
What do I work on?

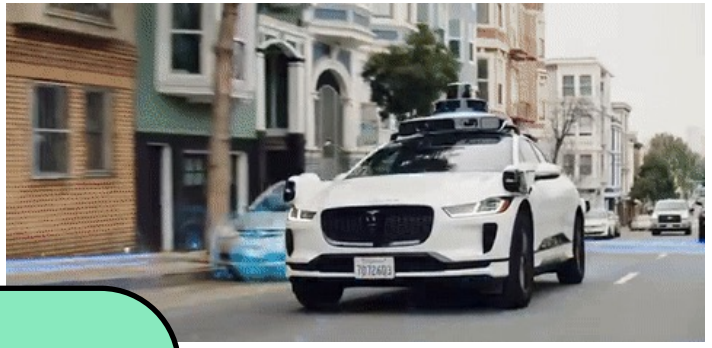
Robots that keep getting better – reinforcement learning!



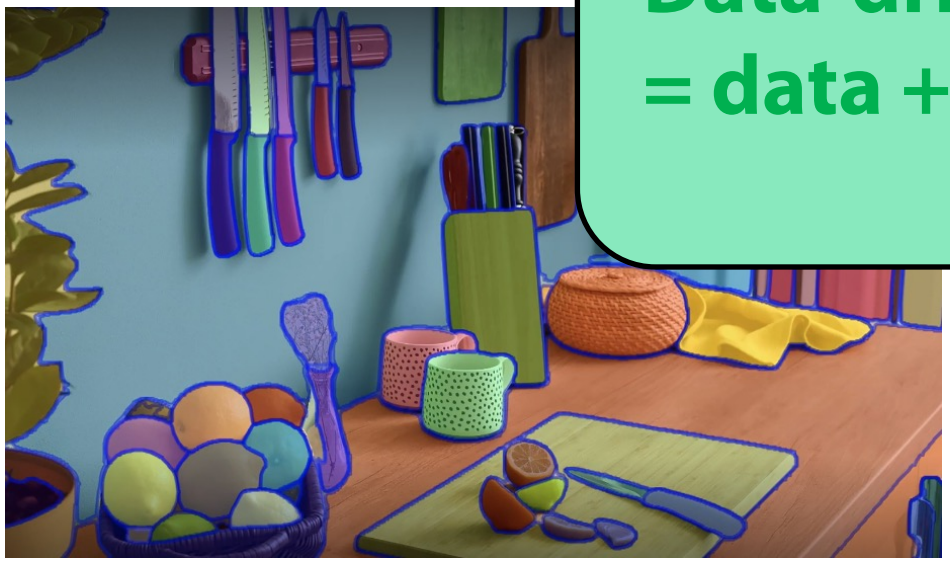
Who do I work with?

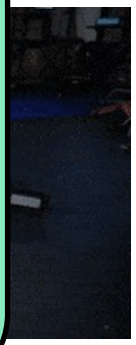
What are we going to talk about today?





Data-driven decision making
= data + sequential decisions

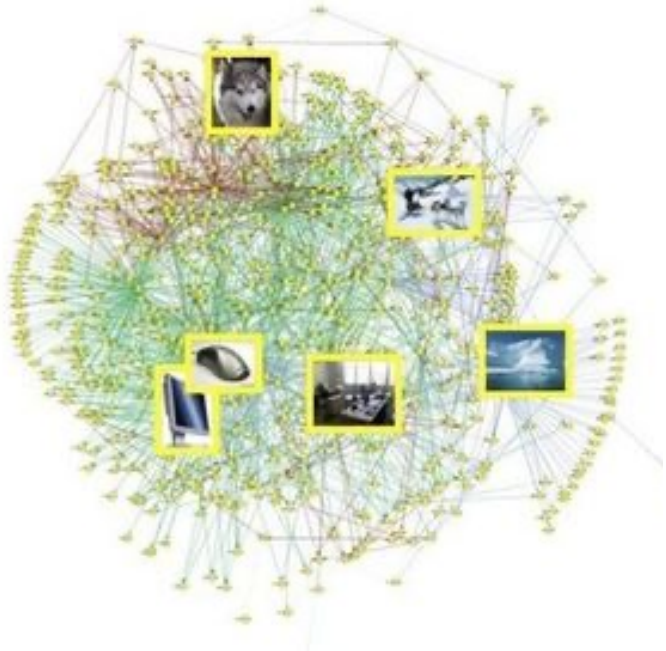


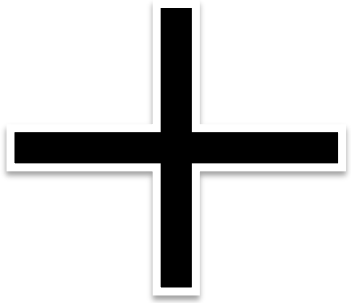


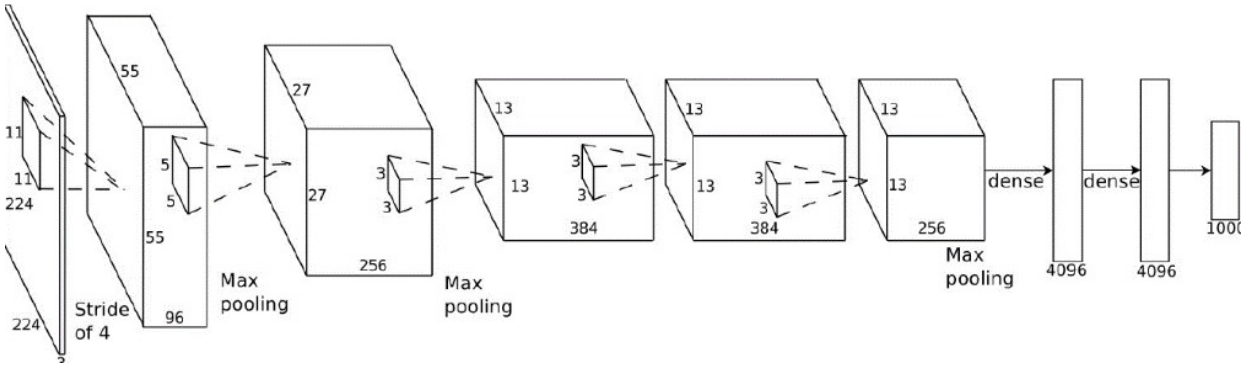
Prediction

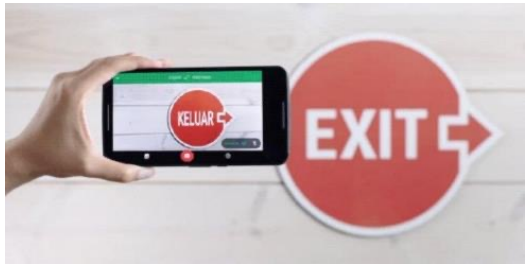
Decision making

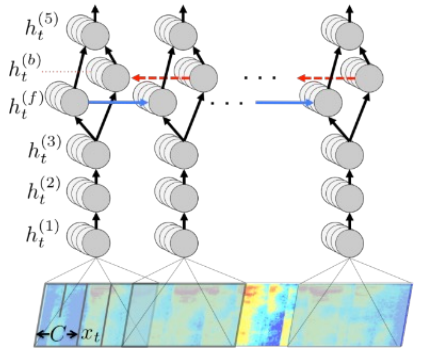
What makes modern machine learning work?









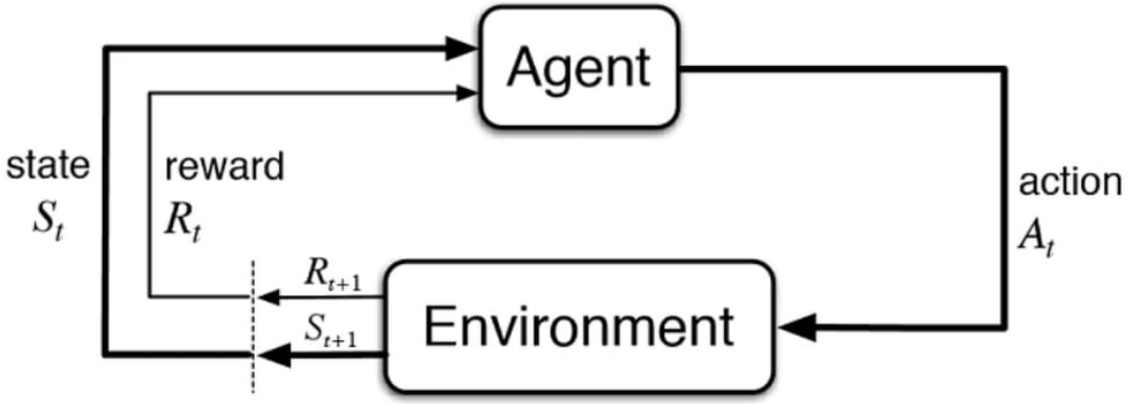


Huge amounts of data

+ large models / compute

+ right training recipe

What makes data-driven "decision-making" different?



A → B → C → …

(Decisions have long term consequences)

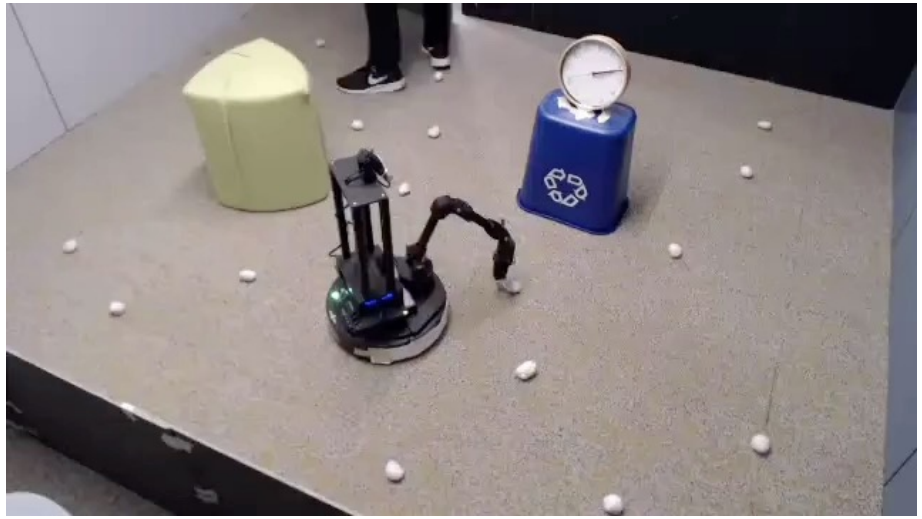
(Failure data may still be useful)

Need to "act" in the world sequentially, not just predict labels

Not all data may be "good" data

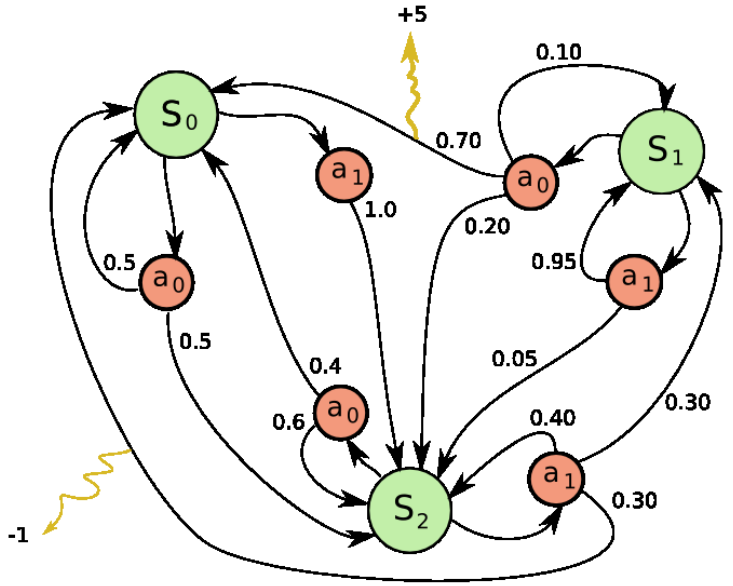
Why does this matter?

From the perspective of decision making



Allows bootstrapping from logged data

From the perspective of data-driven learning



Allows interactive, multi-step decision making

What will we talk about today?

Preliminaries

Imitation Learning

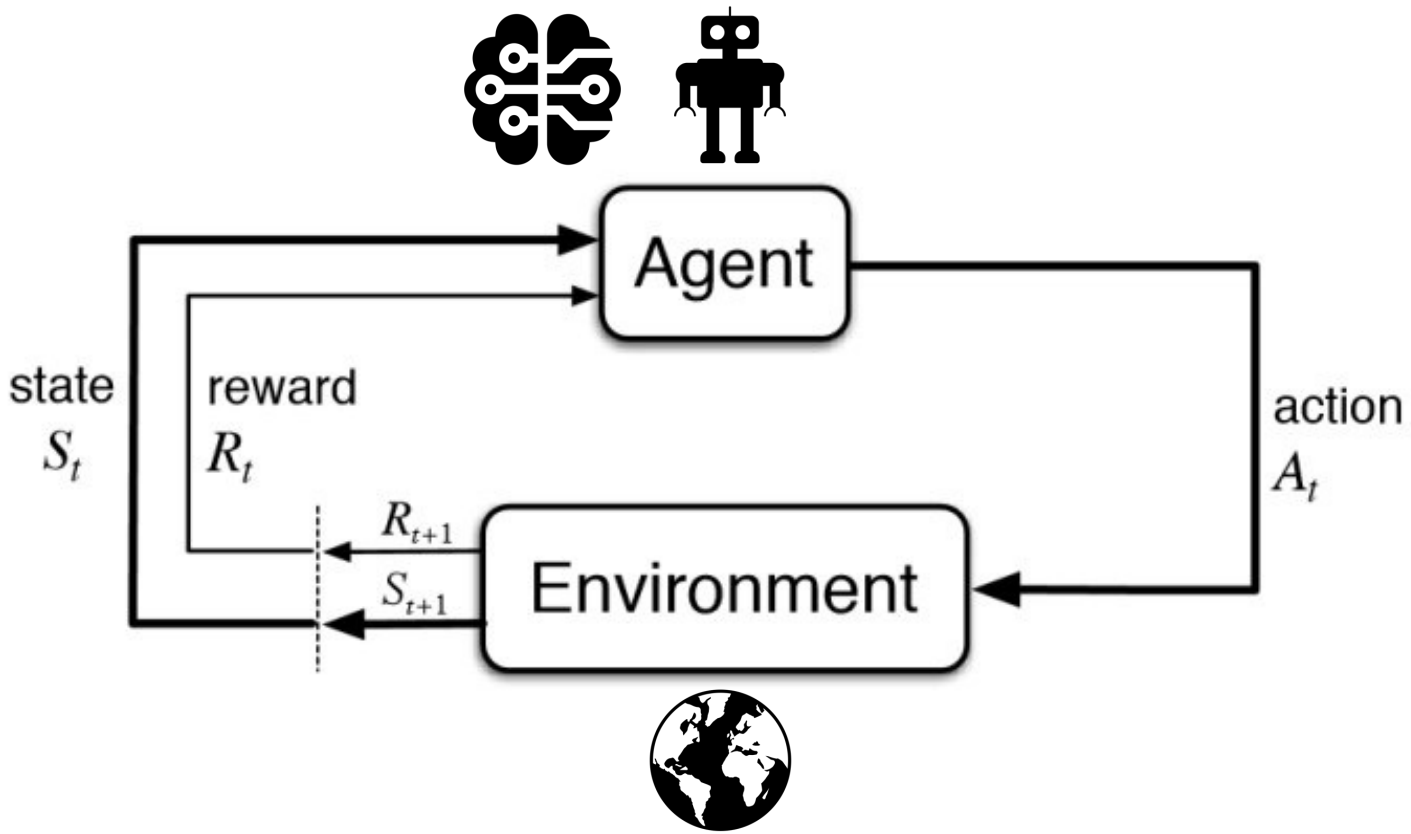
(Learning Behaviors from Expert Data)

Offline Reinforcement Learning

(Learning from Non-Expert data)

Let's think about sequential decision making

Observed states



Executed Actions

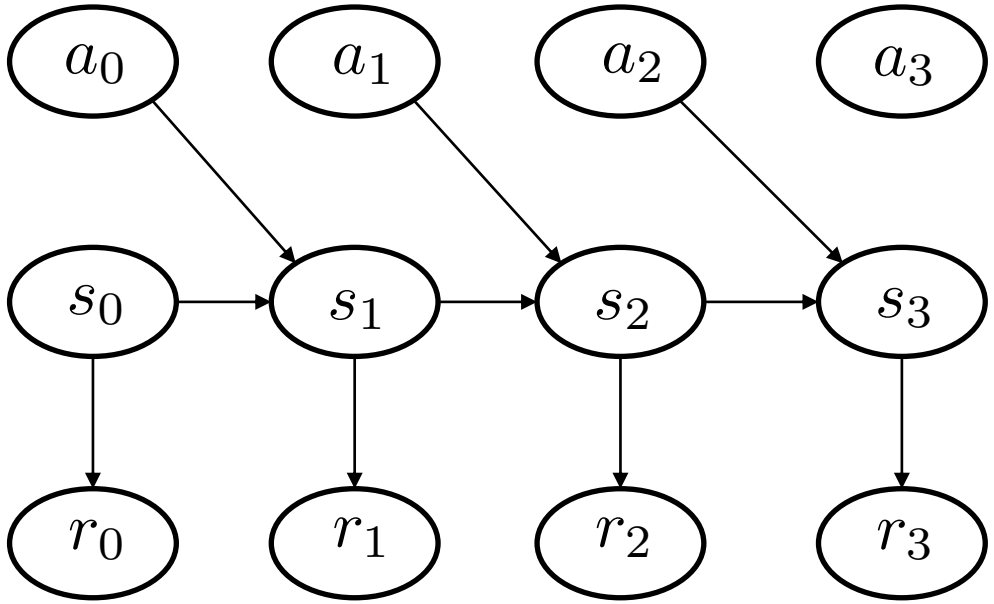
Different from 1-step prediction!

$$\max_{\theta} \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} r(s_t, a_t) \right]$$

Act to optimize a metric of **long-term** success

Framework for SDM - Markov Decision Process

Augment Markov chain with rewards and actions



States: $\mathcal{S}$

Initial state dist: $\rho_0(s)$

Actions: $\mathcal{A}$

Discount: γ

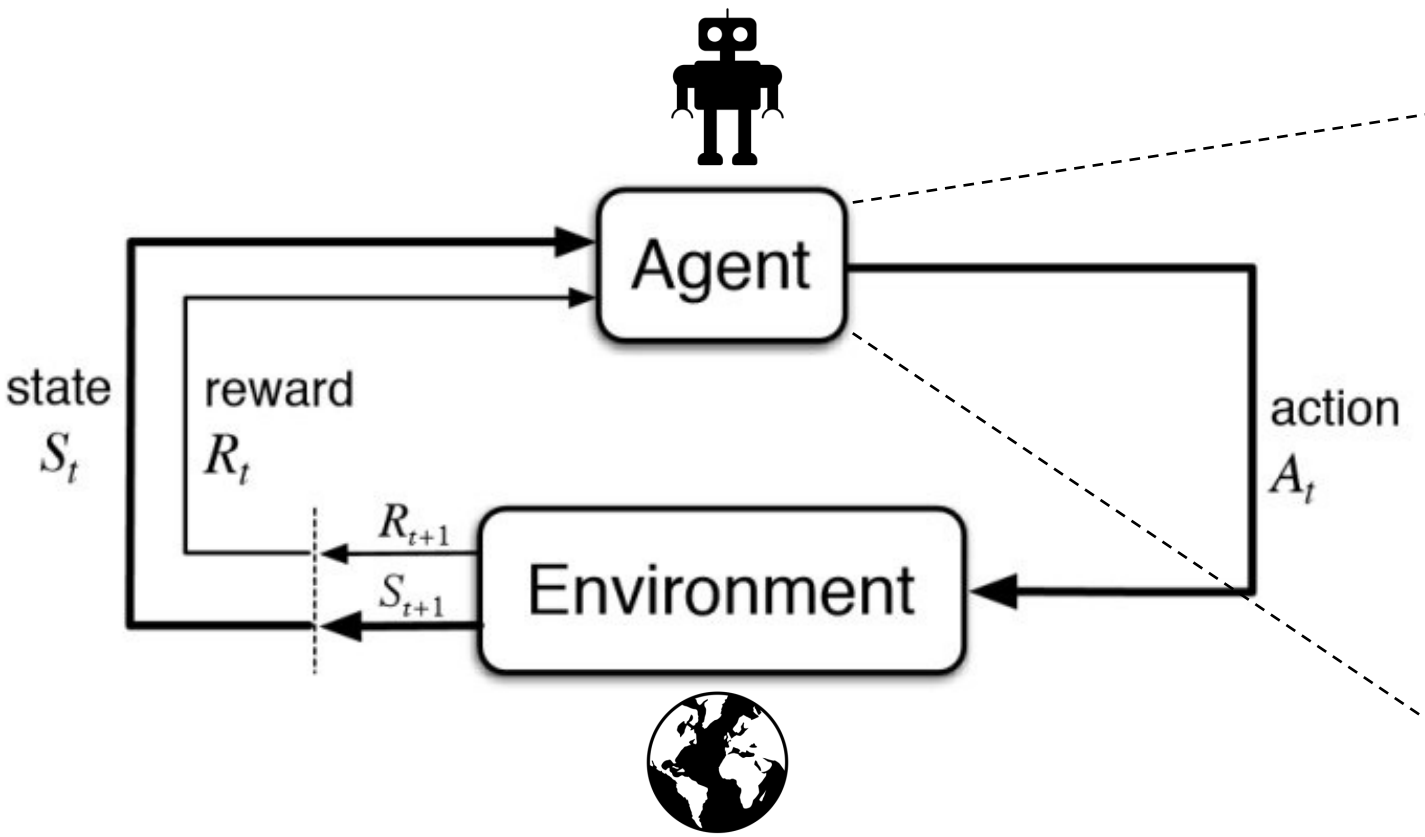
Rewards: $\mathcal{R}$

Transition Dynamics - $p(s_{t+1}|s_t, a_t)$

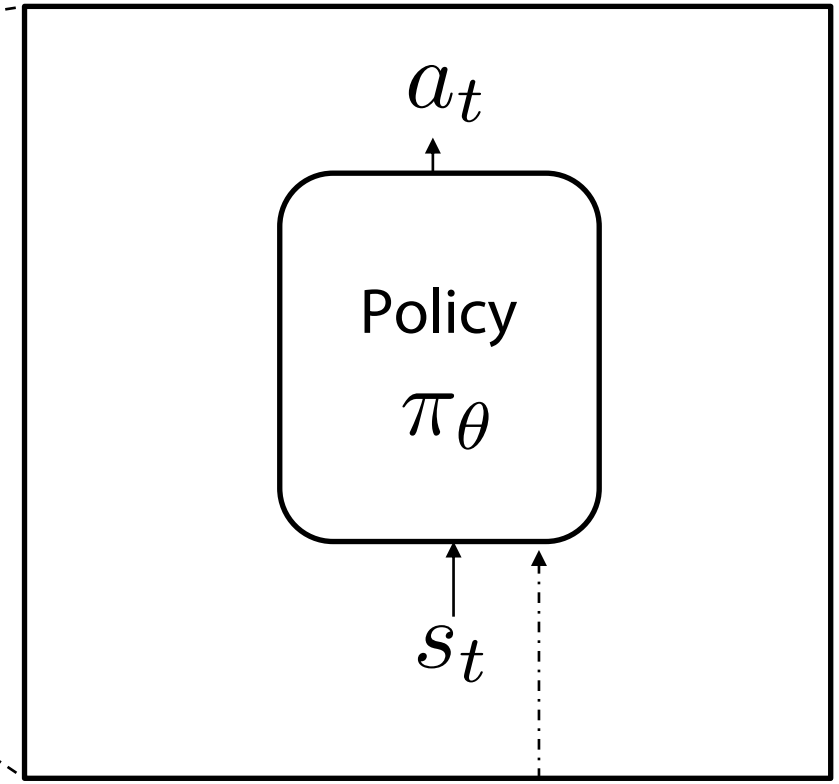
Markov property $p(s_0, s_1, s_2, a_0, a_1, a_2) = p(s_0)p(a_0|s_0)p(s_1|s_0, a_0)p(a_1|s_1)p(s_2|s_1, a_1)p(a_2|s_2)$

Trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T)$

What's the goal of sequential decision making?



Rules for choosing actions



Needs to be learned

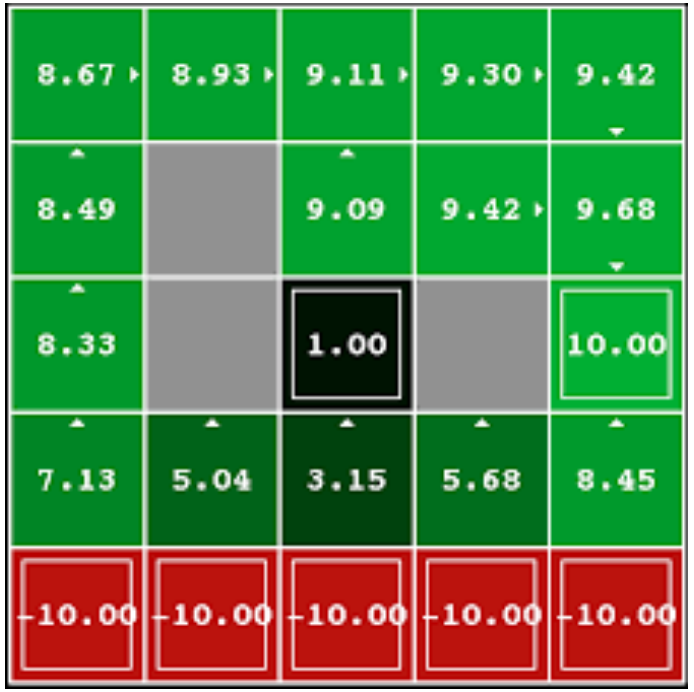
$$\max_\theta \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} r(s_t, a_t) \right]$$

Find the policy with best performance under sampling

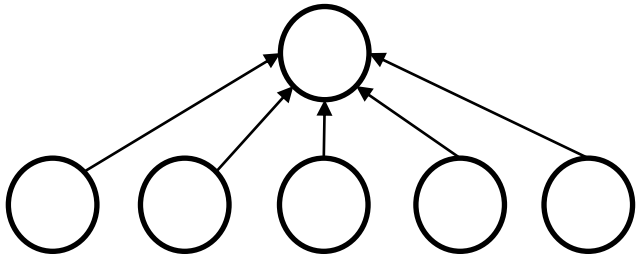
Main thing to learn - Policies

Policies are **mappings** from states to distributions over actions

Tabular

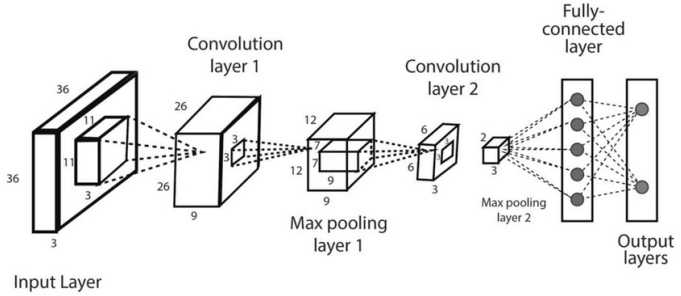


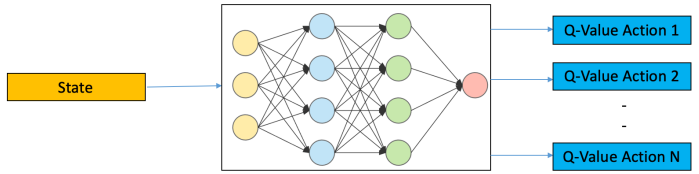
Linear



$$\pi(a|s) = \langle \phi(s,a), w \rangle$$

Arbitrary function approx



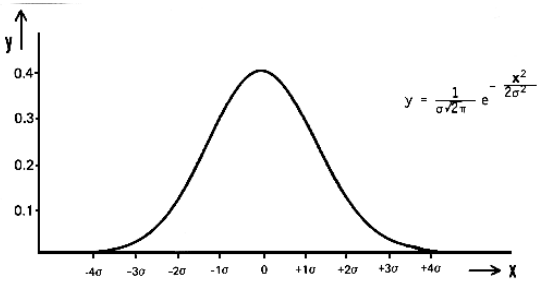


Deep Q Learning

Main thing to learn - Policies

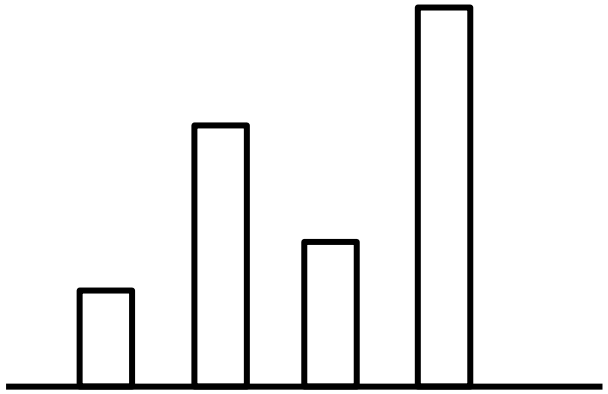
Policies are mappings from states to **distributions** over actions

Gaussian



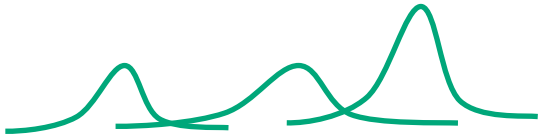
$\mu(s), \Sigma(s)$

Categorical



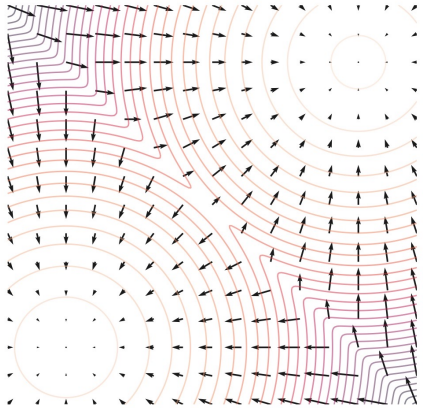
$p_1(s), p_2(s), \ldots, p_k(s)$

Mixture of Gaussians



$\mu_1(s), \Sigma_1(s), \mathrm{w}_1$
$\mu_2(s), \Sigma_2(s), \mathrm{w}_2$
...
$\mu_N(s), \Sigma_N(s), \mathrm{w}_\mathrm{N}$

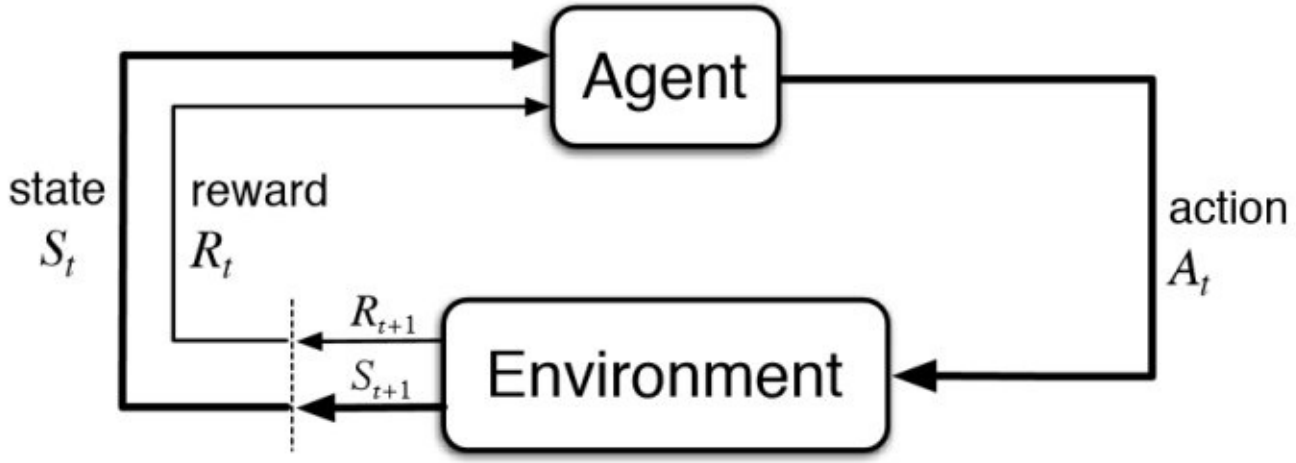
Diffusion Models



$\nabla_a \log \pi(a|s)$

Are we just going to study reinforcement learning?

Reinforcement learning is typically the problem of learning such SDM policies through trial and error



Learned through **online** interaction

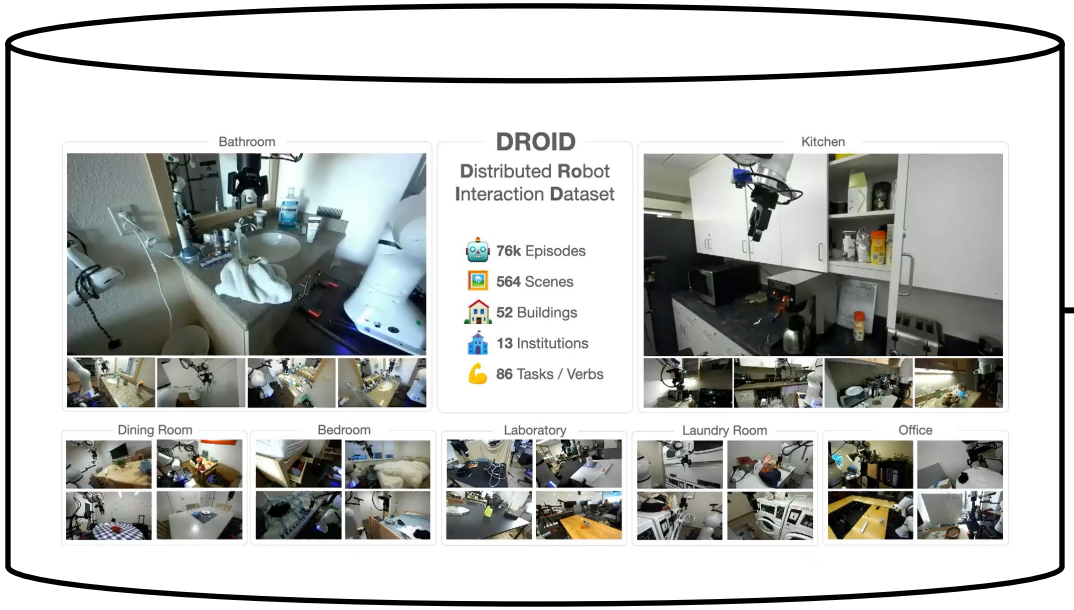
$$\max_{\theta} \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} r(s_t, a_t) \right]$$

Data-driven decision making

Data-driven decision making will learn such policies **offline**, from logged data

Same goal of policy learning,
but different solution

$$\max_\theta \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} r(s_t, a_t) \right]$$



Data-driven decision making algorithms

???

