Carnegie Mellon
School of Computer Science

Deep Reinforcement Learning and Control

Learning from demonstations and task rewards

Katerina Fragkiadaki

Learning from demonstrations

pros:

- Can much accelerate trial-and-error learning by suggested good actions to try
- Can help us train initial safe policies, to deploy in the real world

cons:

- Time consuming
- May include suboptimal, noise and diverse ways to perform the task
- When you imitate, you cannot surpass the ``expert”.

Learning from task rewards

pros:

- Cheap supervision
- Optimizes the right end task, as encoded in the task rewards

cons:

- Super sample inefficient - impossible to have in the real world
- Initial policy random thus unsafe to have in the real world

Learning from demonstations and task rewards

Goals:

- More sample efficient that RL
- Good/safe initial performance
- Outperfom the human expert

Challenges for kinesthetic demonstations:

- Handling expert suboptimality

Additional challenges for learning from video demonstations:

- requires visual perception
- requires handling mismatch between imitator and demonstator action spaces

Learning from demonstations and task rewards

Goals:

- More sample efficient that RL
- Good/safe initial performance
- Outperfom the human expert

Challenges for kinesthetic demonstations:

- Handling expert suboptimality

Additional challenges for learning from video demonstations:

- requires visual perception
- requires handling mismatch between imitator and demonstator action spaces

Learning from demonstrations and RL

Input:

- a set of kinesthetic expert demonstrations in the form of action-state-reward sequences
- an environment that emits task related rewards.

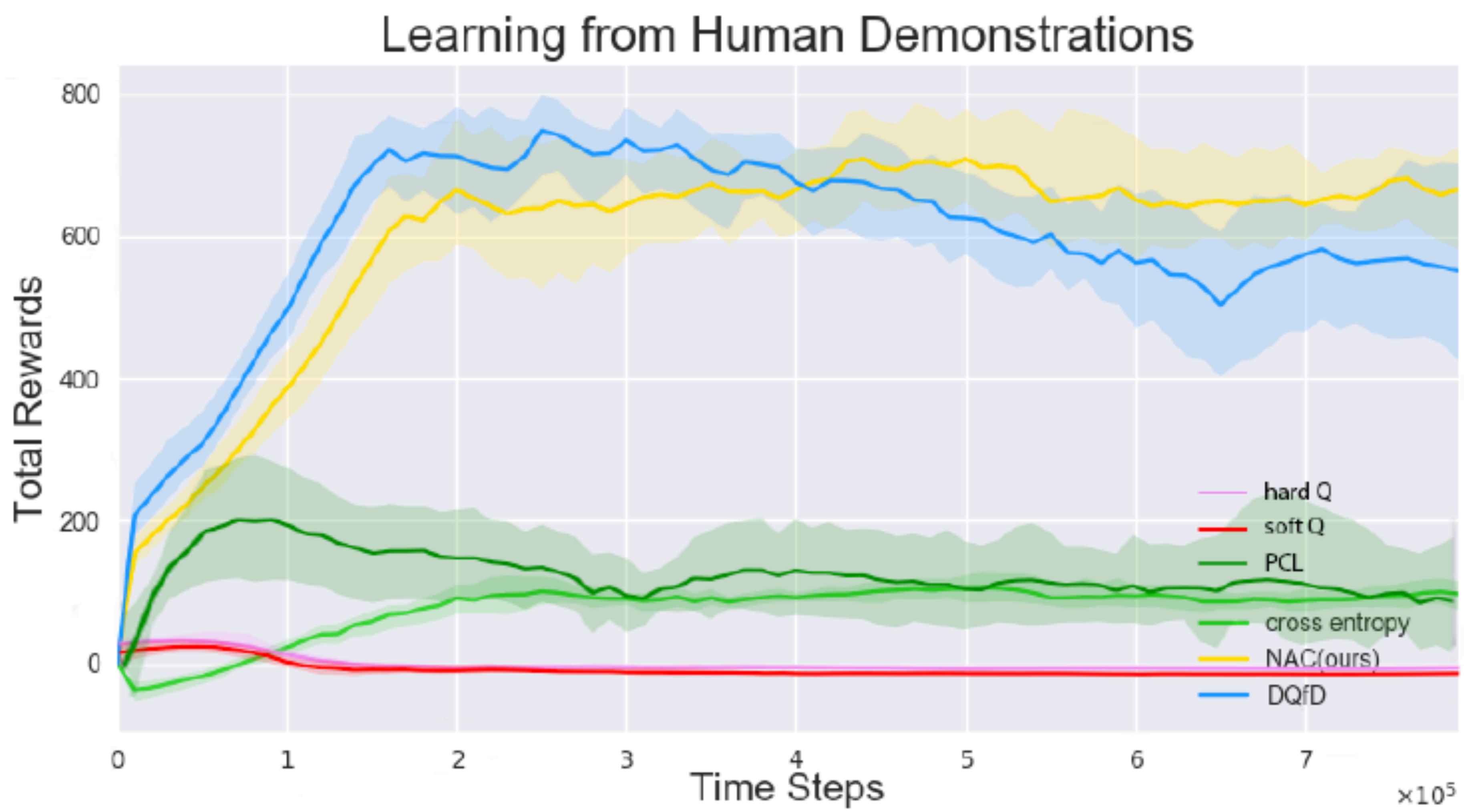
- Idea 0: Augment interaction data with demo trajectories: initialize the replay buffer with demos, (which will be later either removed, or kept forever), and start your model-free RL method
- Idea 1: **Pre-train with demonstrations**, finetune with demonstrations+interacting with the environment.
    - We need to pretrain both a policy and a consistent with it value function (and finetune both later). How?

Apply off-policy model-free RL to demos

- Problem with this?
- Convergence of off policy methods relies on the assumption of visiting each (s,a) pair infinitely many times. Demos are highly biased transitions of the environment, much violate that assumption.
- The states and actions in the demonstrations generally have higher Q-values than other states. Q-learning will push up Q(s; a) values in a sampled state s. However, since the values or the bad actions are not observed, the Q-function has no way of knowing whether the action itself is good, or whether all actions in that state are good, so the demonstrated action will not necessarily have a higher Q-value than other actions in the demonstrated state.

Reinforcement Learning from Imperfect Demonstrations, Gan et al.

Apply off-policy model-free RL to demos



(b) Performances on the Torcs game with human demonstrations. DQfD performs well in the beginning, but overfits in the end. The behavior cloning method is much worse than NAC and DQfD. Our NAC method performs best at convergence.

