# Meta Reinforcement Learning as Task Inference

Jan Humplik, Alexandre Galashov, Leonard Hasenclever,
Pedro A.Ortega, Yee Whye Teh, Nicholas Heess

Topic: Bayesian RL

Presenter: Ram Ananth

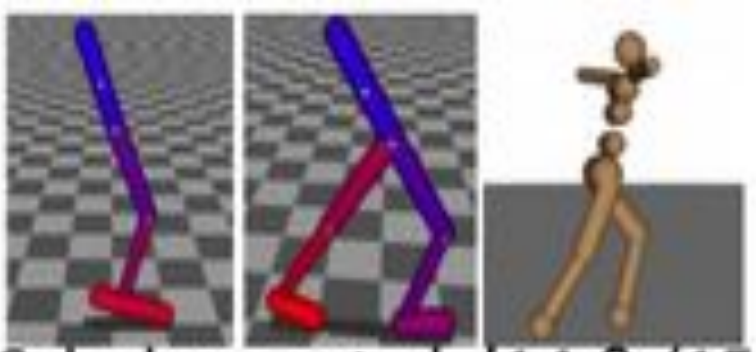
# Why meta Reinforcement Learning?

"First Wave" of Deep Reinforcement Learning algorithms can learn to solve complex tasks and even achieve "superhuman" performance in some cases

Example: Space Invaders

Mnih et al. '13

Example: Continuous Control tasks like Walker and Humanoid

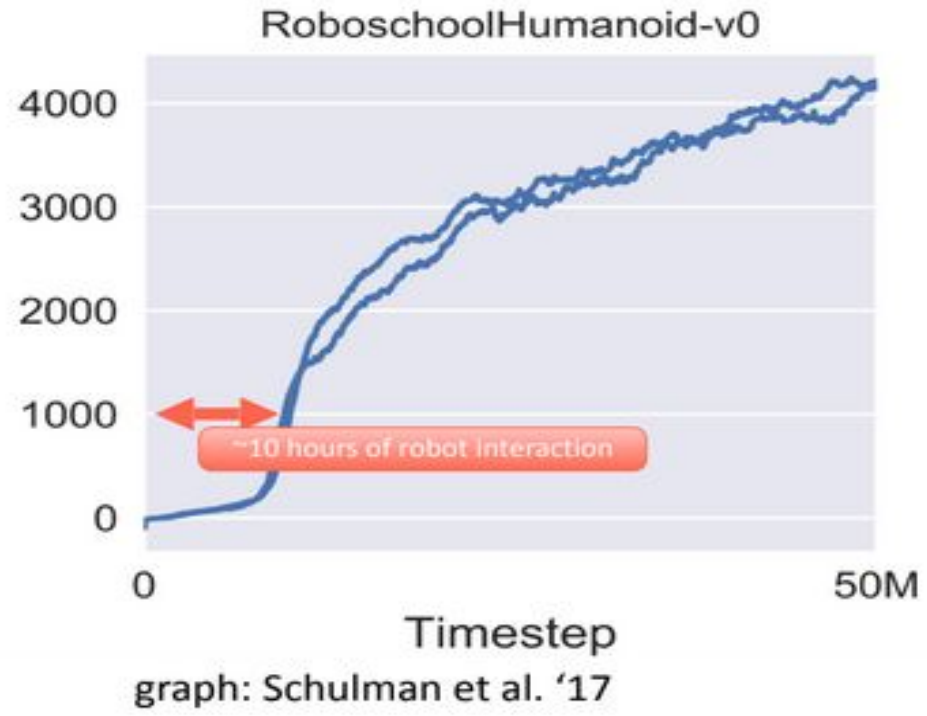


Schulman et al. '14 & '15

Figures adapted from Finn and Levine ICML 19 tutorial on Meta Learning

# Why meta Reinforcement Learning?

However these algorithms are not very efficient in terms of number of samples required to learn (and are “slow”)




graph: Schulman et al. ‘17

Fig adapted from Finn and Levine ICML 19 tutorial on Meta Learning

# Why meta Reinforcement Learning?

Humans (Animals) leverage prior knowledge when learning compared to RL algorithms that learn tabula rasa and hence can learn extremely quickly

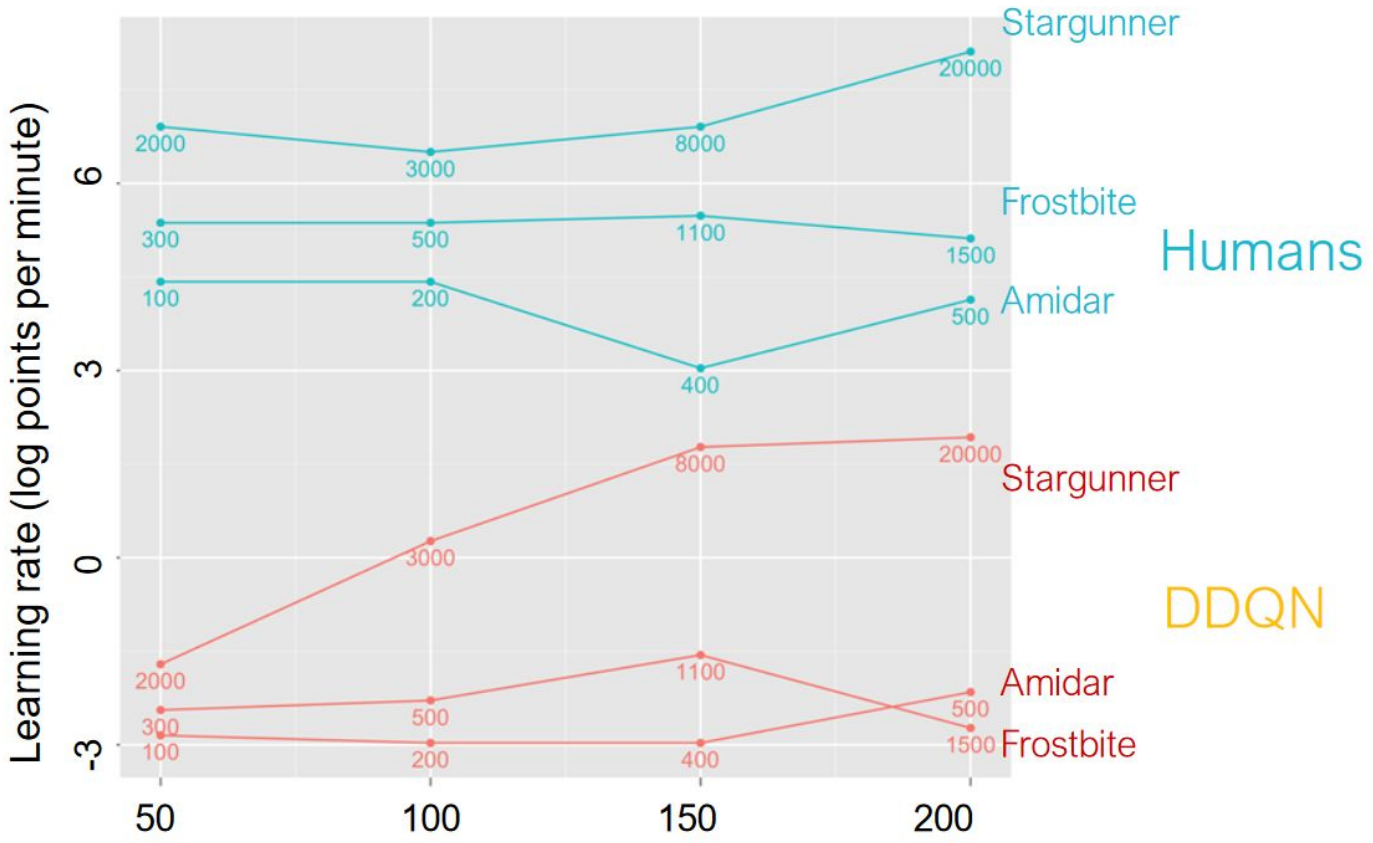
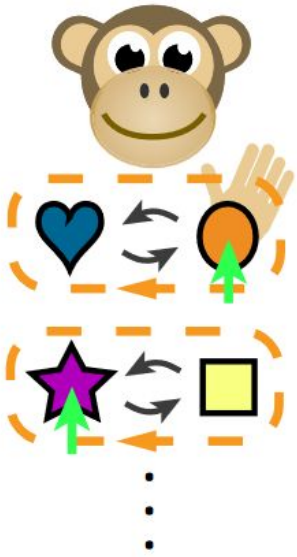
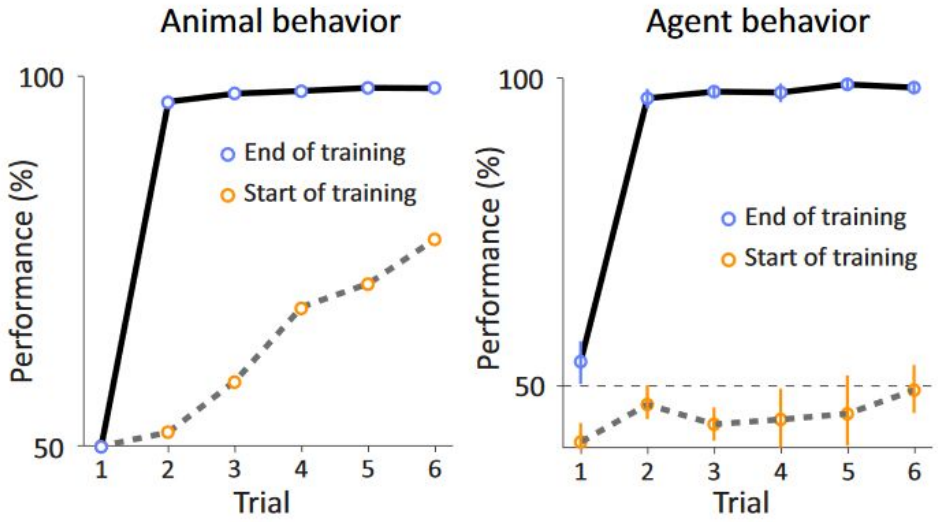



Fig adapted from Animesh Garg 2020 “Human Learning in Atari’

# Why meta Reinforcement Learning?

## The Harlow's Task






Can we "meta-learn" efficient RL algorithms that can leverage prior knowledge about the structure of naturally occuring tasks ?