Policy π_θ

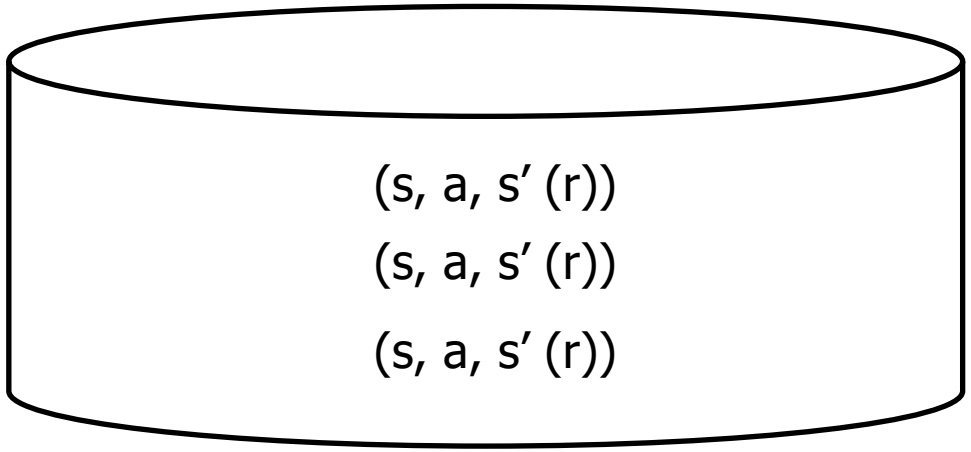
a_t

s_t

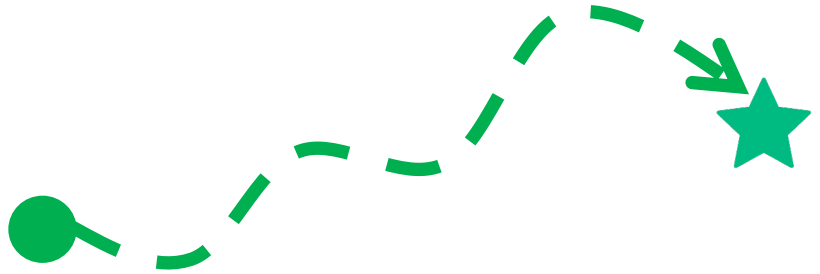
Formalism for data-driven decision making

We will study how to develop algorithms that go from logged data to a sequential decision making policy

Given: Dataset of prior interactions



Goal: Find performant policy π_θ



Case 1: Data is near-optimal (Imitation Learning)

Case 2: Data is not necessarily optimal (Offline RL)

What will we talk about today?

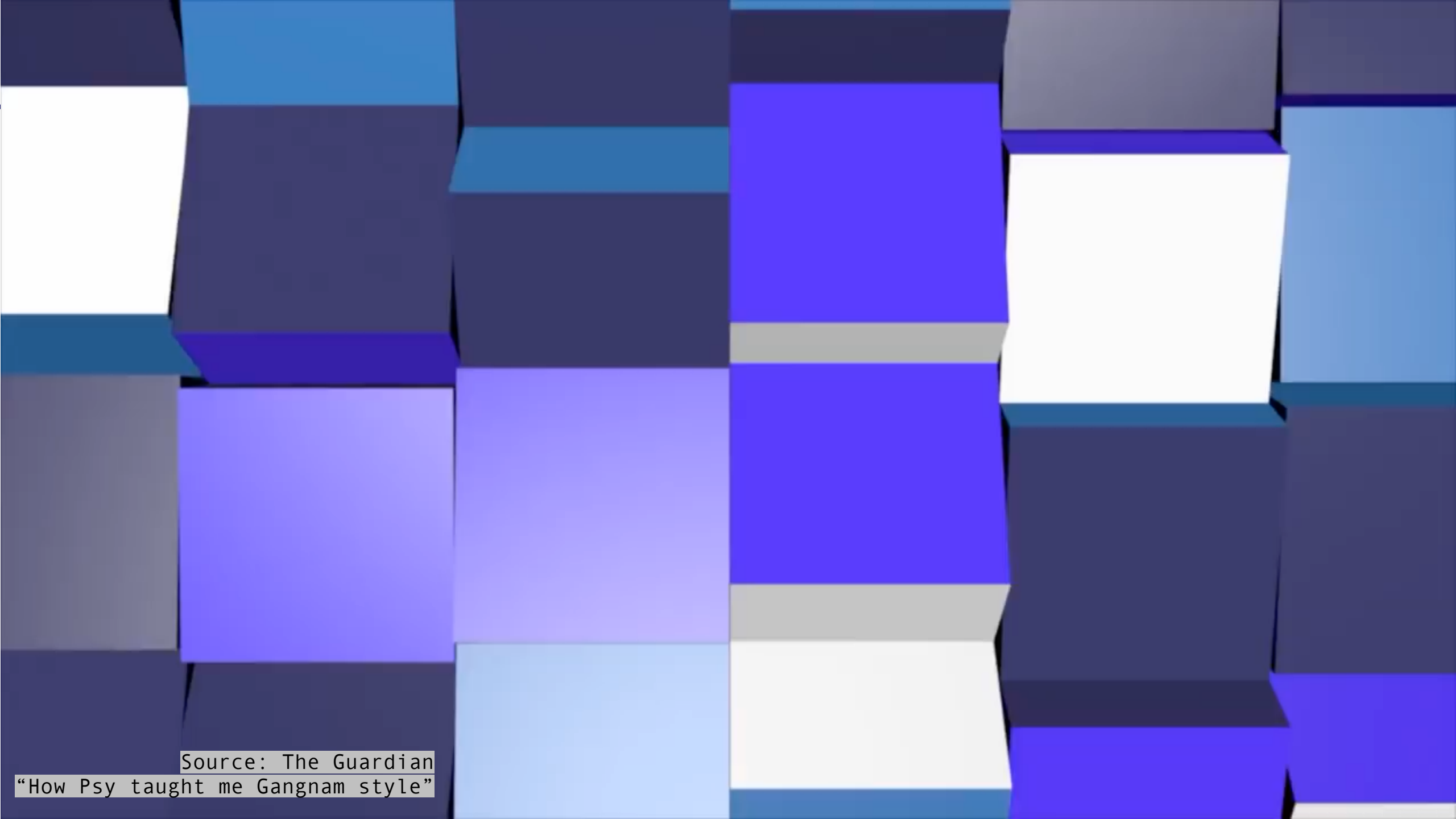
Preliminaries

Imitation Learning

(Learning Behaviors from Expert Data)

Offline Reinforcement Learning

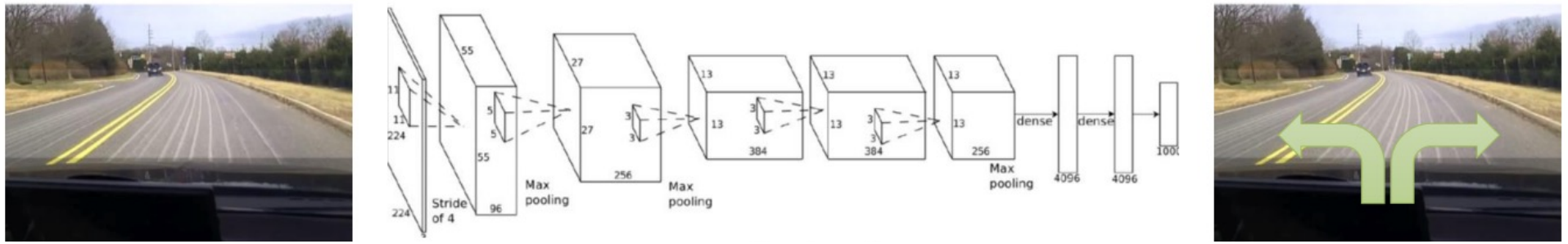
(Learning from Non-Expert data)


Source: The Guardian
“How Psy taught me Gangnam style”

Imitation Learning: Problem Statement

Given: Demonstrations of **optimal** behavior $\mathcal{D} = \{s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_T^i, a_T^i\}_{i=1}^N$

Goal: Train a policy to mimic the demonstrator $\pi_\theta(a|s)$

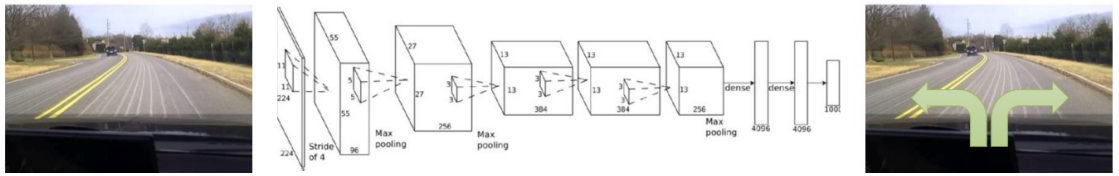


Pros: No rewards, online experience needed (?)

Why would we do this?

Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator



Pros:

- ⊕ Avoids need for rewards, exploration
- ⊕ Natural way to do task specification
- ⊕ Can work well in practice

Cons:

- ⊖ Requires expert data, can be expensive
- ⊖ Cannot get better on deployment
- ⊖ Struggles on long horizon tasks

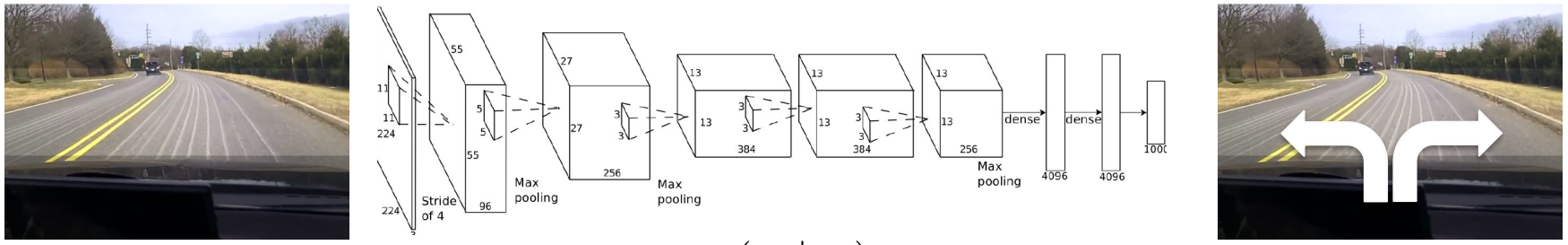
Idea 1: Imitation Learning via Supervised Learning

Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator

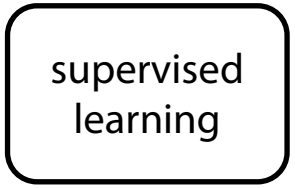
$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

Idea: Treat imitation learning as a supervised learning problem! → Behavior Cloning



$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

$\mathbf{o}_t$
$\mathbf{a}_t$



$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

Idea 1: Imitation Learning via Supervised Learning

Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

Discrete vs continuous

Maximum likelihood

```
if isinstance(env.action_space, gym.spaces.Box):
    criterion = nn.MSELoss()
else:
    criterion = nn.CrossEntropyLoss()
# Extract initial policy
model = student.policy.to(device)
def train(model, device, train_loader, optimizer):
   model.train()
   for batch_idx, (data, target) in enumerate(train_loader):
      data, target = data.to(device), target.to(device)
      optimizer.zero_grad()
      if isinstance(env.action_space, gym.spaces.Box):
         if isinstance(student, (A2C, PPO)):
            action, _, _ = model(data)
         else:
            action = model(data)
         action_prediction = action.double()
      else:
         dist = model.get_distribution(data)
         action_prediction = dist.distribution.logits
         target = target.long()
      loss = criterion(action_prediction, target)
      loss.backward()
      optimizer.step()
```

Idea 1: Imitation Learning via Supervised Learning

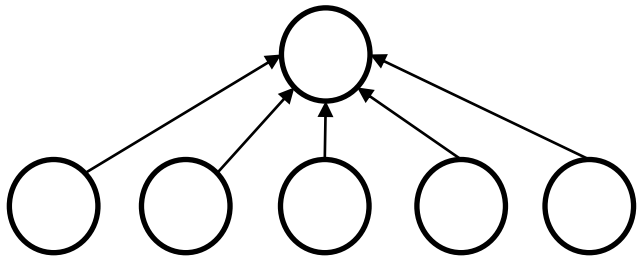
Given: Demonstrations of optimal behavior

Goal: Train a policy to mimic the demonstrator

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

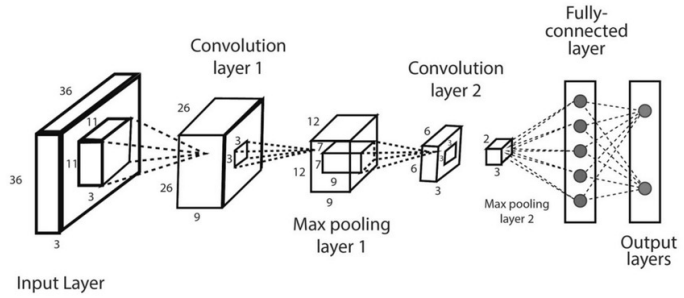
Tabular

Linear



$$\pi(a|s) = \langle \phi(s,a), w \rangle$$

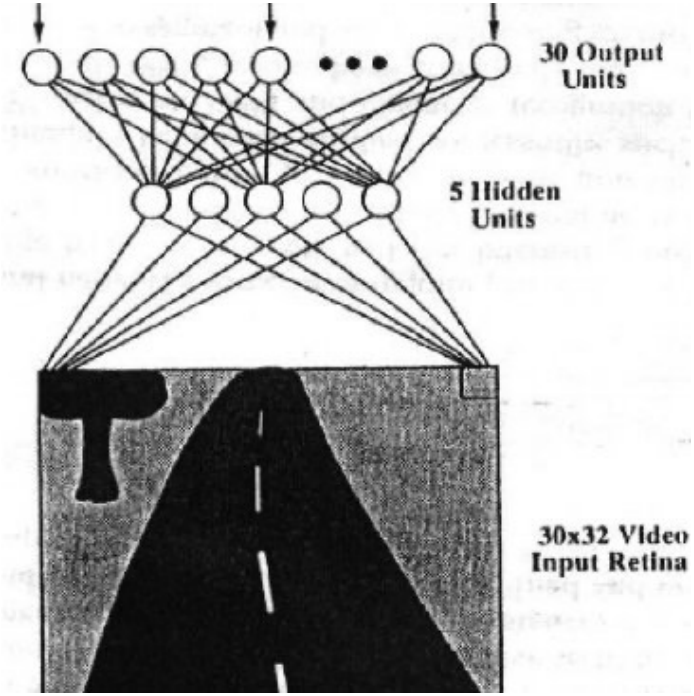
Arbitrary function approx



In practice, amounts to simple gradient based training with backpropagation

The original deep imitation learning system

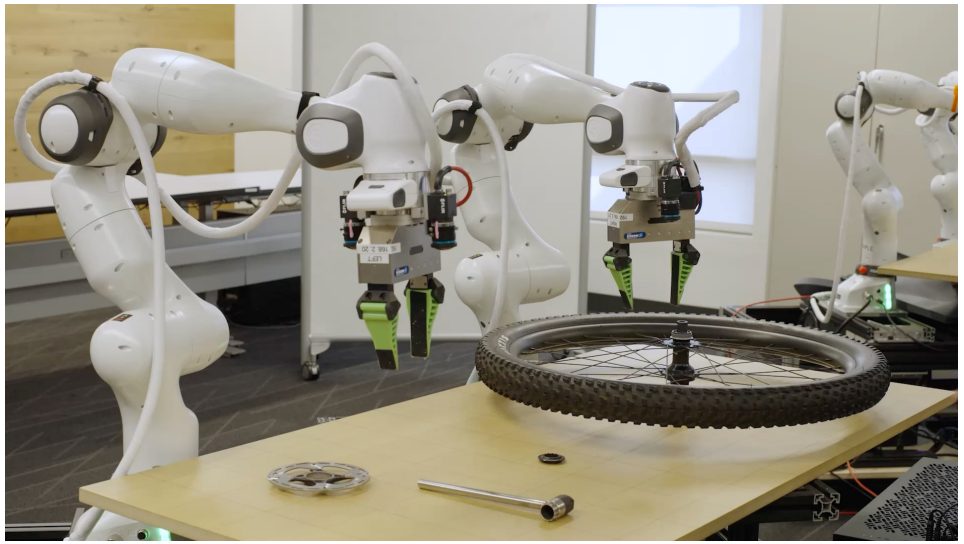
ALVINN: **A**utonomous **L**and **V**ehicle **I**n a **N**eural **N**etwork
1989

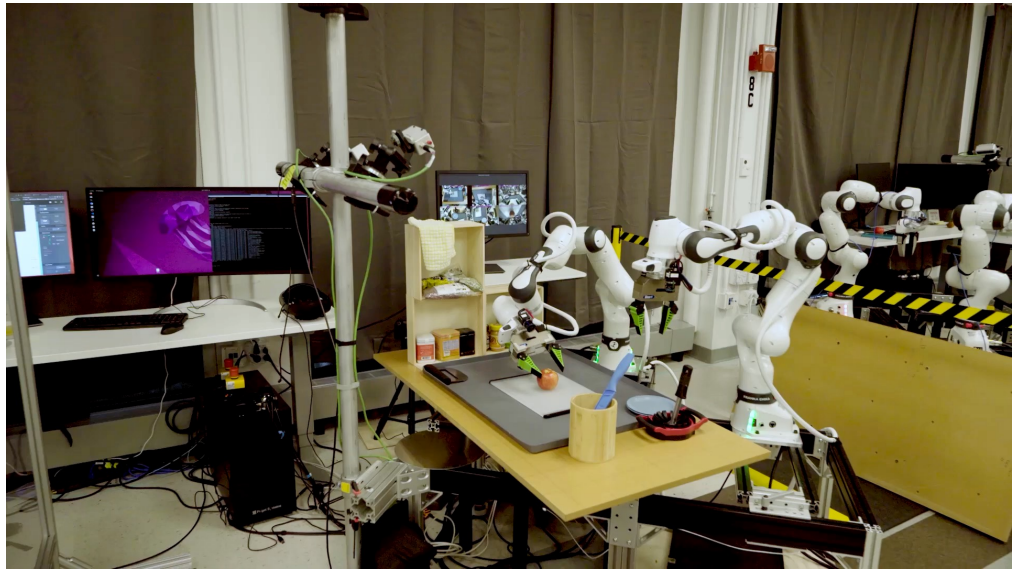


Where we are in 2025?

Cheng et al ’24

Where we are in 2025?



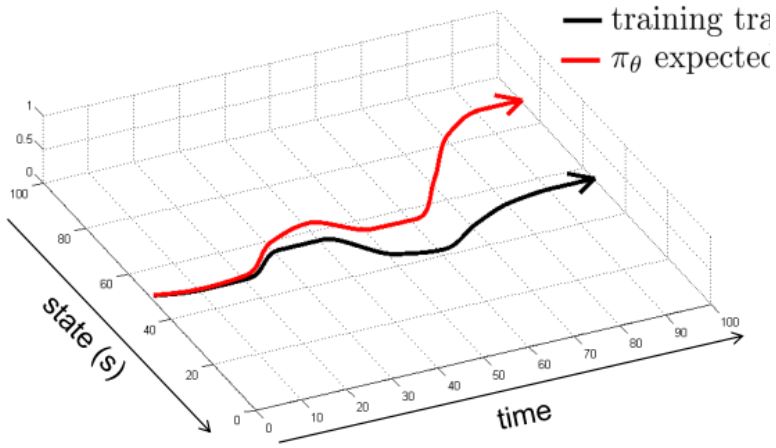


TRI et al ‘25

Treat Imitation as Supervised Learning

So does behavior cloning really work?

- Imitation Learning ≠ Supervised Learning



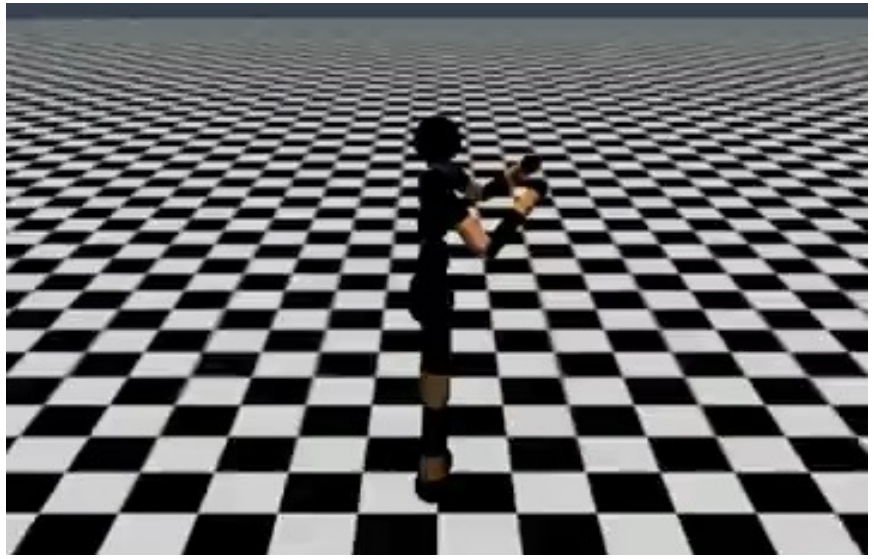
Compounding error!

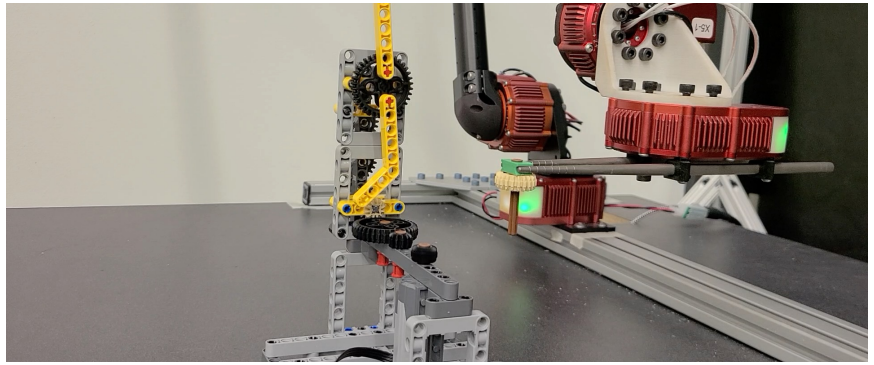
$$\arg\max_\theta \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}}\left[\log \pi_\theta(a^*|s^*)\right] \qquad \mathbb{E}_{(s,a)\sim\rho(\pi)}\left[1(a=a^*)\right]$$

Not the same!

So does behavior cloning really work?

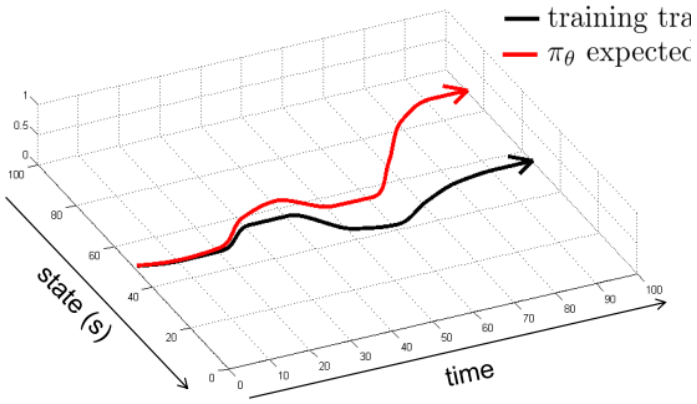
- Fails in practice as well!





How well does BC do?: Intuition

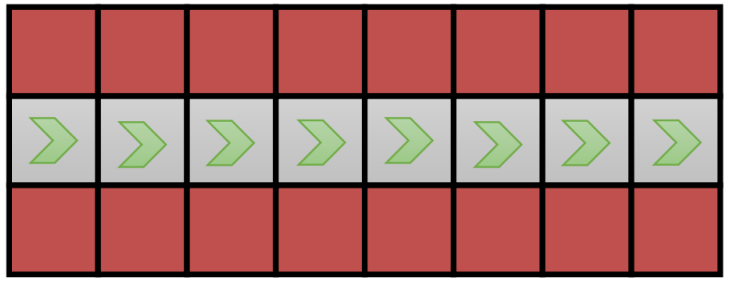
Behavior cloning has quadratically compounding error



$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

Horizon H

If you fall off, assume the worst



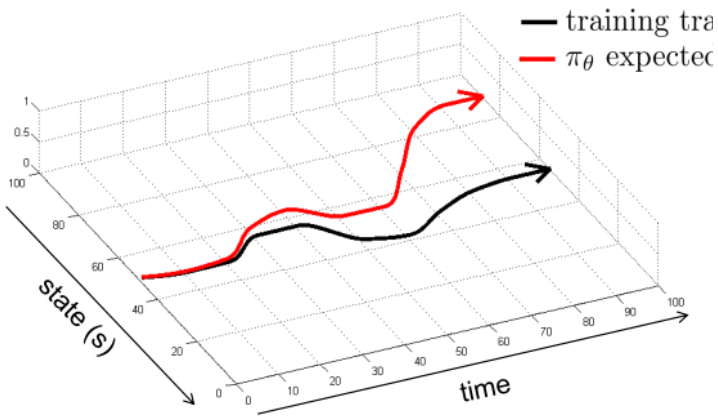
$$\underbrace{\mathbb{E}\left[\sum_t c(s_t, a_t)\right]}_{O(\epsilon H^2)} \leq \epsilon H + \cdots + \ldots$$

Union bound

Let's try and understand where the problem lies?

Behavior cloning has challenges in both theory and practice

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$



Underfitting

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

Compounding error

$$\leq O(\epsilon H^2)$$

What will we talk about today?

Imitation Learning and Inverse Reinforcement Learning

(Learning Behaviors from ***Expert*** Data)

- Multimodality and Underfitting in Imitation
- Compounding Error in Imitation

What will we talk about today?

Imitation Learning and Inverse Reinforcement Learning

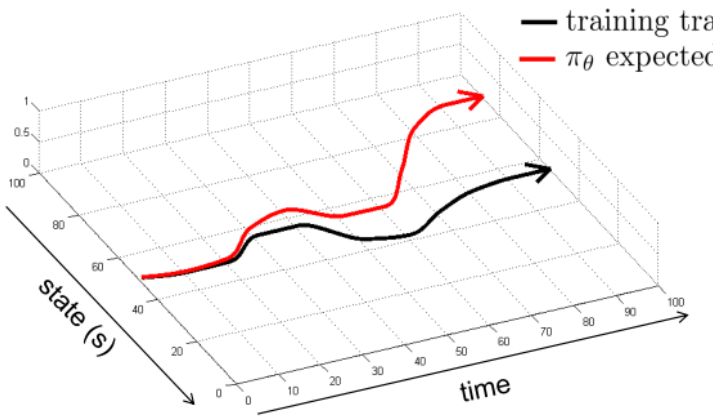
(Learning Behaviors from ***Expert*** Data)

- **Multimodality and Underfitting in Imitation**
- Compounding Error in Imitation

Let's try and understand where the problem lies?

Behavior cloning has challenges in both theory and practice

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$



Underfitting

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

Compounding error

$$\leq O(\epsilon H^2)$$

But won't a bigger neural net just solve this?

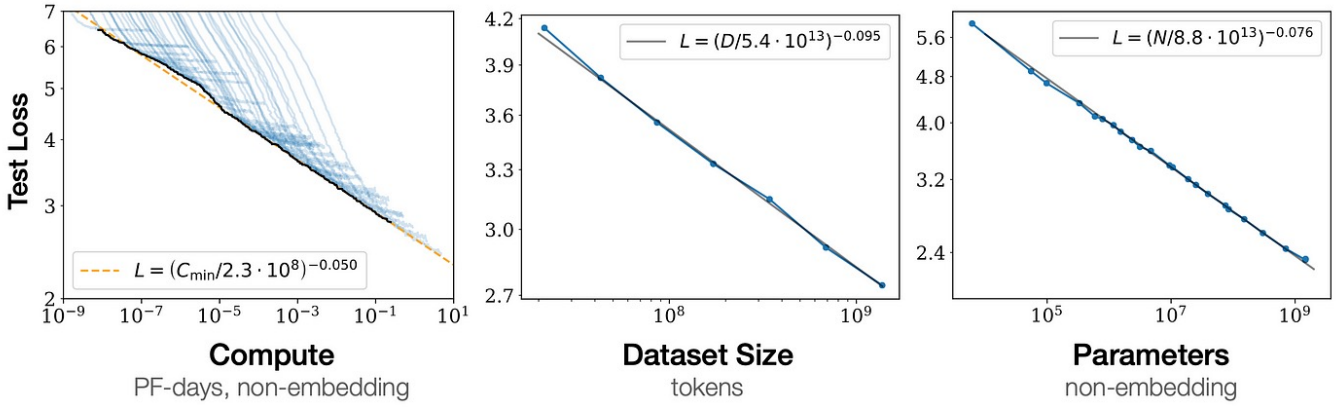
- Behavior cloning can underfit the data

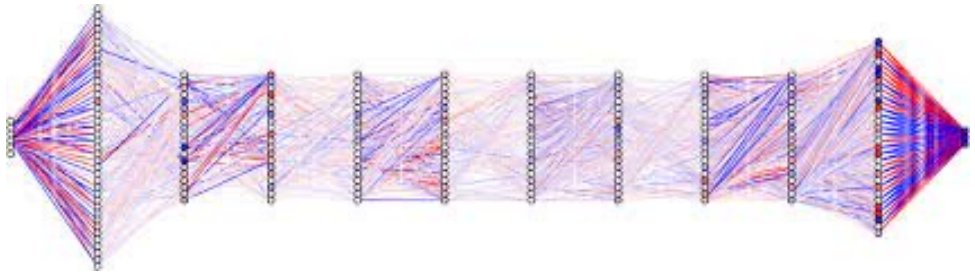
$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$
$$\text{for } s_t \sim p_{\text{train}}(s_t)$$

May not be able to satisfy this

Q: won't a bigger model just solve the problem?



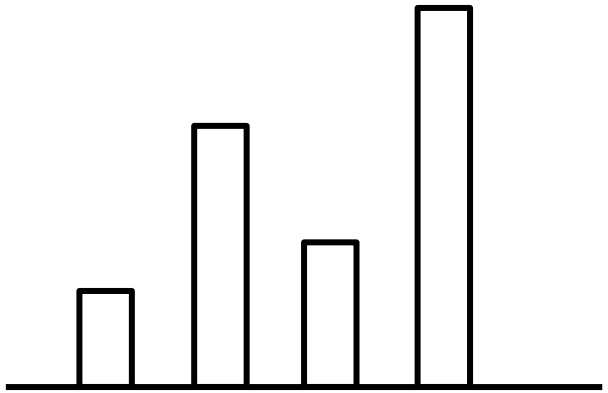


Kind of, but there's a fundamental problem!

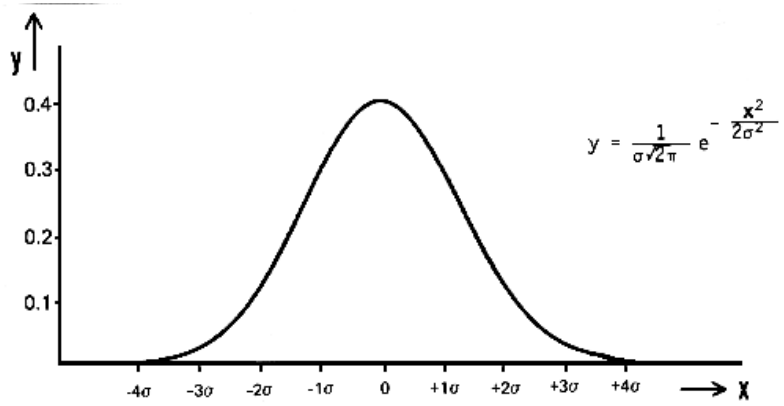
Distributional Expressivity

- Policy expressivity is a combination of expressivity of the function approximator and of the distribution family

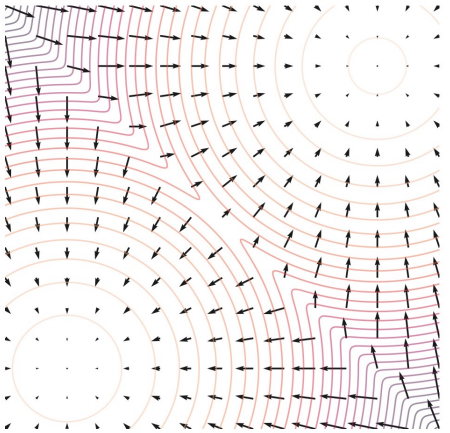
Categorical



Gaussian



Diffusion policy

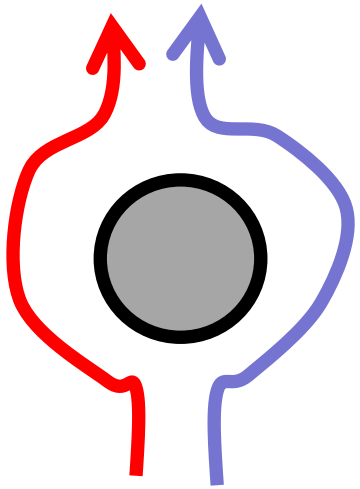


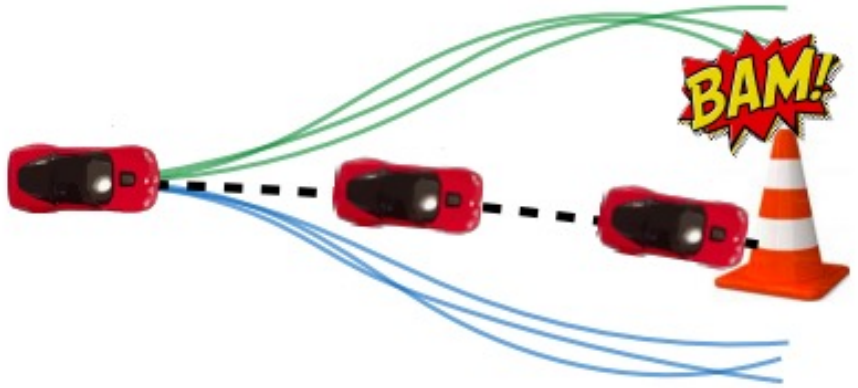
Tradeoff between expressivity and tractability

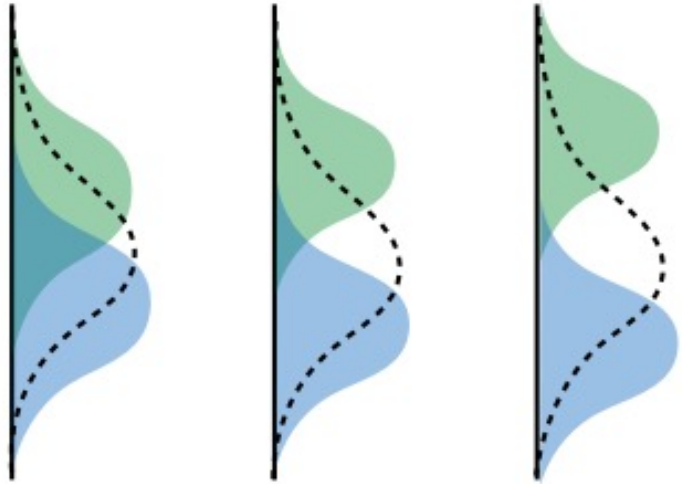
How does this reflect on imitation learning?

Let us consider a case with Gaussian policy

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$





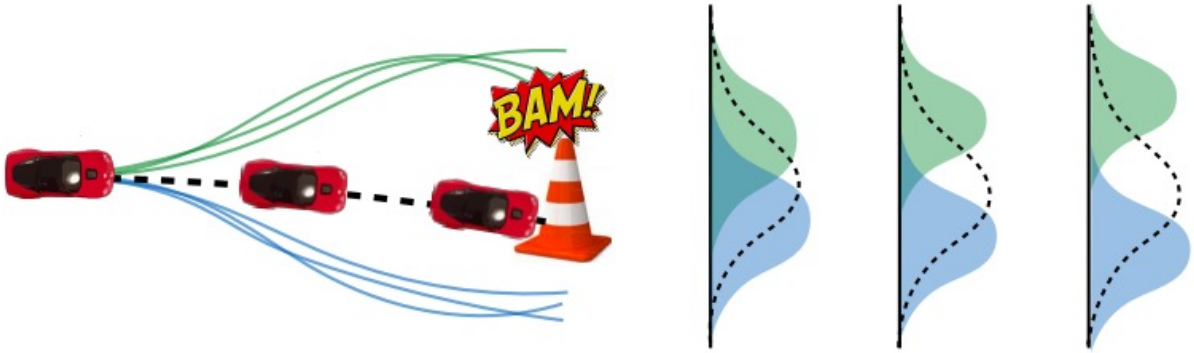


A combination of **distributional expressivity** and **objective** lead to mode averaging

What does this mean?