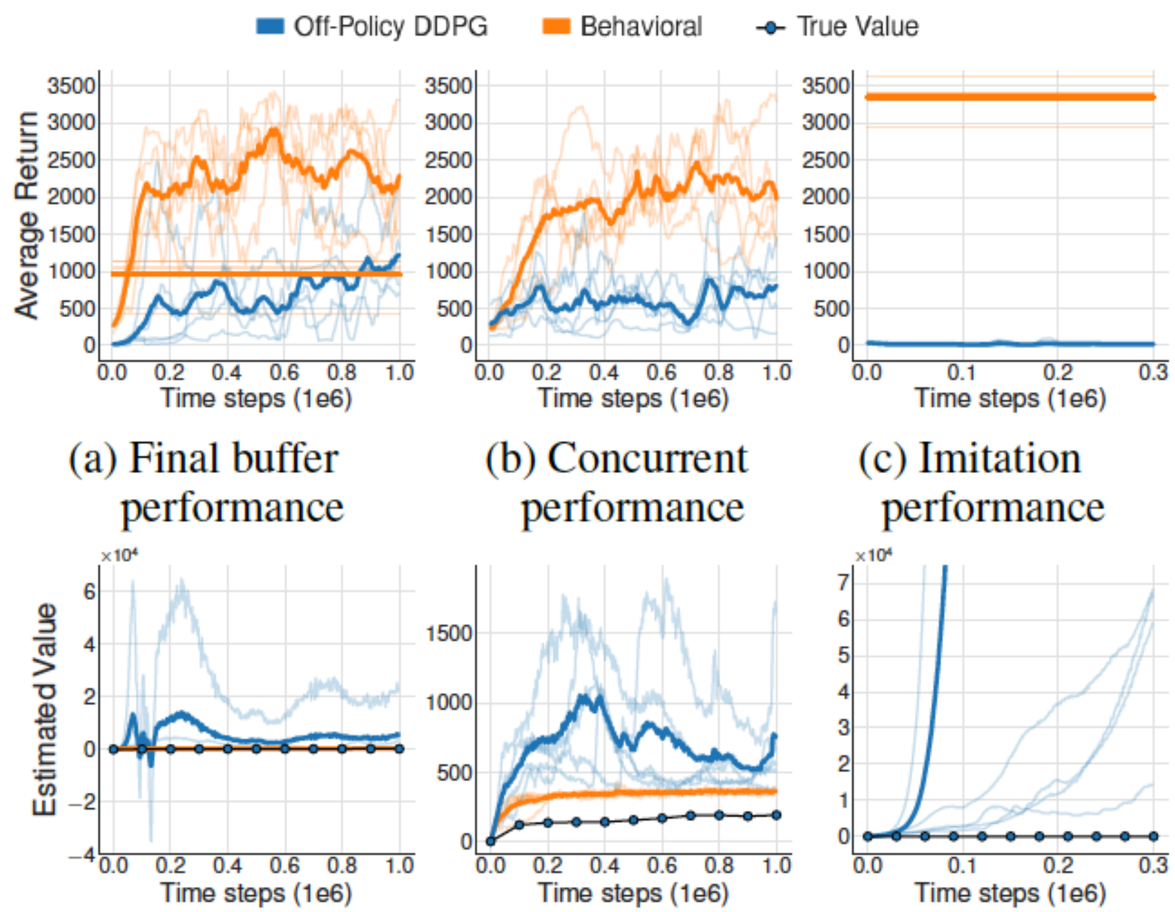
Reinforcement Learning from Imperfect Demonstrations, Gan et al.

Off policy RL

- Off policy RL should be able to learn from data collected under any behavioural policy, colected in a buffer.

- Batch RL: trully off-policy RL: buffer doesn’t grow with data collected from a near on policy exploratory policy. (generated transitions are heavily correlated to the current policy)

Batch RL (or truly off policy RL) does not work



(a) Final buffer performance
(b) Concurrent performance
(c) Imitation performance
(d) Final buffer value estimate
(e) Concurrent value estimate
(f) Imitation value estimate

- Behavioural policy: the policy based on which actions are selected (with small exploration noise) and the experience buffer is polulated
- Off policy: just uses experience tuples from the buffer the behavioural policy generates.
- Q: will off policy agent and behavioural agent have the same Q estimates?

Idea: add a supervised margin loss

Standard DQN loss:

$$\mathscr{L}_{QL}(Q) = \left(\underbrace{\left[R(s,a) + \gamma \max_{a'} Q(s',a') \right]}_{\text{target}} - Q(s,a) \right)^2$$

The margin loss ensures the expert action has higher Q (by a margin) than the rest of the actions.

$$\mathscr{L}_{margin}(Q) = \left(\max_{a \in A} [Q(s,a) + \ell(a_E, a)] - Q(s, a_E) \right)^2$$

$\ell(a_E, a_E) = 0$ and $\ell(a_E, a) > 0,\ a \neq a_E$

Deep Q learning from demonstrations, Hester et al.

V1.0: add a margin loss

Algorithm 1 Deep Q-learning from Demonstrations.

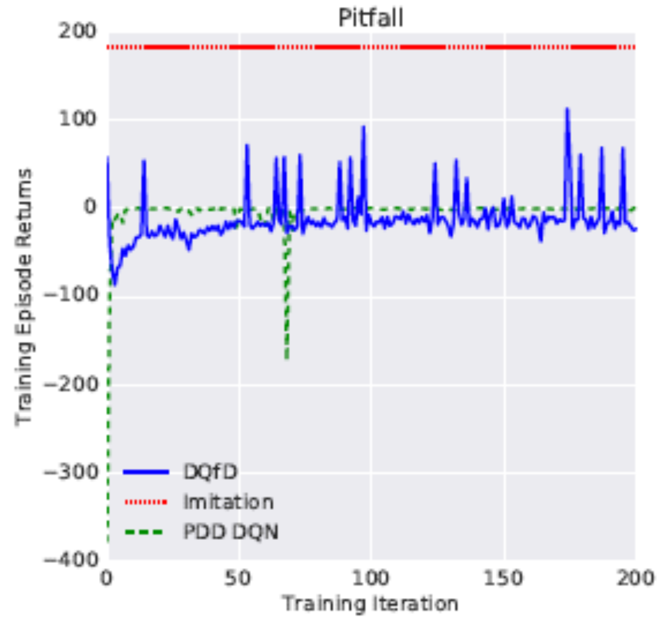
1: Inputs: $\mathcal{D}^{replay}$: initialized with demonstration data set, θ: weights for initial behavior network (random), θ': weights for target network (random), τ: frequency at which to update target net, k: number of pre-training gradient updates
2: **for** steps $t \in \{1, 2, \dots k\}$ **do**
3:     Sample a mini-batch of n transitions from $\mathcal{D}^{replay}$ with prioritization
4:     Calculate loss $J(Q)$ using target network
5:     Perform a gradient descent step to update θ
6:     **if** t **mod** $\tau = 0$ **then** $\theta' \leftarrow \theta$ **end if**
7: **end for**
8: **for** steps $t \in \{1, 2, \dots\}$ **do**
9:     Sample action from behavior policy $a \sim \pi^{\epsilon Q_\theta}$
10:     Play action a and observe (s', r).
11:     Store (s, a, r, s') into $\mathcal{D}^{replay}$, overwriting oldest self-generated transition if over capacity
12:     Sample a mini-batch of n transitions from $\mathcal{D}^{replay}$ with prioritization
13:     Calculate loss $J(Q)$ using target network
14:     Perform a gradient descent step to update θ
15:     **if** t **mod** $\tau = 0$ **then** $\theta' \leftarrow \theta$ **end if**
16:     $s \leftarrow s'$
17: **end for**

Pretraining only with demos

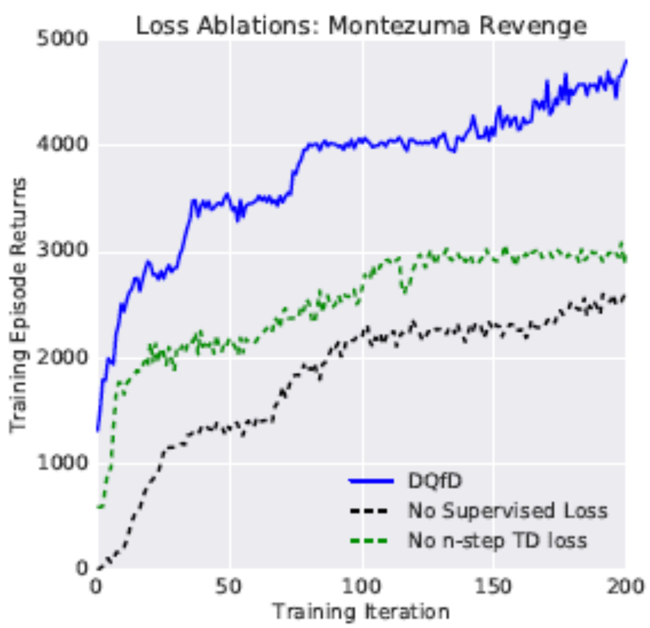
(using DQN and classification losses)