→ Meta Reinforcement Learning

Fig adapted from Botvinick et al 19

# The meta RL problem

Reinforcement learning:

$$\theta^\star = \arg\max_\theta E_{\pi_\theta(\tau)}[R(\tau)]$$

$$= f_{\mathrm{RL}}(\mathcal{M}) \qquad \mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$$

MDP

Finn and Levine ICML 19 tutorial on Meta Learning

# The meta RL problem

Reinforcement learning:

$$\theta^\star = \arg\max_\theta E_{\pi_\theta(\tau)}[R(\tau)]$$

$$= f_{\mathrm{RL}}(\mathcal{M}) \qquad \mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$$

MDP

Meta-reinforcement learning:

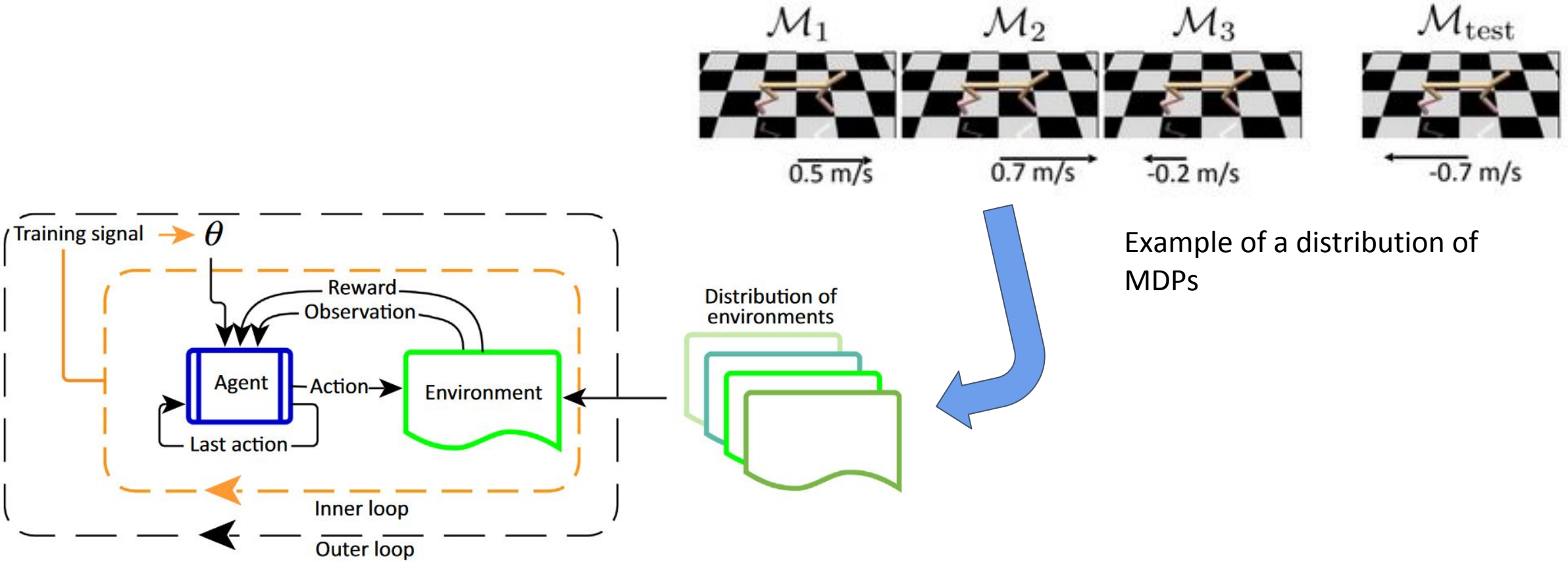
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

where $\phi_i = f_\theta(\mathcal{M}_i)$

MDP for task $i$

Finn and Levine ICML 19 tutorial on Meta Learning

# The meta RL problem : Training framework

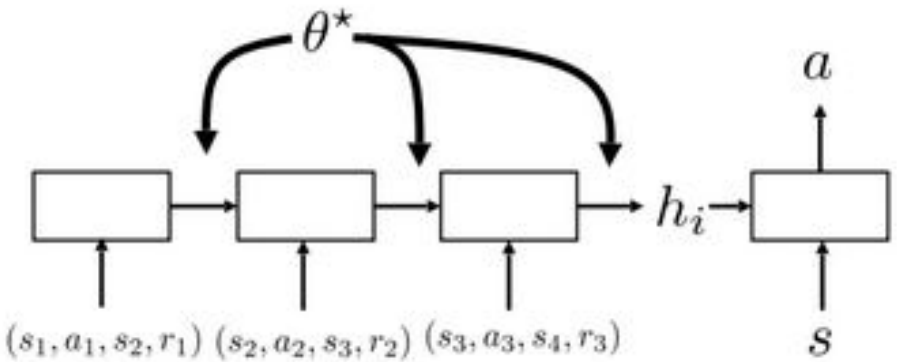



Example of a distribution of MDPs

Fig adapted from Botvinick et al 19

Fig adapted from Finn and Levine ICML 19 tutorial on Meta Learning

## Perspective 1: just RNN it




+ conceptually simple

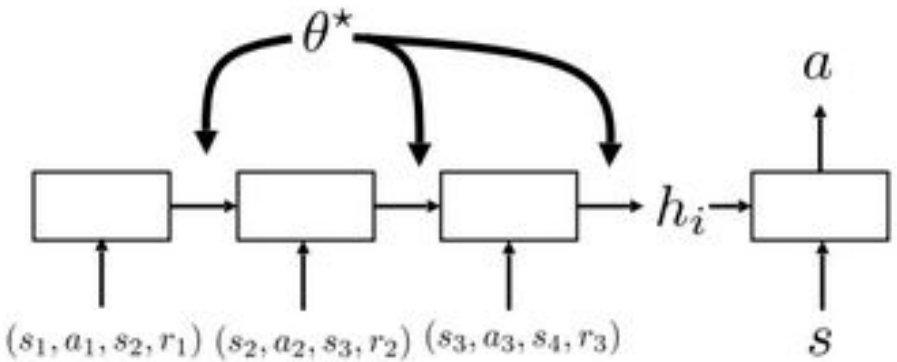
+ relatively easy to apply

- vulnerable to meta-overfitting

- challenging to optimize in practice

Fig adapted from Finn and Levine ICML 19 tutorial on Meta Learning

Perspective 1: just RNN it




+ conceptually simple

+ relatively easy to apply

- vulnerable to meta-overfitting

- challenging to optimize in practice

Perspective 2: bi-level optimization

$$f_\theta(\mathcal{M}_i) = \theta + \alpha \nabla_\theta J_i(\theta)$$

MAML for RL

+ good extrapolation ("consistent")

+ conceptually elegant

- complex, requires many samples

Fig adapted from Finn and Levine ICML 19 tutorial on Meta Learning

# Motivation

- **Alternate Perspective to Meta Reinforcement Learning**

  **(Probabilistic meta Reinforcement Learning)**

  - Simple, effective exploration
  - Elegant reduction to POMDP

  The process of Learning to solve a task can be considered as probabilistically inferring the task given observations

# Motivation

- **Alternate Perspective to Meta Reinforcement Learning**

  **(Probabilistic meta Reinforcement Learning)**

  The process of Learning to solve a task can be considered as probabilistically inferring the task given observations

Why probabilistic inference makes sense?

Need to learn fast from less observations ⟹ Low information regime ⟹ Uncertainty in Task identity

Uncertainty in Task identity can help agent balance exploration and exploitation

# Motivation

- Probabilistic Meta RL : Use a particularly formulated partially observable markov decision processes (POMDP)

# Motivation

- Probabilistic Meta RL : Use a particularly formulated partially observable markov decision processes (POMDP)

If each task is an MDP, the optimal agent (that initially doesn’t know the task ) is one that maximises rewards in a POMDP* with a single unobserved (static) state consisting of task specification

* Referred to meta-RL POMDP (Bayes-adaptive MDP in Bayesian RL literature)

# Motivation

In general for POMDP, optimal policy depends on full history of observations, actions and rewards

Can this dependance on full history be captured by a sufficient statistic?

# Motivation

In general for POMDP, optimal policy depends on full history of observations, actions and rewards

Can this dependance on full history be captured by a sufficient statistic?

Yes, belief state. For our particular POMDP the relevant part of belief state is posterior distribution over the uncertain task specification given the agent's experience thus far. Reasoning about this belief state is at the heart of Bayesian RL

# Motivation

**The given problem can be separated into 2 modules**

1) **Estimating this belief state** ——— Hard problem to solve

# Motivation

**The given problem can be separated into 2 modules**

1) **Estimating this belief state** ——— Hard problem to solve

   Why is this problem hard?

   Estimating the belief state is intractable in most POMDPs

2) **Acting based on this estimate of the belief state**

# Motivation

**The given problem can be separated into 2 modules**

1) **Estimating this belief state** ——— Hard problem to solve

   Why is this problem hard?

   Estimating the belief state is intractable in most POMDPs

2) **Acting based on this estimate of the belief state**

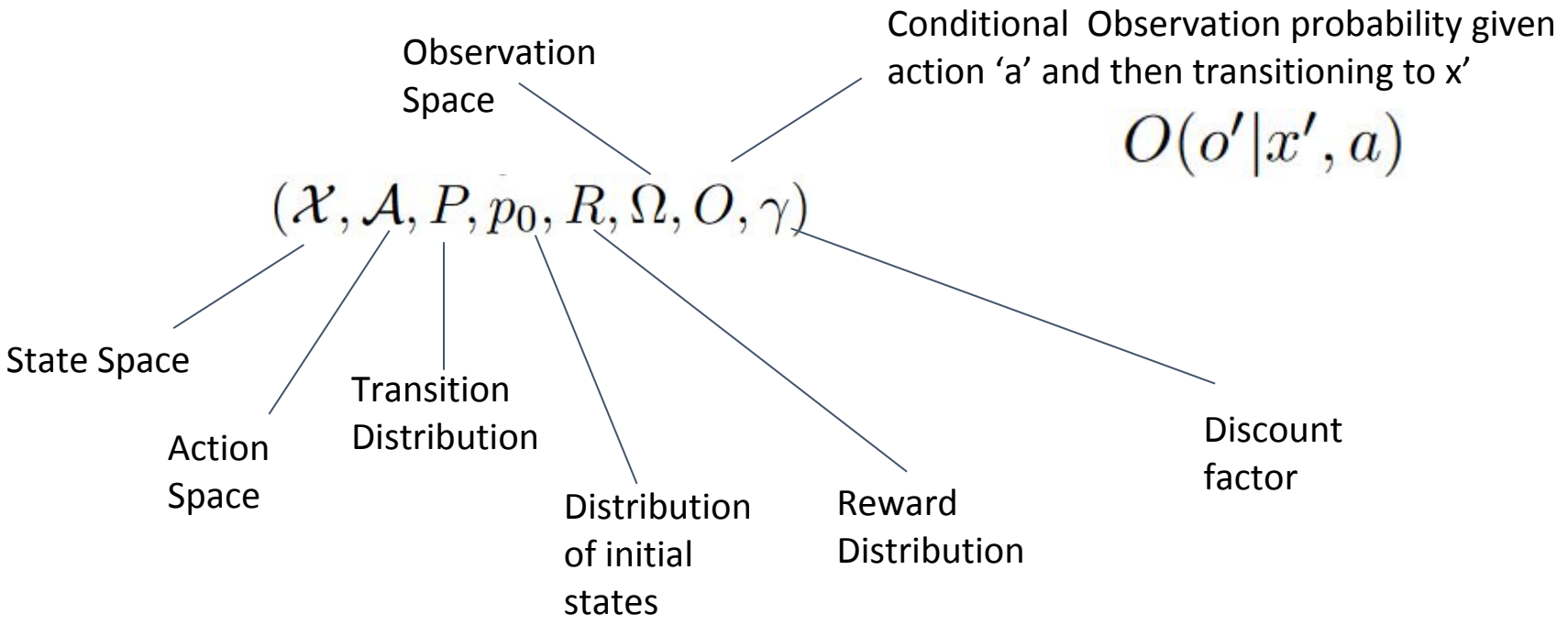
But typically in meta RL, task distribution is under designer's control and also task specification is available at meta-training. Can we take advantage of this privileged information?