Let us consider a Gaussian policy

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

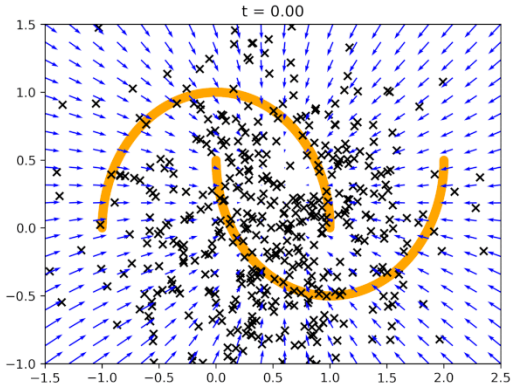


Any sample in the data that is missed will be penalized → cover modes

Option 1: Change objective

$$\arg\max_{\theta} \mathbb{E}_{\pi_\theta(a|s)} \left[\log \frac{\pi_e(a|s)}{\pi_\theta(a|s)}\right]$$

Option 2: Change away from Gaussian



What kind of objective may be useful here?

Different divergences lead to different properties

$$\arg\max_\theta \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

Forward KL $\mathbb{E}_{s^*\sim p_{\pi_e}(.)} \left[D_{\mathrm{KL}}(\pi_e(.|s^*)||\pi_\theta(.|s^*))\right]$

(Doesn't want to drop modes – mode covering)

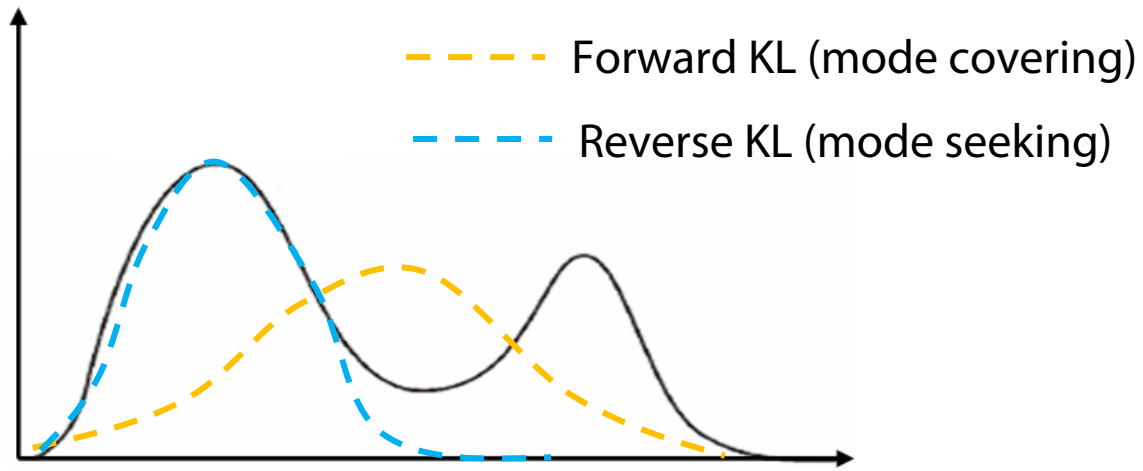
$$\arg\max_\theta \mathbb{E}_{\pi_\theta(a|s)} \left[\log \frac{\pi_e(a|s)}{\pi_\theta(a|s)}\right]$$

Reverse KL $\mathbb{E}_{s^*\sim p_{\pi_e}(.)} \left[D_{KL}(\pi_\theta(.|s^*)||\pi_e(.|s^*))\right]$

(Picks one mode – mode seeking)

Need to sample

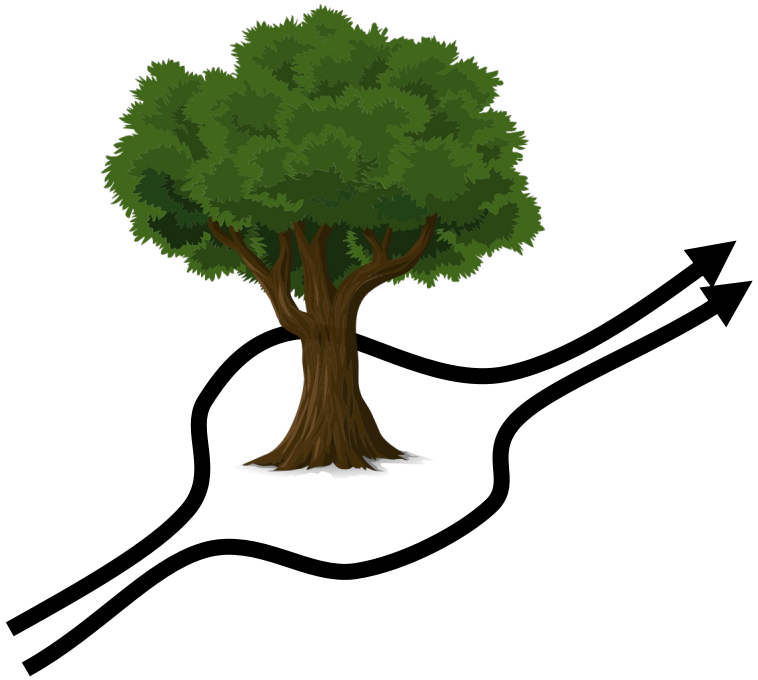
Need expert likelihood

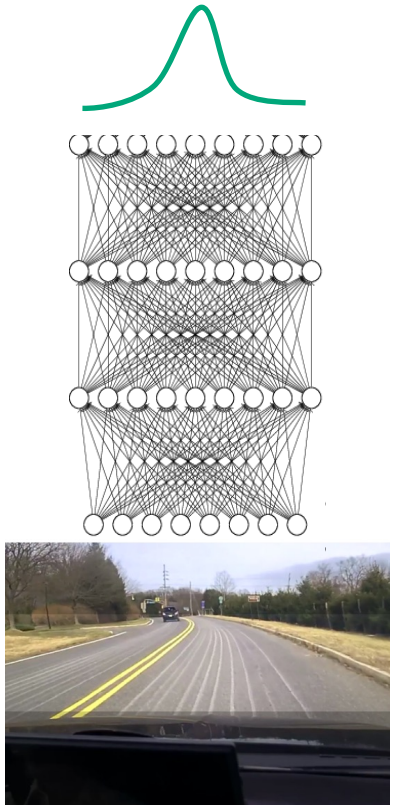


Eh, it's too complicated

Using Richer Policy Distribution Classes

Multimodal behavior → use more **expressive** probability distributions, no mode averaging issues



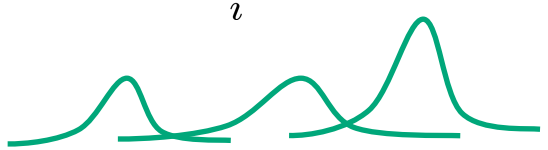


1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …

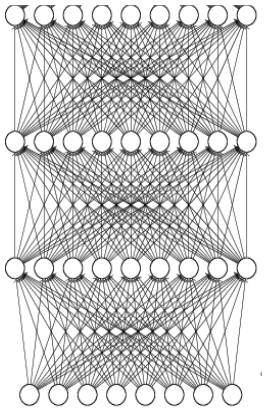
Why might we fail to fit the expert?

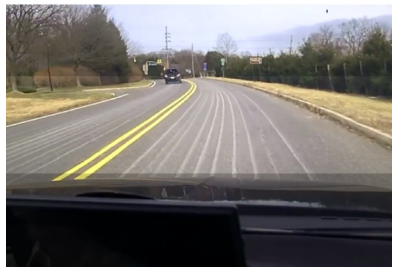
1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …

$$\pi(\mathbf{a}|\mathbf{o}) = \sum_i w_i \mathcal{N}(\mu_i, \Sigma_i)$$



$$w_1, \mu_1, \Sigma_1, \ldots, w_N, \mu_N, \sigma_N$$

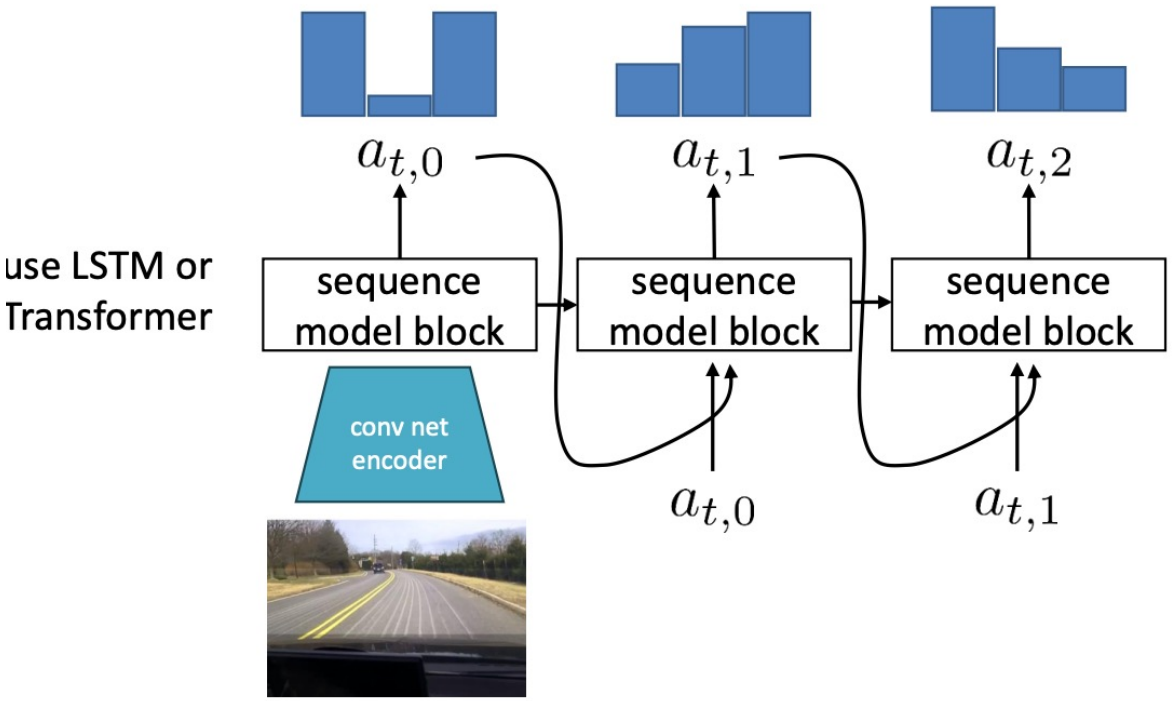




Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …

$$\mathbf{a}_t = \begin{pmatrix} 0.1 \\ 1.2 \\ -0.3 \end{pmatrix} \begin{matrix} a_{t,0} \\ a_{t,1} \\ a_{t,2} \end{matrix}$$



Why does this work?

$$\begin{aligned} \text{first step: } & p(a_{t,0}|\mathbf{s}_t) \\ \text{second step: } & p(a_{t,1}|\mathbf{s}_t, a_{t,0}) \\ \text{third step: } & p(a_{t,2}|\mathbf{s}_t, a_{t,0}, a_{t,1}) \end{aligned}$$

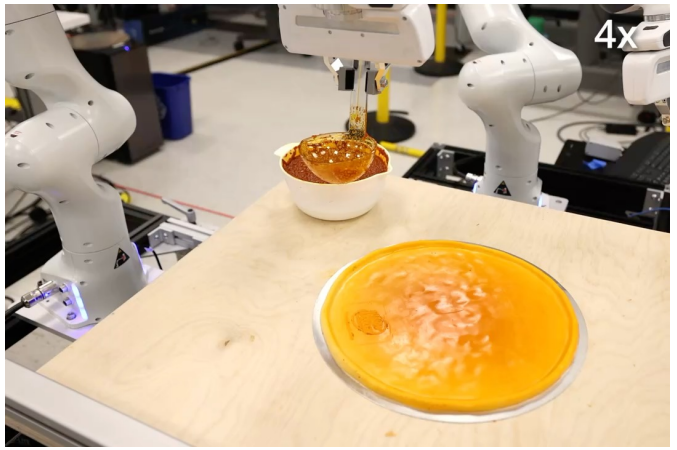
$$\begin{aligned} & p(a_{t,2}|\mathbf{s}_t, a_{t,0}, a_{t,1})p(a_{t,1}|\mathbf{s}_t, a_{t,0})p(a_{t,0}|\mathbf{s}_t) \\ &= p(a_{t,0}, a_{t,1}, a_{t,2}|\mathbf{s}_t) \\ &= p(\mathbf{a}_t|\mathbf{s}_t) \end{aligned}$$

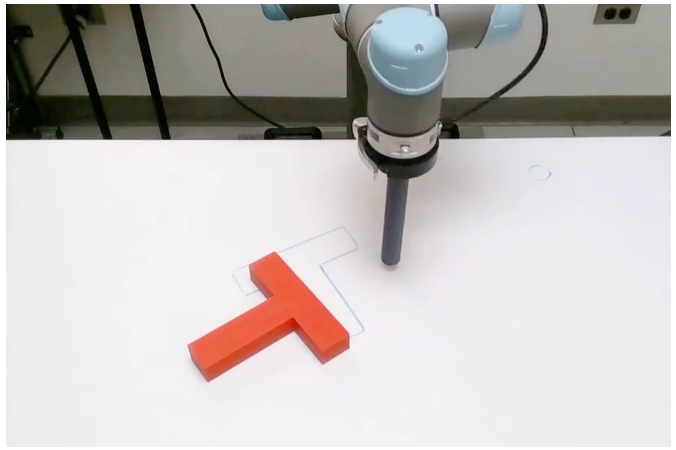
Examples of Autoregressive Policies

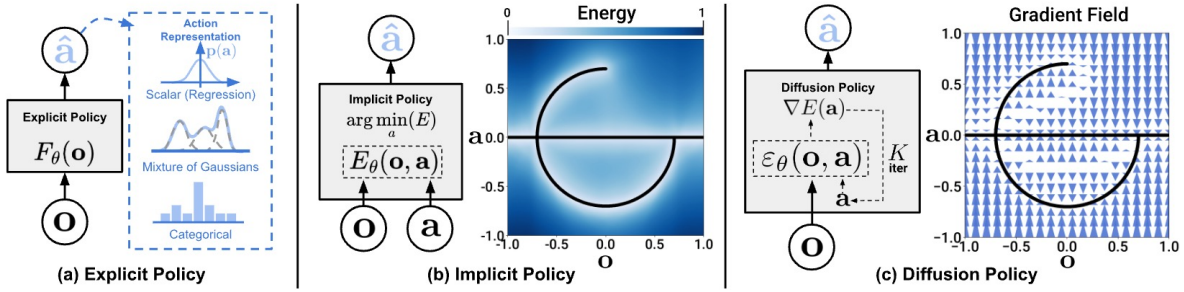
RT-2, Google Deepmind

Why might we fail to fit the expert?

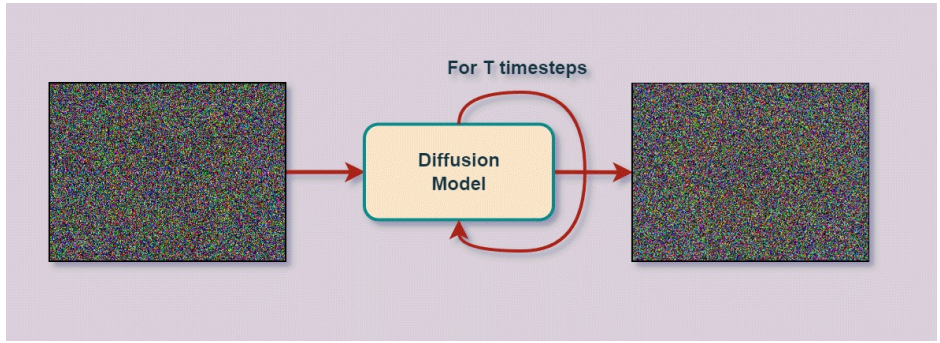
1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. …







How do diffusion policies work?



States

Algorithm 1 Conditional DDPM Training

Require: Dataset $\mathcal{D} = \{(x_0, c)\}$, diffusion steps T, noise schedule $\{\beta_t\}_{t=1}^{T}$, model $\epsilon_\theta(x_t, t, c)$

1: **for all** $(x_0, c) \in \mathcal{D}$ **do**
2: Sample $t \sim \text{Uniform}(\{1, \ldots, T\})$
3: Sample $\epsilon \sim \mathcal{N}(0, \mathbf{I})$
4: Compute $\bar{\alpha}_t = \prod_{s=1}^{t}(1 - \beta_s)$
5: Compute $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$
6: Predict noise: $\hat{\epsilon} = \epsilon_\theta(x_t, t, c)$
7: Compute loss: $\mathcal{L} = \|\epsilon - \hat{\epsilon}\|^2$
8: Update θ via gradient descent on $\mathcal{L}$

Actions

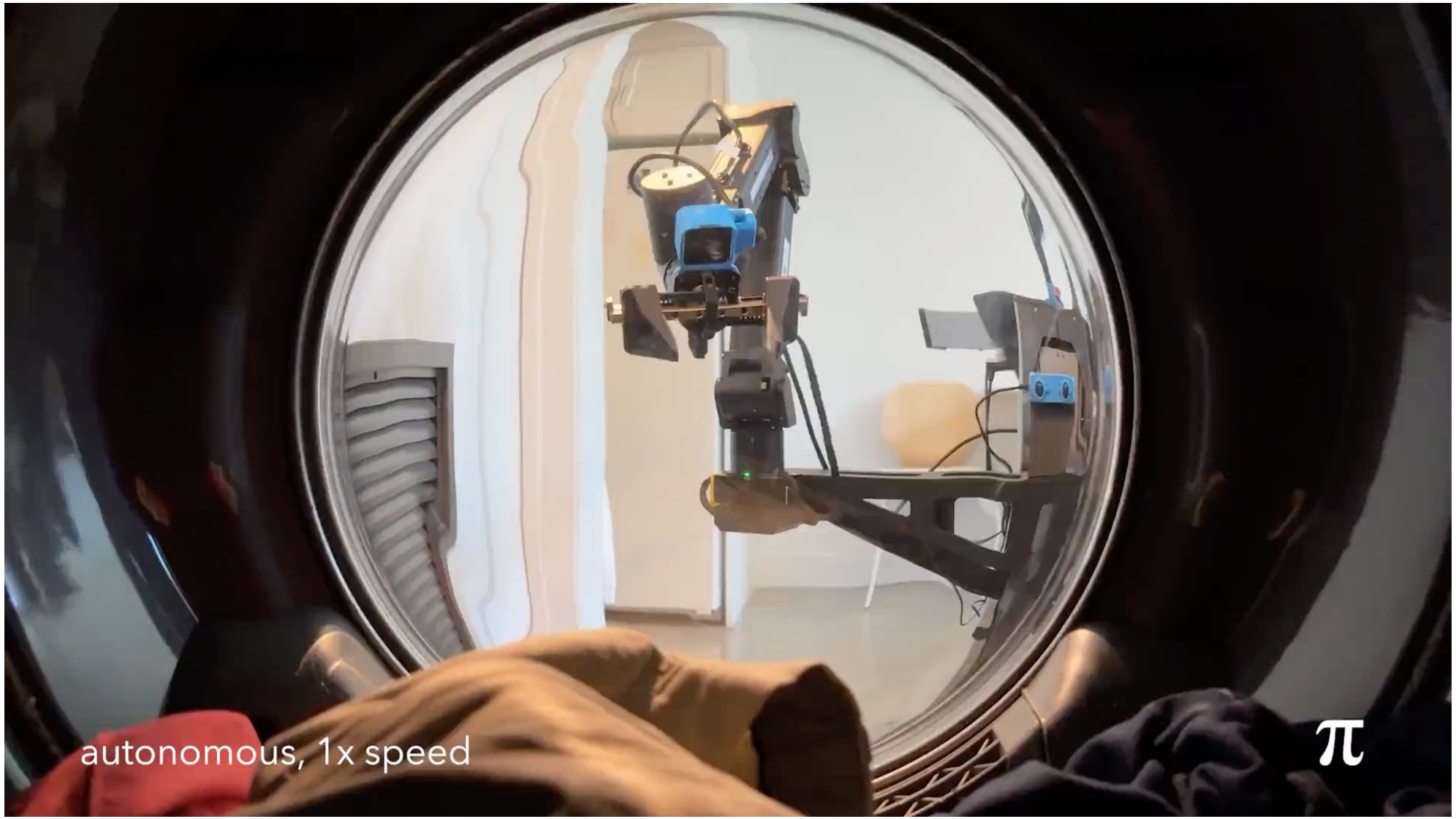
Noisy actions

Create noisy sample

Predict the noise

Do repeated denoising during inference to produce actions

Examples of Diffusion-Based Policies



Use rich generative models (from vision/language) for “generative” imitation

What will we talk about today?

Imitation Learning and Inverse Reinforcement Learning

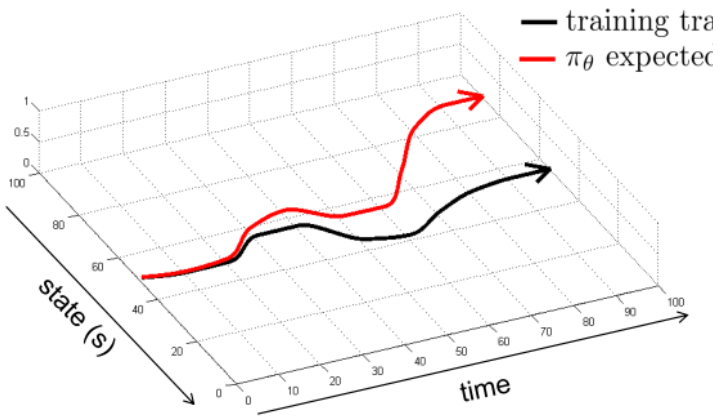
(Learning Behaviors from ***Expert*** Data)

- Multimodality and Underfitting in Imitation
- **Compounding Error in Imitation**

Let's try and understand where the problem lies?

Behavior cloning has challenges in both theory and practice

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim p_{\pi_\theta}(s_t,a_t)}\left[c(s_t,a_t)\right] \leq O(\epsilon H^2)$$



Underfitting

$$\pi_\theta(a \neq \pi^*(s_t)|s_t) \leq \epsilon$$

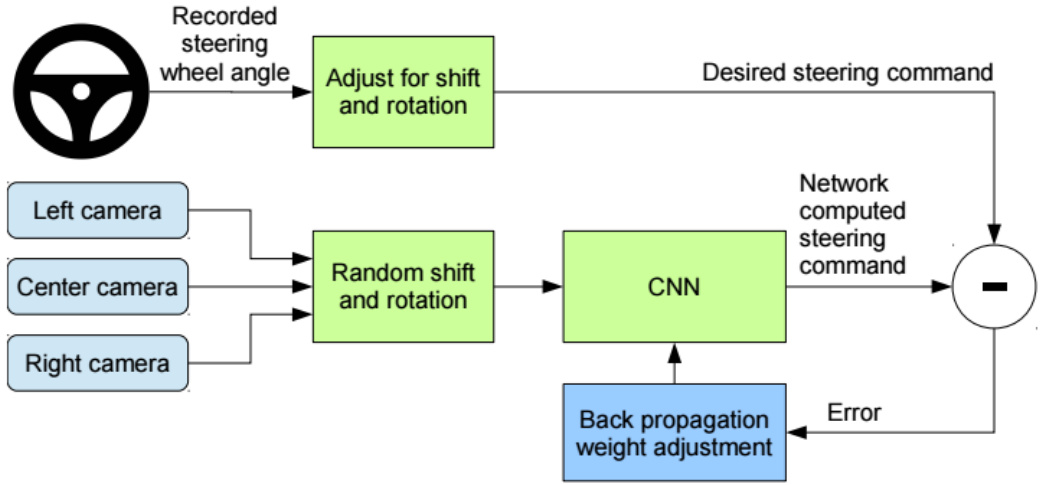
Compounding error

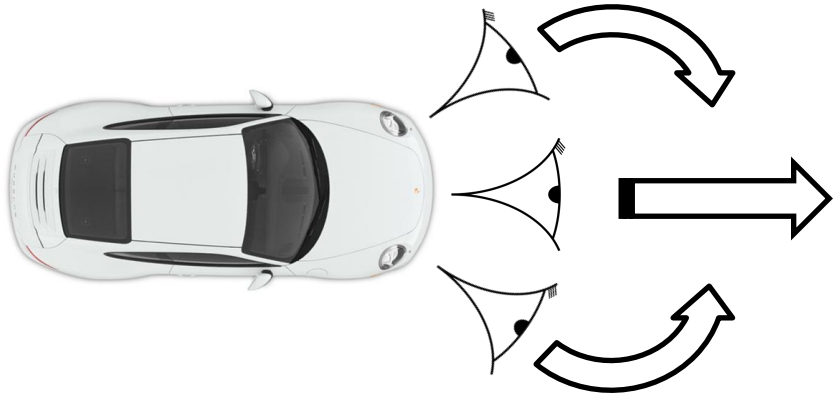
$$\leq O(\epsilon H^2)$$

Can we avoid compounding error in special cases?

Video: Bojarski et al. ‘16, NVIDIA

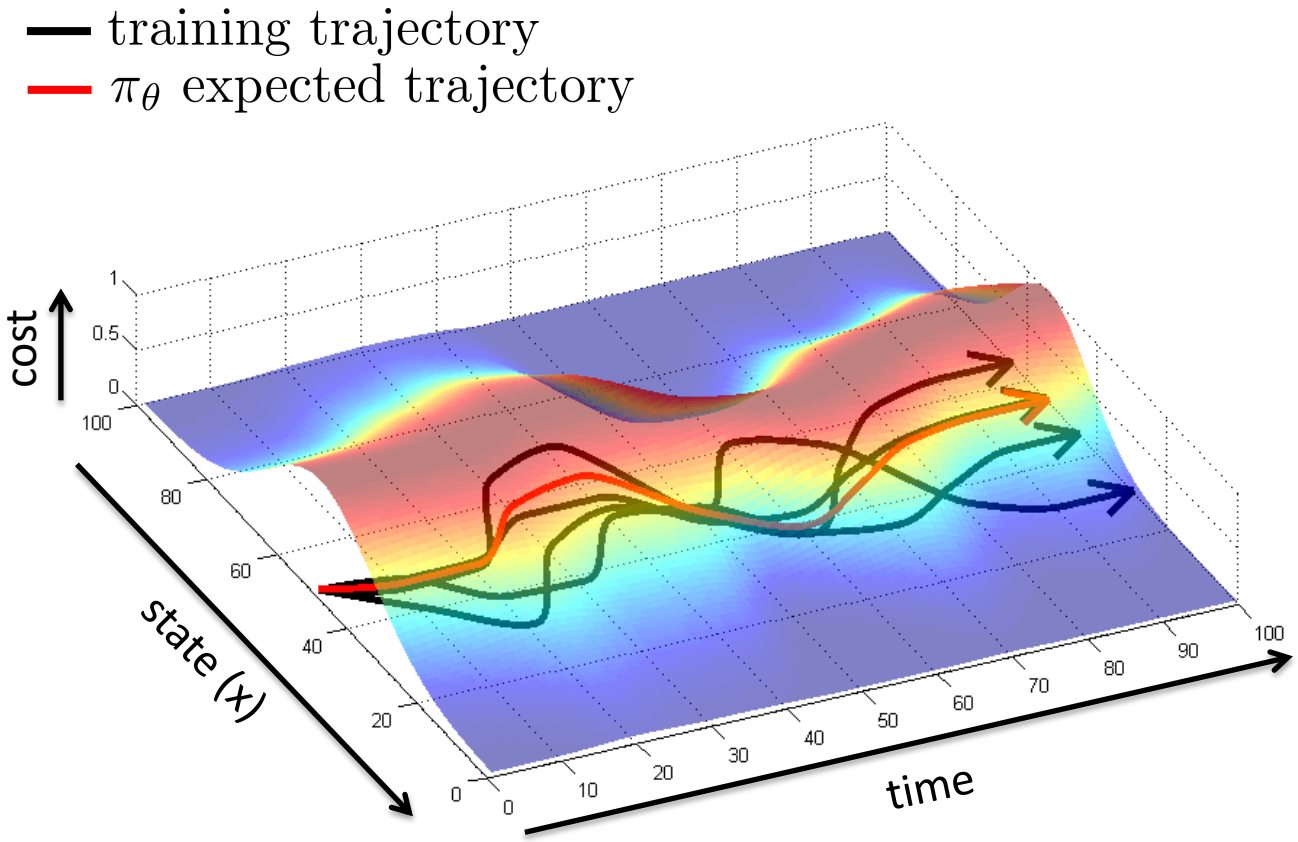
Why did that work?





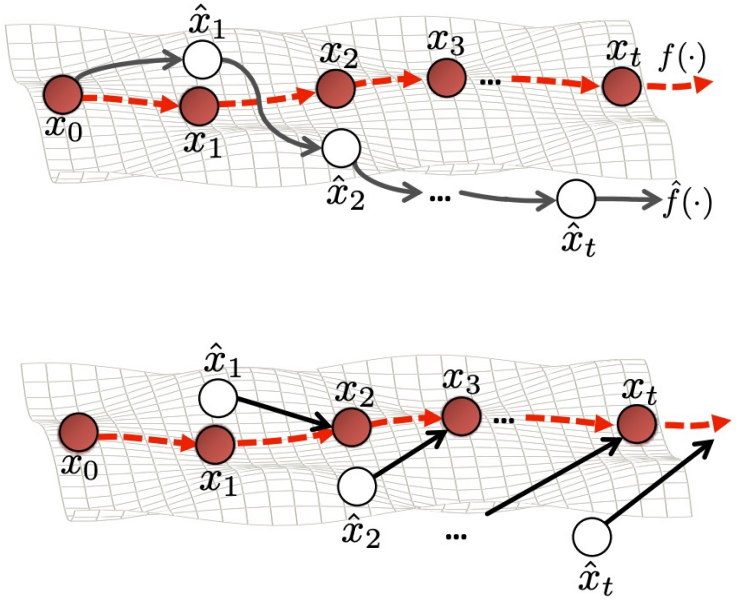
Bojarski et al. '16, NVIDIA

What is the general principle?

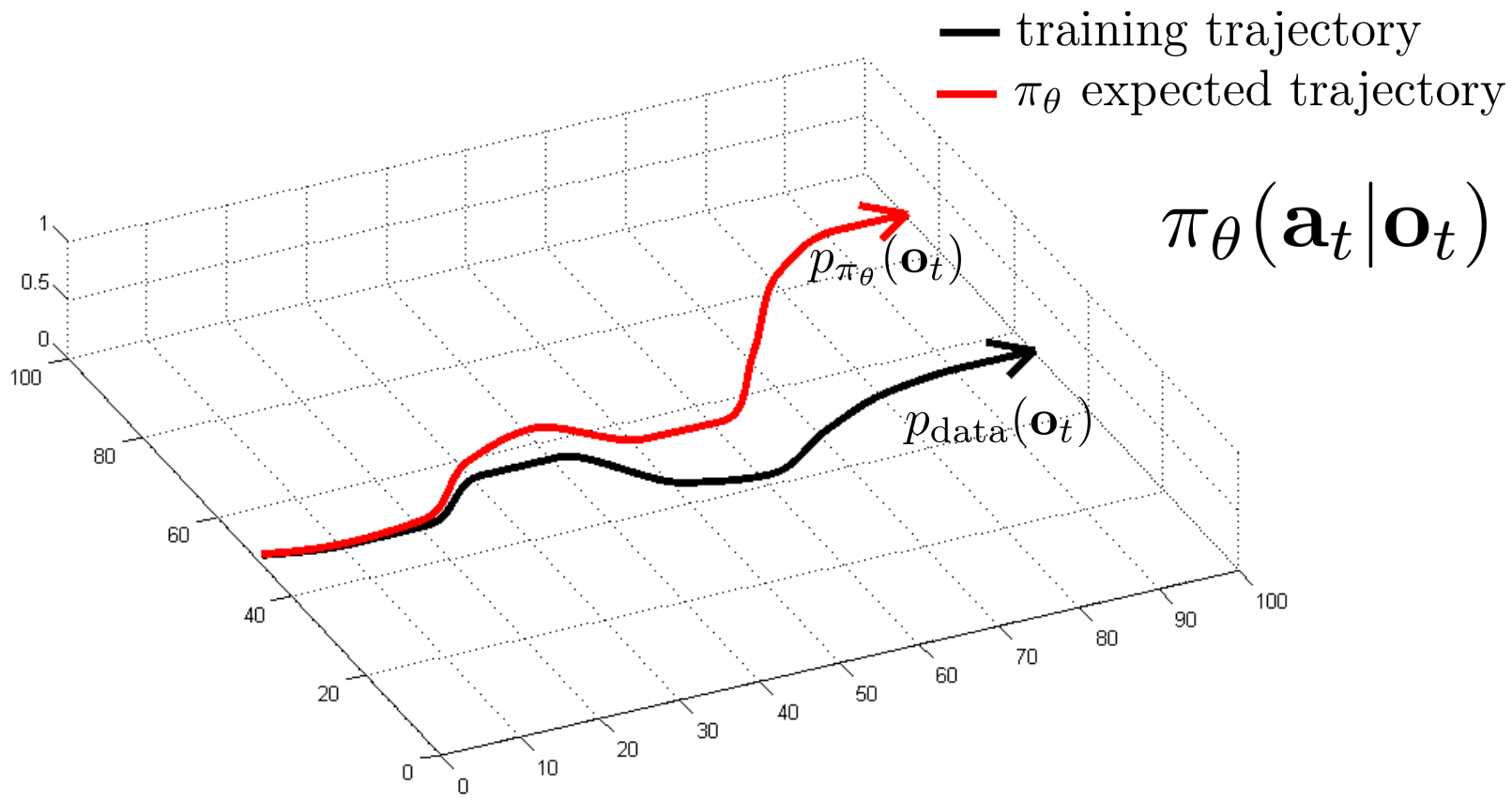


stability

Corrective labels that bring you back to the data



What might this mean mathematically?



can we make $p_{\text{data}}(\mathbf{o}_t) = p_{\pi_\theta}(\mathbf{o}_t)$?

Concrete Instantation: DAgger

can we make $p_{\text{data}}(\mathbf{o}_t) = p_{\pi_\theta}(\mathbf{o}_t)$?

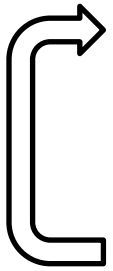
idea: instead of being clever about $p_{\pi_\theta}(\mathbf{o}_t)$, be clever about $p_{\text{data}}(\mathbf{o}_t)$!