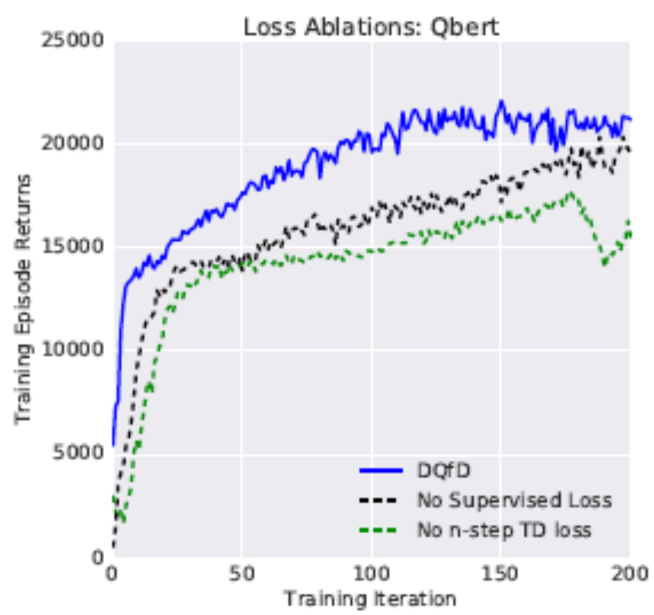
Finetuning jointly with demo+self generated transition in the replay buffer

Deep Q learning from demonstrations, Hester et al.

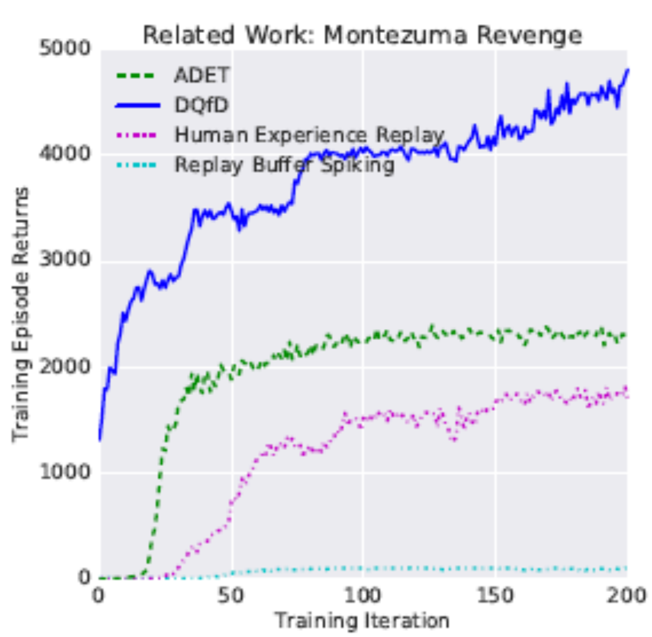


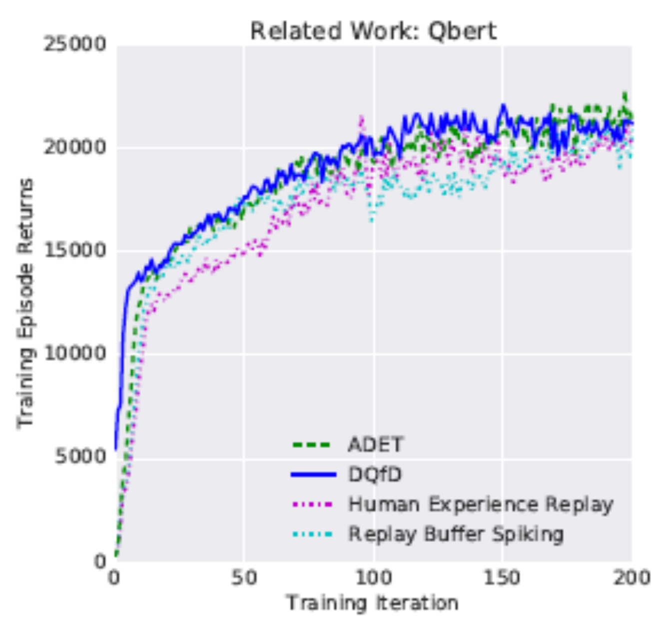
Outperforms both RL alone (PDD DQN) and imitation alone





Margin loss essential





Outperforms both just initializing the replay buffer with demos (RBS) as well as keeping demos around in the buffer (HER).

Deep Q learning from demonstrations, Hester et al.

Combine imitation rewards with task rewards

Input:

- a set of video demonstrations in the form of RGB video sequences
- an environment that emits task related rewards.

Playing hard exploration games by watching YouTube

Yusuf Aytar*, Tobias Pfaff*, David Budden, Tom Le Paine, Ziyu Wang, Nando de Freitas

DeepMind, London, UK
{yusufaytar,tpfaff,budden,tpaine,ziyu,nandodefreitas}@google.com

- Self-supervised visual representation learning to bridge the domain gap between youtube video demonstrations of people playing the game, with the frames the game emits

- Given one video demo, use visual similarity encoded as frame embedding distance as imitation reward, to be added (optionally) to environment rewards.

Playing hard exploration games by watching YouTube

Yusuf Aytar*, Tobias Pfaff*, David Budden, Tom Le Paine, Ziyu Wang, Nando de Freitas

DeepMind, London, UK
{yusufaytar,tpfaff,budden,tpaine,ziyu,nandodefreitas}@google.com

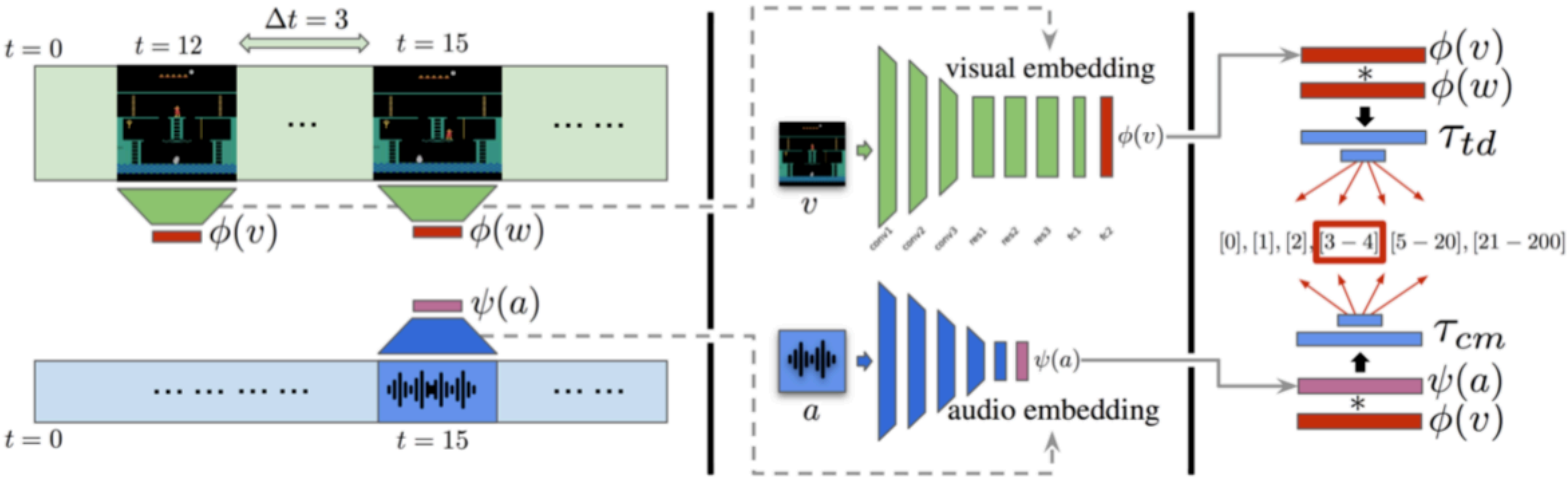
- Self-supervised visual representation learning to bridge the domain gap between youtube video demonstrations of people playing the game, with the frames the game emits
- Given one video demo, use visual similarity encoded as frame embedding distance as imitation reward, to be added (optionally) to environment rewards.