# Contributions

1. Demonstrate that leveraging cheap task specific information during meta-training can boost performance of meta-RL algorithms
2. Train meta-RL agents with recurrent policies efficiently with off-policy RL algorithms
3. Experimentally demonstrate that the agents can solve meta-RL problems in complex continuous control environment with sparse rewards and requiring long term memory
4. Show that the agents can discover Bayes-optimal search strategy

# Preliminaries

**POMDPs**




Sequence of states is denoted by $x_{0:t} = (x_0, \ldots, x_t)$ and similarly for actions and rewards
Observed trajectory is denoted by $\tau_{0:t} = (o_{0:t}, a_{0:t-1}, r_{0:t-1})$

# Preliminaries

**POMDPs**

Optimal policy of POMDP $\pi^*(a_t|\tau_{0:t})$

$$\pi^* = \arg\max_\pi \mathbb{E}_{\tau_{0:\infty}, x_{0:\infty} \sim p_\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

Joint distribution between trajectory and states

$$p_\pi(\tau_{0:t}, x_{0:t})$$

# Preliminaries

**POMDPs**

Optimal policy of POMDP $\pi^*(a_t|\tau_{0:t})$

$$\pi^* = \arg\max_\pi \mathbb{E}_{\tau_{0:\infty}, x_{0:\infty} \sim p_\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

Belief state is given by $b_t(x) \equiv p_\pi(x_t = x|\tau_{0:t})$

Joint distribution between trajectory and states

$$p_\pi(\tau_{0:t}, x_{0:t})$$

# Preliminaries

## POMDPs

Optimal policy of POMDP $\pi^*(a_t|\tau_{0:t})$

$$\pi^* = \arg\max_\pi \mathbb{E}_{\tau_{0:\infty}, x_{0:\infty} \sim p_\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

Belief state is given by $b_t(x) \equiv p_\pi(x_t = x|\tau_{0:t})$

Belief state is sufficient statistic for optimal action $\pi^*(a_t|\tau_{0:t}) = \pi^*(a_t|b_t)$

Joint distribution between trajectory and states

$p_\pi(\tau_{0:t}, x_{0:t})$

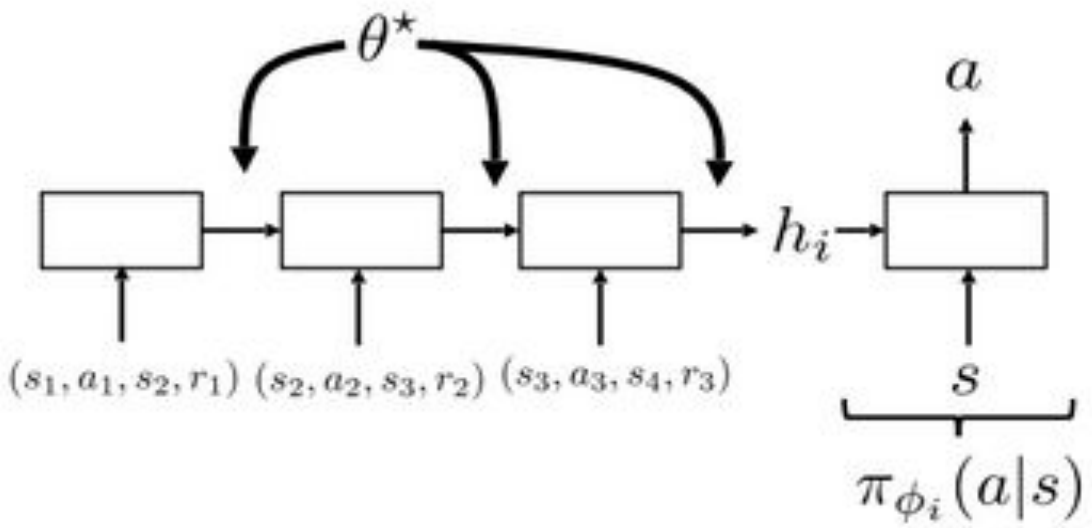
# Preliminaries : Meta-RL with recurrent policies

Meta RL objective

$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

$$\text{where } \phi_i = f_\theta(\mathcal{M}_i)$$

RNN policy




RNN hidden state

meta-learned weights

$$\text{as before, } \phi_i = [h_i, \theta_\pi]$$

Figures adapted from Finn and Levine ICML 2019 talk on Meta Learning

# Preliminaries : Regularisation with Information Bottleneck

In supervised learning the goal is to learn a mapping $g_\theta : X \to Y$ Such that the loss is minimised

$$L(\theta, \mathcal{D}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[l(y, f_\theta(x))]$$

# Preliminaries : Regularisation with Information Bottleneck

In supervised learning the goal is to learn a mapping $g_\theta : X \to Y$ Such that the loss is minimised

$$L(\theta, \mathcal{D}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[l(y, f_\theta(x))]$$

In IB regularization $g_\theta(x) = \mathbb{E}_{z\sim q_\theta(\cdot|x)}[f_\theta(z)]$,

$q_\theta(z|x)$ Is a stochastic encoder and Z is latent embedding of X

# Preliminaries : Regularisation with Information Bottleneck

The new regularised objective is:

Intractable

$$L_{\mathrm{IB}}(\theta, \mathcal{D}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathbb{E}_{z\sim q_\theta(\cdot|x)}[l(y, f_\theta(z))] + \lambda I[Z, X]$$

# Preliminaries : Regularisation with Information Bottleneck

The new regularised objective is:

Intractable

$$L_{\mathrm{IB}}(\theta, \mathcal{D}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathbb{E}_{z\sim q_\theta(\cdot|x)}[l(y, f_\theta(z))] + \lambda I[Z, X]$$

However, it is upper bounded $I[Z, X] < \mathbb{E}_{x\sim\mathcal{D}}[D_{\mathrm{KL}}[q_\theta(\cdot|x)||r]]$

Can be any arbitrary distribution but set to N(0,1) in practice

# Approach : POMDP view of Meta RL

$\mathcal{W}$ : Task space with distribution of tasks $p(w)$

Each task is given by (PO)MDP given by $(\mathcal{X}, \mathcal{A}, P^w, p_0^w, R^w, \gamma)$

POMDP states $(x, w) \in \mathcal{X} \times \mathcal{W}$

POMDP action space is same as each task's action space

POMDP transitions $P(x', w'|x, w, a) = \delta(w' - w)P^w(x'|x, a)$

POMDP initial state distribution $p_0(x, w) = p(w)p_0(x|w)$

POMDP reward distribution $R(r|x, w, a, x', w') = R^w(r|x, a, x')$

POMDP observation distribution is deterministic $O(o'|x', w, a) = \delta(o' - x')$

# Approach : POMDP view of Meta RL

$$b_t(x, w) = p(x, w|\tau_{0:t}) = \delta(x - x_t)p(w|\tau_{0:t})$$

Belief state for meta-RL POMDP

Posterior over tasks given what the agent has observed so far

Objective function to find optimal policy for meta-RL POMDP

$$\max_{\pi} \sum_{w} p(w) \sum_{\tau_{0:\infty}} p_{\pi}(\tau_{0:\infty}|w) \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right]$$

# Proof to facilitate off-policy learning

Objective function can be written as

$$\max_{\pi} \sum_{\tau_{0:\infty}} p_\pi(\tau_{0:\infty}) \left[ \sum_{t=0}^{\infty} \gamma^t \bar{r}(x_t, a_t, x_{t+1}, b_t) \right]$$

where

$$p_\pi(\tau_{0:\infty}) = \sum_w p(w) p_\pi(\tau_{0:\infty}|w),$$

marginal distribution of the trajectory

$$\bar{r}(x_t, a_t, x_{t+1}, b_t) \equiv \sum_w b_t(w) \sum_r R(r|x_t, a_t, x_{t+1}, w) r,$$

$$b_t(w) \equiv p(w|\tau_{0:t}),$$

posterior expected reward

Belief state/posterior distribution over tasks

# Proof to facilitate off-policy learning

(i.e meta-RL POMDP belief state is independent of the policy given the trajectory)

$$b_{t+1}(w) \equiv p(w|\tau_{0:t+1}) = p(w|\tau_{0:t}, a_t, r_t, x_{t+1})$$
$$= \frac{p(w, a_t, r_t, x_{t+1}|\tau_{0:t})}{\sum_w p(w, a_t, r_t, x_{t+1}|\tau_{0:t})}$$

# Proof to facilitate off-policy learning

(i.e meta-RL POMDP belief state is independent of the policy given the trajectory)

$$
\begin{aligned}
b_{t+1}(w) &\equiv p(w|\tau_{0:t+1}) = p(w|\tau_{0:t}, a_t, r_t, x_{t+1}) \\
&= \frac{p(w, a_t, r_t, x_{t+1}|\tau_{0:t})}{\sum_w p(w, a_t, r_t, x_{t+1}|\tau_{0:t})} \\
&= \frac{p(a_t, r_t, x_{t+1}|w, \tau_{0:t})p(w|\tau_{0:t})}{\sum_w p(a_t, r_t, x_{t+1}|w, \tau_{0:t})p(w|\tau_{0:t})} \\
&= \frac{p(a_t, r_t, x_{t+1}|w, \tau_{0:t})b_t(w)}{\sum_w p(a_t, r_t, x_{t+1}|w, \tau_{0:t})b_t(w)}
\end{aligned}
$$

# Proof to facilitate off-policy learning

(i.e meta-RL POMDP belief state is independent of the policy given the trajectory)

$$= \frac{R^w(r_t|x_t,a_t)P^w(x_{t+1}|x_t,a_t)\pi(a_t|\tau_{0:t})b_t(w)}{\sum_w R^w(r_t|x_t,a_t)P^w(x_{t+1}|x_t,a_t)\pi(a_t|\tau_{0:t})b_t(w)},$$

# Proof to facilitate off-policy learning

(i.e meta-RL POMDP belief state is independent of the policy given the trajectory)

Since policy is independent of task

$$= \frac{R^w(r_t|x_t,a_t)P^w(x_{t+1}|x_t,a_t)\pi(a_t|\tau_{0:t})b_t(w)}{\sum_w R^w(r_t|x_t,a_t)P^w(x_{t+1}|x_t,a_t)\pi(a_t|\tau_{0:t})b_t(w)}$$

$$= \frac{R^w(r_t|x_t,a_t)P^w(x_{t+1}|x_t,a_t)b_t(w)}{\sum_w R^w(r_t|x_t,a_t)P^w(x_{t+1}|x_t,a_t)b_t(w)},$$

$$b_t(w) = p(w|\tau_{0:t}) \propto p(w)p_0(x_0|w)\prod_{t'=0}^{t-1} P^w(x_{t'+1}|x_{t'},a_{t'})R^w(r_{t'}|x_{t'},a_{t'},x_{t'+1})$$

Given trajectory, task posterior is independent of policy that generated it

# Approach : Learning belief network

- In general, it is difficult to learn belief representation of POMDPs

Solution : Use the privileged information given as part of the meta RL problem

Similar in purpose as using expert trajectories, natural language instructions or designed curricula to speed up learning

# Approach : Learning belief network

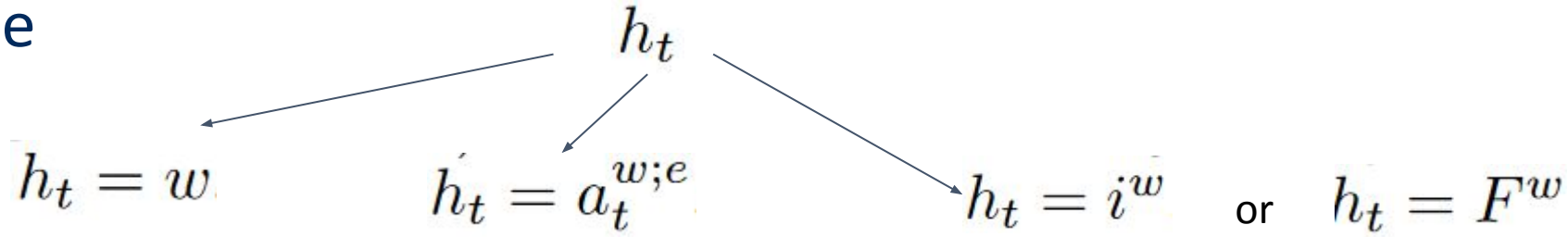
- In general, it is difficult to learn belief representation of POMDPs