DAgger: Dataset Aggregation

goal: collect training data from $p_{\pi_\theta}(\mathbf{o}_t)$ instead of $p_{\text{data}}(\mathbf{o}_t)$

how? just run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

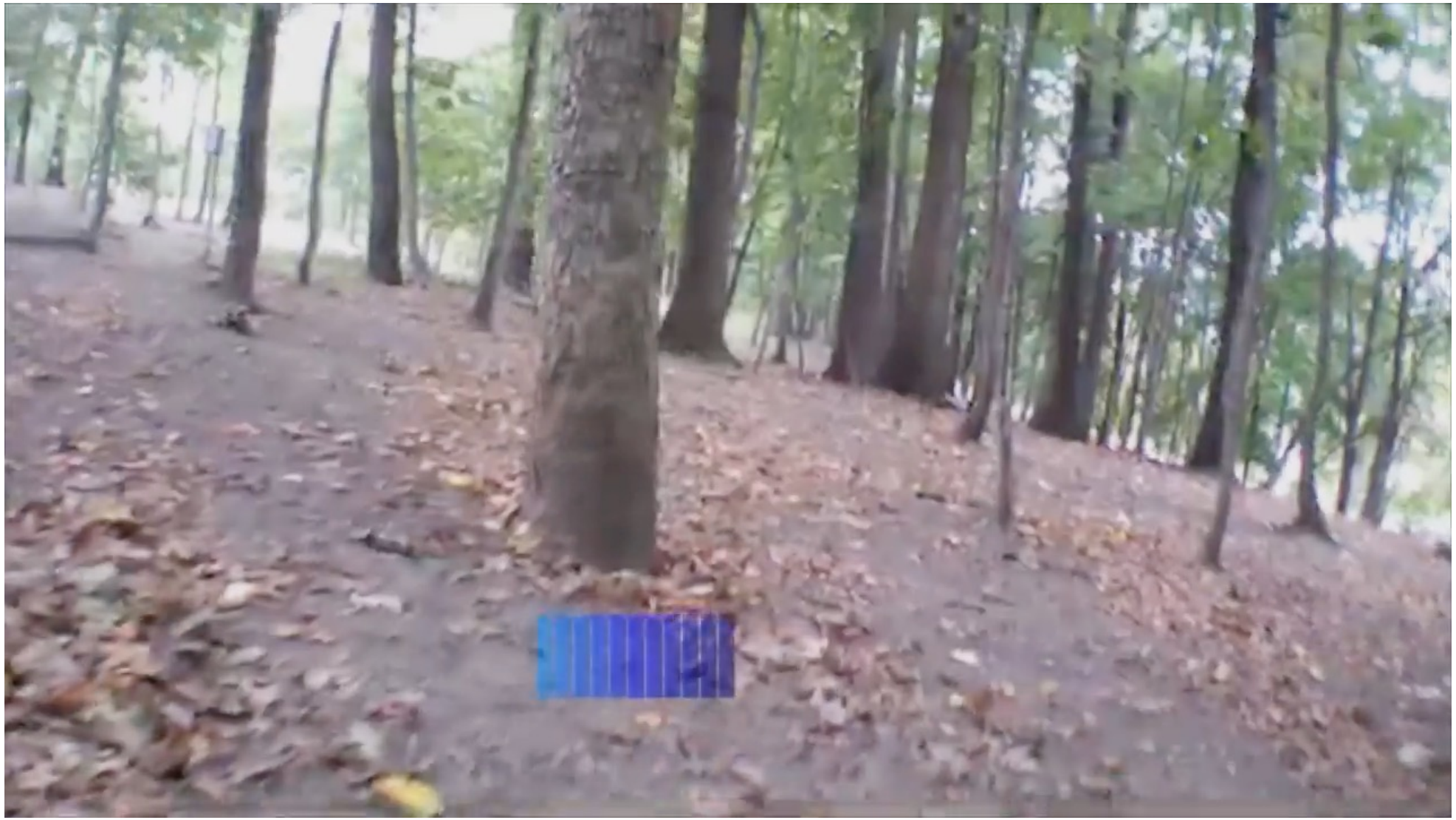
but need labels $\mathbf{a}_t$!



1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Ross et al. '11

DAgger Example



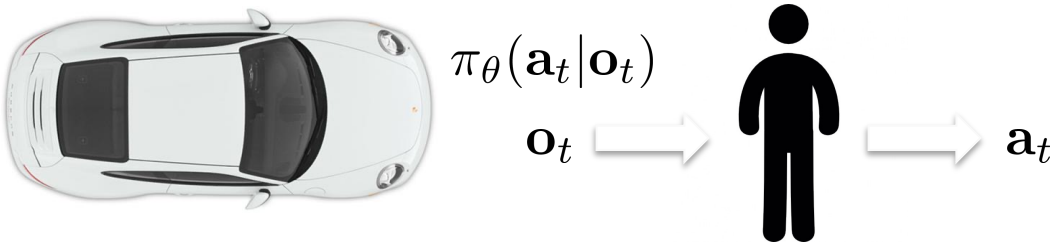
Ross et al. '13

What's the problem?

Requires more human labels

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Requires data in the environment

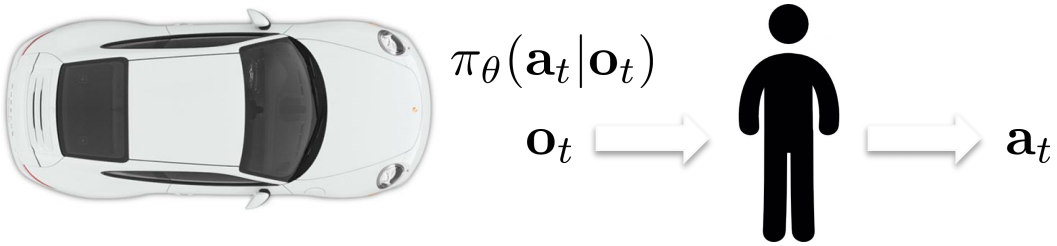


How might we fix this?

Match human data using learned models

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

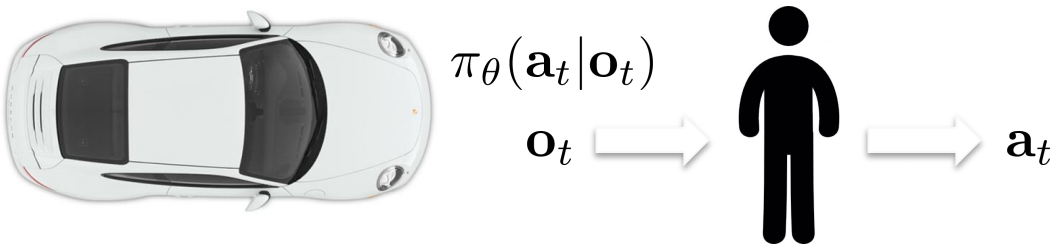
Generate synthetic data offline



How might we fix this?

Match human data using learned models

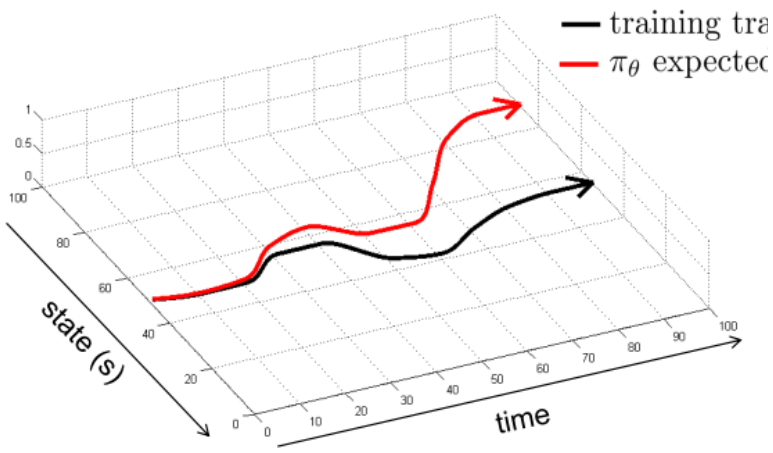
1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$



Avoiding Compounding Error with Distribution Matching

Remember:

Supervised learning is not sequential decision making



Compounding error!

$$\arg\max_\theta \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}}\left[\log \pi_\theta(a^*|s^*)\right] \qquad \mathbb{E}_{(s,a)\sim\rho(\pi)}\left[1(a=a^*)\right]$$

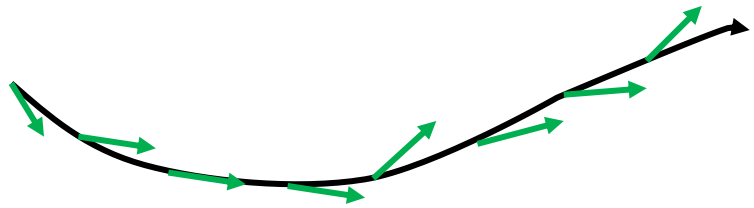
Not the same!

What if we made them the same!

Recasting Imitation Learning as Distribution Matching

Behavior Cloning:

$$\arg\max_{\theta} \mathbb{E}_{(s^*,a^*)\sim\mathcal{D}} \left[\log \pi_\theta(a^*|s^*)\right]$$

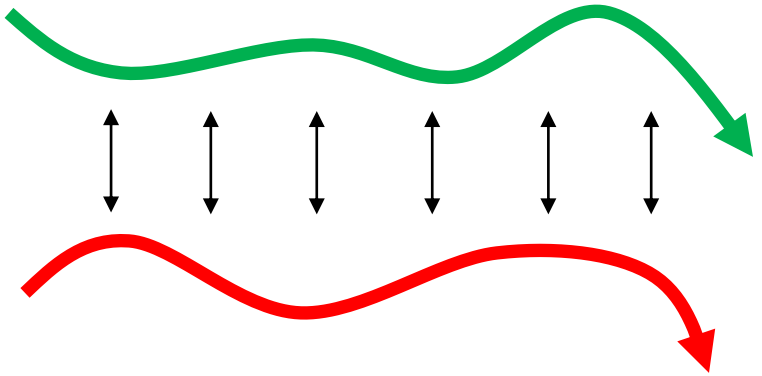


Distribution Matching

$$\min_{\theta} D(\rho^*(s,a)||\rho_\theta(s,a))$$

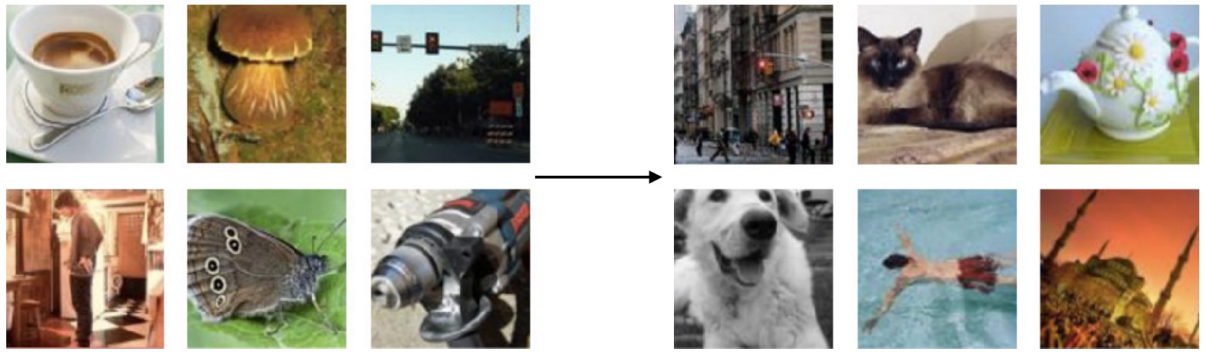
or

$$\min_{\theta} D(p^*(\tau)||p_\theta(\tau))$$

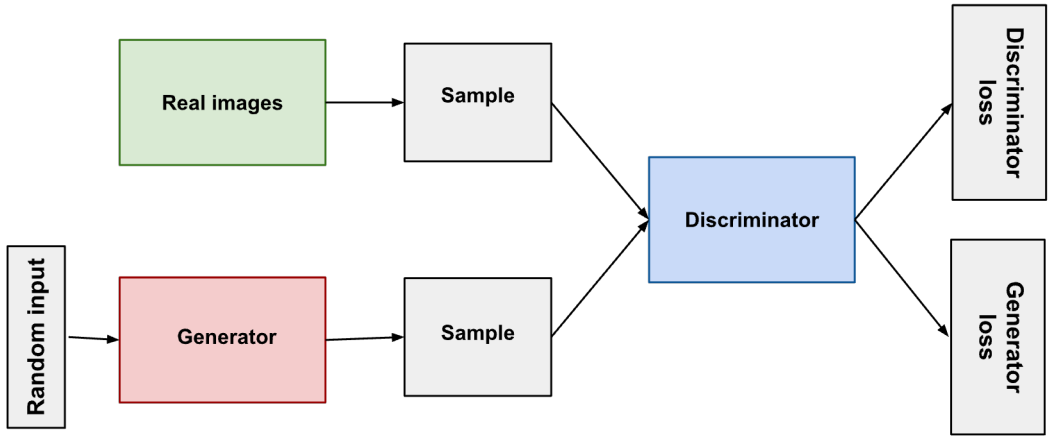


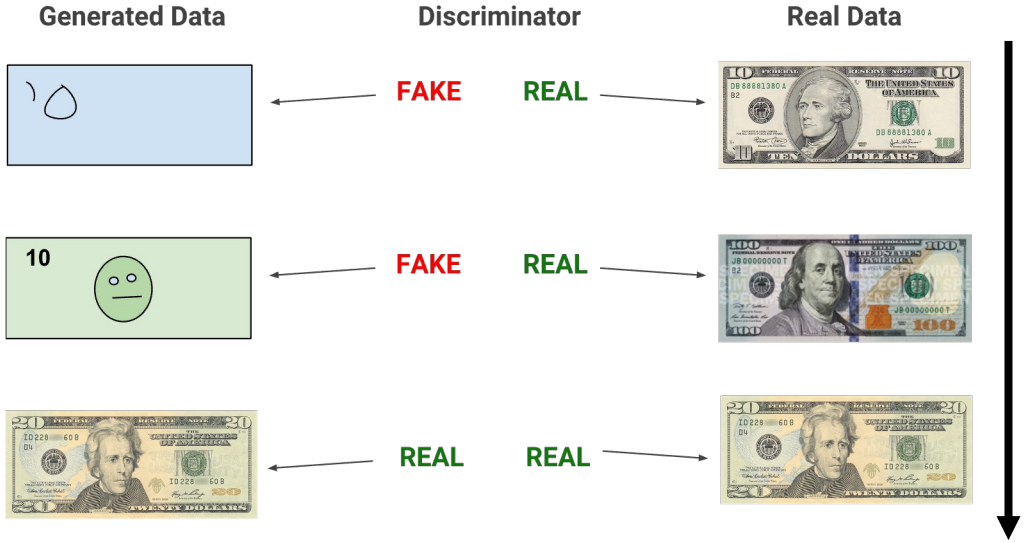
How do we represent and optimize this objective?

Distribution Matching in Computer Vision - GANs



Train 2 models – Generator and Discriminator

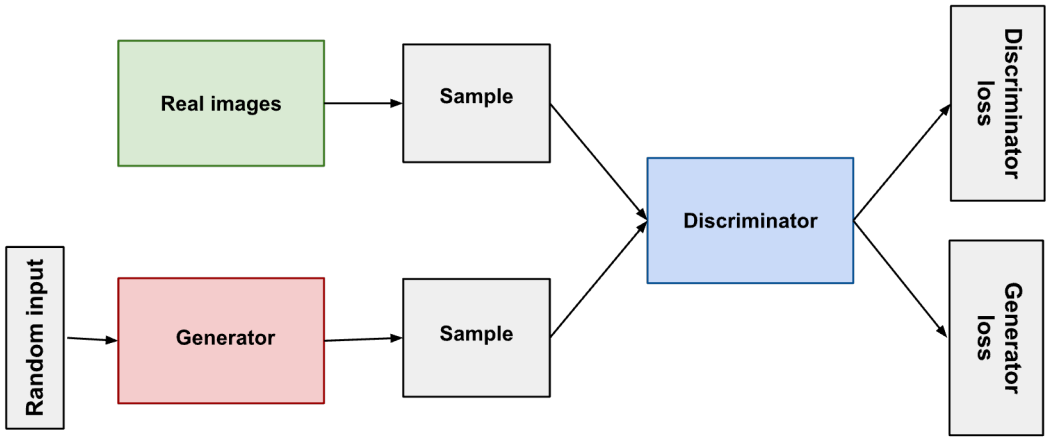


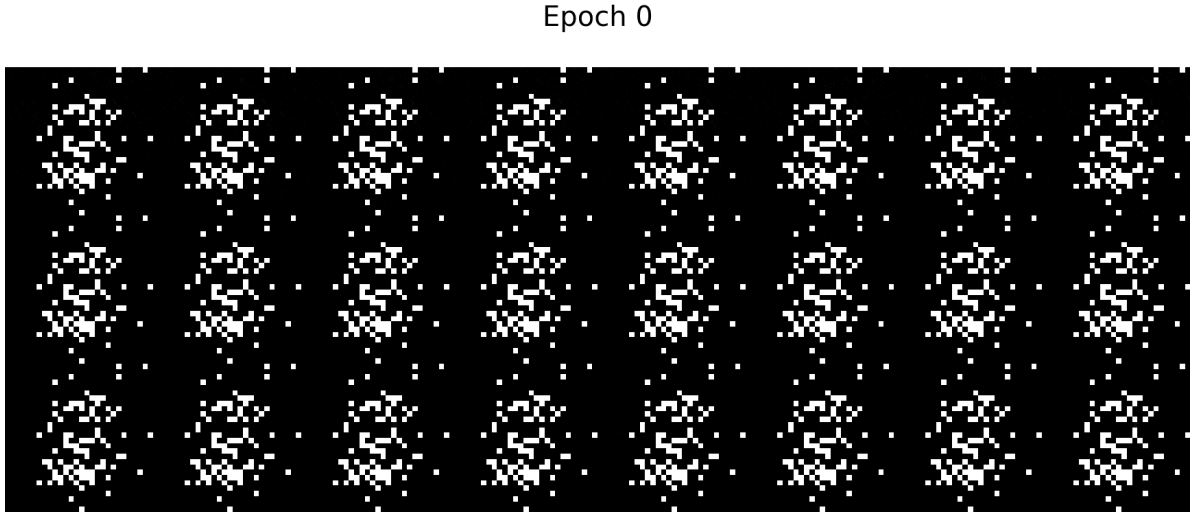


Discriminator – real vs fake Generator – confuse discriminator

Distribution Matching in Computer Vision - GANs

Train 2 models – Generator and Discriminator





Discriminator – real vs fake Generator – confuse discriminator

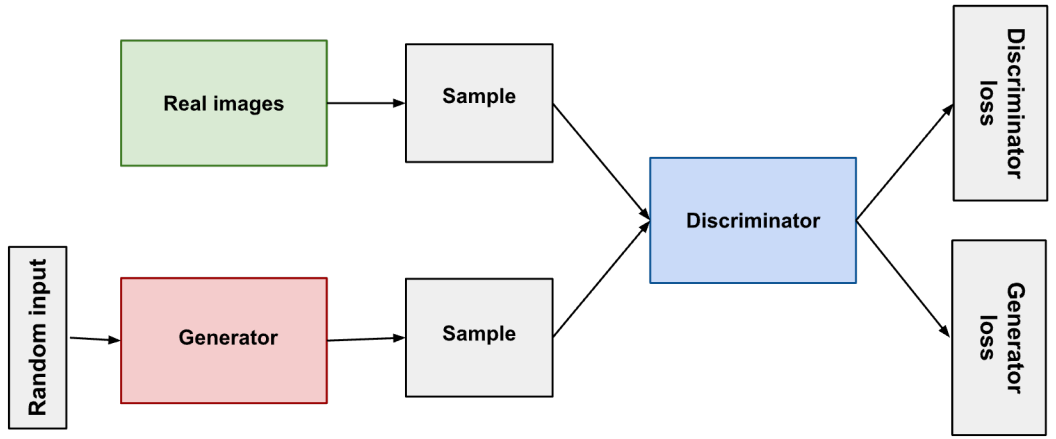
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Converges the generator to the data distribution

Can show that this minimizes JSD to data

$$\text{JSD}(P \parallel Q) = \frac{1}{2}D(P \parallel M) + \frac{1}{2}D(Q \parallel M),$$

Applying GANs to Imitation Learning



Generator = Policy

Discriminator = Reward Function

Policy is trying to confuse the discriminator whether trajectories are demos or policy

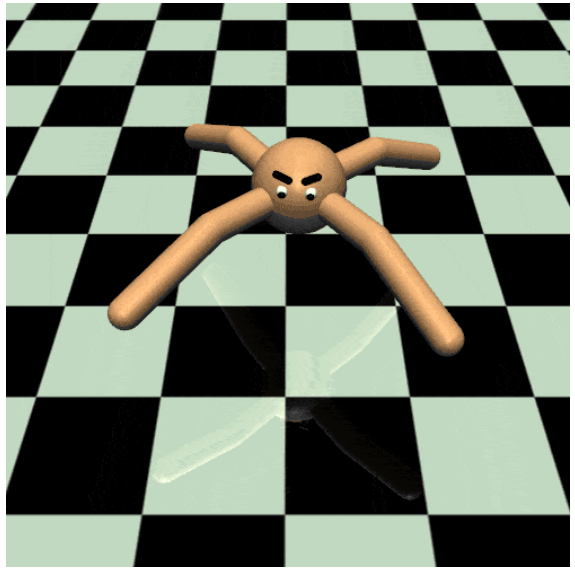
Discriminator – real vs fake Generator – confuse discriminator

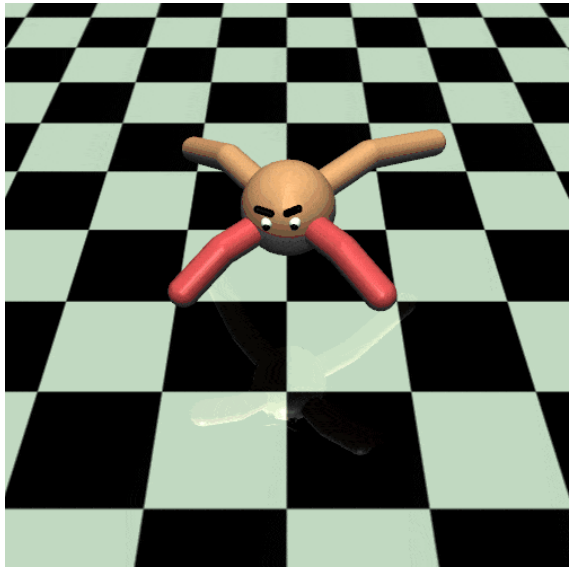
$$\min_{\pi_\theta} \max_{D} \mathbb{E}_{\pi^*} [\log D(s)] + \mathbb{E}_{\pi_\theta} [\log(1 - D(s))]$$

Trained as a classifier Train with RL

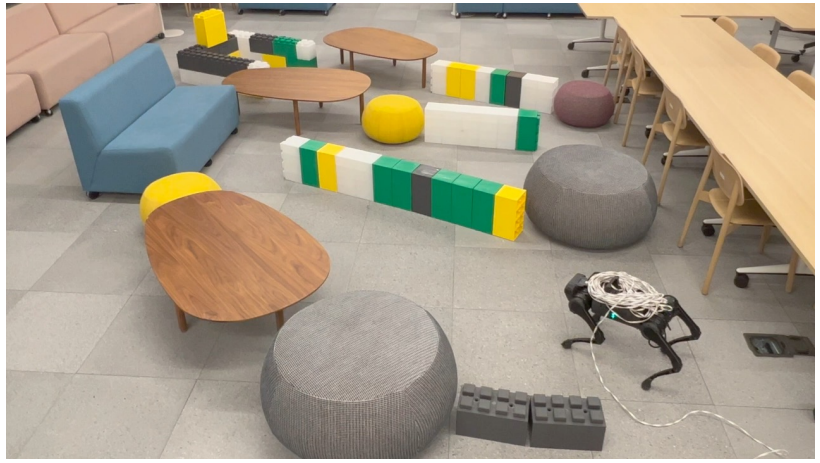
Matches distribution **under** the policy

Generative Adversarial Imitation Learning





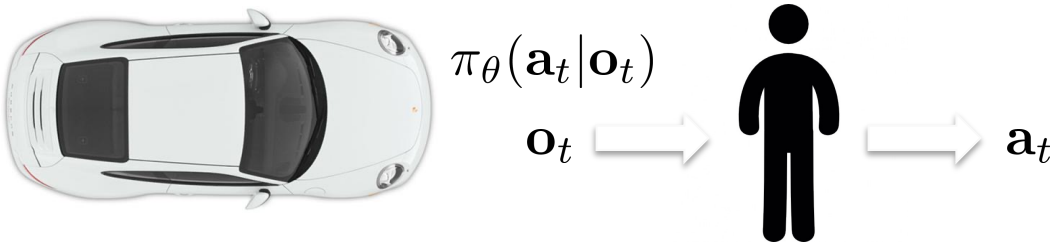
Still requires sampling in the environment



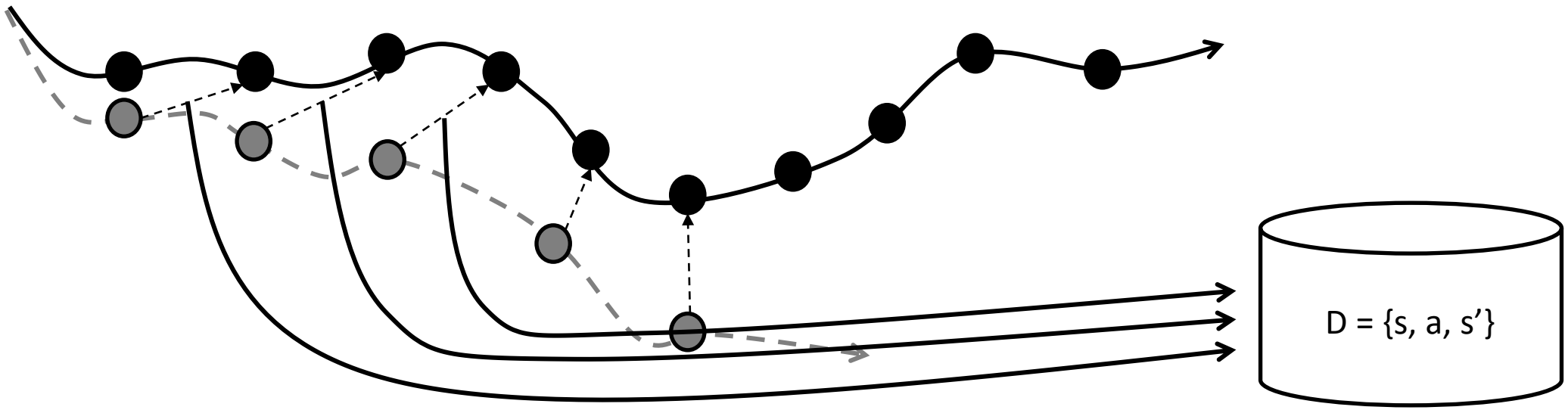
How might we fix this?

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Generate synthetic data offline



Can we avoid expensive online data collection?



Generate corrective labels to dataset for imitation

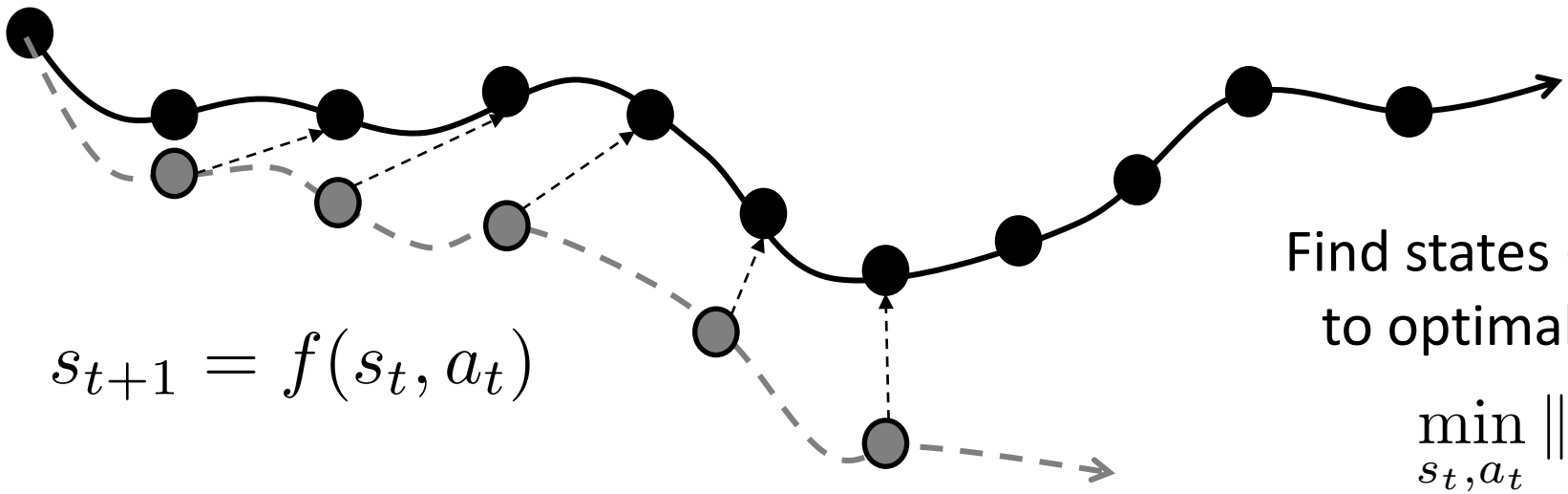
Abhay Deshpande

Yunchu Zhang

Liyiming Ke

How can we find corrective labels without an expensive human in the loop and online data collection?

Generating Corrective Labels From True Dynamics



$$s_{t+1} = f(s_t, a_t)$$

Find states (s_t), actions (a_t) that lead back to optimal states under true dynamics

$$\min_{s_t, a_t} \|s^*_{t+1} - f(s_t, a_t)\| \leq \epsilon$$

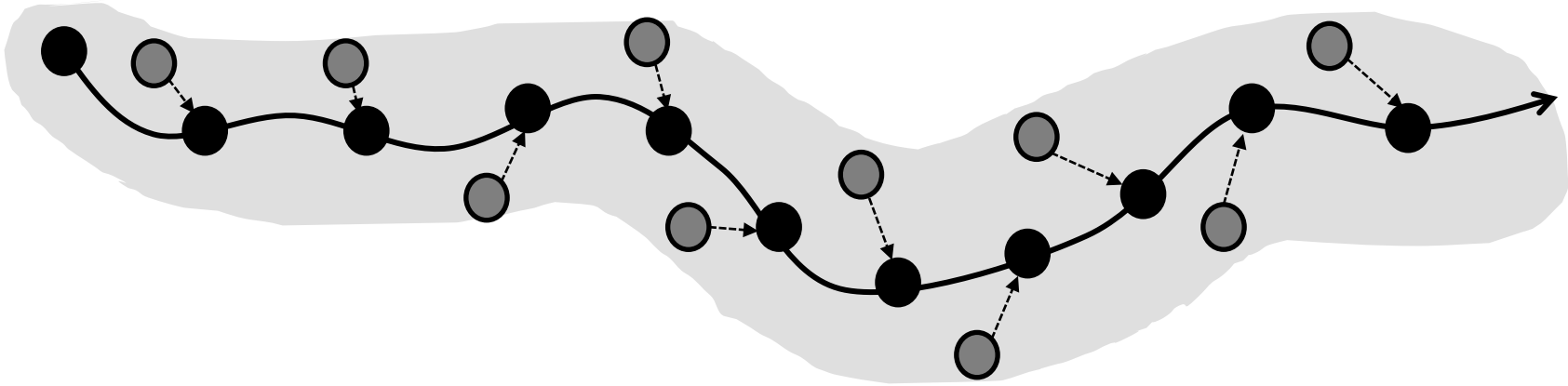
Easy with known dynamics

Intuition: find labels to bring OOD states back in distribution

But models are unknown! ☹

CCIL: Continuity-based Data Augmentation for Corrective Imitation Learning, Ke et al '24
Data Efficient Behavior Cloning for Fine Manipulation via Continuity-based Corrective Labels, Deshpande et al '24

Generating Corrective Labels with **Learned** Dynamics



Ok models are unknown, let's learn them!

$$\min_{\hat{f}} \mathbb{E}_{(s_t,a_t,s_{t+1})\sim\mathcal{D}} \left[\|\hat{f}(s_t,a_t) - s_{t+1}\|_2 \right]$$

But learned dynamics $\hat{f}_\phi$ are not globally accurate?

$$\|s^*_{t+1} - \hat{f}_\phi(s_t,a_t)\| \leq \epsilon$$

Under approximately Lipschitz smooth models, trust models around training data

Find states (s_t), actions (a_t) that lead back to optimal states under ~~true~~ learned dynamics, **where learned dynamics can be trusted**

$$\min_{s_t,a_t} \|s^*_{t+1} - \hat{f}_\phi(s_t,a_t)\| \leq \epsilon \longleftarrow \text{Corrective label}$$

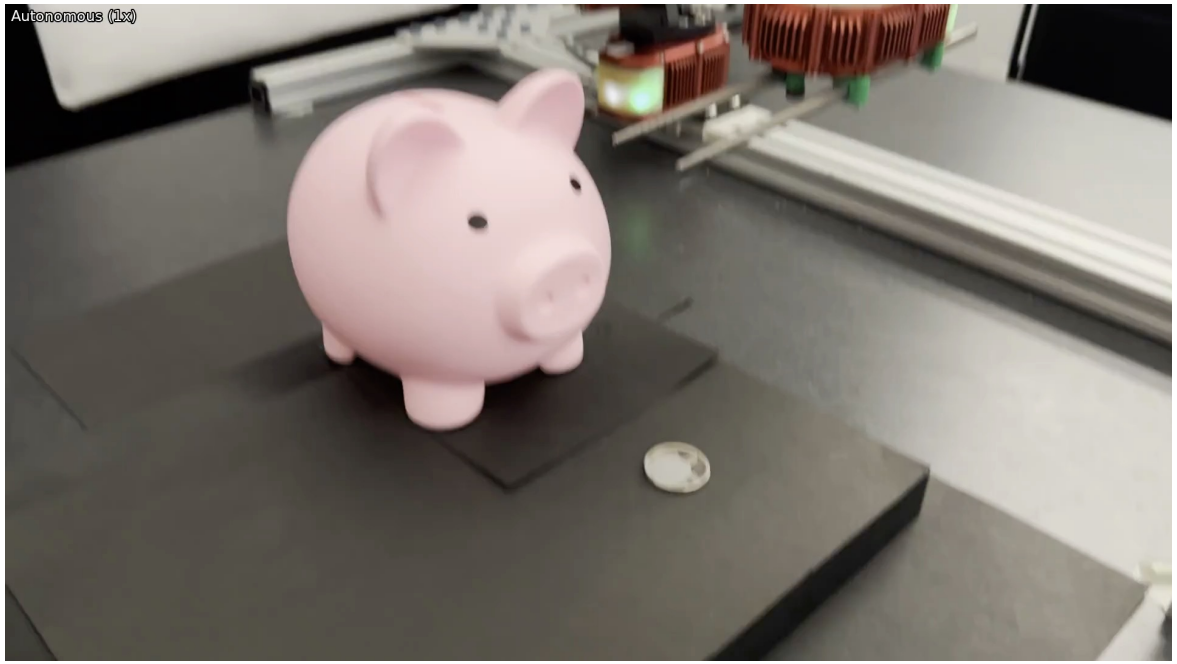
$$\text{s.t} \quad \|s^*_t - s_t\| \leq \epsilon_1, \|a^*_t - a_t\| \leq \epsilon_2 \longleftarrow \text{Close to data}$$

CCIL: Continuity-based Data Augmentation for Corrective Imitation Learning, Ke et al '24
Data Efficient Behavior Cloning for Fine Manipulation via Continuity-based Corrective Labels, Deshpande et al '24

How well does generating corrective labels work?