Closing the visual domain gap

(a) ALE frame (b) Frames from different YouTube videos

- Not a huge domain gap, but nonetheless needs to be bridged for comparing frames across the two domains. How?

Closing the visual domain gap



(a) Temporal and cross-modal pair selection (b) Embedding networks (c) Classification networks

- Not a huge domain gap, but nonetheless needs to be bridged for comparing frames across the two domains. How:
- Temporal distance classification: given two frames, clarify their temporal distance into one of k intervals, e.g., {[0],[1],[2],[3–4],[5–20],[21–200]}
- Cross-modal temporal distance classification: given a video frame and an audio snippet, classify their temporal distance into two categories: matching, non-matching.

Playing hard exploration games by watching YouTube

Yusuf Aytar*, Tobias Pfaff*, David Budden, Tom Le Paine, Ziyu Wang, Nando de Freitas

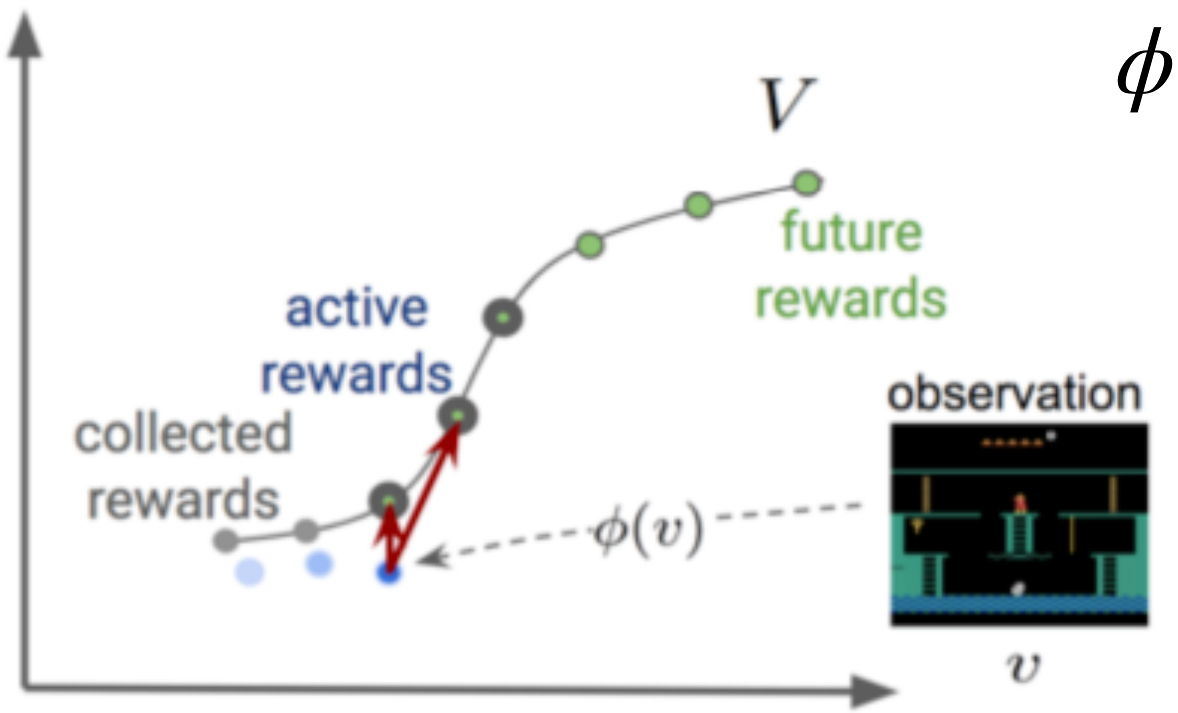
DeepMind, London, UK
{yusufaytar,tpfaff,budden,tpaine,ziyu,nandodefreitas}@google.com

- Self-supervised visual representation learning to bridge the domain gap between youtube video demonstrations of people playing the game, with the frames the game emits

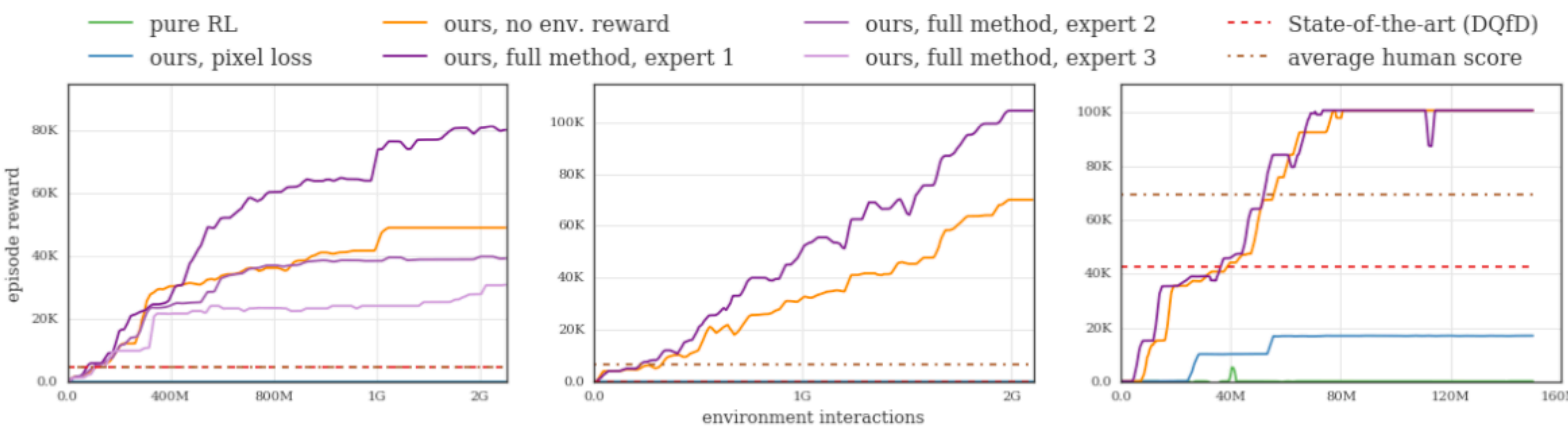
- Given one video demo, use visual similarity encoded as frame embedding distance as imitation reward, to be added (optionally) to environment rewards.