Solution : Use the privileged information given as part of the meta RL problem



Similar in purpose as using expert trajectories, natural language instructions or designed curricula to speed up learning

- Different types of task information are used with varying levels of privilege




Predict true task information

Predict action chosen by expert trained only on that task $\pi^{w;e}(a_t|\tau_{0:t})$

Predict index of task

Predict task embedding if available

# Approach : Learning belief network

- We need to train belief module $b_\theta(h_t|\tau_{0:t})$

Minimize auxiliary log loss

$$\mathbb{E}_{p(h_t|\tau 0:t)}[-\log b_\theta(h_t|\tau_{0:t})]$$

Posterior distribution of task information given the trajectory

Minimizing auxiliary log loss is equivalent to minimizing

$$\mathbb{KL}(p(h_t|\tau 0:t)\|b_\theta(h_t|\tau_{0:t}))$$

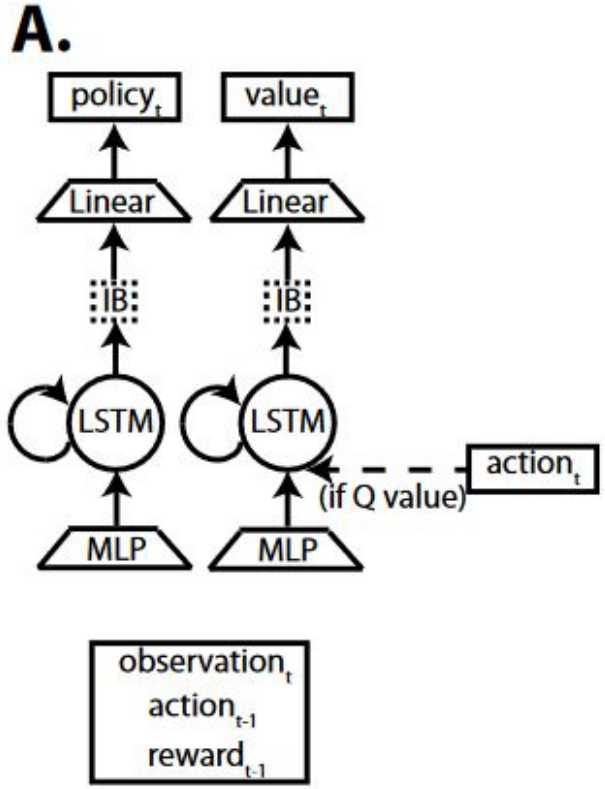
- Although, we don't know the posterior distribution we can still get samples in our meta-RL setting and since belief state is independent of policy given the trajectory so is the task information. It can be trained with off-policy data

Note: This is backward KL which is different than the one used in variational inference

# Approach: Architectures and Algorithms

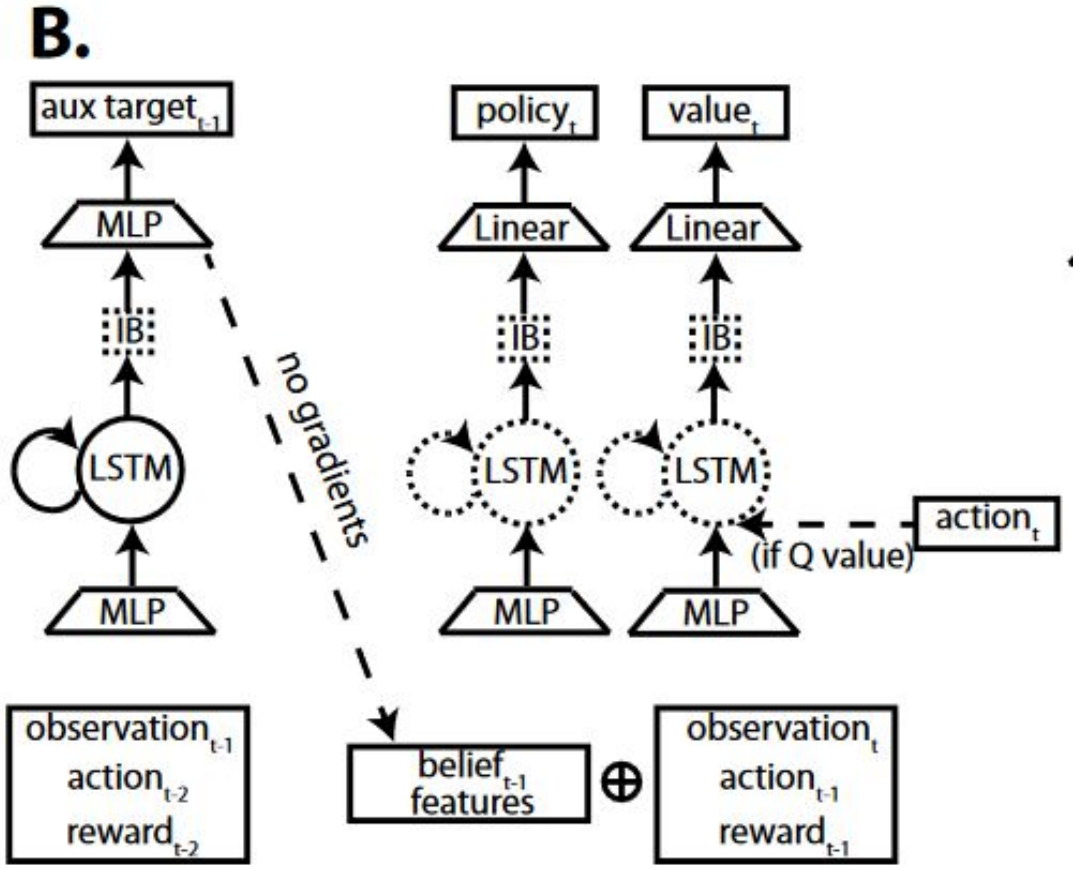
The paper proposes the use of entropy-regularised (distributed) SVG(0) for off-policy learning and the use of PPO for on-policy version for all different architectures used

Baseline architecture



# Approach: Architectures and Algorithms

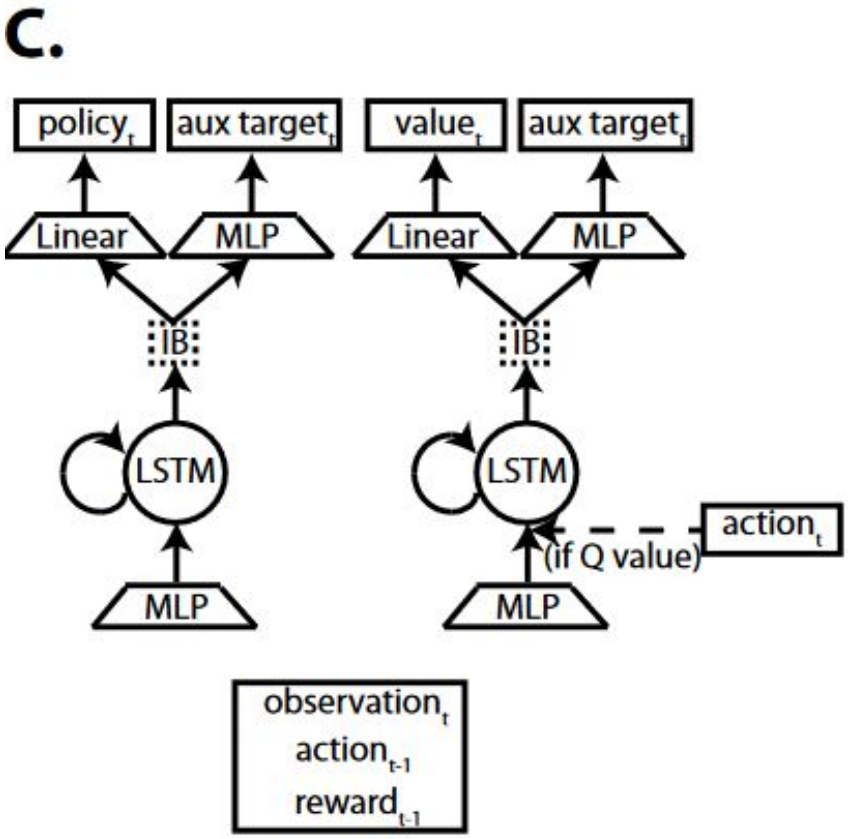
The proposed belief network architecture



# Approach: Architectures and Algorithms

Alternate auxiliary head agent, whereby the auxiliary loss directly shapes the representations learnt

(AuxHead)



**Algorithm 1** Belief net SVG(0) with IB (learner)

initial states: $h_0^\pi, h_0^Q, h_0^b$
policy: $q_{t,\theta}^\pi \equiv q_\theta^\pi(z_t|\tau_{0:t}, h_0^\pi), \pi_{t,\theta} \equiv \pi_\theta(a_t|z_t)$
Q: $q_{t,\psi}^Q(a_t) \equiv q_\psi^Q(z_t|\tau_{0:t}, a_t, h_0^Q), Q_{t,\psi} \equiv Q_\psi(z_t)$
belief: $q_{t,\phi}^b \equiv q_\phi^b(z_t|\tau_{0:t}, h_0^b), b_{t,\phi} \equiv b_\phi(w|z_t)$
online parameters: $\theta_O, \psi_O$
target parameters: $\theta_T = \theta_O, \psi_T = \psi_O$
replay buffer: $\mathcal{B}$
batch size: $B$
unroll length: $U$
target update period: $M$
**for** $step = 1$ **to** $\infty$ **do**
  $\{h_0^{i,\pi}, h_0^{i,Q}, h_0^{i,b}, \tau_{0:U}^i, w^i\}_{i=1}^B \leftarrow B$ samples from $\mathcal{R}$
  $(x_0^i, a_0^i, r_0^i, \ldots, r_{U-1}^i, x_U^i) \leftarrow \tau_{0:U}^i$
  $L_\pi = 0, L_Q = 0, L_b = 0$
  **for** $i = 1$ **to** $B$ **do**
    $L_{t,b}^i = \mathbb{E}_{z\sim q_{t,\phi}^b}[\log b(w^i|z)] + \lambda D_{\mathrm{KL}}[q_{t,\phi}^b||N(0,1)]$
    $(f_0^i, \ldots, f_{U-1}^i) \leftarrow q_{t,\phi}^b$ features
    $\tau_{0:U}^i \leftarrow \tau_{0:U}^i \cup (f_0^i, \ldots, f_{U-1}^i)$
    $V_t^i = \mathbb{E}_{z^\pi\sim q_{t,\theta_T}^\pi} \mathbb{E}_{a\sim\pi_{\theta_T}(\cdot|z^\pi)} \mathbb{E}_{z^Q\sim q_{t,\psi_T}^Q(a)} \left[Q_{\psi_T}(z^Q)\right]$
    $Q_t^i = r_t^i + \gamma V_{t+1}^i$
    $Q_t^{R,i} = Retrace(Q_t^i)$
    $L_{t,Q}^i = \mathbb{E}_{z\sim q_{t,\psi_O}^Q(a_t^i)} \left[(Q_{\psi_O}(z) - Q_t^{R,i})^2\right]$
    $L_{t,Q}^i \mathrel{+}= \lambda D_{\mathrm{KL}}[q_{t,\psi_O}^Q(a_t^i)||N(0,1)]$
    $L_{t,\pi}^i = \mathbb{E}_{z^\pi\sim q_{t,\theta_O}^\pi} \mathbb{E}_{a\sim\pi_{\theta_O}(\cdot|z^\pi)} \mathbb{E}_{z^Q\sim q_{t,\psi_T}^Q(a)} \left[Q_{\psi_T}(z^Q)\right]$
    $L_{t,\pi}^i \mathrel{+}= \lambda D_{\mathrm{KL}}[q_{t,\theta_O}^\pi||N(0,1)] - \alpha \mathsf{H}[\pi_{t,\theta_O}]$
    $L_Q \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,Q}^i$
    $L_\pi \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,\pi}^i$
    $L_b \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,b}^i$
  **end for**
  $\theta_O = Adam[\nabla_{\theta_O} L_\pi]$
  $\psi_O = Adam[\nabla_{\psi_O} L_Q]$
  $\phi = Adam[\nabla_\phi L_b]$
  **if** $step \,\%\, M = 0$ **then**
    $\theta_T = \theta_O$
    $\psi_T = \psi_O$
  **end if**
**end for**