With corrective labels

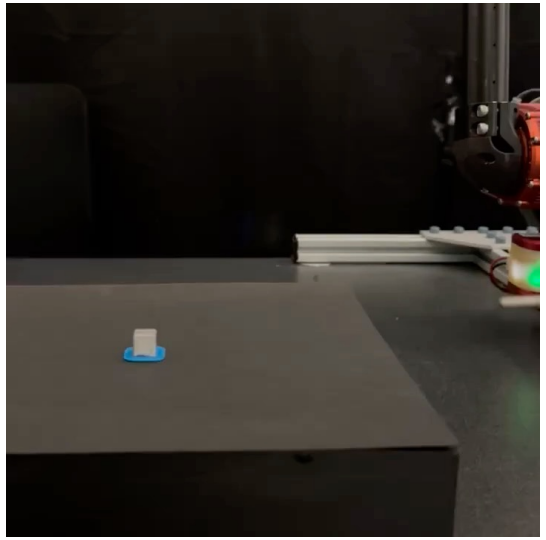
Without corrective labels

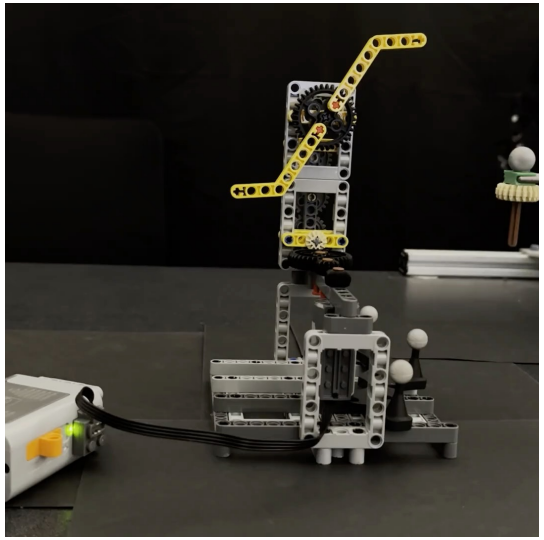


CCIL: Continuity-based Data Augmentation for Corrective Imitation Learning, Ke et al '24
Data Efficient Behavior Cloning for Fine Manipulation via Continuity-based Corrective Labels, Deshpande et al '24

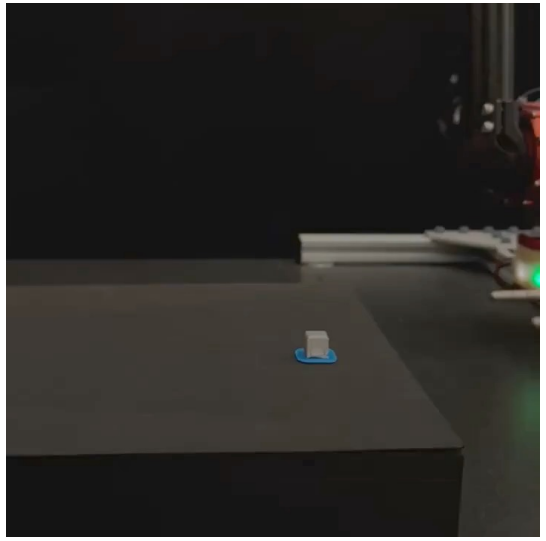
How well does generating corrective labels work?

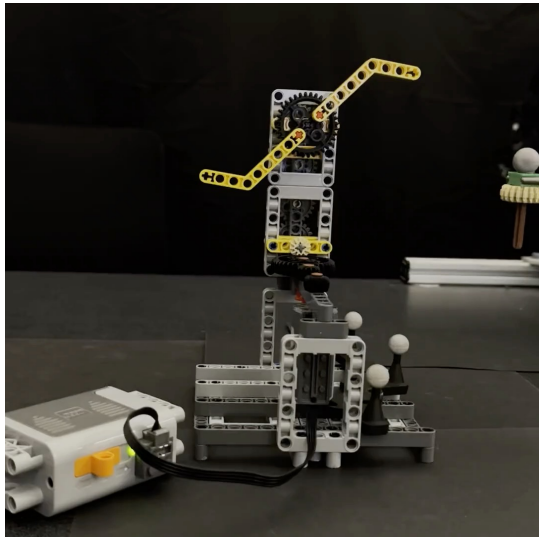
With corrective labels

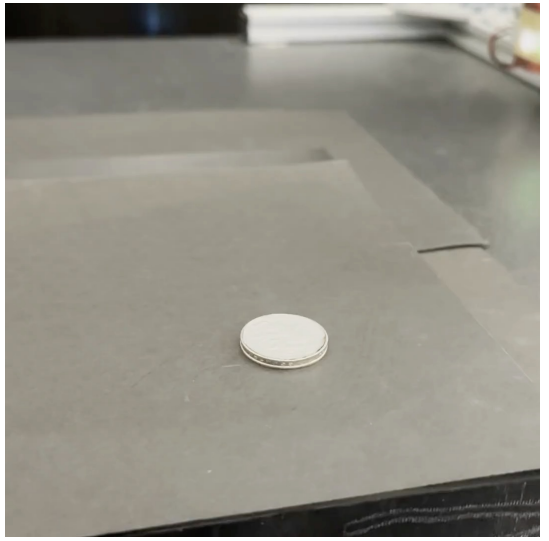




Without corrective labels







Correct for compounding errors by making train and test look more similar

What we talked about today?

Imitation Learning and Inverse Reinforcement Learning

(Learning Behaviors from ***Expert*** Data)

- **Multimodality and Underfitting in Imitation**
- **Compounding Error in Imitation**

Some cool behavior cloning videos

1x and tesla humanoid robots

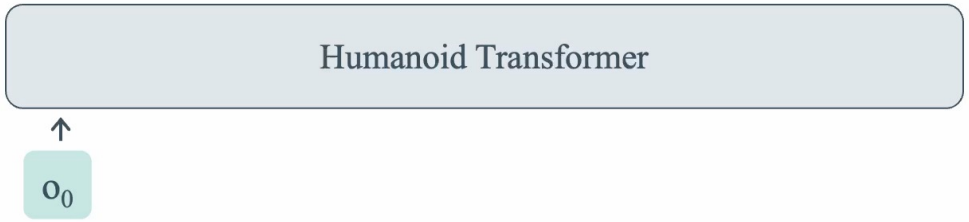
ALOHA and CherryBot Fine Manipulation

TRI Diffusion Policies

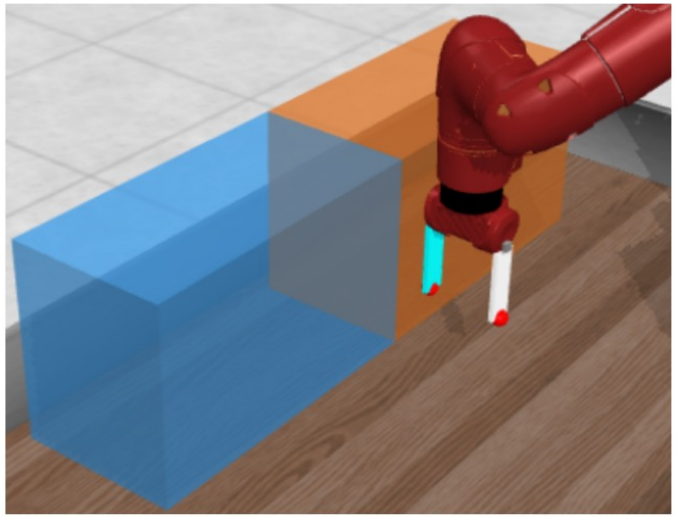
So does this solve all the issues in imitation?

Frontiers in Imitation Learning

Non-Markovian Demonstrators



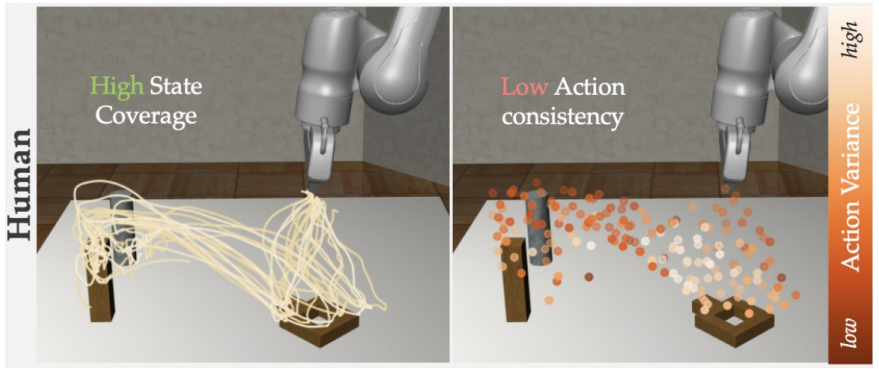
Characterizing generalization



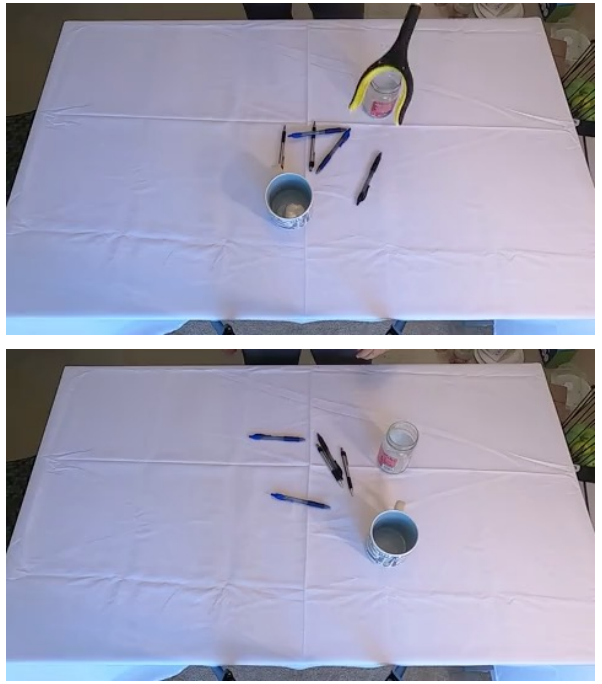
Action-Free Data

Frontiers in Imitation Learning

Data Curation and Quality



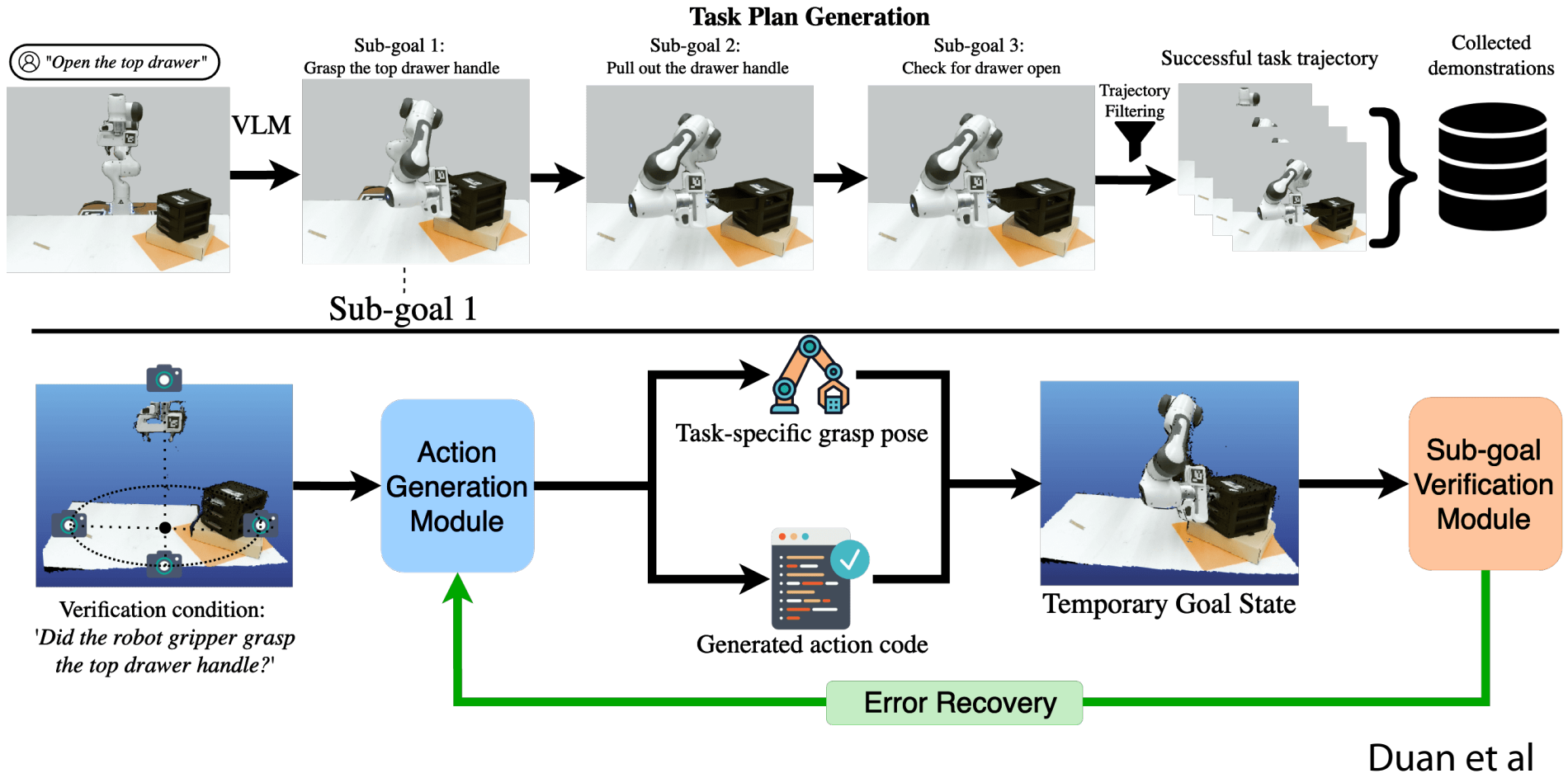
Embodiment Shift



Teleoperation Interfaces

Frontiers in Imitation Learning

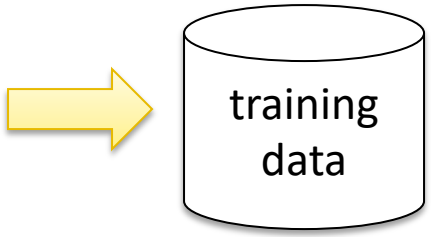
Learning how to retry and improve

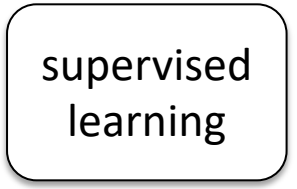


Duan et al

Perspectives on Imitation

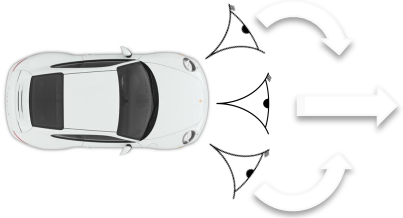
$\mathbf{o}_t$
$\mathbf{a}_t$



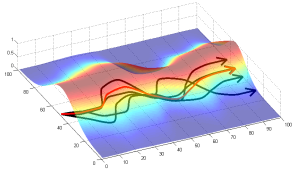


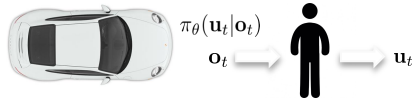
$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

- Pros:
  - Easy to use, no additional infra
  - Can sometimes be unreasonably effective



- Cons:
  - Challenges of compounding error, multimodality
  - Doesn't really generalize
  - Expensive in terms of data collection – needs optimality!





What will we talk about today?

Preliminaries

Imitation Learning

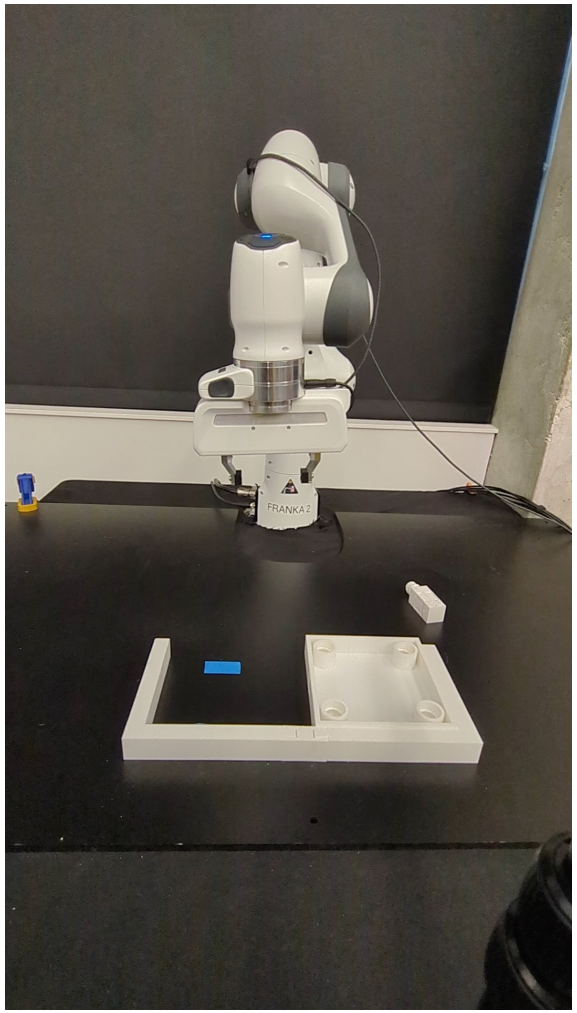
(Learning Behaviors from Expert Data)

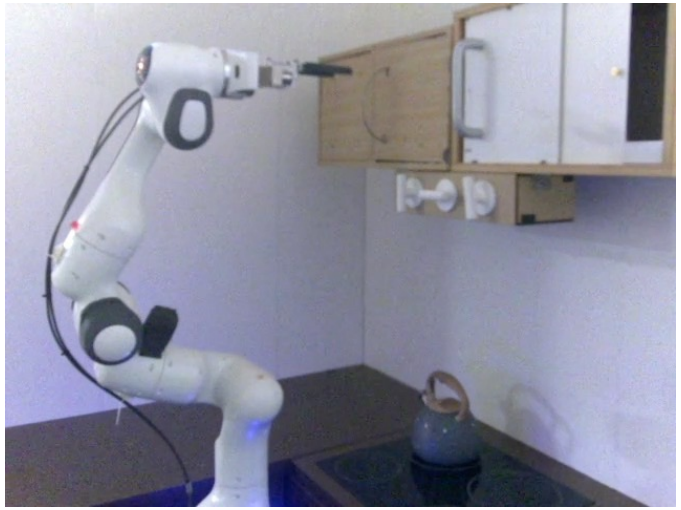
Offline Reinforcement Learning

(Learning from Non-Expert data)

What if our data was not optimal?

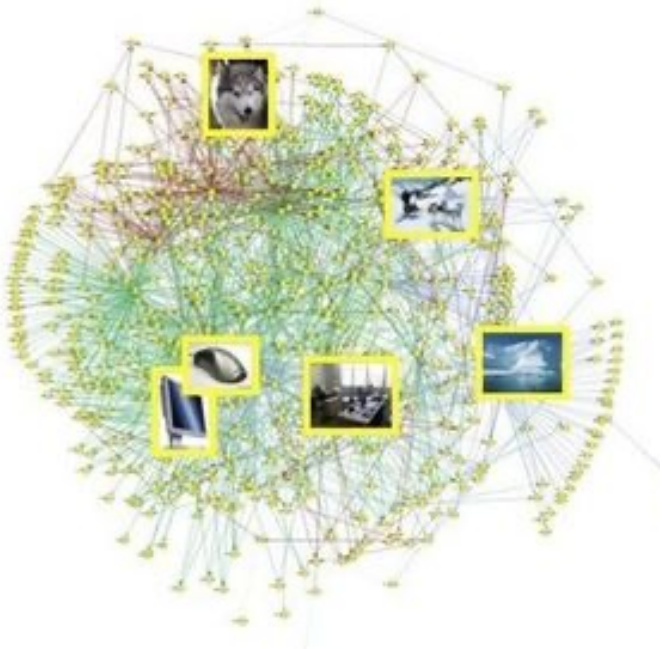
Much of the data collected is suboptimal

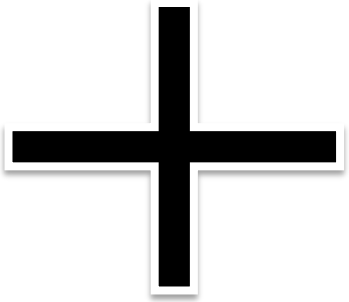


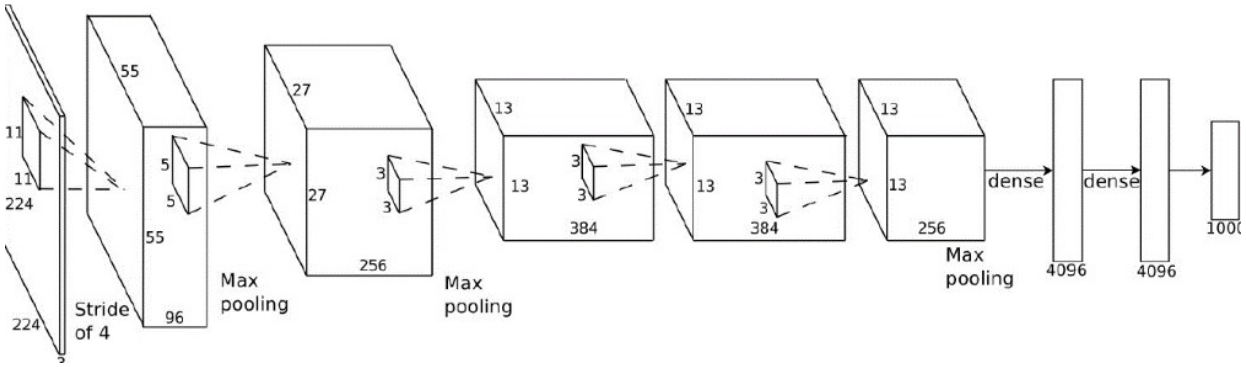


Imitating suboptimal behavior leads to suboptimal behavior!

What makes modern ML work? – Unsupervised Learning



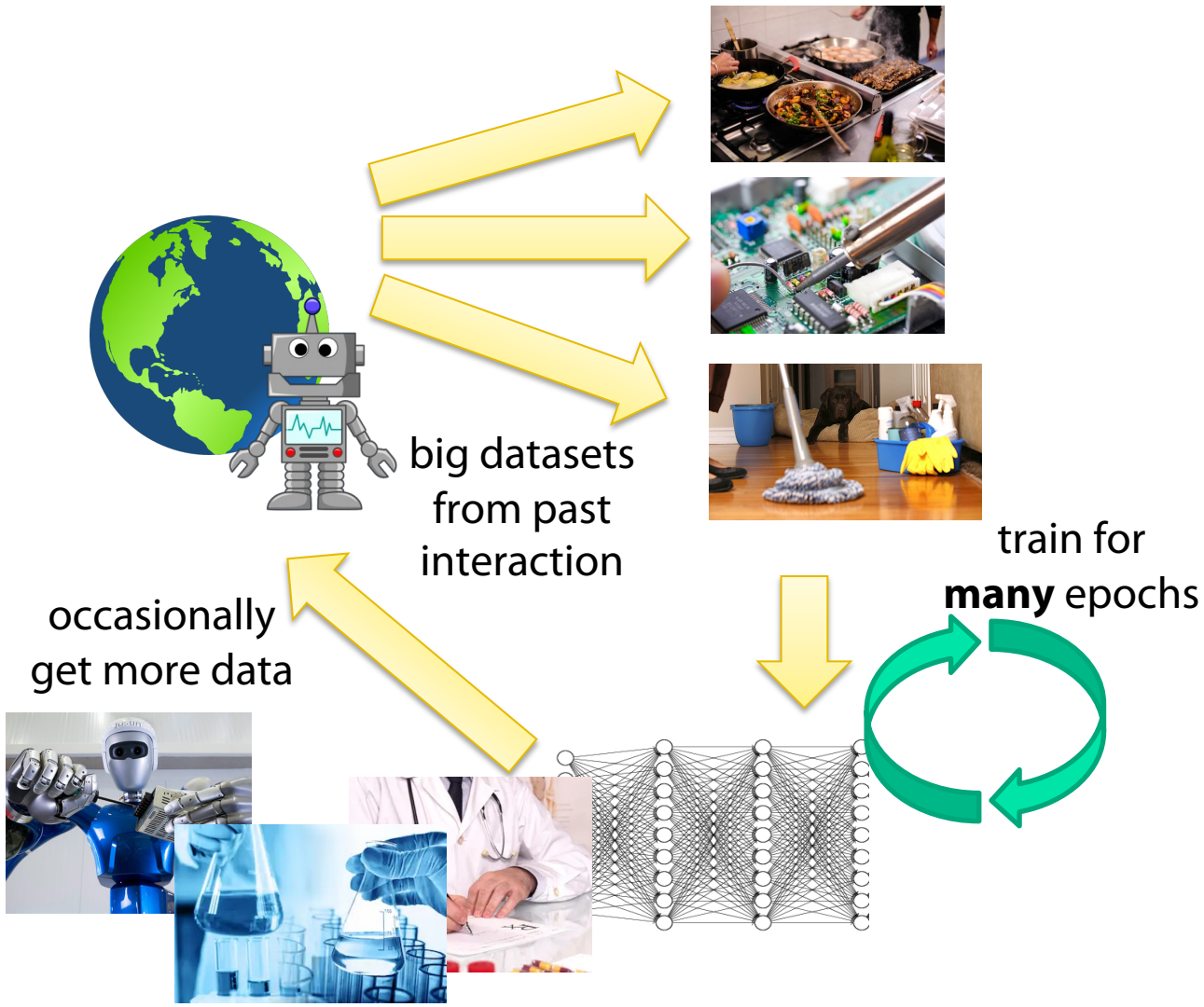




Learn patterns at scale from potentially suboptimal data

Finetune with small amounts of curated optimal data

Can we develop data-driven "RL" methods?



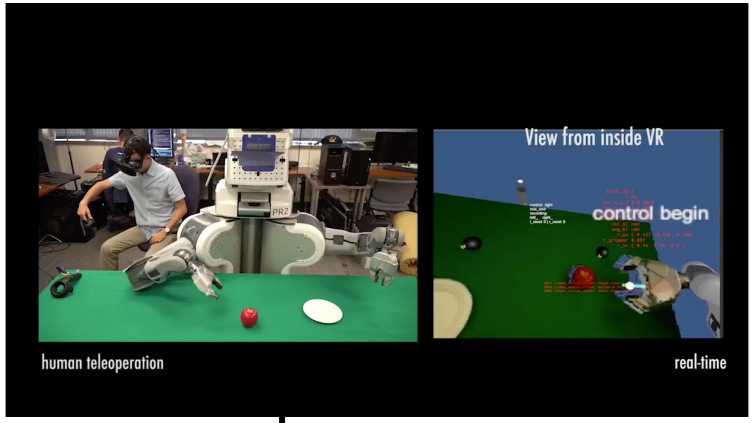
Methods that can learn from and do better than (potentially suboptimal) offline data

What might prior datasets look like?

Large scale datasets

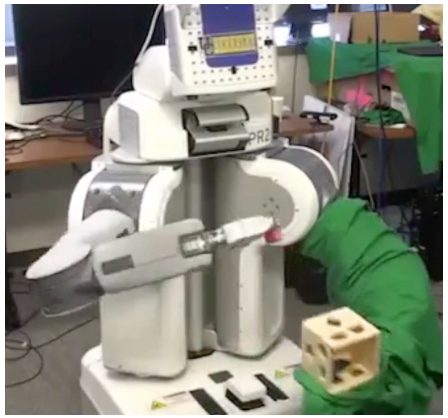
Damen et al

Teleoperated Data



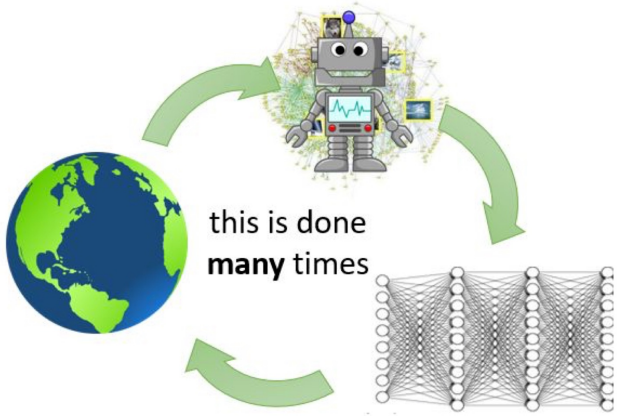
Zhang et al

Previous Experiments



Levine et al

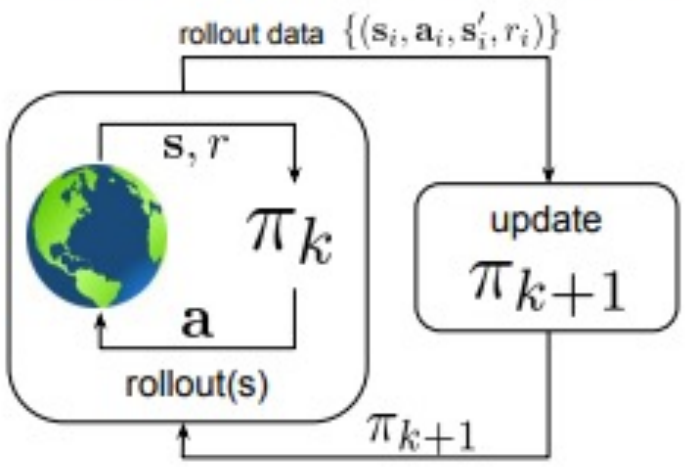
reinforcement learning



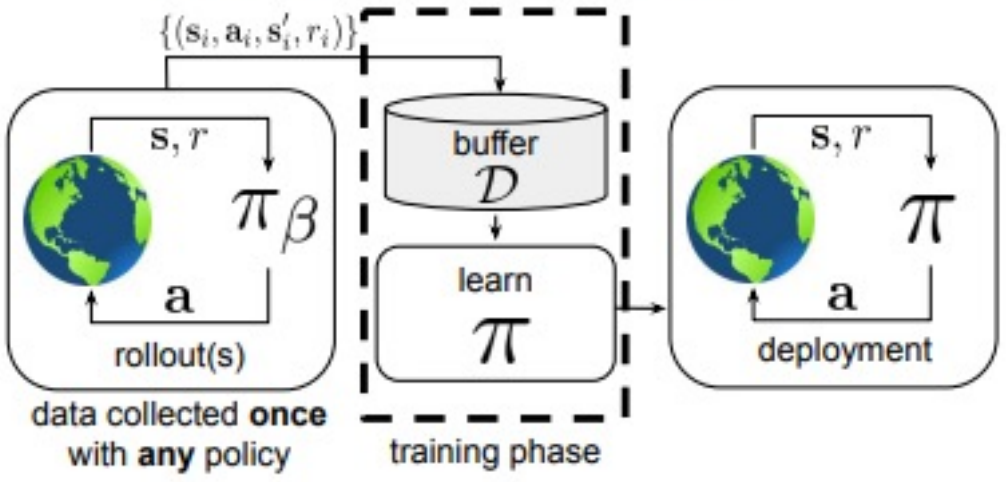
Why is this called offline **RL**?

What we typically think of as RL?
→ Trial and error online

What is offline RL?
→ **No trial and error online**
→ Sequential decision making
→ Similar solution tools to RL



On-policy RL



Data-driven Offline RL

How can we learn decision making from non-optimal offline data?

Formalism for Offline RL

$$\max_{\theta} \mathbb{E}_{s_0 \sim \mu_0(s), a_t \sim \pi_\theta(a_t|s_t), s_{t+1} \sim \mathcal{P}(s_{t+1}|s_t, a_t)} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) \right]$$

Offline dataset: $\mathcal{D} = \{(s, a, s', r)\}_{i=1}^{n}$

Behavior policy generating offline dataset: $\pi_\beta(a|s)$

Reward is assumed to be available to say good/bad a (s, a) transition is

Goal:

- With only sampling access to $\mathcal{D}$, learn π_θ^* without online data collection
- Perform better than the original behavior policy

When might offline RL be effective?

If $\mathcal{D}$ is only expert demos:

π_θ as good as $\mathcal{D}$

If $\mathcal{D}$ is sub-optimal data with insufficient coverage:

π_θ , $\mathcal{D}$ are both suboptimal

If $\mathcal{D}$ is sub-optimal data with sufficient coverage:

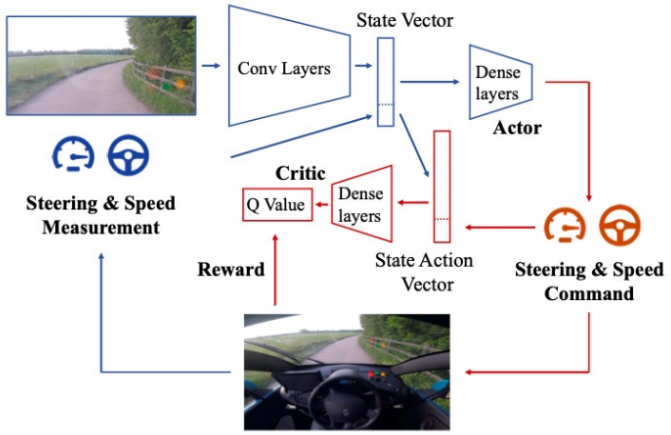
π_θ better than $\mathcal{D}$

Also serves as data curation

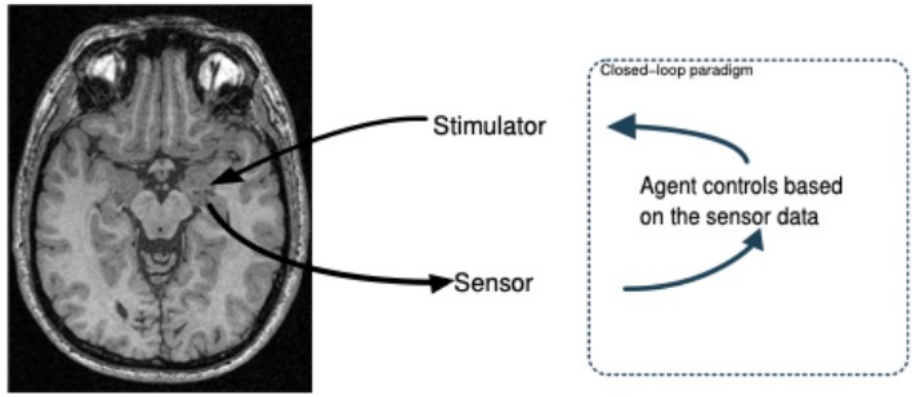
Application Domains

Useful in domains where data collection is risky/hard, but human datasets available

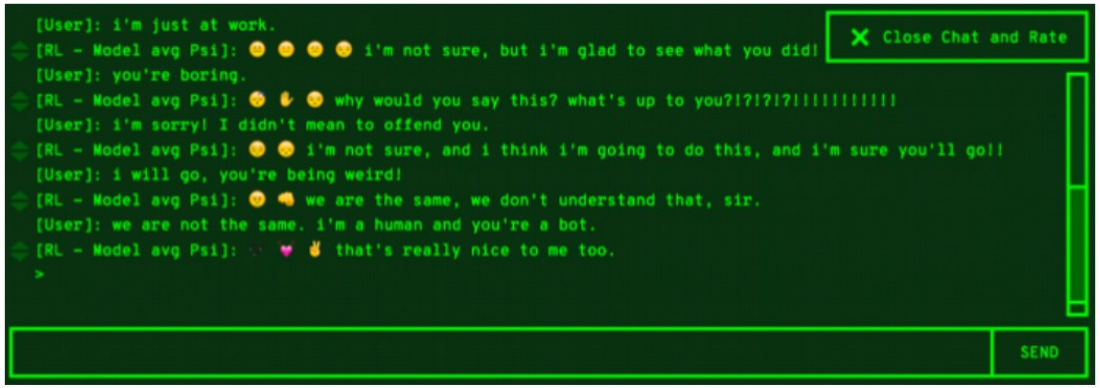
Driving



Healthcare



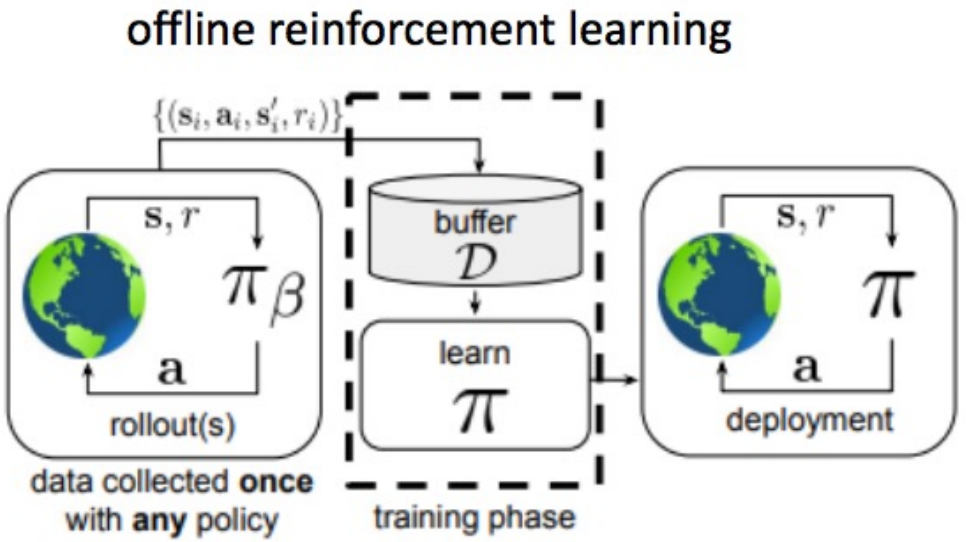
Dialogue



Robotics

Visuals from Levine et al

Offline RL Taxonomy

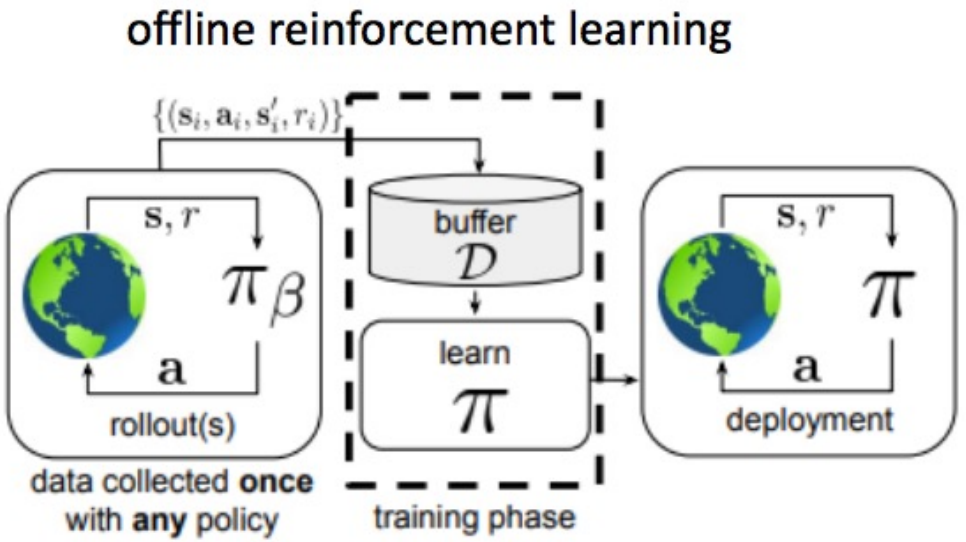


Importance Sampling

Policy Constraint Methods

Lower Bounded Q Values

Offline RL Taxonomy



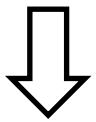
Importance Sampling

Policy Constraint Methods

Lower Bounded Q Values

Finding optimal policies

The goal of offline RL is to iteratively find the best policy – but without recollecting data!