Single shot visual imitation

$\phi(v)$: the visual feature encoding



(b) One shot imitation

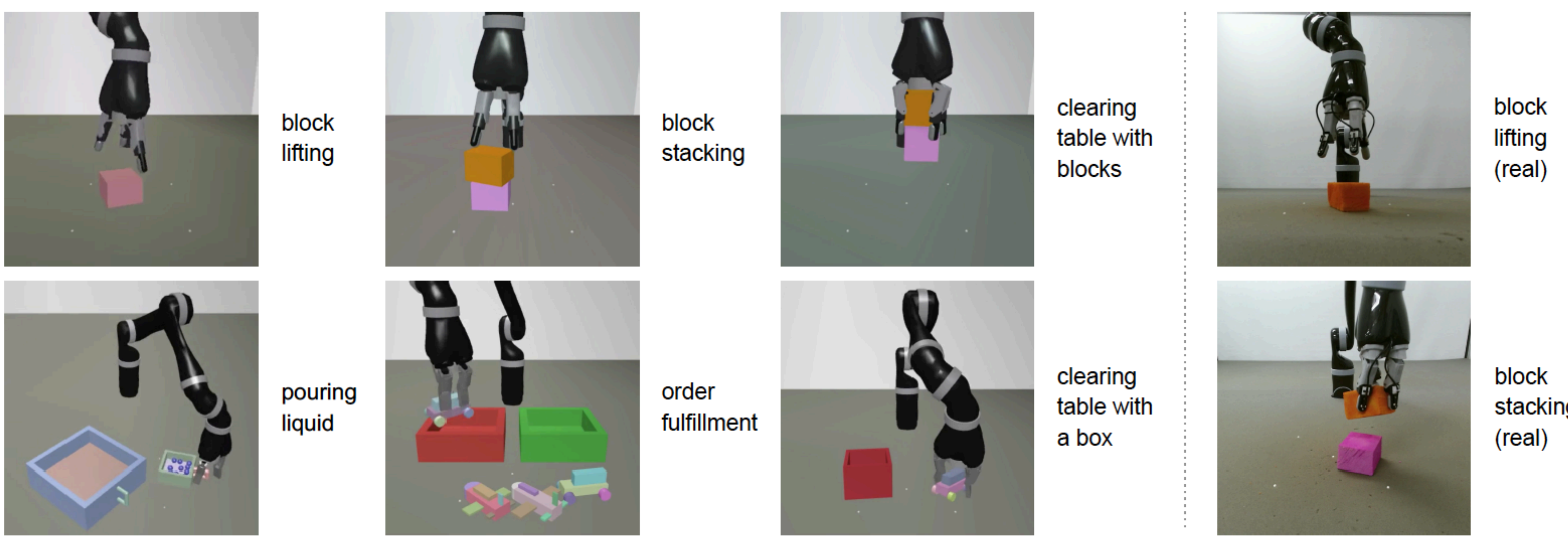


MONTEZUMA'S REVENGE PITFALL! PRIVATE EYE

Reinforcement and Imitation Learning for Diverse Visuomotor Skills

Yuke Zhu[†] Ziyu Wang[‡] Josh Merel[‡] Andrei Rusu[‡] Tom Erez[‡] Serkan Cabi[‡]
Saran Tunyasuvunakool[‡] János Kramár[‡] Raia Hadsell[‡] Nando de Freitas[‡] Nicolas Heess[‡]
[†]Computer Science Department, Stanford University, USA
[‡]DeepMind, London, UK

We are given 30 kinesthetic trajectories in terms of s,a,r for each of the tasks.



Reinforcement and Imitation Learning for Diverse Visuomotor Skills

Yuke Zhu[†] Ziyu Wang[‡] Josh Merel[‡] Andrei Rusu[‡] Tom Erez[‡] Serkan Cabi[‡]
Saran Tunyasuvunakool[‡] János Kramár[‡] Raia Hadsell[‡] Nando de Freitas[‡] Nicolas Heess[‡]
[†]Computer Science Department, Stanford University, USA
[‡]DeepMind, London, UK

- Combine imitation and task rewards.
- Start episods by setting the world in states of the demonstration trajectories.
- Asymetric actor-critic: the value network takes as input the low-dim state of the system and the policy is trained from pixels.
- Only scene state info to the discriminator
- Co-train the policy CNN with auxiliary task
- Sim2REAL via domain randomization.

Reinforcement and Imitation Learning for Diverse Visuomotor Skills

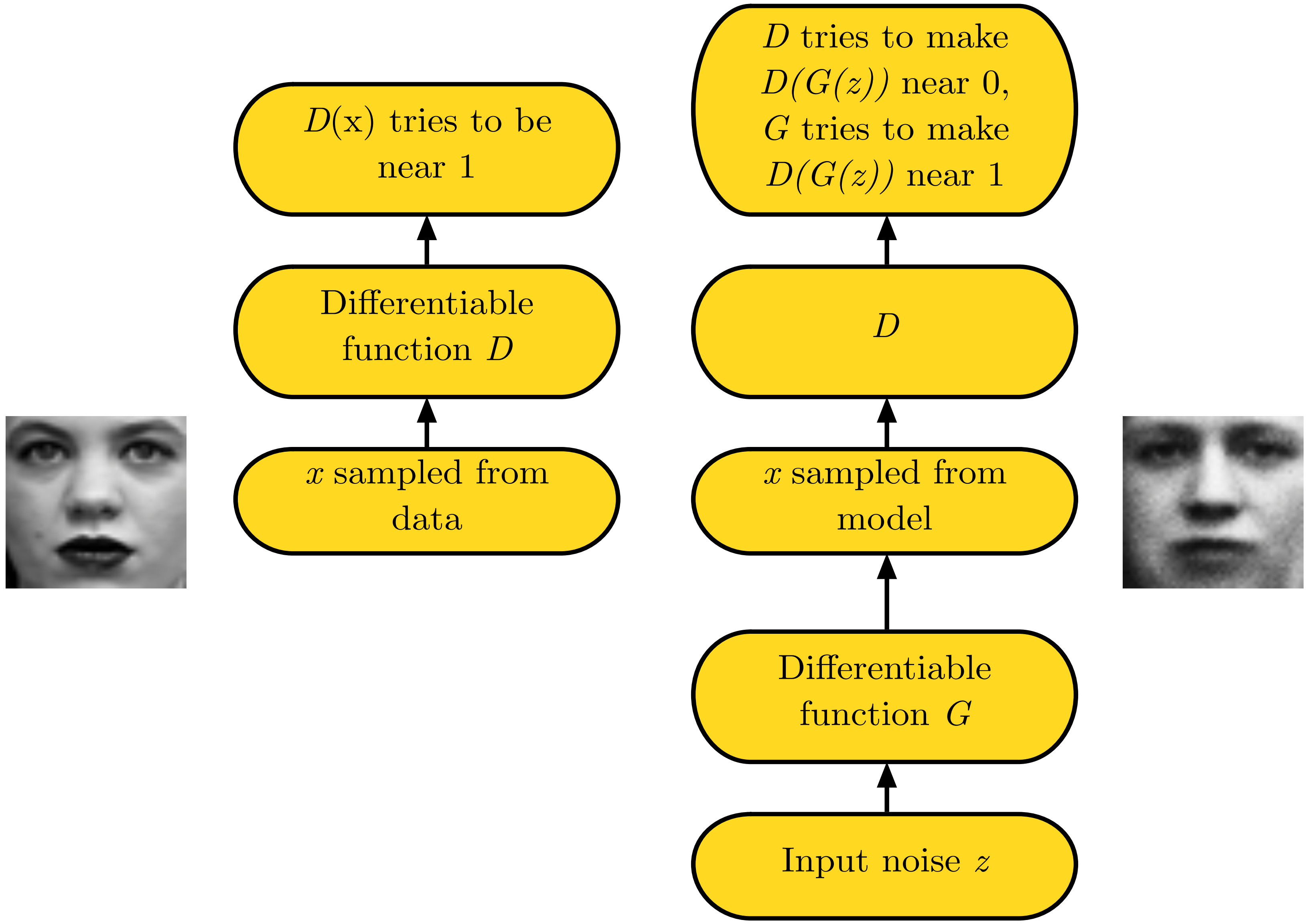
Yuke Zhu† Ziyu Wang‡ Josh Merel‡ Andrei Rusu‡ Tom Erez‡ Serkan Cabi‡
Saran Tunyasuvunakool‡ János Kramár‡ Raia Hadsell‡ Nando de Freitas‡ Nicolas Heess‡
†Computer Science Department, Stanford University, USA
‡DeepMind, London, UK

- Combine imitation and task rewards.
- Start episods by setting the world in states of the demonstration trajectories.
- Assymetric actor-critic: the value network takes as input the low-dim state of the system and the policy is trained from pixels.
- Only scene state info to the discriminator
- Co-train the policy CNN with auxiliary task
- Sim2REAL via domain randomization.