**Belief network loss (with IB)**

**Algorithm 1** Belief net SVG(0) with IB (learner)

initial states: $h_0^\pi, h_0^Q, h_0^b$
policy: $q_{t,\theta}^\pi \equiv q_\theta^\pi(z_t|\tau_{0:t}, h_0^\pi), \pi_{t,\theta} \equiv \pi_\theta(a_t|z_t)$
Q: $q_{t,\psi}^Q(a_t) \equiv q_\psi^Q(z_t|\tau_{0:t}, a_t, h_0^Q), Q_{t,\psi} \equiv Q_\psi(z_t)$
belief: $q_{t,\phi}^b \equiv q_\phi^b(z_t|\tau_{0:t}, h_0^b), b_{t,\phi} \equiv b_\phi(w|z_t)$
online parameters: $\theta_O, \psi_O$
target parameters: $\theta_T = \theta_O, \psi_T = \psi_O$
replay buffer: $\mathcal{B}$
batch size: $B$
unroll length: $U$
target update period: $M$
**for** $step = 1$ **to** $\infty$ **do**
  $\{h_0^{i,\pi}, h_0^{i,Q}, h_0^{i,b}, \tau_{0:U}^i, w^i\}_{i=1}^B \leftarrow B$ samples from $\mathcal{R}$
  $(x_0^i, a_0^i, r_0^i, \ldots, r_{U-1}^i, x_U^i) \leftarrow \tau_{0:U}^i$
  $L_\pi = 0, L_Q = 0, L_b = 0$
  **for** $i = 1$ **to** $B$ **do**
    $L_{t,b}^i = \mathbb{E}_{z\sim q_{t,\phi}^b}[\log b(w^i|z)] + \lambda D_{\mathrm{KL}}[q_{t,\phi}^b||N(0,1)]$
    $(f_0^i, \ldots, f_{U-1}^i) \leftarrow q_{t,\phi}^b$ features
    $\tau_{0:U}^i \leftarrow \tau_{0:U}^i \cup (f_0^i, \ldots, f_{U-1}^i)$
    $V_t^i = \mathbb{E}_{z^\pi\sim q_{t,\theta_T}^\pi} \mathbb{E}_{a\sim\pi_{\theta_T}(\cdot|z^\pi)} \mathbb{E}_{z^Q\sim q_{t,\psi_T}^Q(a)} \left[Q_{\psi_T}(z^Q)\right]$
    $Q_t^i = r_t^i + \gamma V_{t+1}^i$
    $Q_t^{R,i} = Retrace(Q_t^i)$
    $L_{t,Q}^i = \mathbb{E}_{z\sim q_{t,\psi_O}^Q(a_t^i)} \left[(Q_{\psi_O}(z) - Q_t^{R,i})^2\right]$
    $L_{t,Q}^i \mathrel{+}= \lambda D_{\mathrm{KL}}[q_{t,\psi_O}^Q(a_t^i)||N(0,1)]$
    $L_{t,\pi}^i = \mathbb{E}_{z^\pi\sim q_{t,\theta_O}^\pi} \mathbb{E}_{a\sim\pi_{\theta_O}(\cdot|z^\pi)} \mathbb{E}_{z^Q\sim q_{t,\psi_T}^Q(a)} \left[Q_{\psi_T}(z^Q)\right]$
    $L_{t,\pi}^i \mathrel{+}= \lambda D_{\mathrm{KL}}[q_{t,\theta_O}^\pi||N(0,1)] - \alpha \mathrm{H}[\pi_{t,\theta_O}]$
    $L_Q \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,Q}^i$
    $L_\pi \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,\pi}^i$
    $L_b \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,b}^i$
  **end for**
  $\theta_O = Adam[\nabla_{\theta_O} L_\pi]$
  $\psi_O = Adam[\nabla_{\psi_O} L_Q]$
  $\phi = Adam[\nabla_\phi L_b]$
  **if** $step \,\%\, M = 0$ **then**
    $\theta_T = \theta_O$
    $\psi_T = \psi_O$
  **end if**
**end for**

**Belief network loss (with IB)**

**Critic network loss (with IB)**

**Algorithm 1** Belief net SVG(0) with IB (learner)

initial states: $h_0^\pi, h_0^Q, h_0^b$
policy: $q_{t,\theta}^\pi \equiv q_\theta^\pi(z_t|\tau_{0:t}, h_0^\pi), \pi_{t,\theta} \equiv \pi_\theta(a_t|z_t)$
Q: $q_{t,\psi}^Q(a_t) \equiv q_\psi^Q(z_t|\tau_{0:t}, a_t, h_0^Q), Q_{t,\psi} \equiv Q_\psi(z_t)$
belief: $q_{t,\phi}^b \equiv q_\phi^b(z_t|\tau_{0:t}, h_0^b), b_{t,\phi} \equiv b_\phi(w|z_t)$
online parameters: $\theta_O, \psi_O$
target parameters: $\theta_T = \theta_O, \psi_T = \psi_O$
replay buffer: $\mathcal{B}$
batch size: $B$
unroll length: $U$
target update period: $M$
**for** $step = 1$ **to** $\infty$ **do**
  $\{h_0^{i,\pi}, h_0^{i,Q}, h_0^{i,b}, \tau_{0:U}^i, w^i\}_{i=1}^B \leftarrow B$ samples from $\mathcal{R}$
  $(x_0^i, a_0^i, r_0^i, \ldots, r_{U-1}^i, x_U^i) \leftarrow \tau_{0:U}^i$
  $L_\pi = 0, L_Q = 0, L_b = 0$
  **for** $i = 1$ **to** $B$ **do**
    $L_{t,b}^i = \mathbb{E}_{z\sim q_{t,\phi}^b}[\log b(w^i|z)] + \lambda D_{\mathrm{KL}}[q_{t,\phi}^b||N(0,1)]$
    $(f_0^i, \ldots, f_{U-1}^i) \leftarrow q_{t,\phi}^b$ features
    $\tau_{0:U}^i \leftarrow \tau_{0:U}^i \cup (f_0^i, \ldots, f_{U-1}^i)$
    $V_t^i = \mathbb{E}_{z^\pi\sim q_{t,\theta_T}^\pi} \mathbb{E}_{a\sim\pi_{\theta_T}(\cdot|z^\pi)} \mathbb{E}_{z^Q\sim q_{t,\psi_T}^Q(a)} \left[Q_{\psi_T}(z^Q)\right]$
    $Q_t^i = r_t^i + \gamma V_{t+1}^i$
    $Q_t^{R,i} = Retrace(Q_t^i)$
    $L_{t,Q}^i = \mathbb{E}_{z\sim q_{t,\psi_O}^Q(a_t^i)} \left[(Q_{\psi_O}(z) - Q_t^{R,i})^2\right]$
    $L_{t,Q}^i \mathrel{+}= \lambda D_{\mathrm{KL}}[q_{t,\psi_O}^Q(a_t^i)||N(0,1)]$
    $L_{t,\pi}^i = \mathbb{E}_{z^\pi\sim q_{t,\theta_O}^\pi} \mathbb{E}_{a\sim\pi_{\theta_O}(\cdot|z^\pi)} \mathbb{E}_{z^Q\sim q_{t,\psi_T}^Q(a)} \left[Q_{\psi_T}(z^Q)\right]$
    $L_{t,\pi}^i \mathrel{+}= \lambda D_{\mathrm{KL}}[q_{t,\theta_O}^\pi||N(0,1)] - \alpha \mathsf{H}[\pi_{t,\theta_O}]$
    $L_Q \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,Q}^i$
    $L_\pi \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,\pi}^i$
    $L_b \mathrel{+}= \frac{1}{B}\sum_{t=0}^{U-1} L_{t,b}^i$
  **end for**
  $\theta_O = Adam[\nabla_{\theta_O} L_\pi]$
  $\psi_O = Adam[\nabla_{\psi_O} L_Q]$
  $\phi = Adam[\nabla_\phi L_b]$
  **if** $step \,\%\, M = 0$ **then**
    $\theta_T = \theta_O$
    $\psi_T = \psi_O$
  **end if**
**end for**