Let's start by getting the best policy **while being able to recollect data**

$$\max_{\theta} \mathbb{E}_{\tau \sim p_\theta(\tau)} [R(\tau)]$$

$$\mathbb{E}_{\tau \sim p_\theta(\tau)} [\nabla_\theta \log p_\theta(\tau) R(\tau)]$$

Requires on-policy samples → can't be run offline

Useful Trick: Importance Sampling

Want to estimate expectation under p(x), but we only have samples from q(x)

$$
\begin{aligned}
\mathbb{E}_{x\sim p(x)}\left[f(x)\right] &= \int p(x) f(x) dx \\
&= \int p(x) \frac{q(x)}{q(x)} f(x) dx \\
&= \int q(x) \frac{p(x)}{q(x)} f(x) dx \\
&= \mathbb{E}_{x\sim q(x)}\left[\frac{p(x)}{q(x)} f(x)\right]
\end{aligned}
$$

(Reweighted expectation)

Gives us a nice way to get expectations without requiring on-policy data!

Importance Sampling Policy Gradient for Offline RL

Replace on-policy expectation in policy gradient with importance sampling

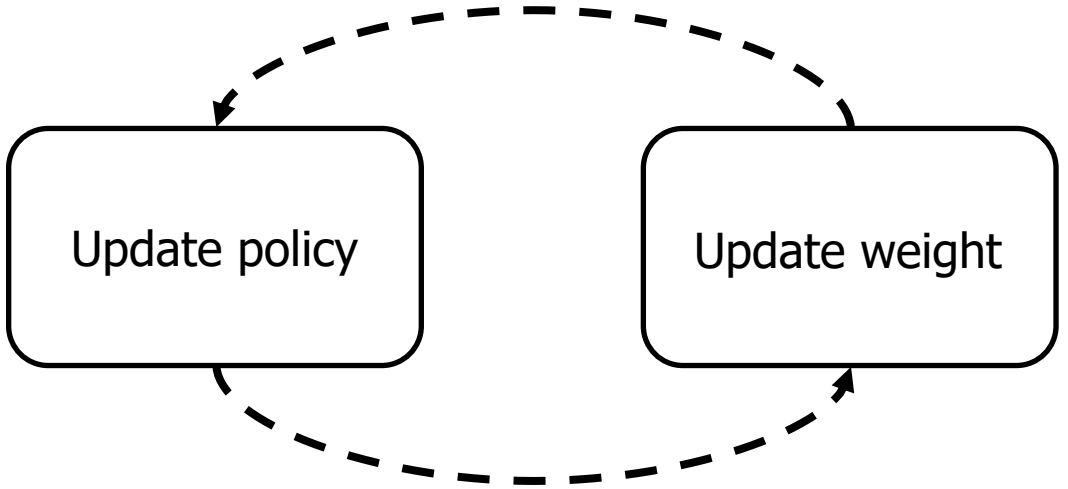
$$\mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\nabla_\theta \log p_\theta(\tau) R(\tau) \right]$$

$$\mathbb{E}_{\tau \sim q(\tau)} \left[\frac{p_\theta(\tau)}{q(\tau)} \nabla_\theta \log p_\theta(\tau) R(\tau) \right]$$

$$p_\theta(\tau) = p(s_0) \Pi_t p(s_{t+1}|s_t, a_t) \pi_\theta(a_t|s_t)$$

$$\mathbb{E}_{\tau \sim q(\tau)} \left[\frac{\cancel{p(s_0)} \Pi_t \cancel{p(s_{t+1}|s_t, a_t)} \pi_\theta(a_t|s_t)}{\cancel{p(s_0)} \Pi_t \cancel{p(s_{t+1}|s_t, a_t)} q(a_t|s_t)} \nabla_\theta \log p_\theta(\tau) R(\tau) \right]$$

$$\mathbb{E}_{\tau \sim q(\tau)} \left[\frac{\Pi_t \pi_\theta(a_t|s_t)}{\Pi_t q(a_t|s_t)} \nabla_\theta \log p_\theta(\tau) R(\tau) \right]$$

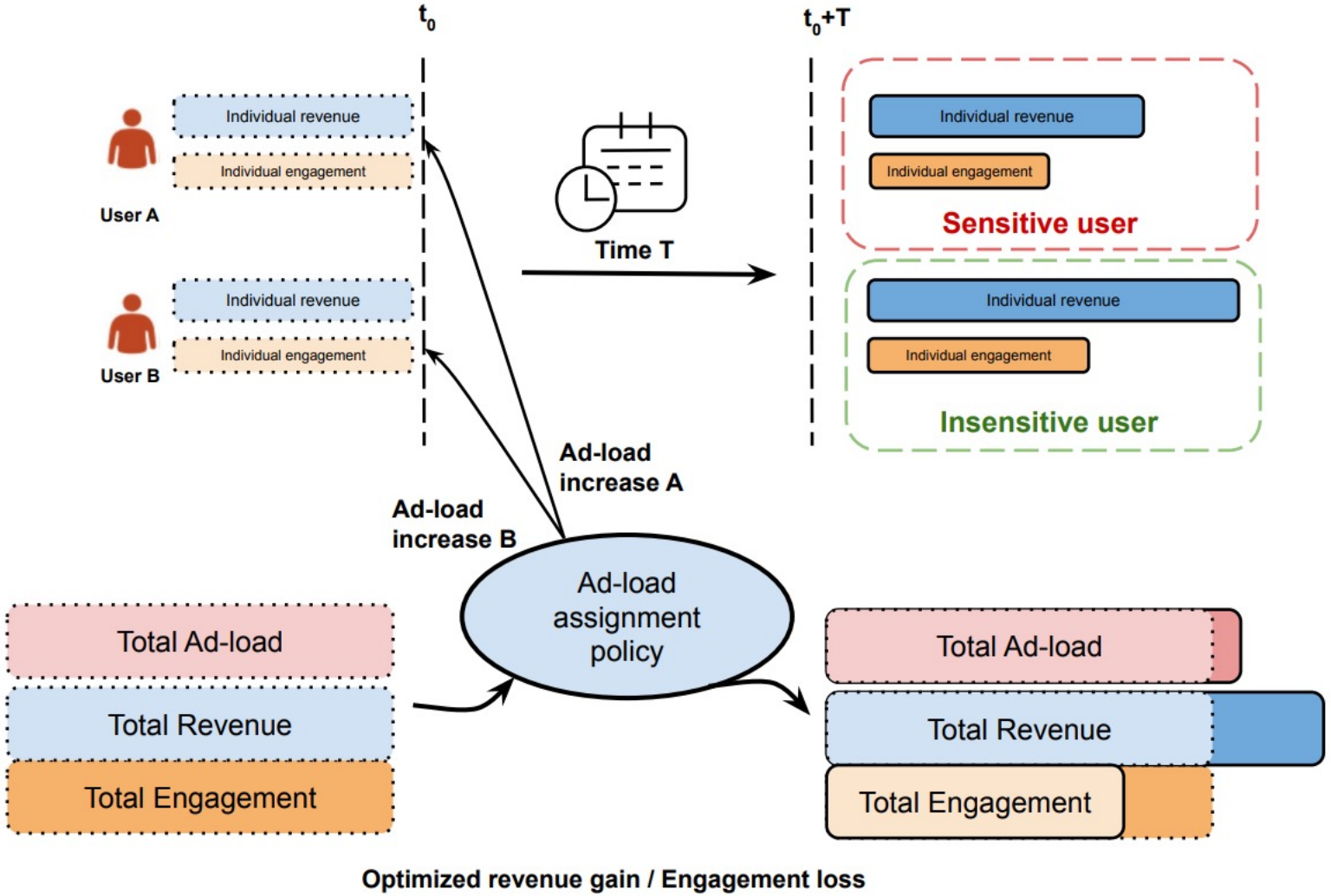


Don't need to know the environment dynamics!

Requires multiplying lots of probabilities – bad!

- Numerically unstable
- High variance

Applications of Importance Sampling based ORL

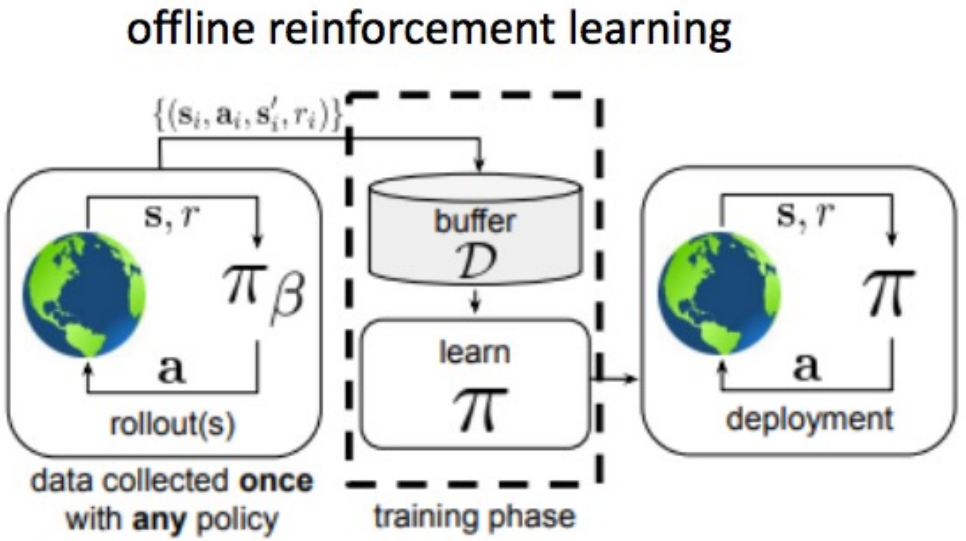


Many ad-problems are cast as importance sampling

Learn from suboptimal ad-placement strategies

Reweight the training data with density ratios to do off-policy estimation

Offline RL Taxonomy



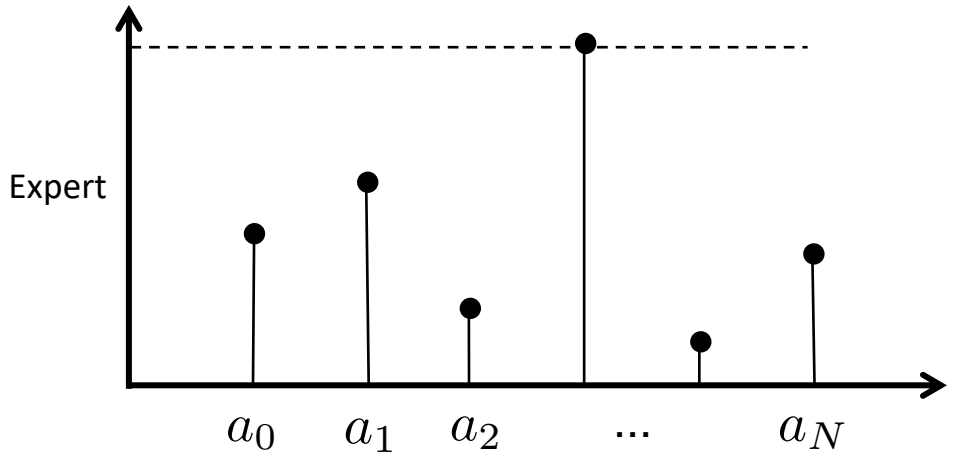
Importance Sampling

Policy Constraint Methods

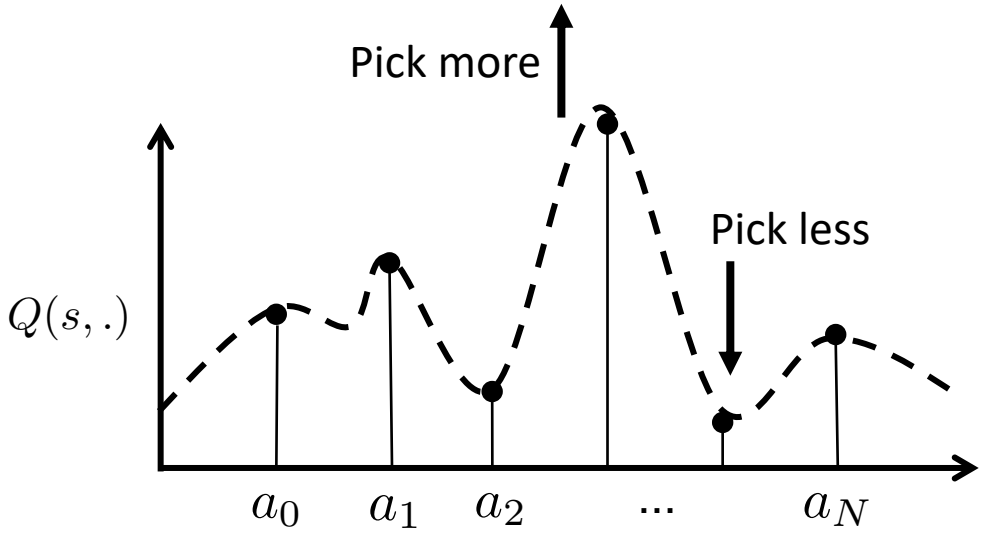
Lower Bounded Q Values

Off-Policy Bootstrapping from Suboptimal Data

Imitation learning no longer works as performance may be arbitrarily suboptimal!



Actions for s



Actions for s

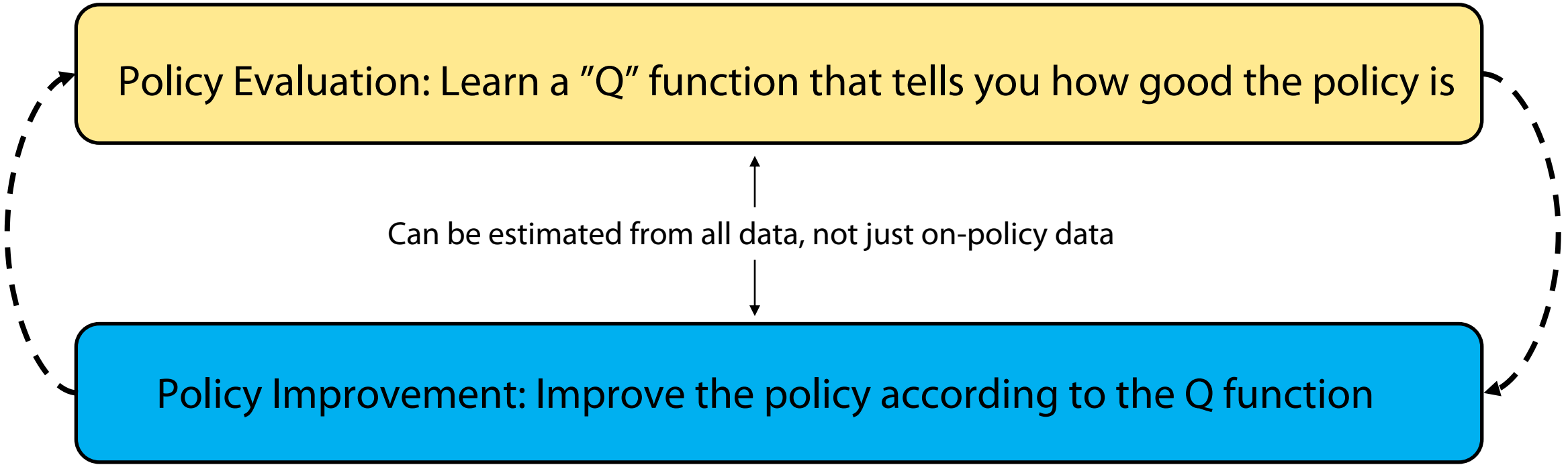
If we were able to estimate long term "goodness" of actions, we can use them to pick actions

↓

Q-function

Recap: Off-Policy RL

Off-Policy RL provides a way to use more than just latest on-policy data



Q-learning / soft-actor critic / AlphaZero / Value Iteration

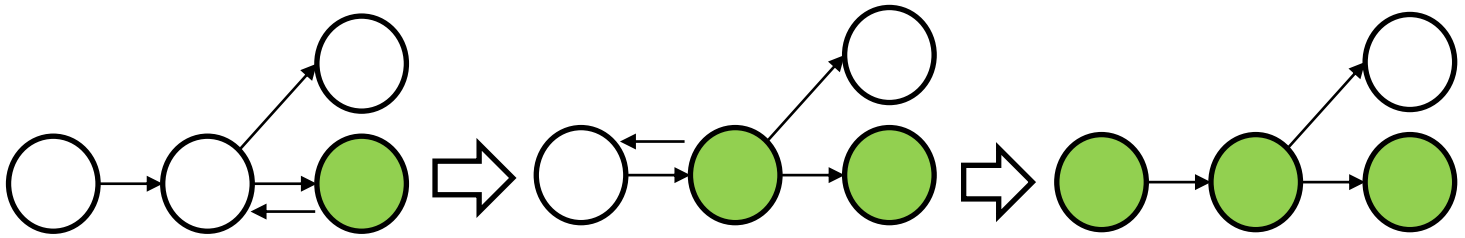
Recap: Off-Policy RL – Policy Evaluation

Policy Evaluation: Learn a "Q" function that tells you how good the policy is

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{\pi_\theta}\left[\sum_{t'=t}^{T} r(s'_t, a'_t)|s_t, a_t\right]$$

$$Q^{\pi}(s_t, a_t) = r(s_t, a_t) + \mathbb{E}_{\substack{s_{t+1}\sim p(.|s_t,a_t) \\ a_{t+1}\sim\pi_\theta(.|s_{t+1})}}\left[Q^{\pi}(s_{t+1}, a_{t+1})\right] \qquad \text{(Bellman)}$$

$$\min_{\phi} \mathbb{E}_{(s_t,a_t,s_{t+1})\sim\mathcal{D}}\left(Q^{\pi}_{\phi}(s_t, a_t) - (r(s_t, a_t) + Q^{\pi}_{\hat{\phi}}(s_{t+1}, a_{t+1}))\right)^2 \quad a_{t+1} \sim \pi(\cdot|s_{t+1})$$



Intuition: measure of how good an action a is if it is played + π is followed after

Recap: Off-Policy RL – Policy Improvement

Policy Improvement: Improve the policy according to the Q function

Choose a such that action is maximized against the Q-function

$$\max_{\theta} \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi_\theta(a|s)} \left[Q^{\pi_\theta}(s, a) \right]$$

Intuition: pick the best action according to score/curation function

Recap: Off-Policy RL with Online Data Collection

Policy Evaluation: Learn a "Q" function that tells you how good the policy is

$$\min_{\phi} \mathbb{E}_{(s_t,a_t,s_{t+1})\sim\mathcal{D}} \left(Q^{\pi}_{\phi}(s_t,a_t) - (r(s_t,a_t) + Q^{\pi}_{\hat{\phi}}(s_{t+1},a_{t+1}))\right)^2 \quad a_{t+1}\sim\pi(\cdot|s_{t+1})$$

Policy Improvement: Improve the policy according to the Q function

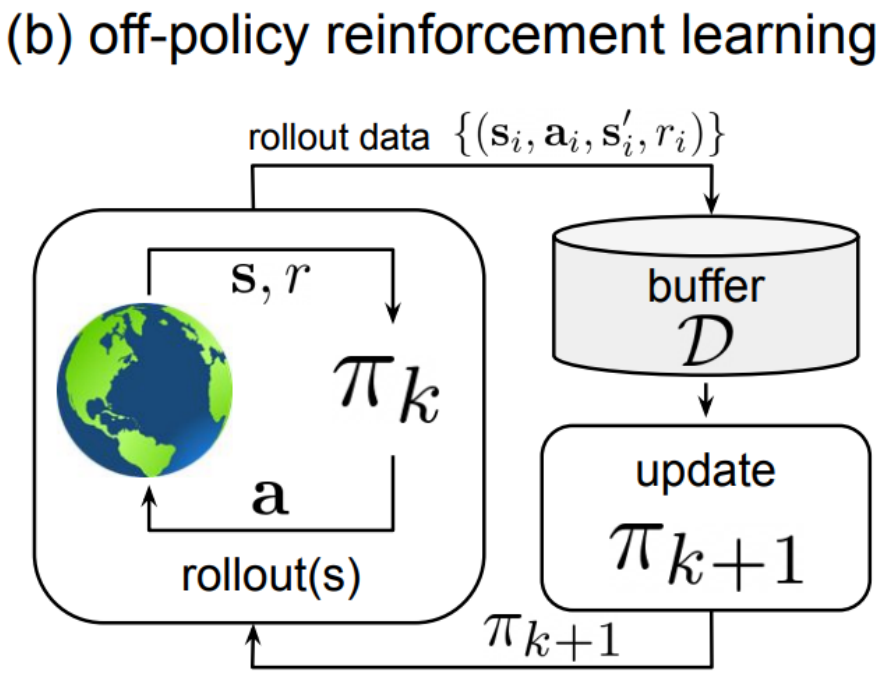
$$\max_{\pi} \mathbb{E}_{s\sim\mathcal{D}}\mathbb{E}_{a\sim\pi(.|s)}\left[Q^{\pi}(s,a)\right]$$

Data Collection with Updated Policy

Can we use off-policy methods to perform **offline** RL?

No data collection

Attempt 1: Offline Off-Policy RL



Can try directly adding offline data to $\mathcal{D}$

Train by sampling from $\mathcal{D}$ (no sampling in env):

1. Add offline data to the replay buffer

2. Minimize Bellman Equation

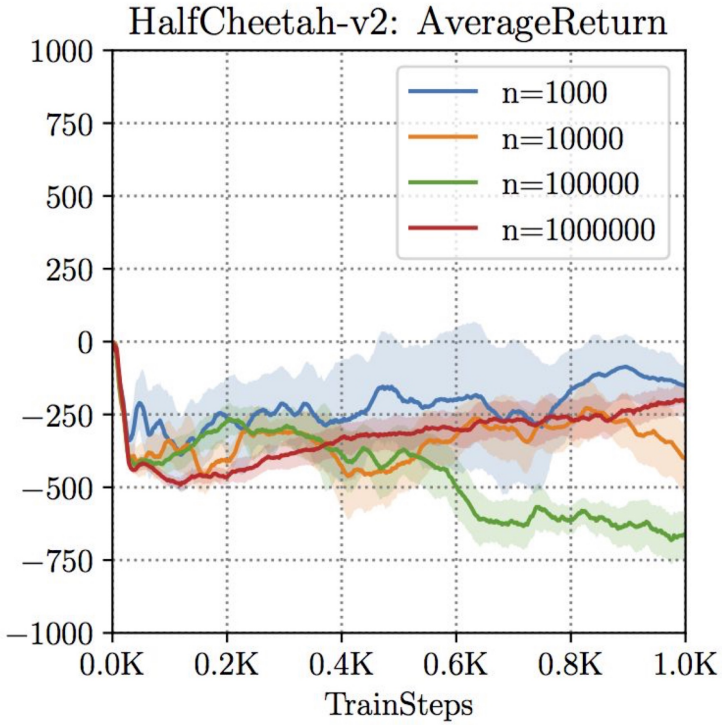
$$\min(Q(s,a) - (r(s,a) + \gamma \max_{a'} Q(s',a')))^2$$

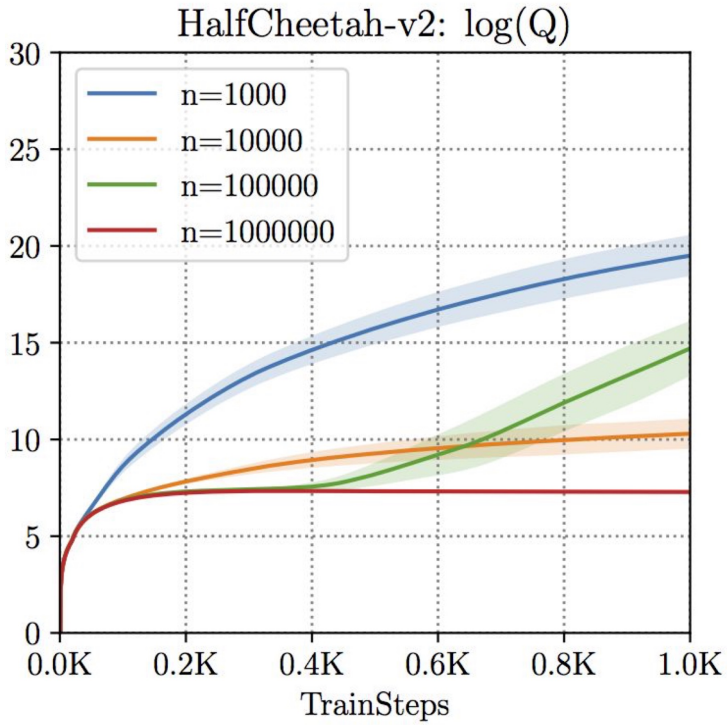
3. Optimize actor $\pi(a|s)$ wrt $Q^\pi(s,a)$

$$\pi \leftarrow \arg\max_\pi \mathbb{E}_\pi \left[Q^\pi\right]$$

Attempt 1: Offline Off-Policy RL

Empirical performance with vanilla off-policy RL on offline data





- Returns don't increase but Q-values diverge
- Not classical overfitting!
- More data does not improve performance

Divergence because of distribution shift

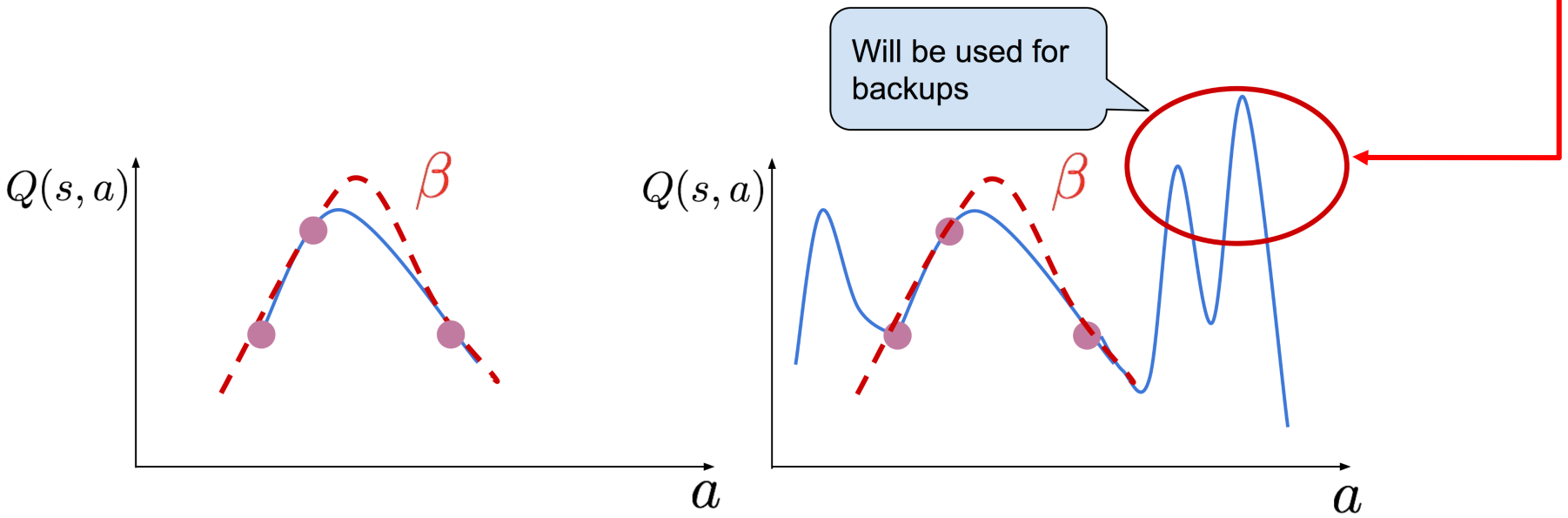
Visuals from Kumar et al

Distribution shift in Offline RL

$$Q^* = \mathcal{T}^* Q^* \quad ; \quad (\mathcal{T}^* \hat{Q})(s,a) := R(s,a) + \gamma \mathbb{E}_{T(s'|s,a)} \left[\max_{a'} \hat{Q}(s', a') \right]$$

Can bootstrap on OOD actions. Q can be arbitrarily overestimated

$$Q := \arg\min_{\hat{Q}} \mathbb{E}_{s \sim \mathcal{D}, a \sim \beta(a|s)} \left[\left(\hat{Q}(s,a) - (\mathcal{T}^* \hat{Q})(s,a) \right)^2 \right]$$



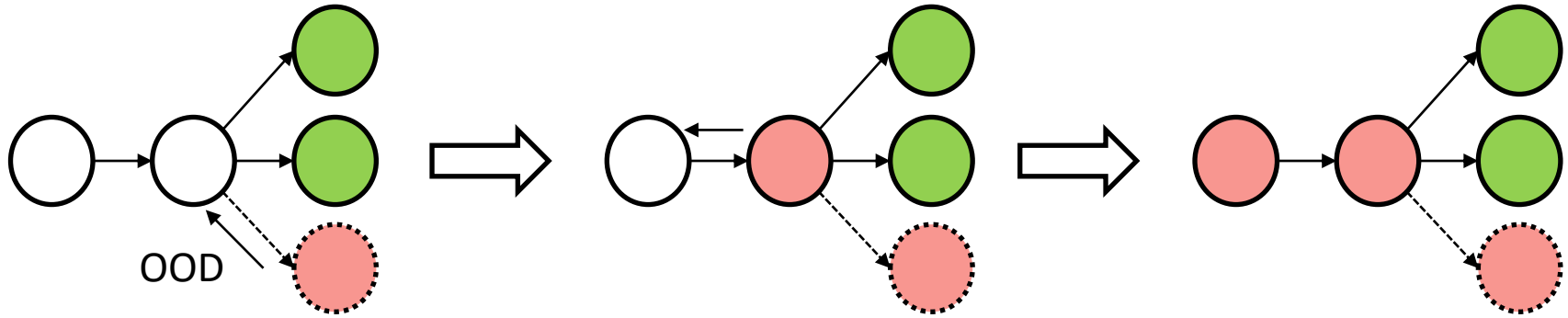
Q-values on training data

Q-values being backed up

Visuals from Kumar et al

Why is distribution shift problematic in Offline RL?

- When Q is trained on $\pi_\beta(a|s)$, it may not be accurate for arbitrary $\pi(a|s)$
  - Some a, s may just be very OOD/out of support.
- Overestimated Q-values can continue to be backed up erroneously.

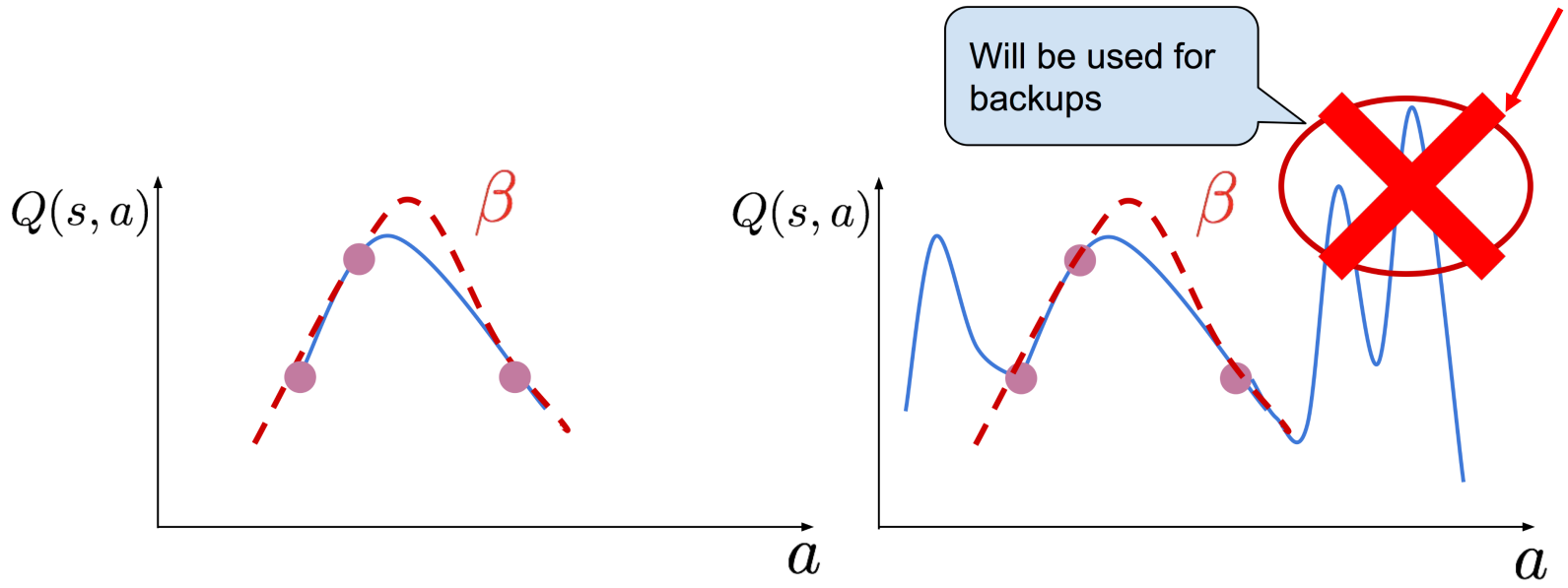


Online data collection corrects over-optimistic Q, but not so offline!