Combining imitation and task rewards

$$r(s, a) = \lambda r_{GAIL}(s, a) + (1 - \lambda) r_{task}(s, a), \quad \lambda \in [0,1] \, .$$

Generative adversarial networks



(Goodfellow 2016)

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

Generative Adversarial Imitation Learning

Jonathan Ho
Stanford University
hoj@cs.stanford.edu

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

NIPS 2016

Algorithm 1 Generative adversarial imitation learning

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

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

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

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

6: **end for**

Combining imitation and task rewards

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

$$r(s, a) = \lambda r_{GAIL}(s, a) + (1 - \lambda) r_{task}(s, a), \quad \lambda \in [0,1] \, .$$

$$r_{GAIL}(s, a) = -\log(1 - D(s, a))$$

Reinforcement and Imitation Learning for Diverse Visuomotor Skills

Yuke Zhu[†] Ziyu Wang[‡] Josh Merel[‡] Andrei Rusu[‡] Tom Erez[‡] Serkan Cabi[‡]
Saran Tunyasuvunakool[‡] János Kramár[‡] Raia Hadsell[‡] Nando de Freitas[‡] Nicolas Heess[‡]
[†]Computer Science Department, Stanford University, USA
[‡]DeepMind, London, UK

- Combine imitation and task rewards.
- Start episods by setting the world in states of the demonstration trajectories. This means we can reset the world however we like, and that we have full state information to be able to set our simulator to it. (Have we done this earlier?)
- Asymetric actor-critic: the value network takes as input the low-dim state of the system and the policy is trained from pixels.
- Only scene state info to the discriminator
- Co-train the policy CNN with auxiliary task
- Sim2REAL via domain randomization.

Reinforcement and Imitation Learning for Diverse Visuomotor Skills

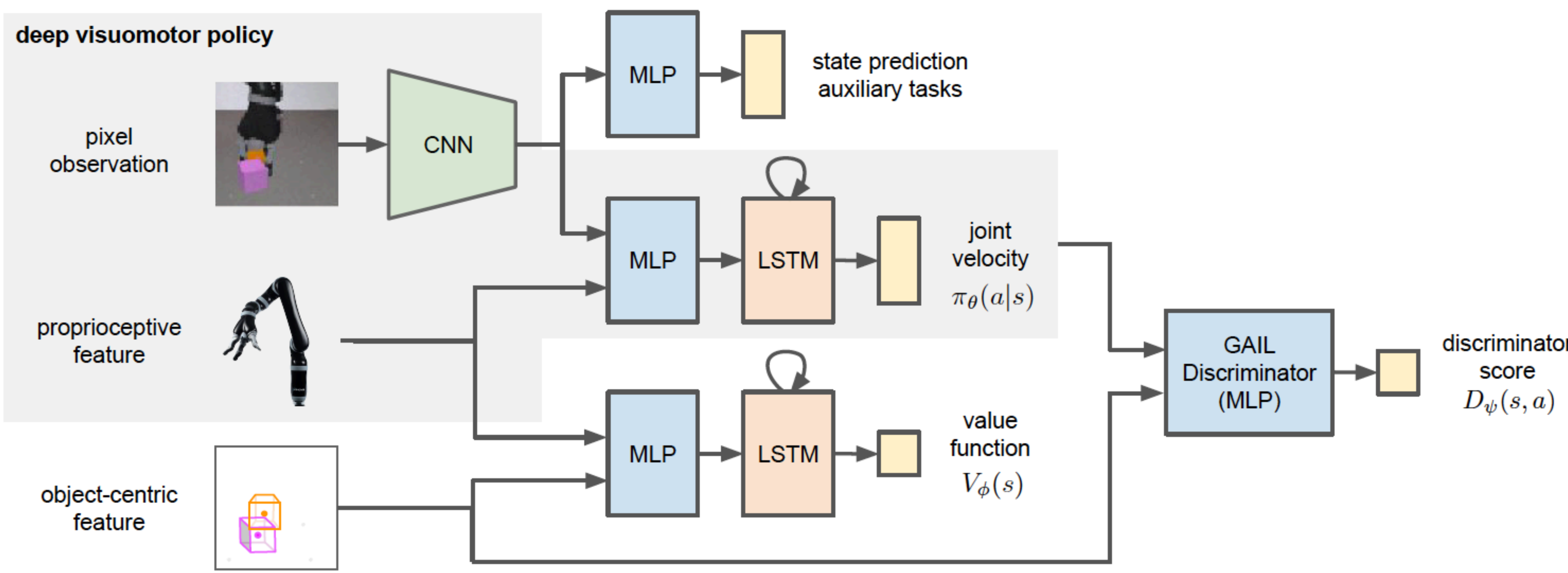
Yuke Zhu† Ziyu Wang‡ Josh Merel‡ Andrei Rusu‡ Tom Erez‡ Serkan Cabi‡
Saran Tunyasuvunakool‡ János Kramár‡ Raia Hadsell‡ Nando de Freitas‡ Nicolas Heess‡
†Computer Science Department, Stanford University, USA
‡DeepMind, London, UK

- Combine imitation and task rewards.
- Start episods by setting the world in states of the demonstration trajectories. This means we can reset the world however we like, and that we have full state information to be able to set our simulator to it. (Have we done this earlier?)
- Asymetric actor-critic: the value network takes as input the low-dim state of the system (3D object location and velocities and relative distances between objects and the gripper) and the policy is trained from pixels directly. This means we need to have access to such state information at training time, but not at test time.
- Only scene state info to the discriminator
- Co-train the policy CNN with auxiliary task
- Sim2REAL via domain randomization.

Reinforcement and Imitation Learning for Diverse Visuomotor Skills

Yuke Zhu[†] Ziyu Wang[‡] Josh Merel[‡] Andrei Rusu[‡] Tom Erez[‡] Serkan Cabi[‡]
Saran Tunyasuvunakool[‡] János Kramár[‡] Raia Hadsell[‡] Nando de Freitas[‡] Nicolas Heess[‡]
[†]Computer Science Department, Stanford University, USA
[‡]DeepMind, London, UK

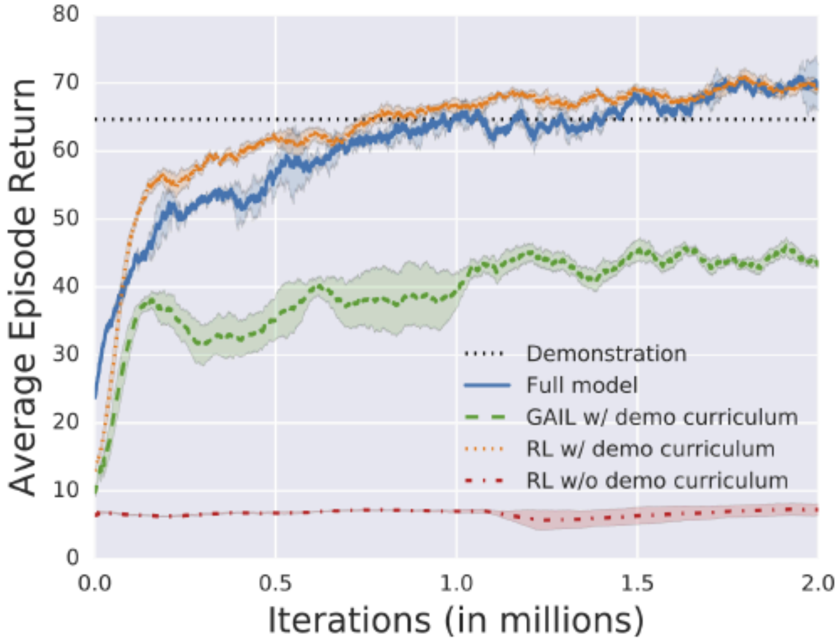
- Combine imitation and task rewards.
- Start episods by setting the world in states of the demonstration trajectories.
- Assymetric actor-critic: the value network takes as input the low-dim state of the system and the policy is trained from pixels.
- Only scene state info to the discriminator
- Co-train the policy CNN with auxiliary task: map images to object locationswith regression and minimize L2 loss. Any object detection/semantic labelling task would work, e.g., learning to detect the robot's gripper is also a useful auxiliary task for training the visual features.
- Sim2REAL via domain randomization.