**Belief network loss (with IB)**

**Critic network loss (with IB)**

**Policy network loss (with IB and entropy regularization)**

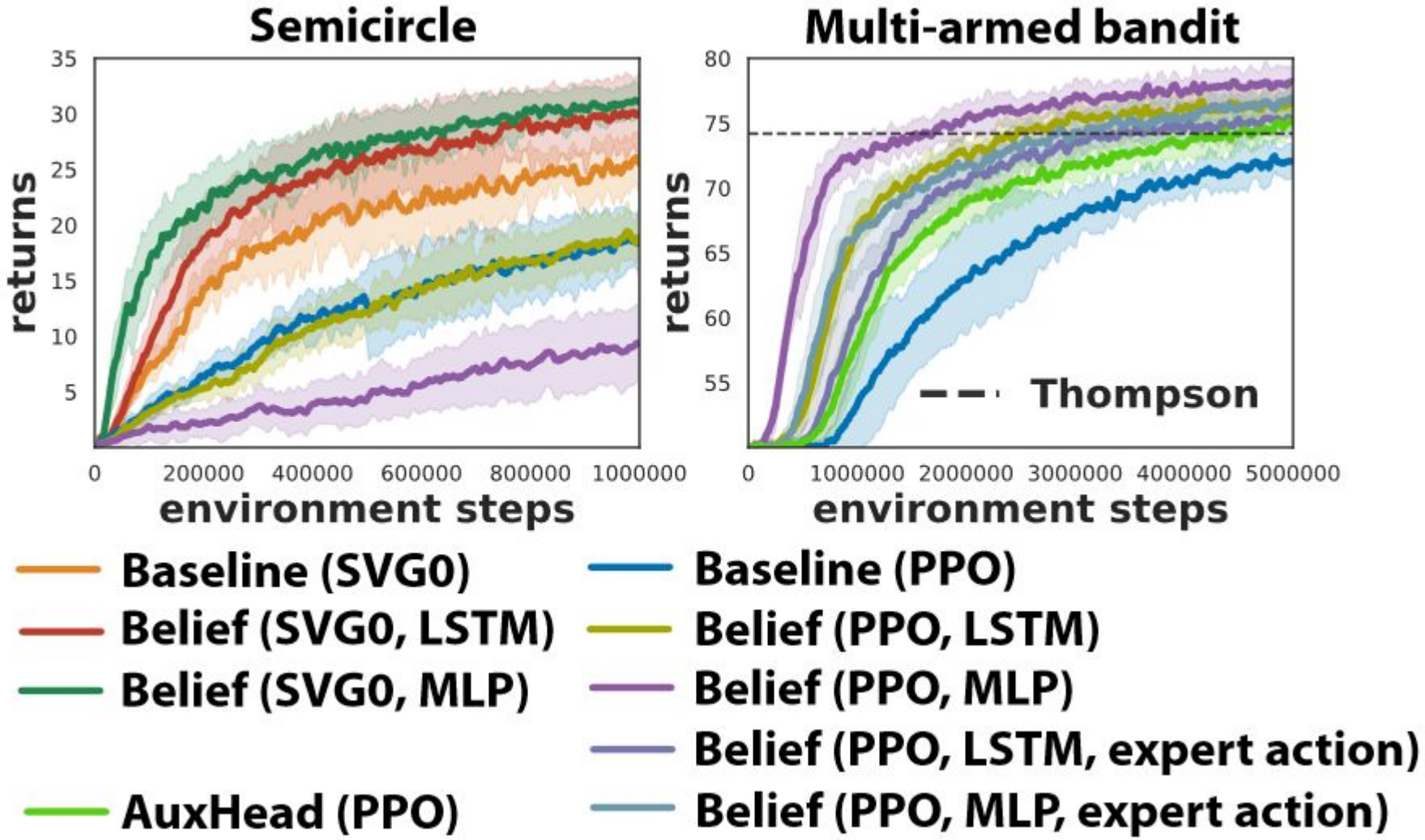
# Experimental Results

**Off-policy and on-policy learning**

Multi Arm bandit : 20 arms and horizon 100

Semicircle: Reach a target on semicircle

For semicircle task where the comparison was done, off policy SVG(0) performs better than PPO

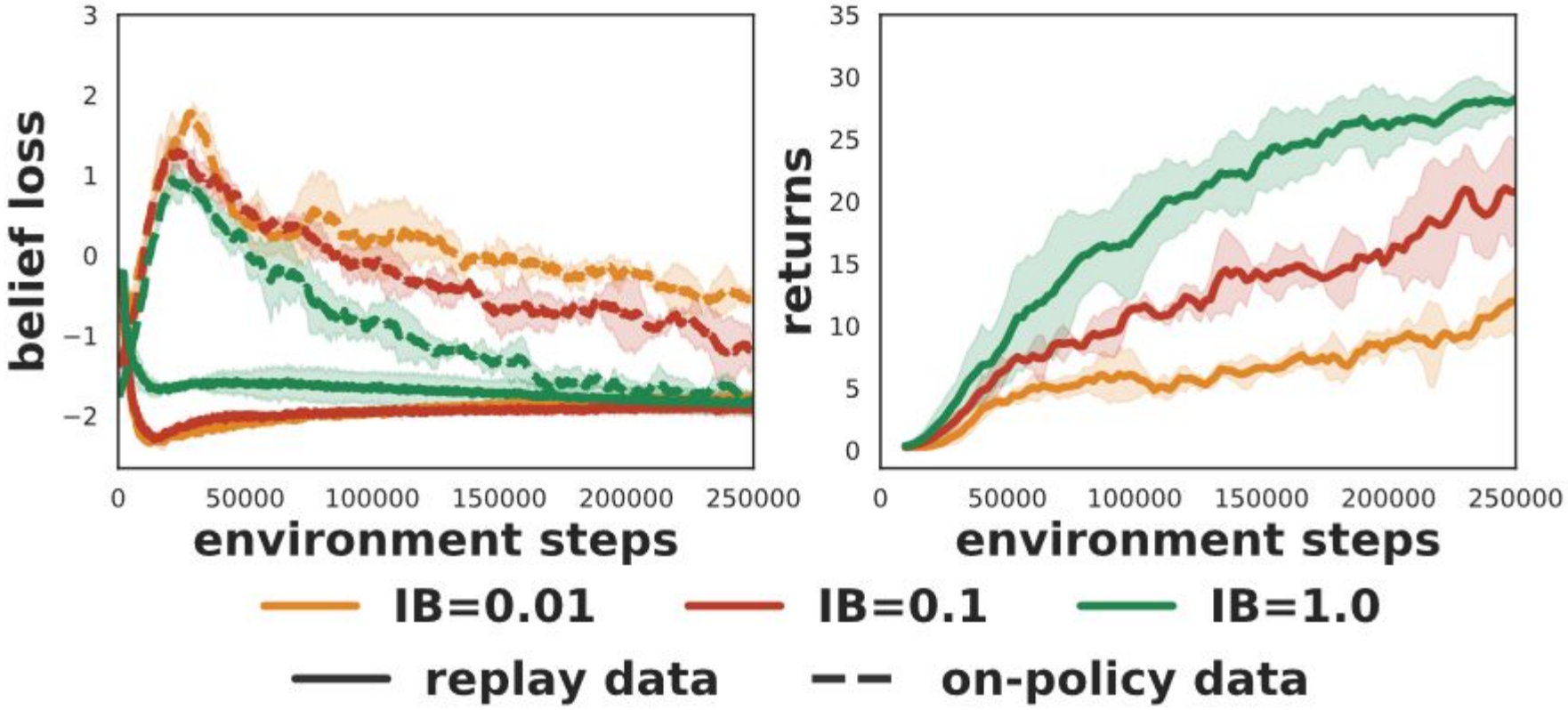


# Experimental Results

For semicircle task where the comparison was done, off policy SVG(0) performs better than PPO

Increasing the regularization strength of IB decreases the generalization gap, and it increases the sample efficiency of the agent

**Effect of Information bottleneck**

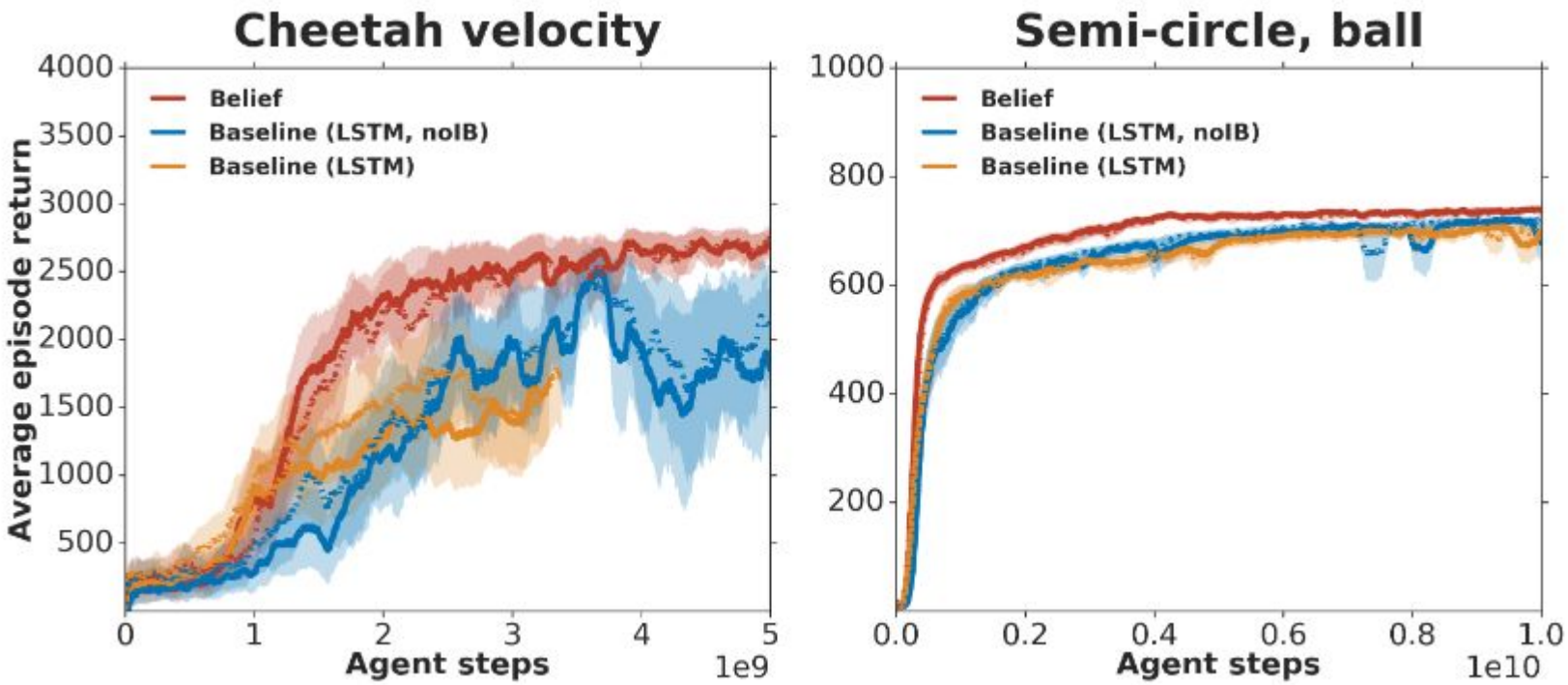


# Experimental Results

## Role of supervision

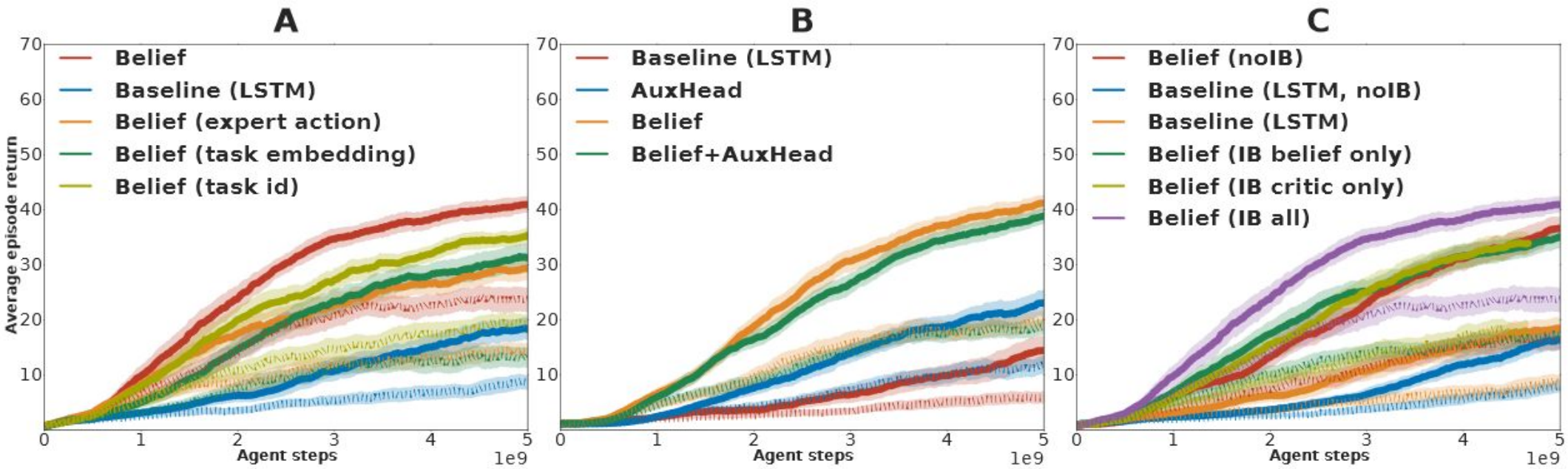
For Cheetah (simulated cheetah has to run at a particular speed), The use of supervision proved to be beneficial