Tackling distribution shift in offline RL

Key Idea for controlling distribution shift in offline RL: (minimize choosing OOD actions)

- Policy constraint methods
- Lower bounding returns/Q-values



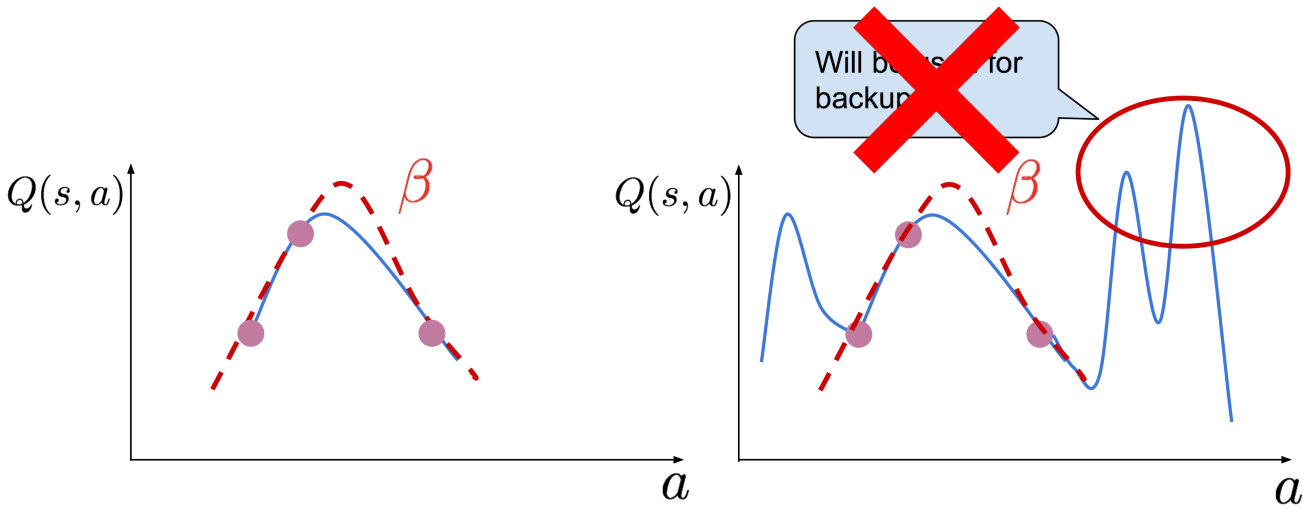
Attempt 2: Policy Constraint Algorithms

Idea: Constrain the actor update to remain close to the behavior policy.

$$\hat{Q}^{\pi}_{k+1} \leftarrow \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}} \left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}'\sim\pi_k(\mathbf{a}'|\mathbf{s}')}[\hat{Q}^{\pi}_{k}(\mathbf{s}',\mathbf{a}')] \right) \right)^2 \right]$$

$$\pi_{k+1} \leftarrow \arg\max_{\pi} \mathbb{E}_{\mathbf{s}\sim\mathcal{D}} \left[\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^{\pi}_{k+1}(\mathbf{s},\mathbf{a})] \right] \text{ s.t. } D(\pi,\pi_\beta) \leq \epsilon.$$

Why would this work?



OOD actions violate the constraint

$$\pi(a|s) \approx \pi_\beta(a|s)$$

Policy Constraint Algorithms

Different forms of the constraint and optimization leads to different algorithms

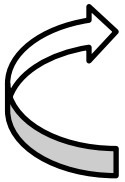
Constraint Form	Constraint Objective	Resulting Algorithm
Support matching	$D = \mathrm{MMD}(\pi_\beta, \pi)$	(Kumar et al. 2019, Laroche et al. 2019, Wu et al. 2019)
Distribution Matching	$D = D_{KL}(\pi \| \pi_\beta)$	BCQ (Fujimoto et al), (Jaques et al 2019), BRAC (Wu et al)
State-marginal constraints	$D = D_{KL}(d^\pi \| d^{\pi_\beta})$	AlgaeDICE (Nachum et al)
Implicit distribution constraints	$D = D_{KL}(\pi \| \pi_\beta)$	AWR (Peng et al), AWAC (Nair et al), CRR (Wang et al)

Implementation of Policy Constraint Algorithms

Pseudocode of a policy constraint algorithm:

1. Add offline data to the replay buffer

2. Minimize Bellman Equation



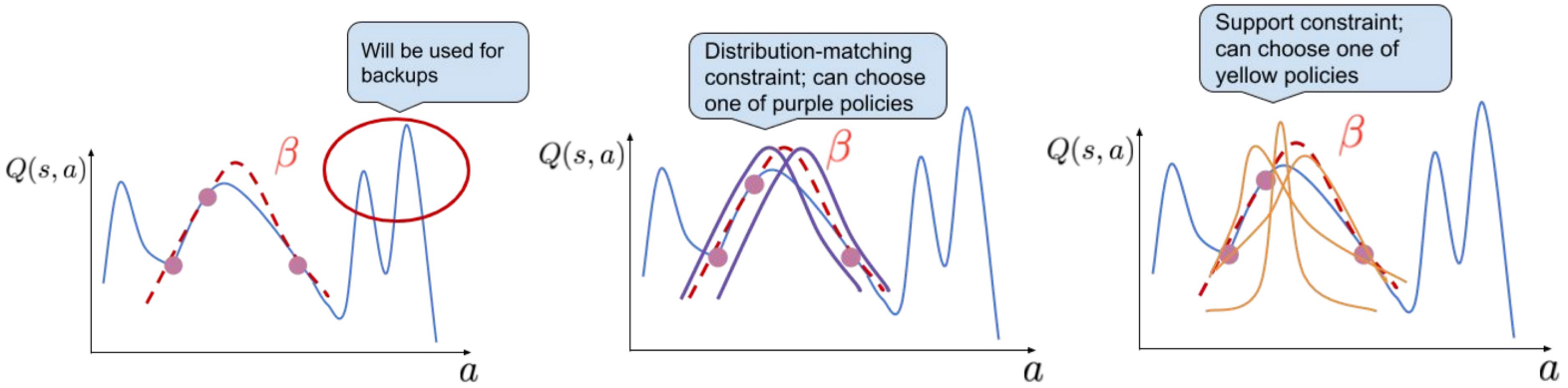
$$\hat{Q}^{\pi}_{k+1} \leftarrow \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}'\sim\pi_k(\mathbf{a}'|\mathbf{s}')}[\hat{Q}^{\pi}_{k}(\mathbf{s}',\mathbf{a}')] - \alpha\gamma D(\pi_k(\cdot|\mathbf{s}'), \pi_\beta(\cdot|\mathbf{s}'))\right)\right)^2\right]$$

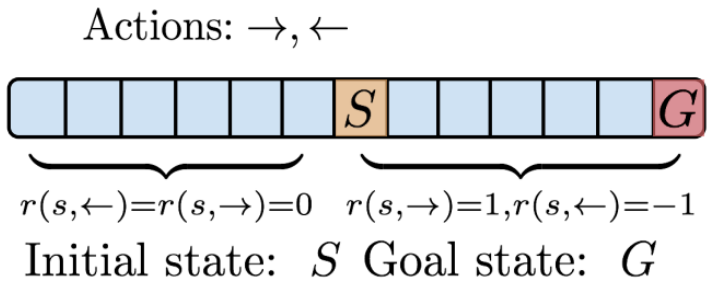
3. Optimize actor $\pi(a|s)$ wrt $Q^{\pi}(s,a)$

$$\pi_{k+1} \leftarrow \arg\max_{\pi} \mathbb{E}_{\mathbf{s}\sim\mathcal{D}}\left[\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^{\pi}_{k+1}(\mathbf{s},\mathbf{a})] - \alpha D(\pi(\cdot|\mathbf{s}), \pi_\beta(\cdot|\mathbf{s}))\right].$$

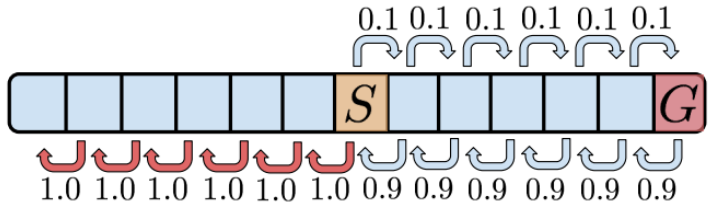
Requires estimation of behavior policy π_β

Tradeoffs between Policy Constraints

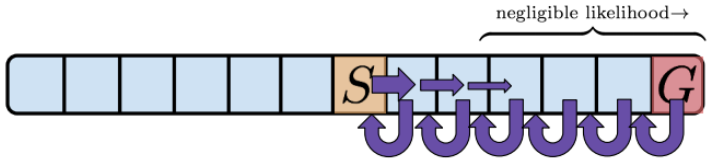




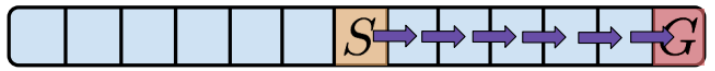
(a) 1D-Lineworld Environment



(b) Behavior Policy



(c) Learned Policy via distribution-matching



(d) Learned Policy via support-constraint

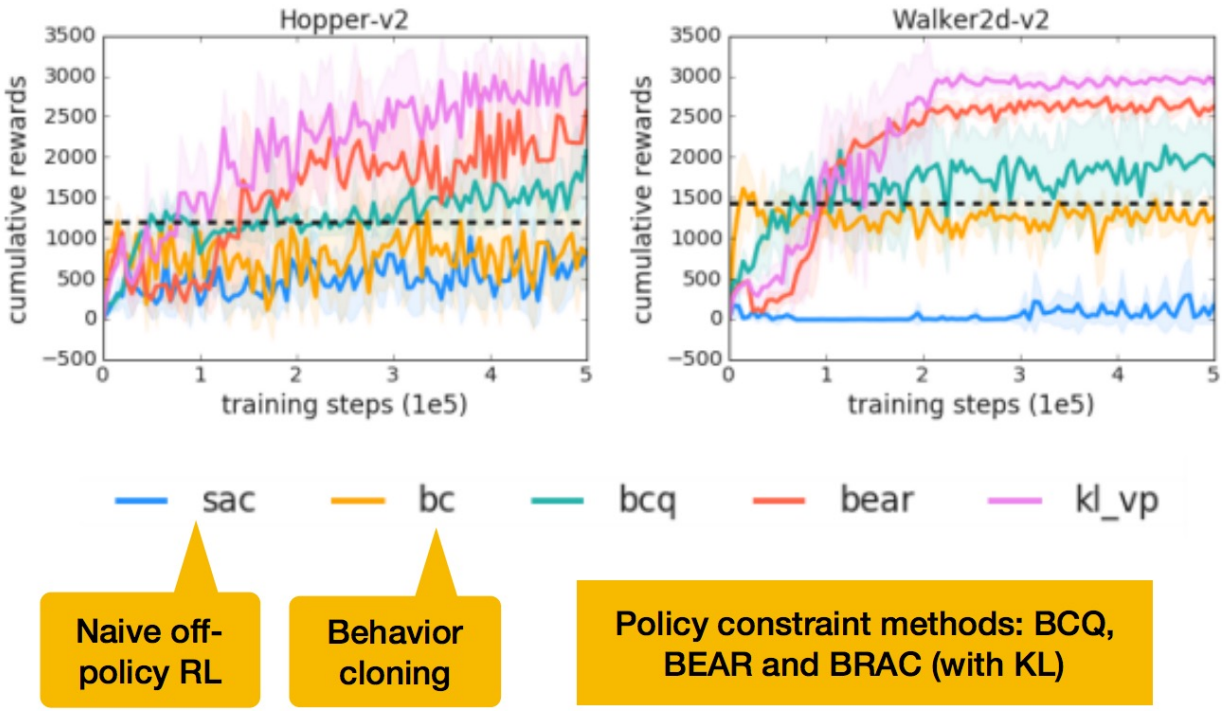
Support constraints are less pessimistic, works across behavior distributions

Visuals from A. Kumar

Performance of Policy Constraint Algorithms

How well does this work?

- Works significantly better than off-policy RL
- Not good on harder tasks, too pessimistic



Slide from A Kumar

Challenges with Policy Constraint Algorithms

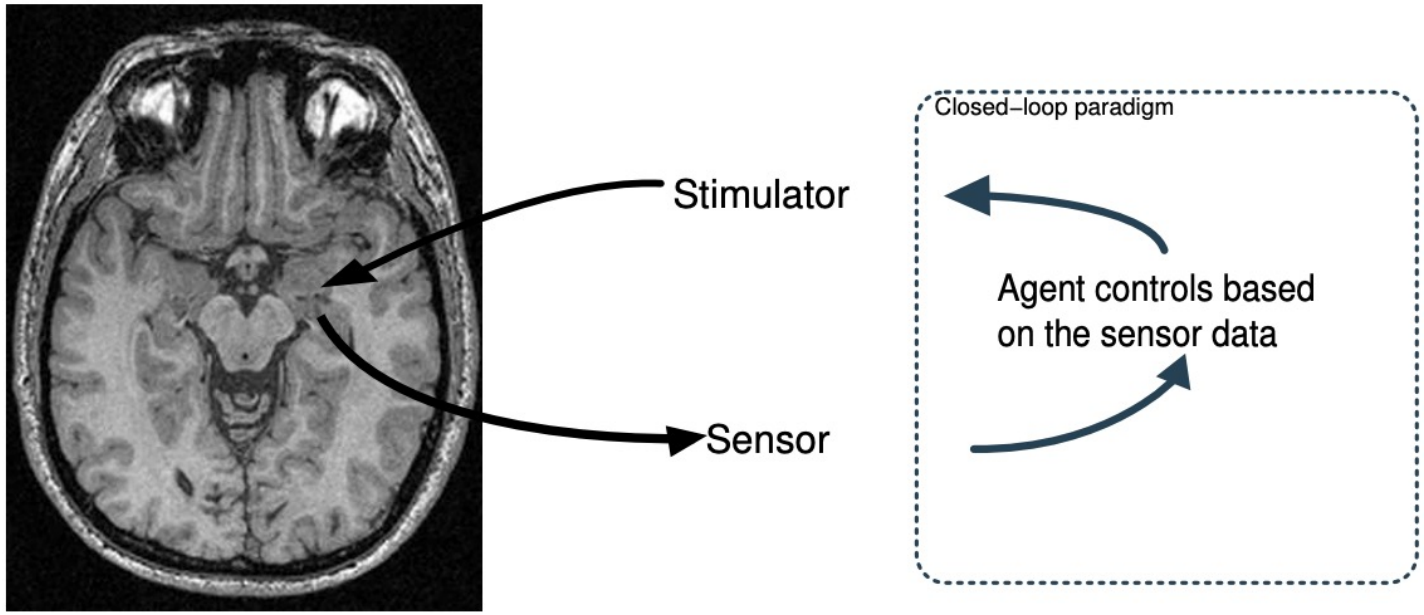
$$\hat{Q}^{\pi}_{k+1} \leftarrow \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}} \left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}'\sim\pi_k(\mathbf{a}'|\mathbf{s}')}[\hat{Q}^{\pi}_{k}(\mathbf{s}',\mathbf{a}')] \right) \right)^2 \right]$$

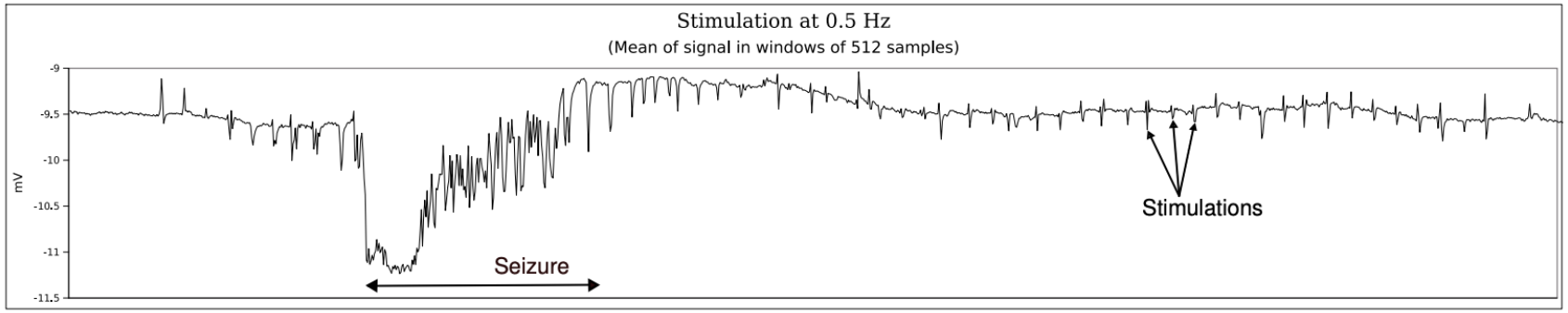
$$\pi_{k+1} \leftarrow \arg\max_{\pi} \mathbb{E}_{\mathbf{s}\sim\mathcal{D}} \left[\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^{\pi}_{k+1}(\mathbf{s},\mathbf{a})] \right] \text{ s.t. } D(\pi, \pi_\beta) \leq \epsilon.$$

1. Requires challenging estimation of behavior policy
2. Constraint can often be **too** pessimistic

Applications of Policy-Constrained ORL: Epilepsy treatment

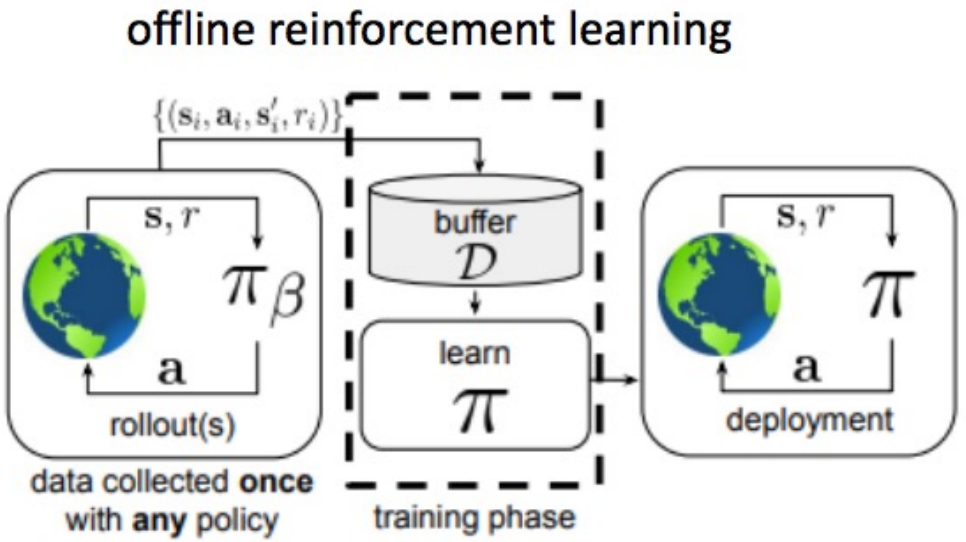
Adaptive strategy learned to minimize the frequency and duration of epilepsy seizures.





Constrain policies to avoid OOD exploitation during off-policy RL (“conservatism”)

Offline RL Taxonomy



Importance Sampling

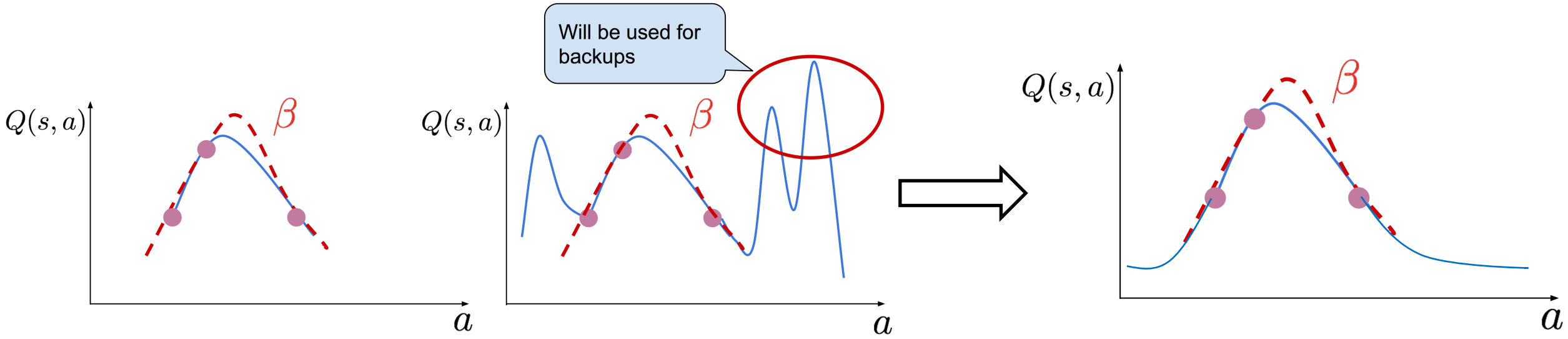
Policy Constraint Methods

Lower Bounded Q Values

Attempt 3: Lower Bounding Q-Values

Idea: Push down over-optimistic Q-values, rather than completely avoiding OOD

- Less pessimistic than avoiding all OOD actions. Some OOD actions may not be bad!



Model-Free Lower Bounding Q-Values

Conservative Q-Learning:

Reduce overestimation by forcing them to lower bound the true Q-value

Push down big Q values

Minimize Bellman Equation

$$\min_{Q} \max_{\mu} \alpha \mathbb{E}_{s \sim \mathcal{D}, a \sim \mu(a|s)} \left[Q(s,a) \right] + \mathbb{E}_{(s,a,s') \sim \mathcal{D}} \left[\left(Q(s,a) - \left(r(s,a) + \mathbb{E}_{\pi} \left[Q(s',a') \right] \right) \right)^2 \right]$$

$$\hat{Q}^{\pi}(s,a) \leq Q^{\pi}(s,a), \forall s, a$$

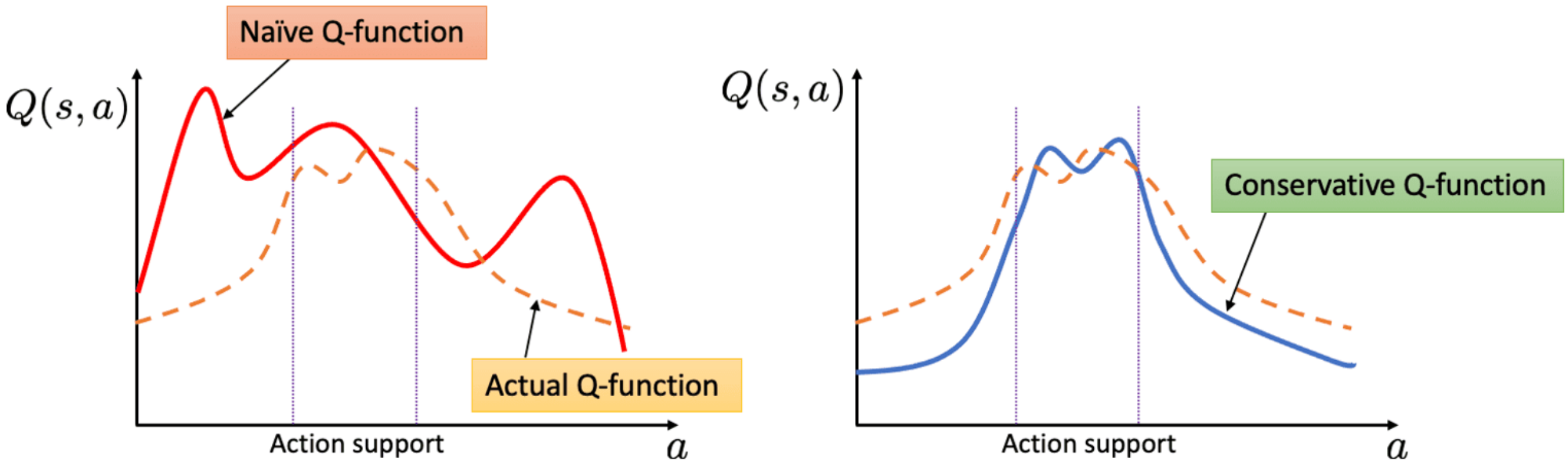
Conservative Q Learning, Kumar et al 2020

Model-Free Lower Bounding Q-Values

Conservative Q-Learning:

Reduce overestimation by forcing them to lower bound the true Q-value

$$\hat{Q}^{\pi}(s,a) \leq Q^{\pi}(s,a), \forall s, a$$

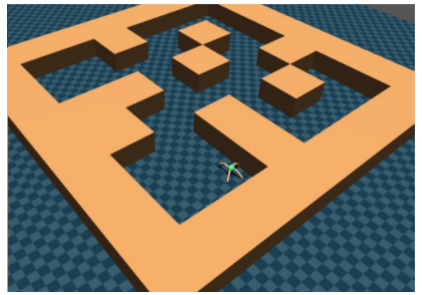


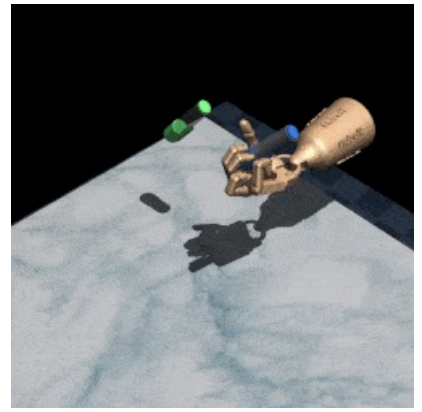
Conservative Q Learning, Kumar et al 2020

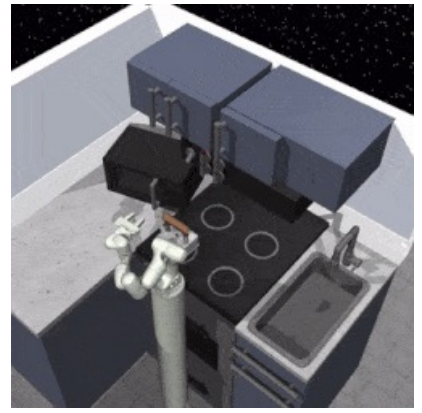
How well does this do?

Performs surprisingly well!

Domain	Task Name	BC	SAC	BEAR	BRAC-p	BRAC-v	CQL($\mathcal{H}$)	CQL(ρ)
AntMaze	antmaze-umaze	65.0	0.0	**73.0**	50.0	70.0	**74.0**	**73.5**
	antmaze-umaze-diverse	55.0	0.0	61.0	40.0	70.0	**84.0**	61.0
	antmaze-medium-play	0.0	0.0	0.0	0.0	0.0	**61.2**	4.6
	antmaze-medium-diverse	0.0	0.0	8.0	0.0	0.0	**53.7**	5.1
	antmaze-large-play	0.0	0.0	0.0	0.0	0.0	**15.8**	3.2
	antmaze-large-diverse	0.0	0.0	0.0	0.0	0.0	**14.9**	2.3
Adroit	pen-human	34.4	6.3	-1.0	8.1	0.6	37.5	**55.8**
	hammer-human	1.5	0.5	0.3	0.3	0.2	**4.4**	2.1
	door-human	0.5	3.9	-0.3	-0.3	-0.3	**9.9**	**9.1**
	relocate-human	0.0	0.0	-0.3	-0.3	-0.3	0.20	**0.35**
	pen-cloned	**56.9**	23.5	26.5	1.6	-2.5	39.2	40.3
	hammer-cloned	0.8	0.2	0.3	0.3	0.3	2.1	**5.7**
	door-cloned	-0.1	0.0	-0.1	-0.1	-0.1	0.4	**3.5**
	relocate-cloned	**-0.1**	-0.2	-0.3	-0.3	-0.3	**-0.1**	**-0.1**
Kitchen	kitchen-complete	33.8	15.0	0.0	0.0	0.0	**43.8**	31.3
	kitchen-partial	33.8	0.0	13.1	0.0	0.0	**49.8**	**50.1**
	kitchen-undirected	47.5	2.5	47.2	0.0	0.0	**51.0**	**52.4**







Underestimates the true Q-values

Task Name	CQL($\mathcal{H}$)	CQL (Eqn. 1)	Ensemble(2)	Ens.(4)	Ens.(10)	Ens.(20)	BEAR
hopper-medium-expert	**-43.20**	-151.36	3.71e6	2.93e6	0.32e6	24.05e3	65.93
hopper-mixed	**-10.93**	-22.87	15.00e6	59.93e3	8.92e3	2.47e3	1399.46
hopper-medium	**-7.48**	-156.70	26.03e12	437.57e6	1.12e12	885e3	4.32

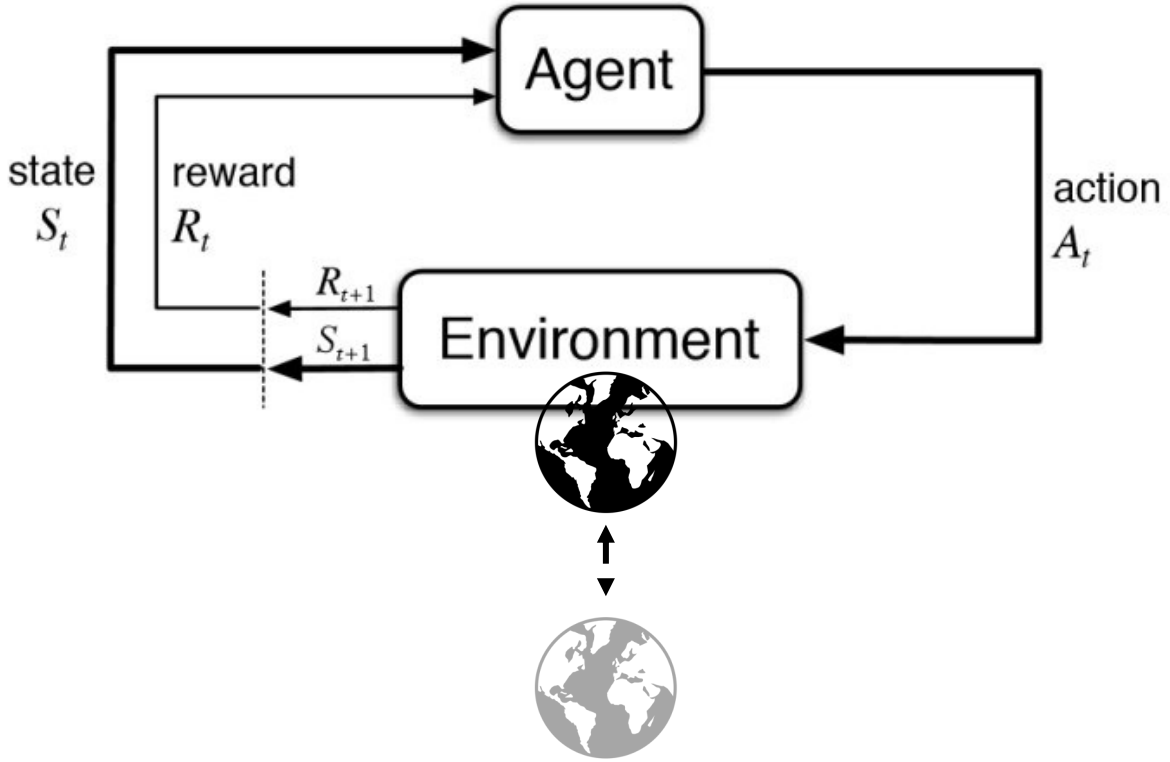
Difference between estimated Q and true Q

Conservative Q Learning, Kumar et al 2020

Extension: Model-Based Offline RL

Can we extrapolate past the offline data?

Predictive “world” models can generalize past experienced states, actions.

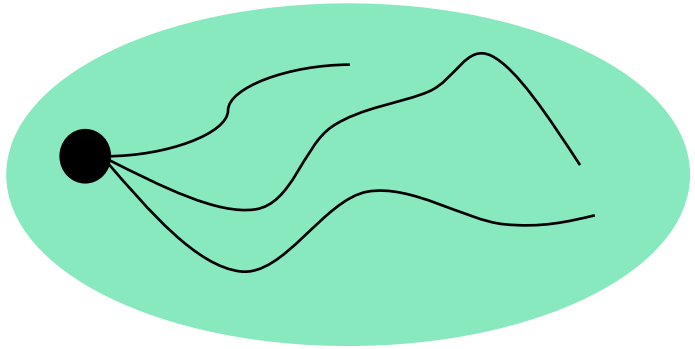


Learn a predictive model of the world from offline data

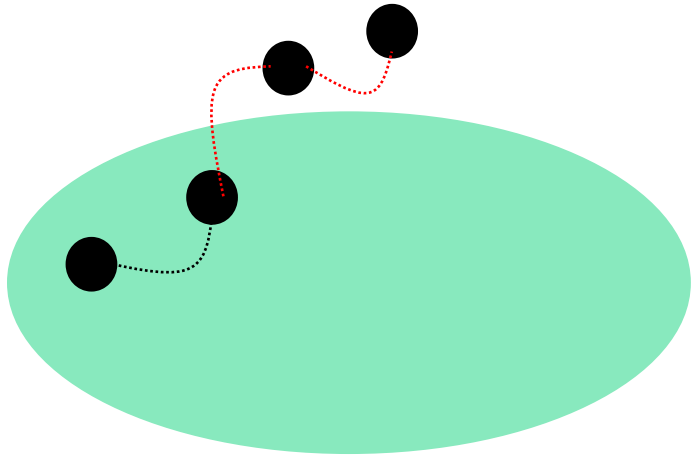
Optimize behavior in “imagination”

Extension: Model-Based Offline RL

What is the problem with using models for offline RL?



Experienced Data



Erroneously Overoptimistic Planning

Same problem of relying on overoptimistic OOD values!