deep visuomotor policy
pixel observation
CNN
MLP
state prediction auxiliary tasks
MLP
LSTM
joint velocity $\pi_\theta(a|s)$
proprioceptive feature
MLP
LSTM
value function $V_\phi(s)$
object-centric feature
GAIL Discriminator (MLP)
discriminator score $D_\psi(s,a)$

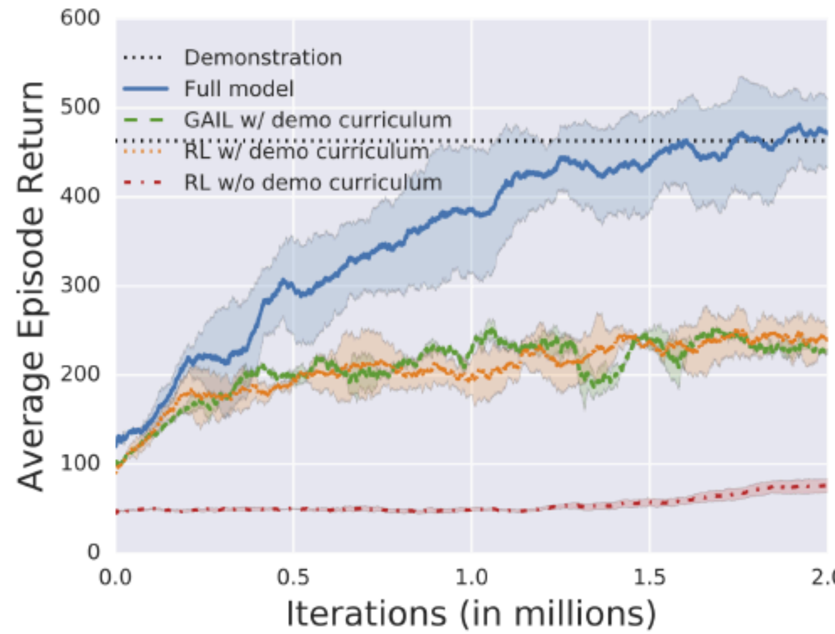
Reinforcement and Imitation Learning for Diverse Visuomotor Skills

Yuke Zhu[†] Ziyu Wang[‡] Josh Merel[‡] Andrei Rusu[‡] Tom Erez[‡] Serkan Cabi[‡]
Saran Tunyasuvunakool[‡] János Kramár[‡] Raia Hadsell[‡] Nando de Freitas[‡] Nicolas Heess[‡]
[†]Computer Science Department, Stanford University, USA
[‡]DeepMind, London, UK

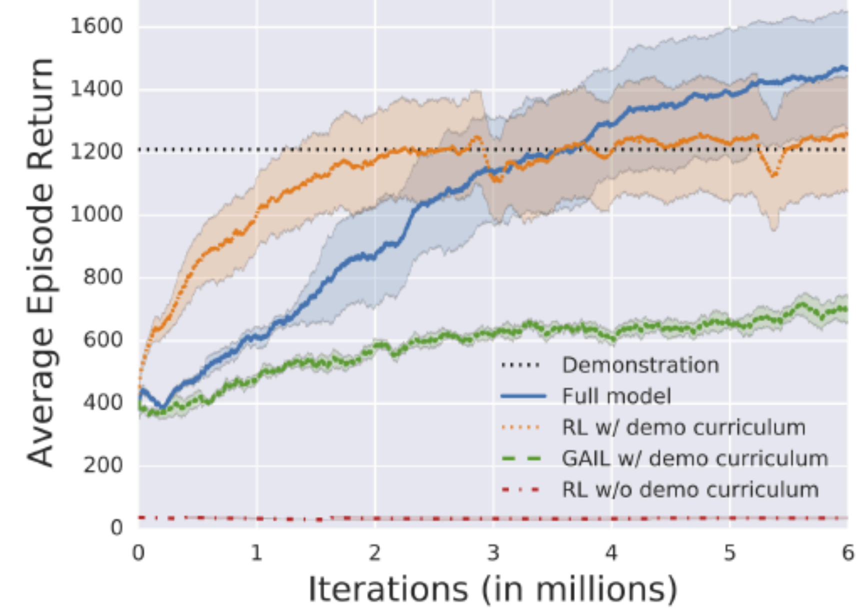
- Combine imitation and task rewards.
- Start episods by setting the world in states of the demonstration trajectories.
- Assymetric actor-critic: the value network takes as input the low-dim state of the system and the policy is trained from pixels.
- Only scene state info to the discriminator
- Co-train the policy CNN with auxiliary task.
- Sim2REAL via domain randomization: randomize camera placement, lighting, background colo, robot arm dynamics, object properties