For semi-circle ball, The use of supervision proved to be beneficial but not very significant



# Experimental Results

## Complex continuous control tasks

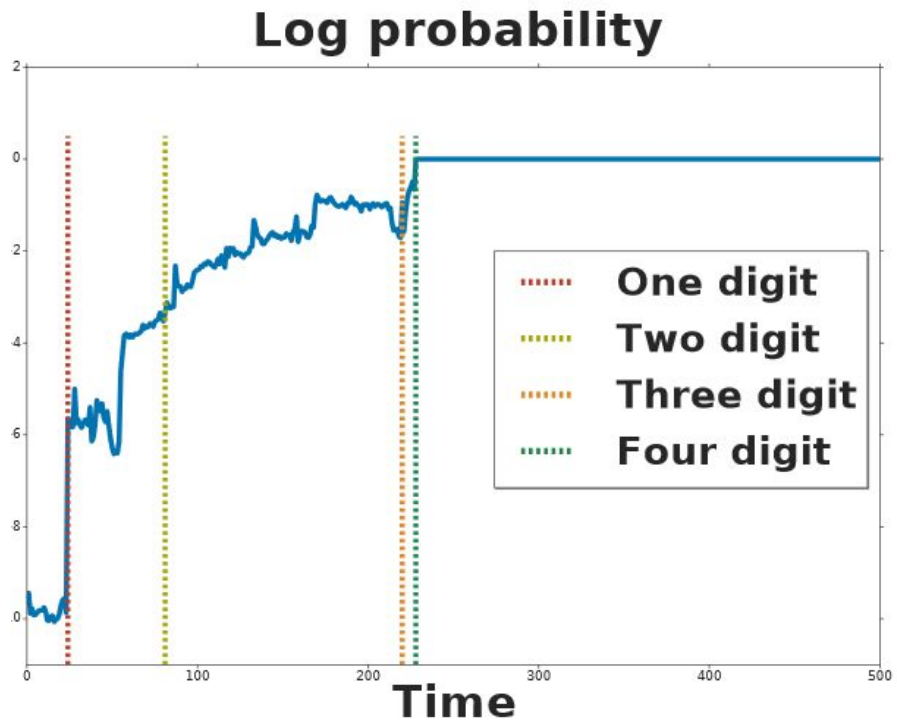



Results for NumPad, a complex continuous control task with sparse rewards and requires long term memory in order to solve with each task as a POMDP

# Experimental Results

The likelihood that the agent assigns to the true task sequence increases rapidly with each new tile in the sequence that is discovered in NumPad environment

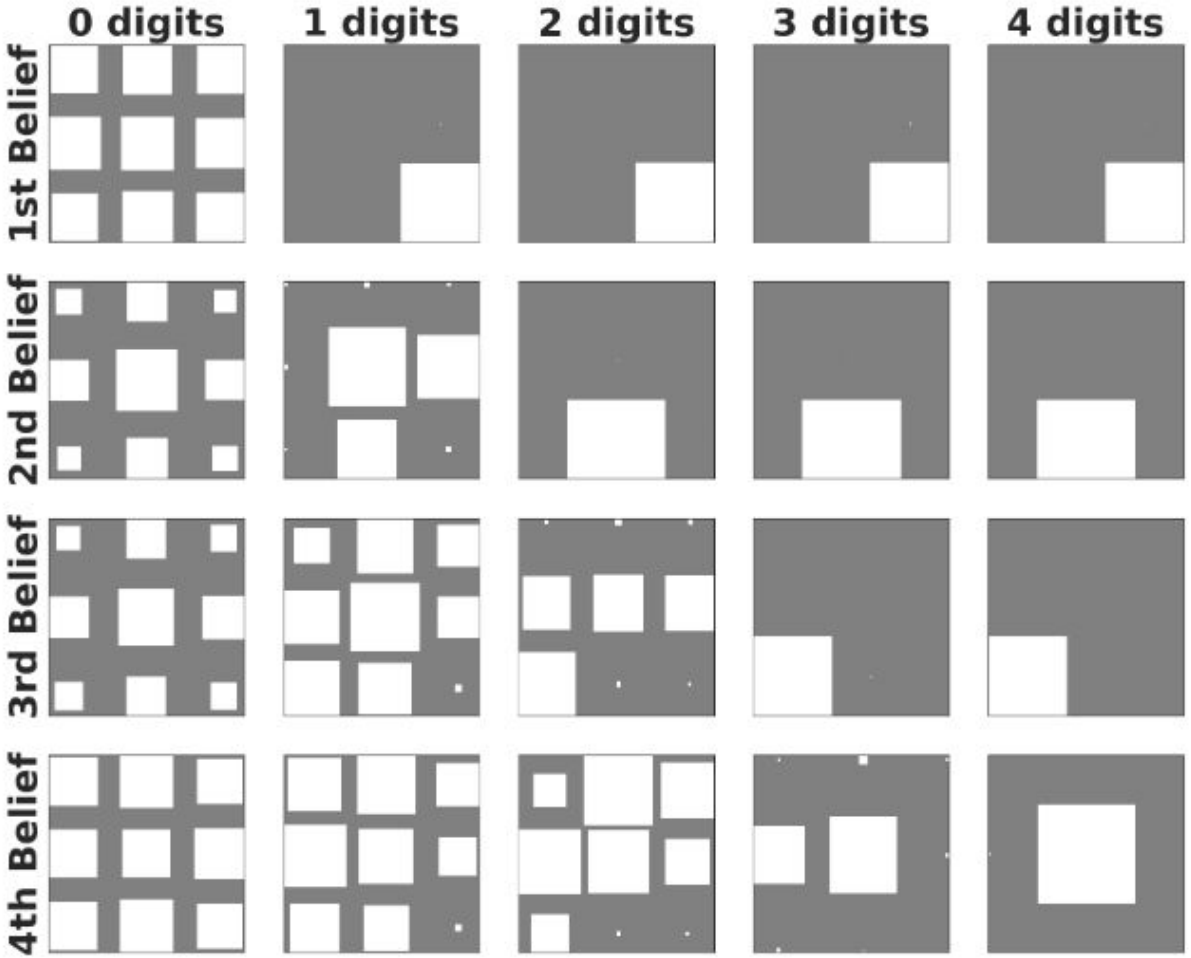
## Behavior Analysis



# Experimental Results

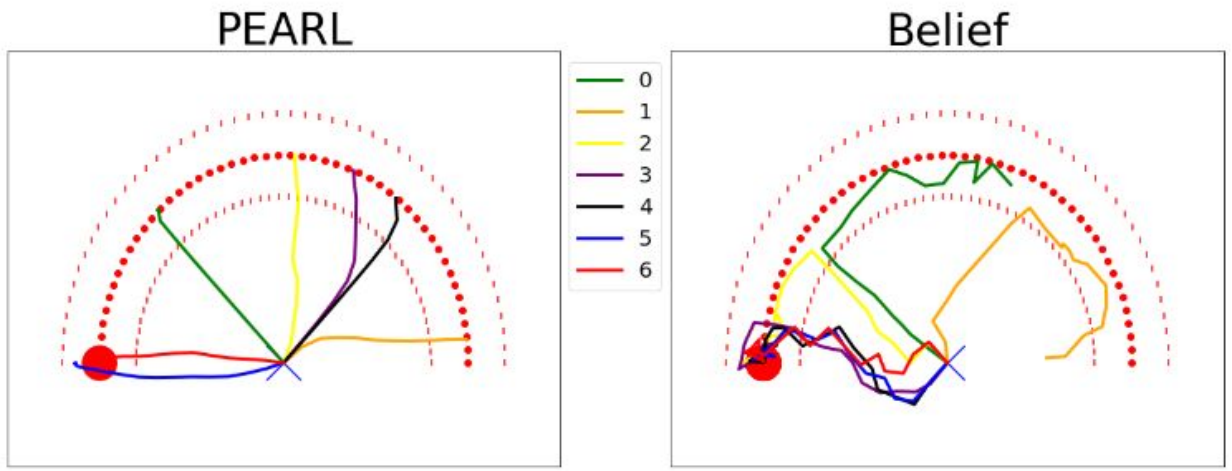
Hinton diagrams visualizing beliefs about a 4 digit sequence. Each row shows the marginal probabilities for each digit. We visualize these marginals at times (columns) in an episode just before the agent discovers a new digit in the unknown task sequence (the last one is after discovering all digits). The belief of this agent reflects the contiguous structure of the allowed sequences: for example, in 3rd column, knowing that the first tile is in the lower left corner (1st row) and the second is at the center on the bottom (2nd row) makesthe agent infer that the third tile (3rd row) is one of the tiles which neighbor these two

## Behavior Analysis

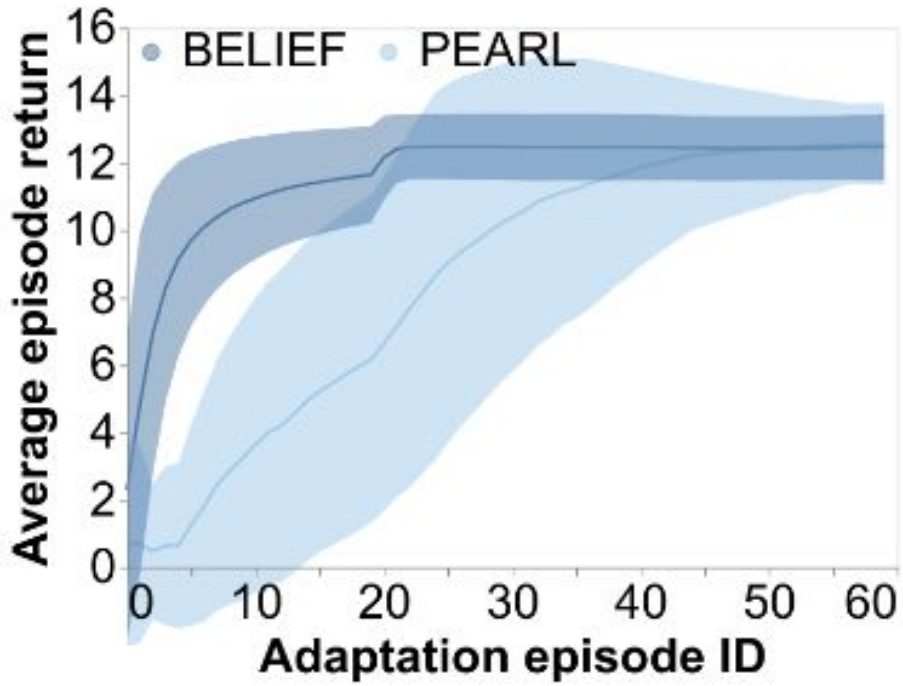


# Experimental Results

This figure shows comparison to PEARL that uses the more heuristic suboptimal “Thompson Sampling” search strategy. The Belief agent adapts much faster. The reason is depicted below