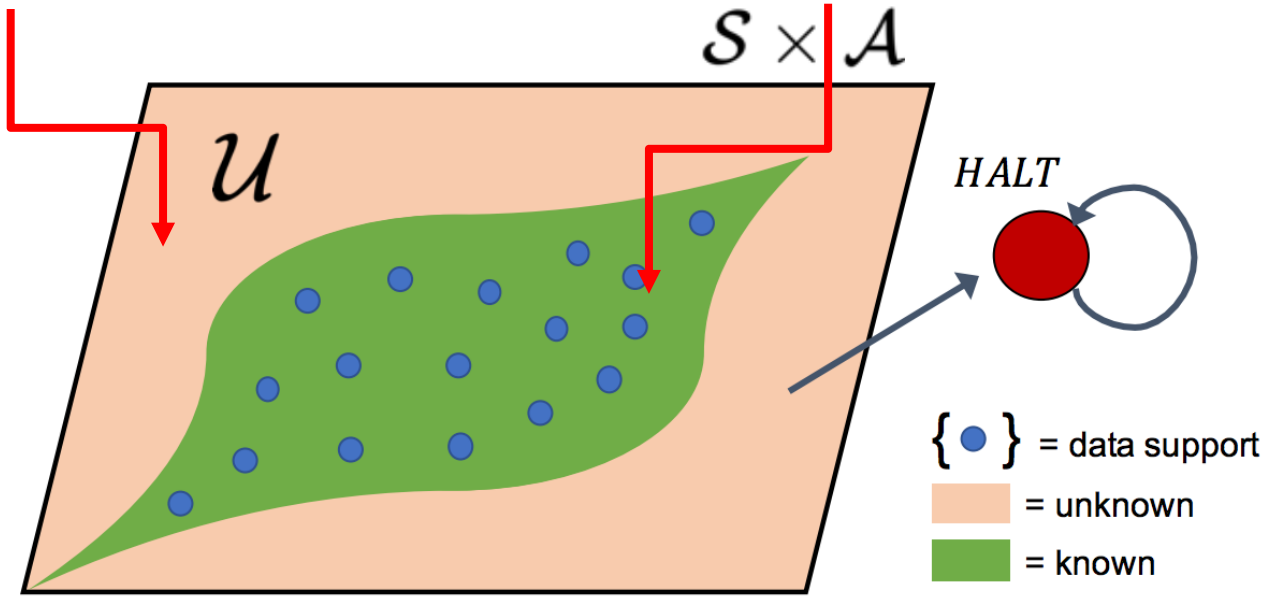
Model-Based Lower Bounding Q-Values

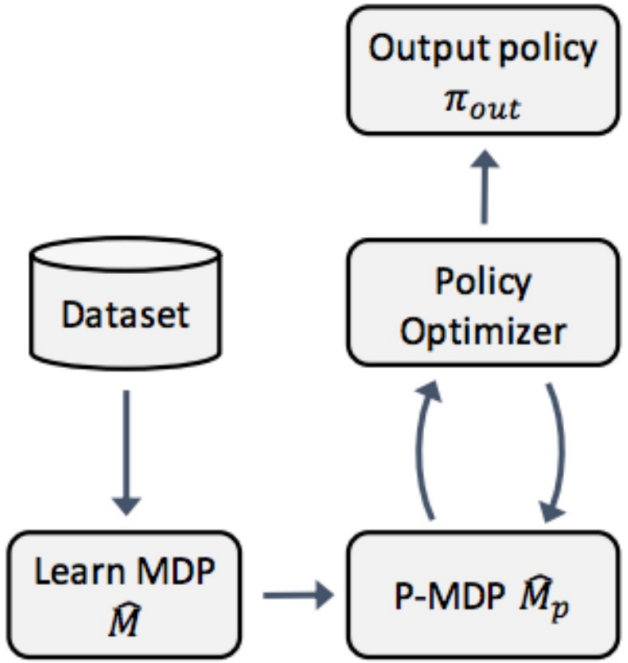
Idea: Down-weight rewards for uncertain model transitions to avoid overoptimism

Modified pessimistic MDP with $r(s,a) = r(s,a) - \lambda u(s,a)$

Penalize OOD (s, a) via reward

Leave in distribution r unchanged

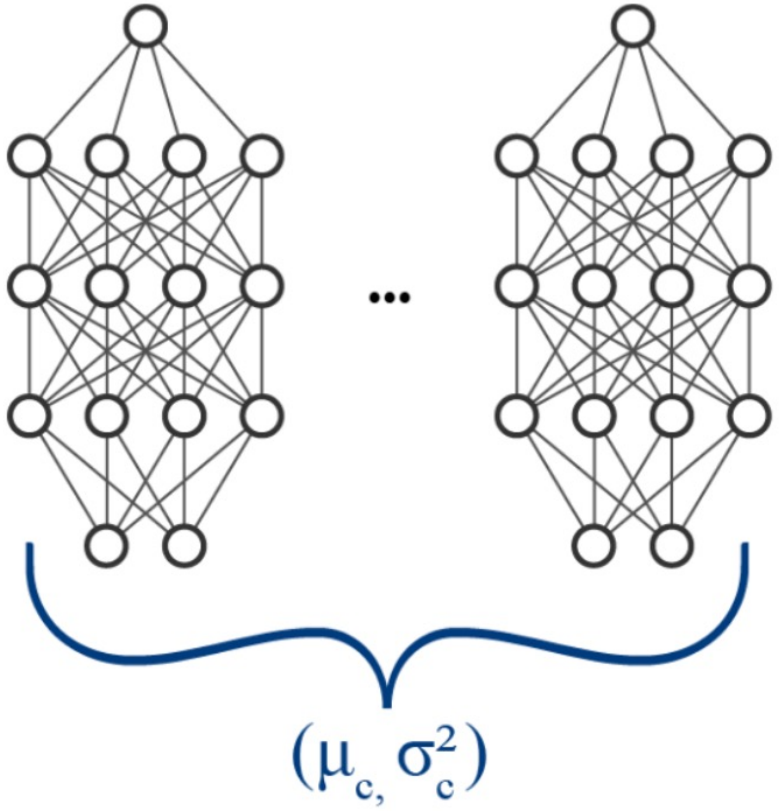




MoREL (Kidambi et al 2020), MOPO (Yu et al 2020)

Model-Based Lower Bounding Q-Values

How to implement $u(s, a)$?



MOPO (Yu et al):

$$\tilde{r} = r_j - \lambda \max_{i=1}^{N} \left\| \Sigma^i(s_j, a_j) \right\|_F$$

MoREL (Kidambi et al):

$$\text{disc}(s, a) = \max_{i,j} \left\| f_{\phi_i}(s, a) - f_{\phi_j}(s, a) \right\|_2$$

$$\tilde{r}(s, a) = -R_{\max} \quad \text{if} \quad \text{disc}(s, a) > \text{threshold}$$

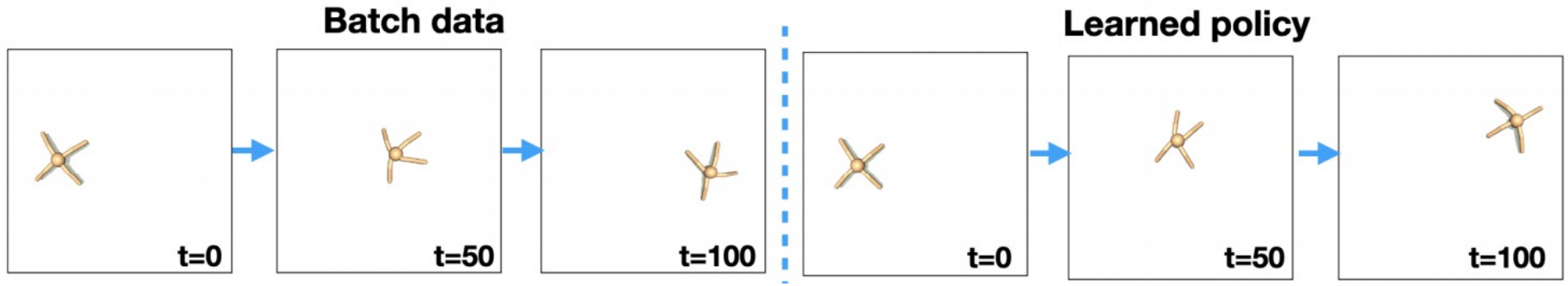
MoREL (Kidambi et al 2020), MOPO (Yu et al 2020)

Model-Based Lower Bounding Q-Values

Works well on random data and mixed data, but less so on medium data

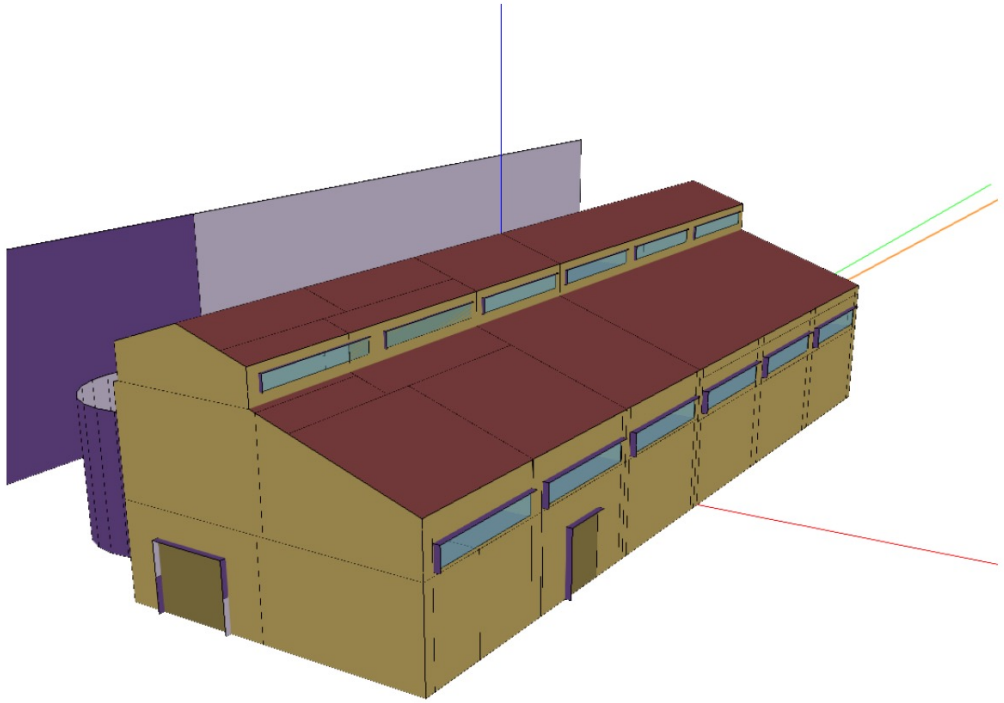
Dataset type	Environment	BC	MOPO (ours)	MBPO	SAC	BEAR	BRAC-v
random	halfcheetah	2.1	**31.9** ± 2.8	30.7 ± 3.9	30.5	25.5	28.1
random	hopper	1.6	**13.3** ± 1.6	4.5 ± 6.0	11.3	9.5	12.0
random	walker2d	9.8	**13.0** ± 2.6	8.6 ± 8.1	4.1	6.7	0.5
medium	halfcheetah	36.1	40.2 ± 2.7	28.3 ± 22.7	-4.3	38.6	**45.5**
medium	hopper	29.0	26.5 ± 3.7	4.9 ± 3.3	0.8	**47.6**	32.3
medium	walker2d	6.6	14.0 ± 10.1	12.7 ± 7.6	0.9	33.2	**81.3**
mixed	halfcheetah	38.4	**54.0** ± 2.6	47.3 ± 12.6	-2.4	36.2	45.9
mixed	hopper	11.8	**92.5** ± 6.3	49.8 ± 30.4	1.9	10.8	0.9
mixed	walker2d	11.3	**42.7** ± 8.3	22.2 ± 12.7	3.5	25.3	0.8
med-expert	halfcheetah	35.8	**57.9** ± 24.8	9.7 ± 9.5	1.8	51.7	45.3
med-expert	hopper	111.9	51.7 ± 42.9	**56.0** ± 34.5	1.6	4.0	0.8
med-expert	walker2d	6.4	55.0 ± 19.1	7.6 ± 3.7	-0.1	26.0	**66.6**

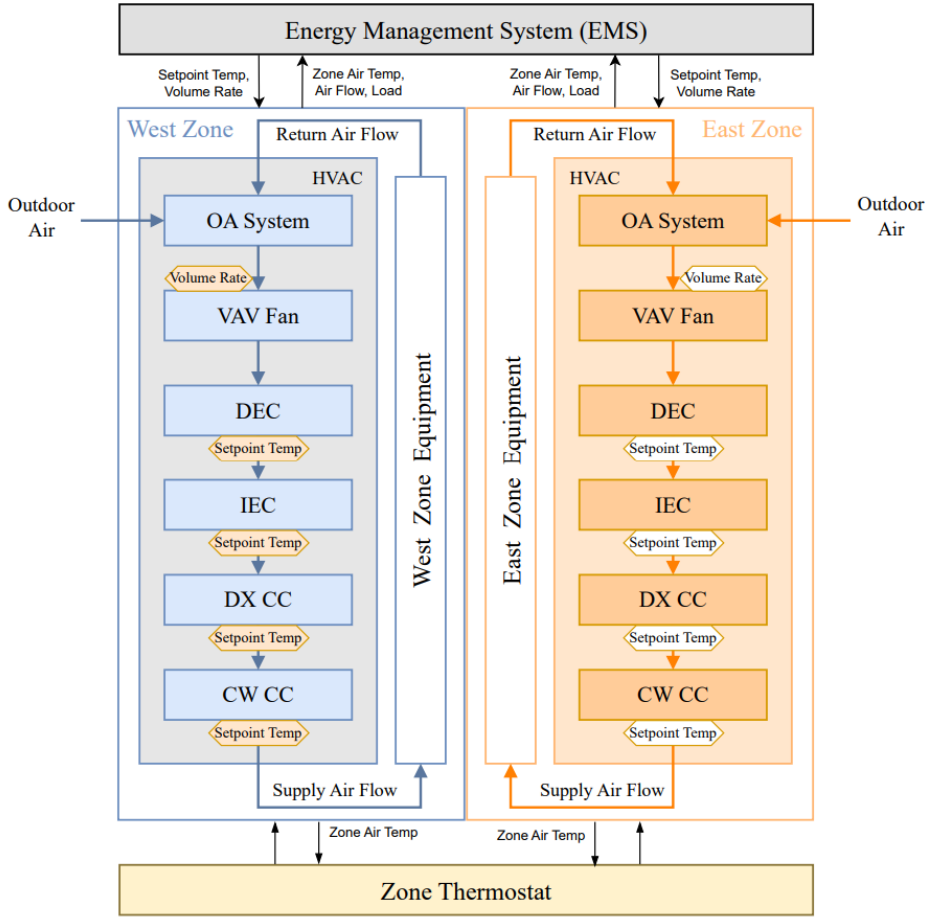
Can extrapolate beyond the offline data coverage.



MoREL (Kidambi et al 2020), MOPO (Yu et al 2020)

Applications of Lower-Bounded Q for ORL



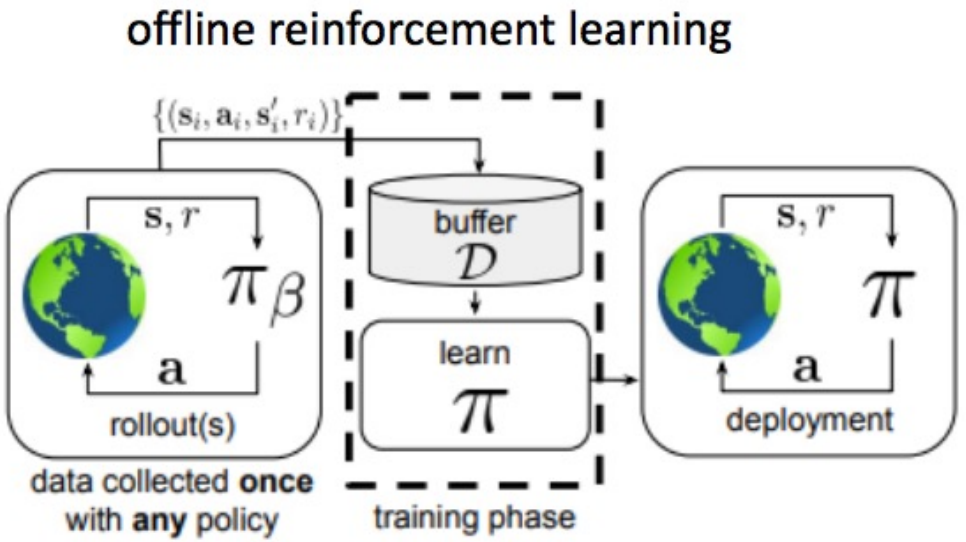


CQL was used for controlling an HVAC system
- Controls setpoint temperatures, flow rates

Experimental evaluation of offline reinforcement learning for HVAC control in buildings, Wang et al.

Push down any OOD values to avoid them getting chosen and exploited

Landscape of Offline RL methods



Importance Sampling

- Unbiased solution
- High variance

Policy Constraint Methods

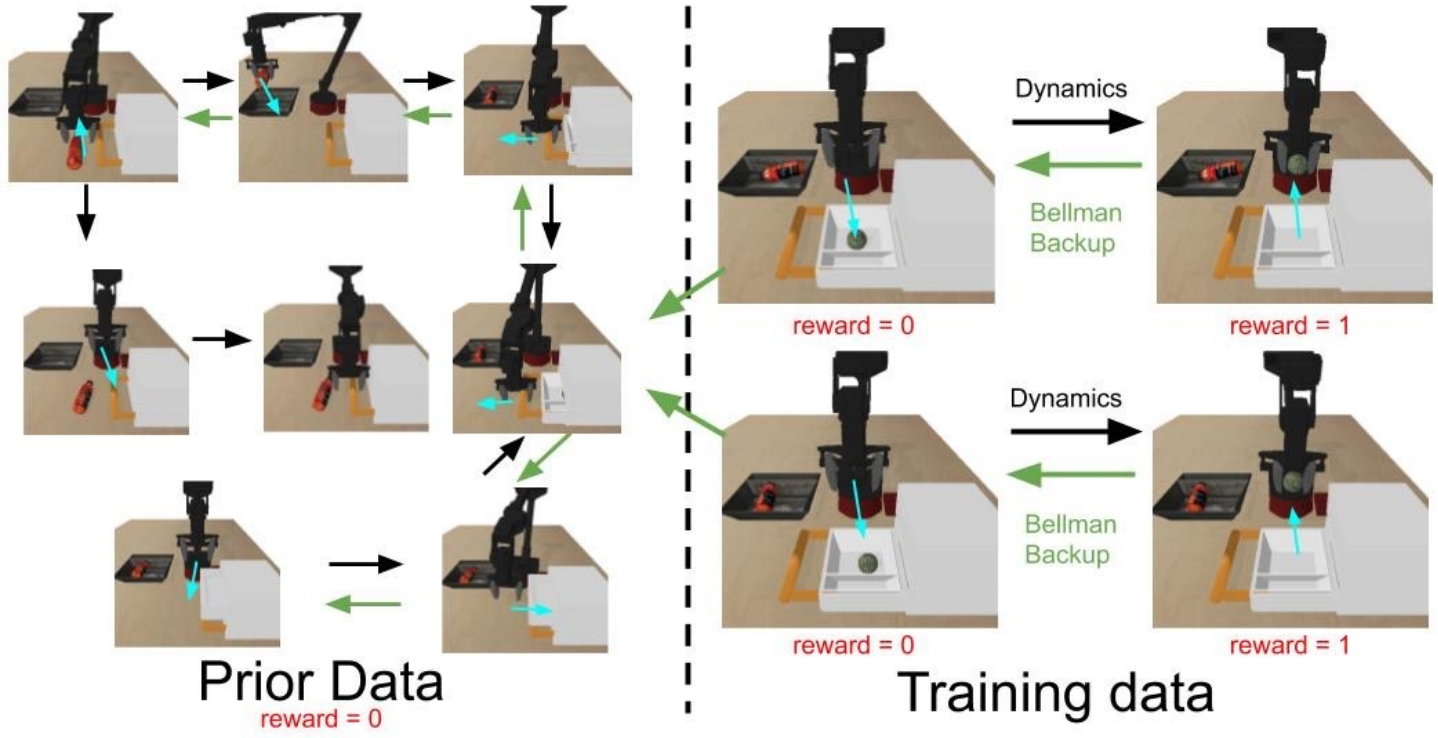
- Simple, works well offline
- Too pessimistic

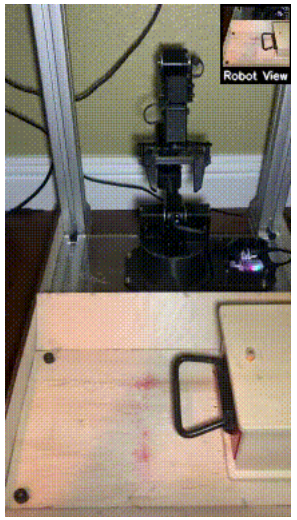
Lower Bounded Q Values

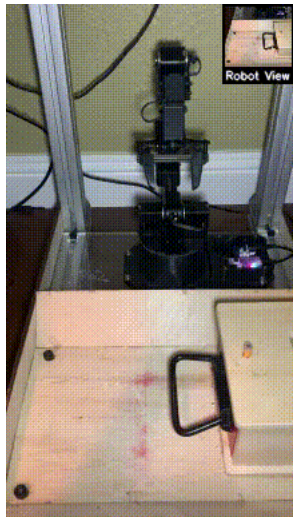
- Less conservative
- More complex min-max optimization

Offline RL on Real Problems

Robotics:







Can stitch together behaviors from prior data

COG: Connecting New Skills to Past Experience with Offline Reinforcement Learning, Singh et al 2020

Offline RL on Real Problems

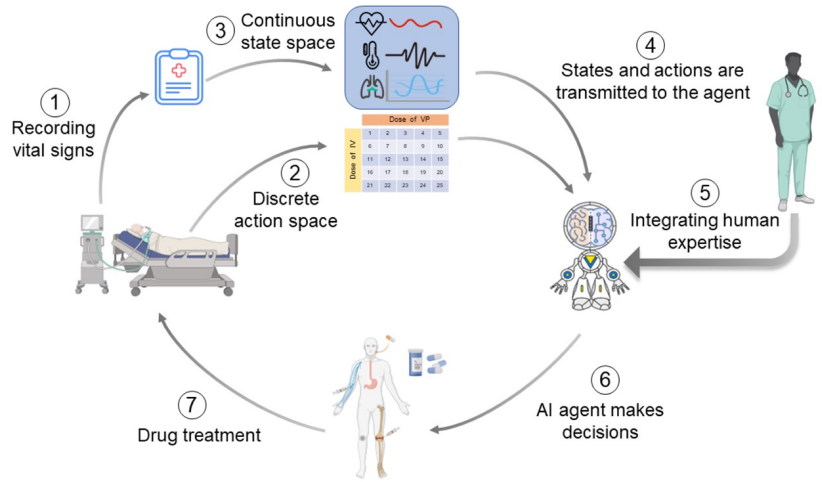
Dialogue systems:

[User]: hey, what's up?
[KL-control]: i'm not sure, but i'm glad to see you, and i'm so happy that you're not alone
[User]: oh thanks, that's very nice of you!
[KL-control]: i'm sorry, but i'm glad to see what you think.
[User]: don't be sorry, you did a good job.
[KL-control]: that's really nice to me.

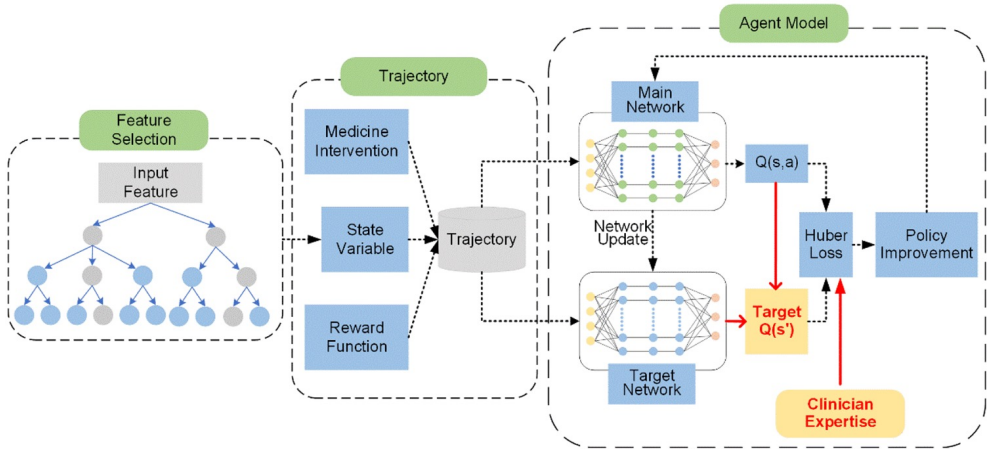
Way Off Policy Batch RL, Jaques et al 2019

Offline RL on Real Problems

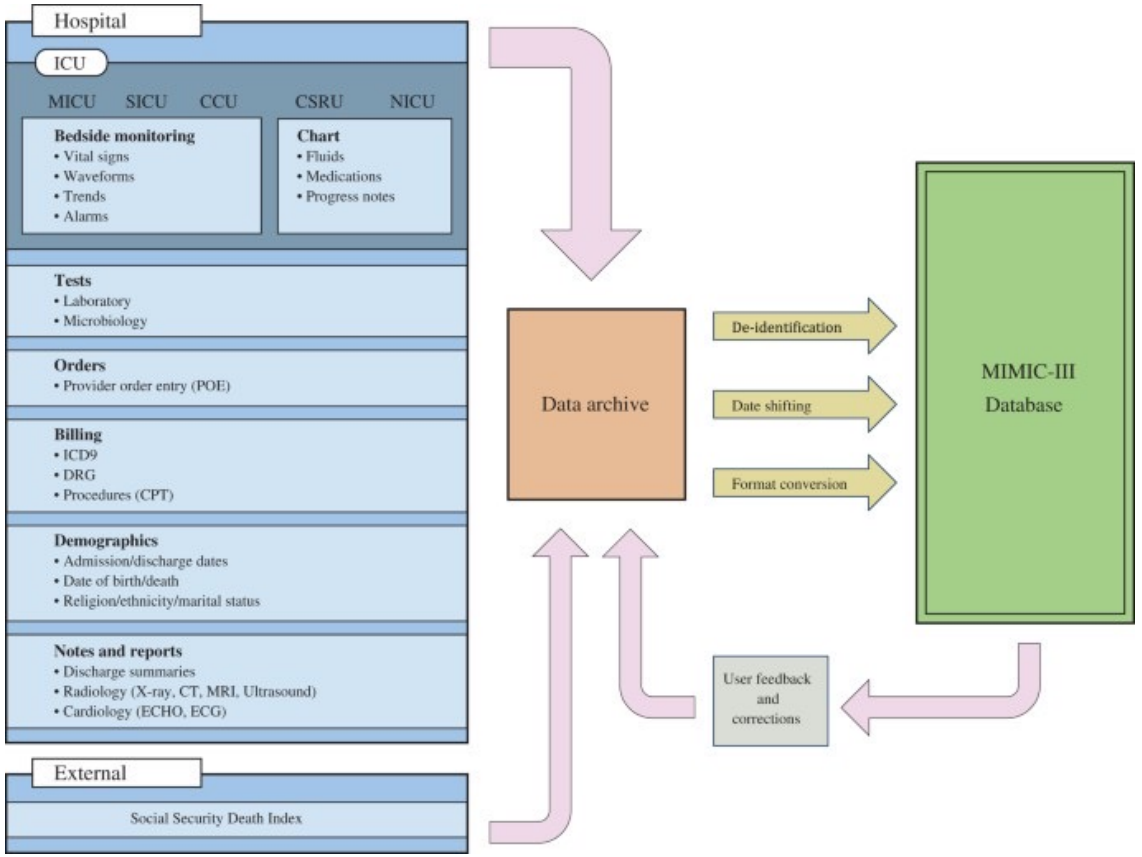
a



b



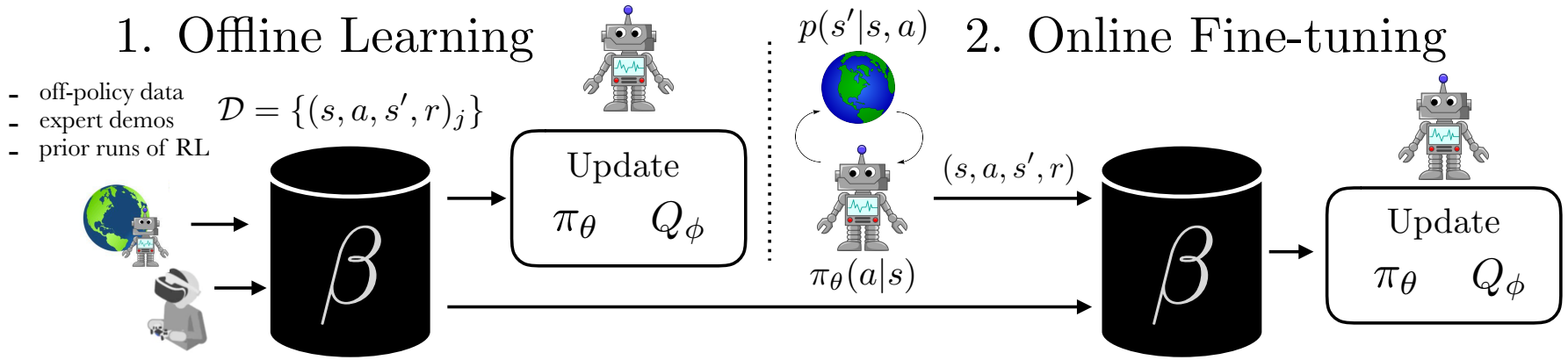
Offline RL decides the optimal solution and dose of fluid and vasopressor therapy

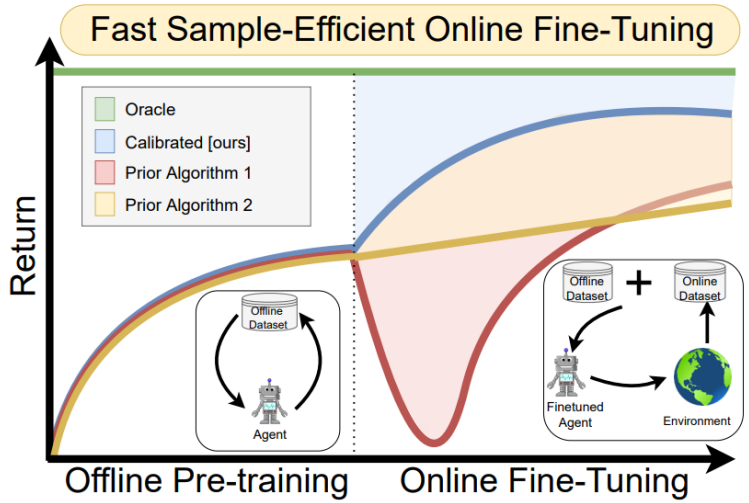


A value-based deep reinforcement learning model with human expertise in optimal treatment of sepsis, Wu et al

Some Frontiers of Offline RL: Finetuning

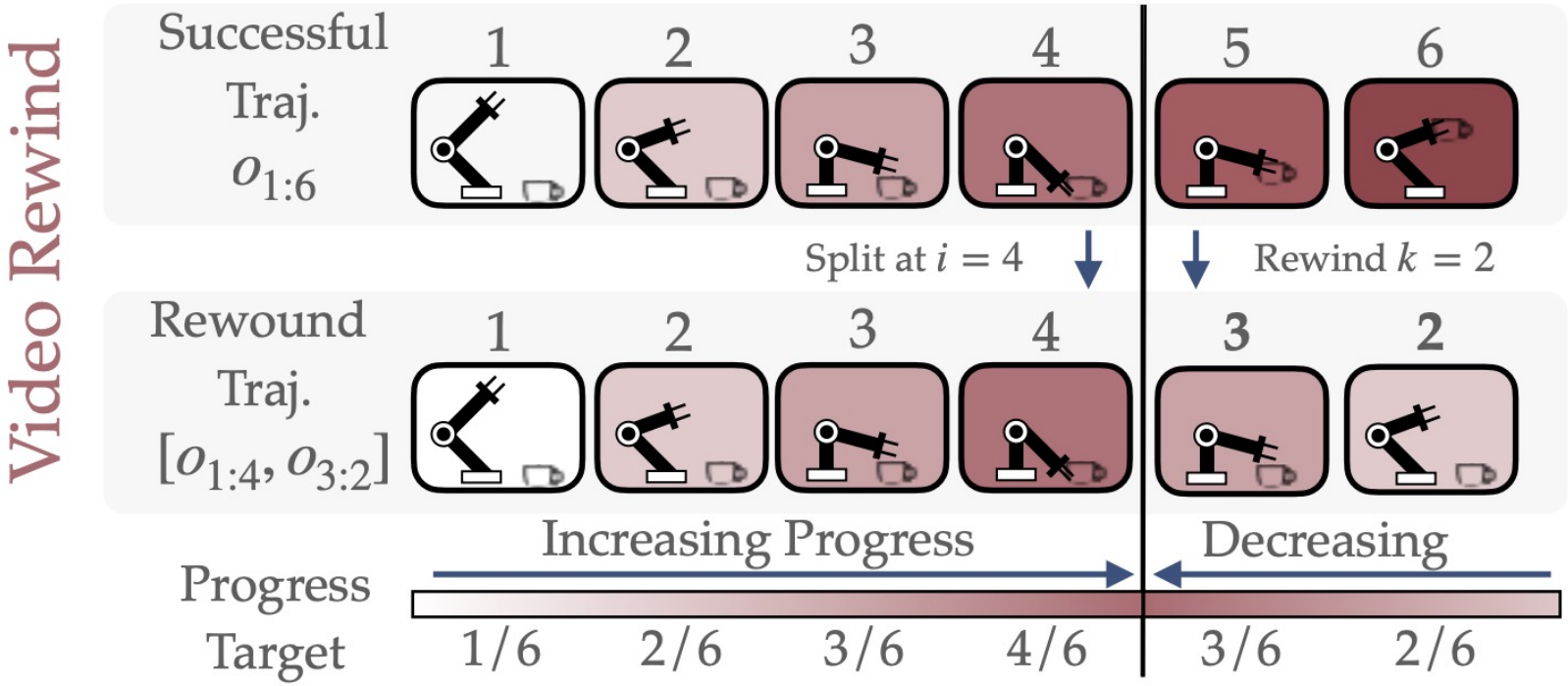
Offline-to-online finetuning is a challenging, yet important problem
Balances exploration with conservatism





Some Frontiers of Offline RL: Reward Functions

Offline RL learns from reward signals – where does this come from?



Learn rewards from videos or preferences

Some Frontiers of Offline RL: Action-Free Video

Lots of logged data does not have actions - how can we learn from this?

No notion of actions – need to infer them from data!

Takeaways from Offline RL

- Offline RL allows us to leverage large offline datasets to improve RL
- Key challenge in offline RL is distribution shift
  - Avoid OOD actions
  - Push down OOD Q/R values
  - Importance sampling
- Offline RL methods can allow for doing “better than data” → essentially serves the role of data driven **curation**

What will we talk about today?

Preliminaries

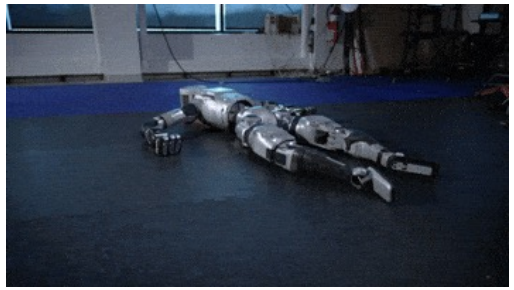
Imitation Learning
(Learning Behaviors from Expert Data)

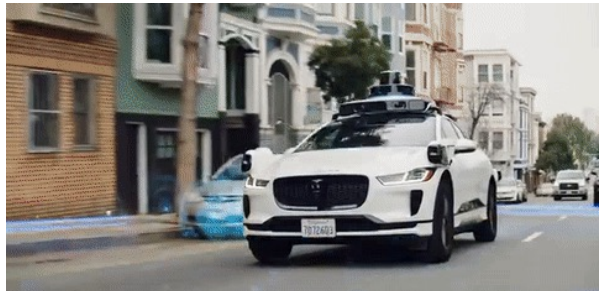
Offline Reinforcement Learning
(Learning from Non-Expert data)

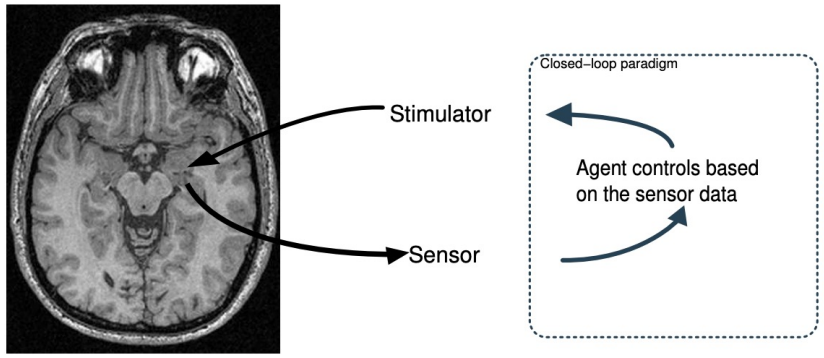
Why should you care about data-driven decision making?

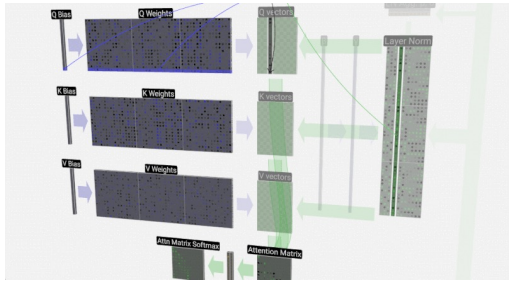
As there will be more data collected offline, we need to be able to absorb it for decision making

Will power the next generation of interactive AI systems









Most problems can be converted into offline RL problems one way or another

Thanks!