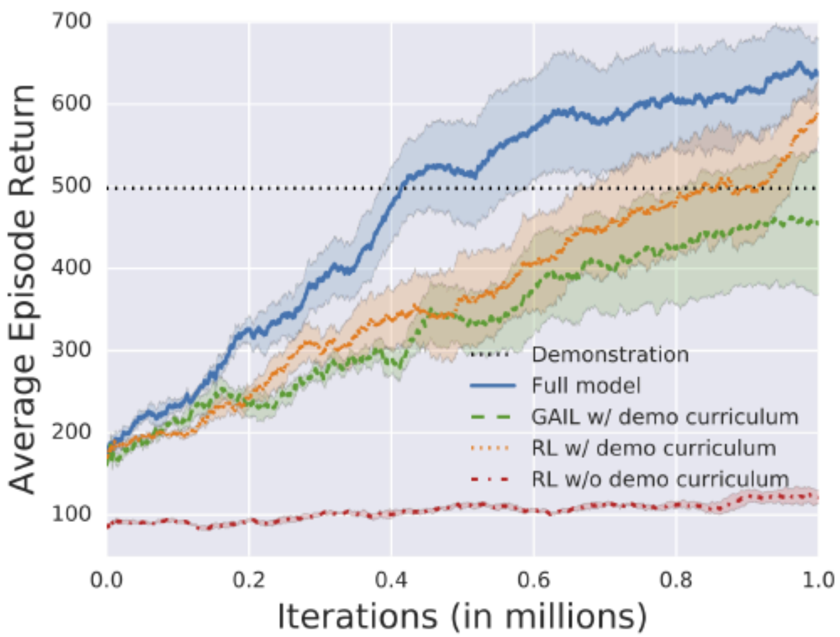
(a) Block lifting



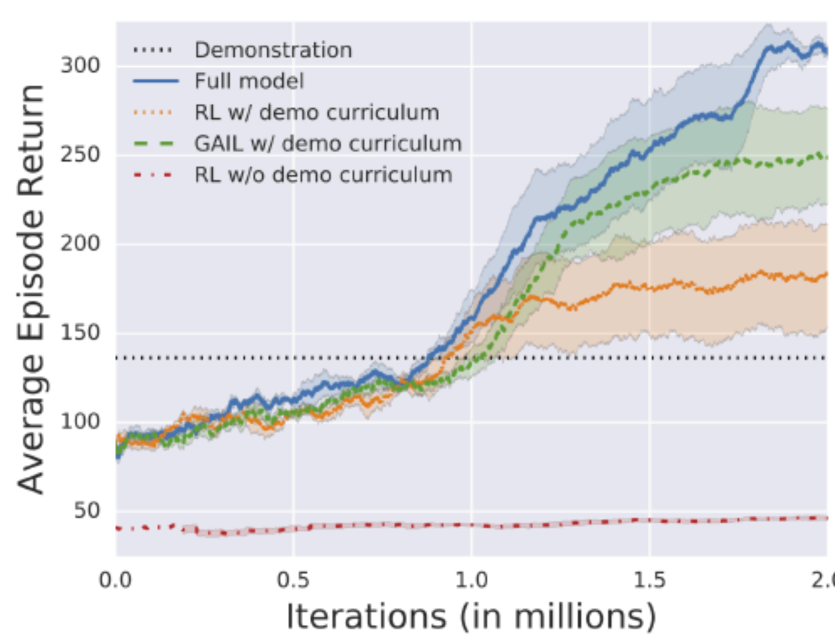
(b) Block stacking



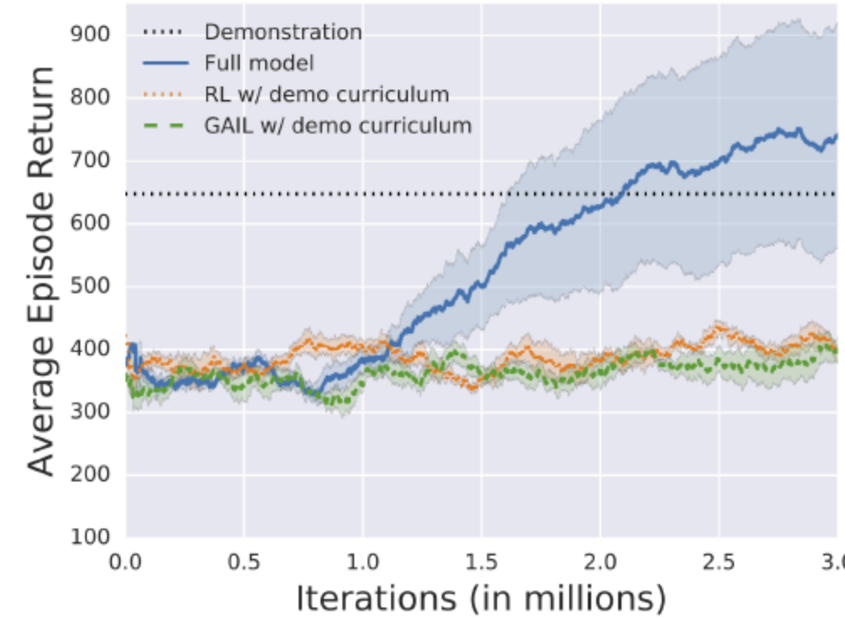
(c) Clearing table with blocks



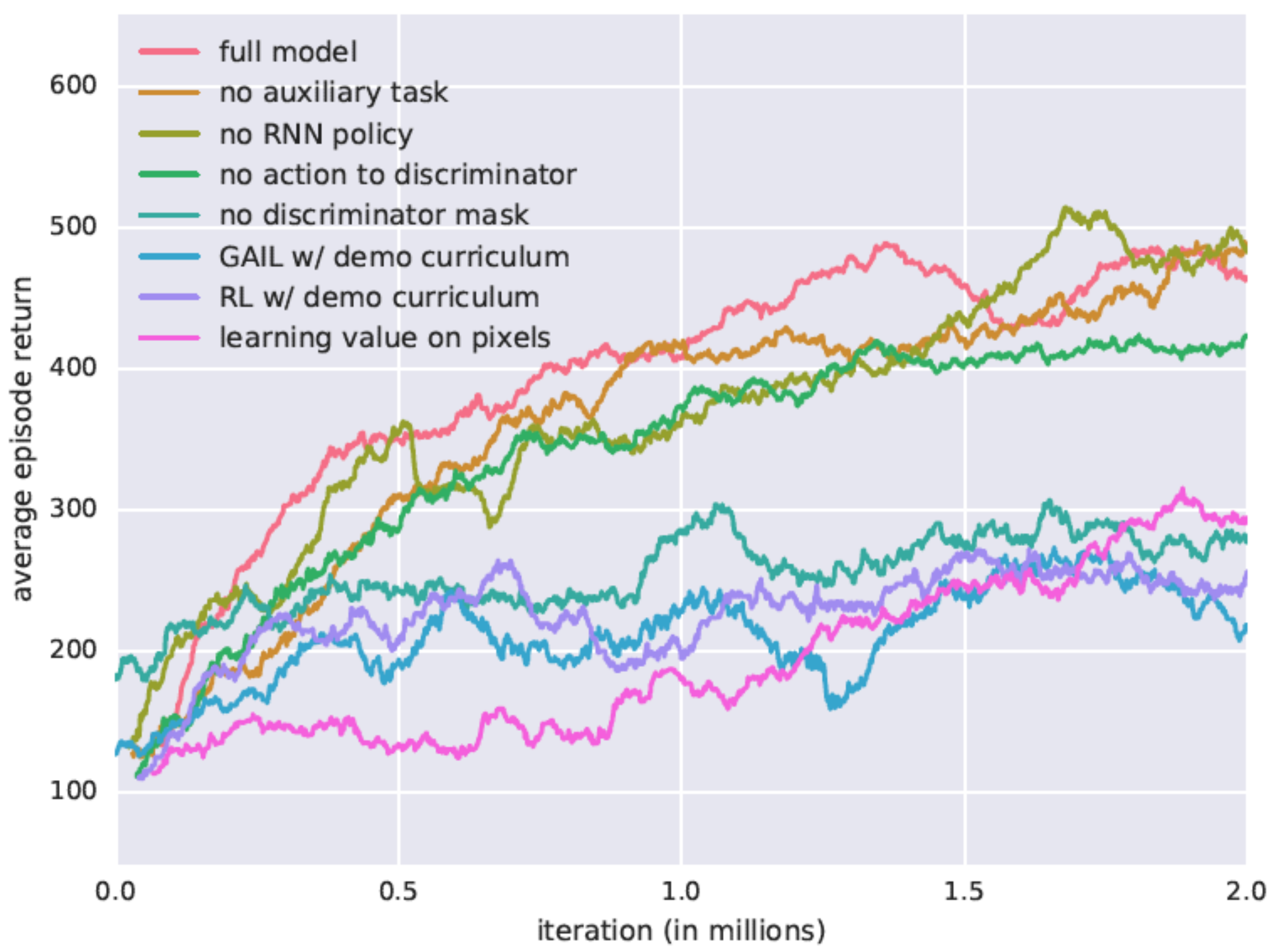
(d) Clearing table with a box



(e) Pouring liquid



(f) Order fulfillment



(a) Ablation study of model components

(b) Model sensitivity to λ values

- Learning value function from pixels directly is slow
- Not using the GAIL imitation reward but rather using demos just to start episods in demo states is slow
- No task reward (just imitation) seems not to work. Why?
- No RNN policy: no problem, RNNs are not great way to integrate info over visual frames.
- No auxiliary task: not big problem.
- Not masking arm info from the discriminator creates problems