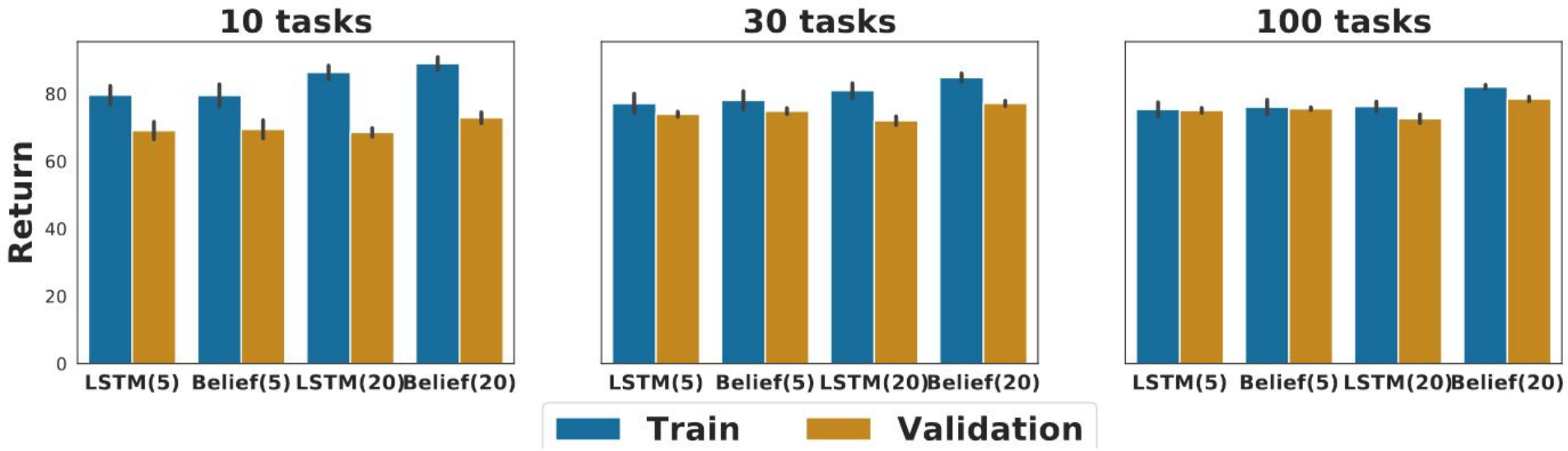



## Behavior Analysis



# Experimental Results

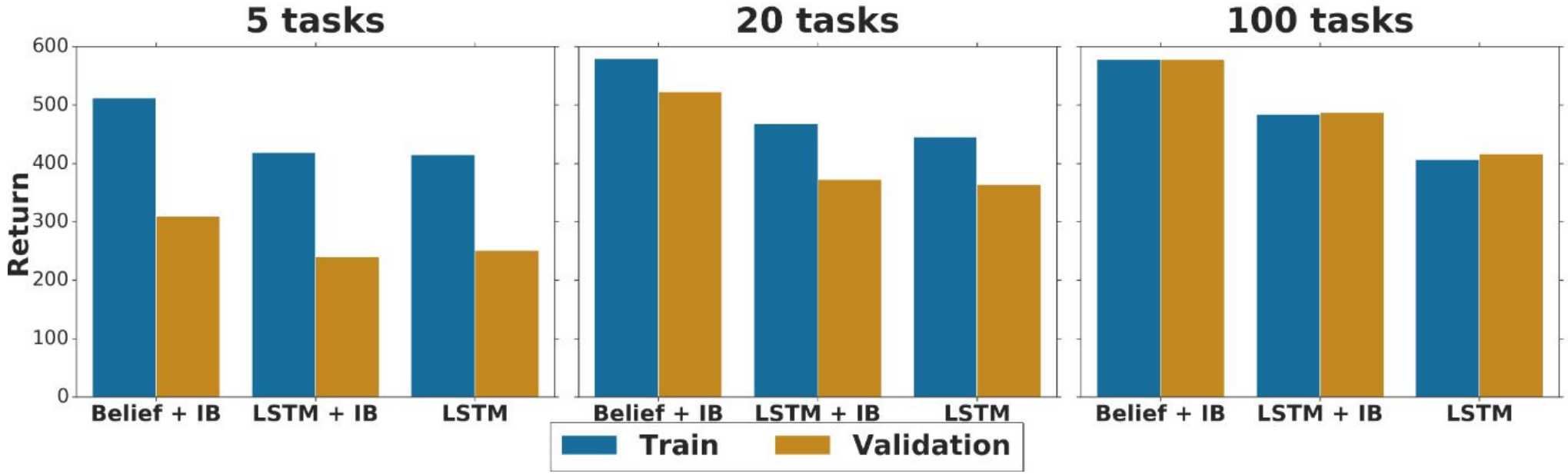
**Generalisation across tasks**




Training vs validation performance on the multi-arm bandit environment

# Experimental Results

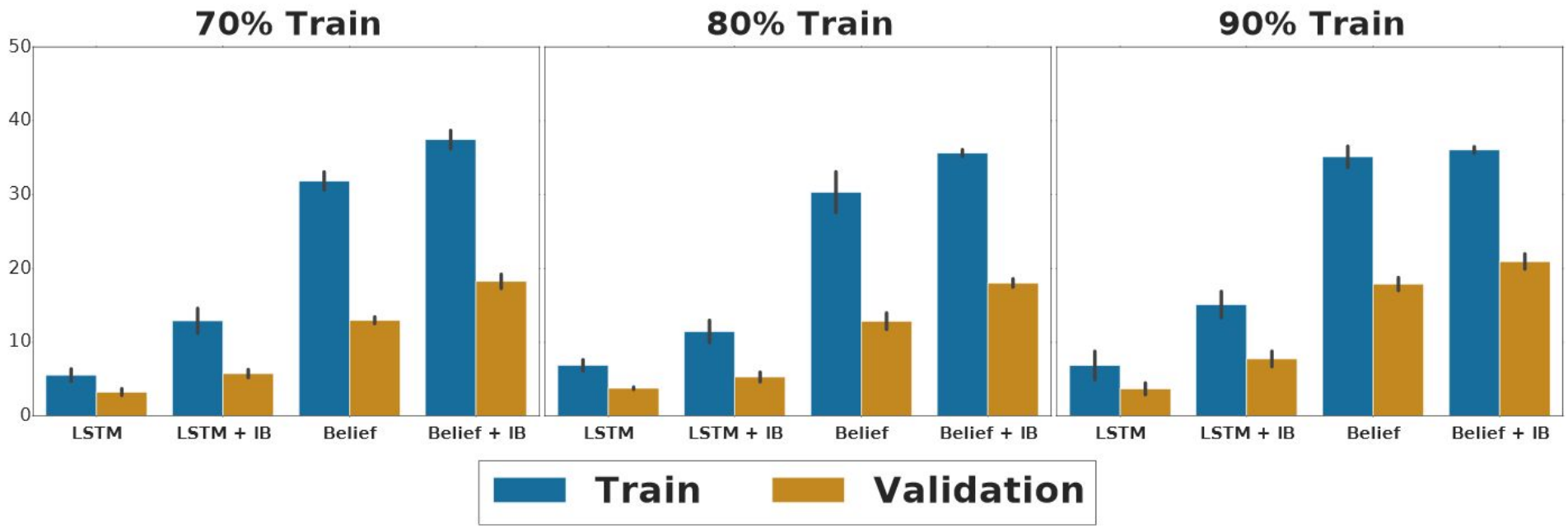
**Generalisation across tasks**




Dependance of generalisation gap on number of tasks on Quadraped Semi-Circle environment

# Experimental Results

**Generalisation across tasks**




Dependance of generalisation gap on training set size in Numpad environment

# Discussion of results

- Privileged information that is available in meta-RL can boost performance of both on-policy and off-policy meta-RL algorithms

# Discussion of results

- Privileged information that is available in meta-RL can boost performance of both on-policy and off-policy meta-RL algorithms
- The belief state can be estimated from off-policy data and thus the module can be combined with efficient off-policy algorithms

# Discussion of results

- Privileged information that is available in meta-RL can boost performance of both on-policy and off-policy meta-RL algorithms
- The belief state can be estimated from off-policy data and thus the module can be combined with efficient off-policy algorithms
- IB regularization helps prevent overfitting and leads to more efficient off-policy learning

# Discussion of results

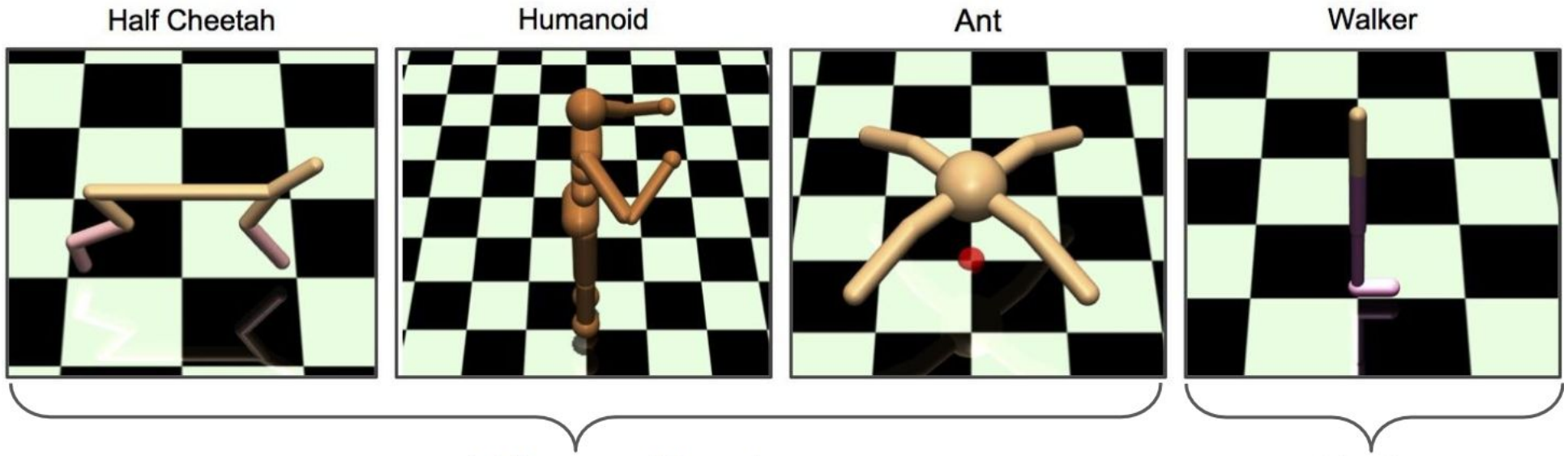
- Privileged information that is available in meta-RL can boost performance of both on-policy and off-policy meta-RL algorithms
- The belief state can be estimated from off-policy data and thus the module can be combined with efficient off-policy algorithms
- IB regularization helps prevent overfitting and leads to more efficient off-policy learning
- Natural structured information in the task(eg instruction or goal location) is more helpful than unstructured information like task ID

# Critique / Limitations / Open Issues

- Justification for the choice of off-policy algorithm used : (distributed SVG(0)). Why not other off-policy algorithms like SAC given its benefits and use in other algorithms like PEARL?

- Need for more comparison with other meta-RL algorithms like PEARL on other environments as well.

# Critique / Limitations / Open Issues




Most of the experiments were under this regime

Need for more experiments in this regime

Fig adapted from http://web.stanford.edu/class/cs330/slides/Exploration%20in%20Meta-RL.pdf

# Critique / Limitations / Open Issues

- Many real-world tasks go beyond easy tasks controlled by single variable(like goal location,velocity,etc). For example, opening a drawer requires the ability to reach and pull which are 2 separate independent tasks (multi-modal task distributions) - Ren et al 2019.

  Can the paper mentioned in the framework work in these settings?

- According to Wang et al 2020, the framework ignores the role of exploration in task inference

# Contributions (Recap)

- The paper aims at taking advantage of privileged information in meta-RL to boost performance of meta-RL algorithms
- The framework in the paper allows for efficient training using off-policy data
- The paper demonstrates experimentally the ability to solve complex continuous control tasks with sparse reward and requiring long term memory