Meta
Reinforcement Learning

Kate Rakelly
11/13/19

Questions we seek to answer

Motivation: What problem is meta-RL trying to solve?

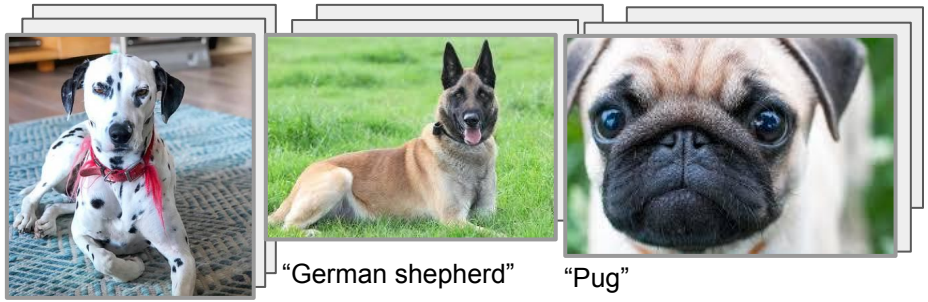
Context: What is the connection to other problems in RL?

Solutions: What are solution methods for meta-RL and their limitations?

Open Problems: What are the open problems in meta-RL?

Meta-learning problem statement

supervised learning



"Dalmation" "German shepherd" "Pug"

corgi

???

reinforcement learning

Robot art by Matt Spangler, *mattspangler.com*

Meta-RL problem statement

Regular RL: learn policy for single task

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

$$= f_{\mathrm{RL}}(\mathcal{M})$$

MDP

Meta-RL: learn adaptation rule

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

Meta-training / Outer loop

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

Adaptation / Inner loop

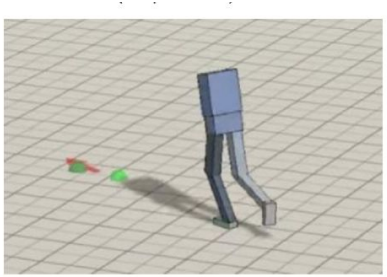
MDP for task i

$\mathcal{M}_1$ $\mathcal{M}_2$ $\mathcal{M}_3$

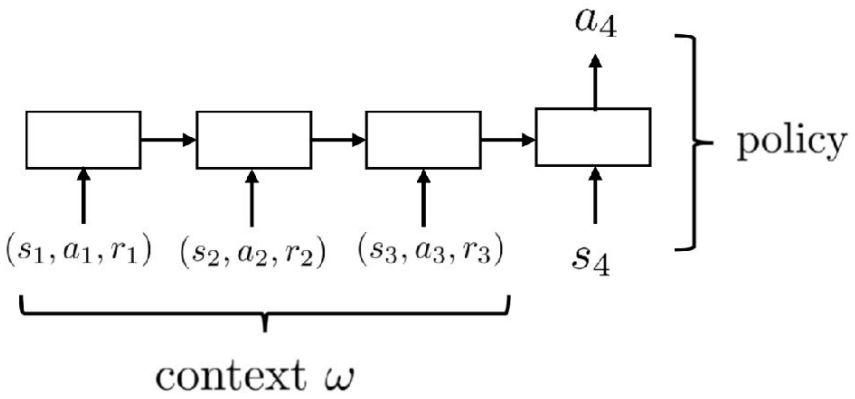
$\mathcal{M}_{test}$

Relation to goal-conditioned policies

ω: stack location



ω: walking direction



Meta-RL can be viewed as a goal-conditioned policy where the task information is inferred from *experience*

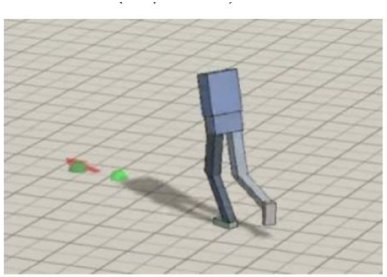
Task information could be about the dynamics or reward functions

Rewards are a strict generalization of goals

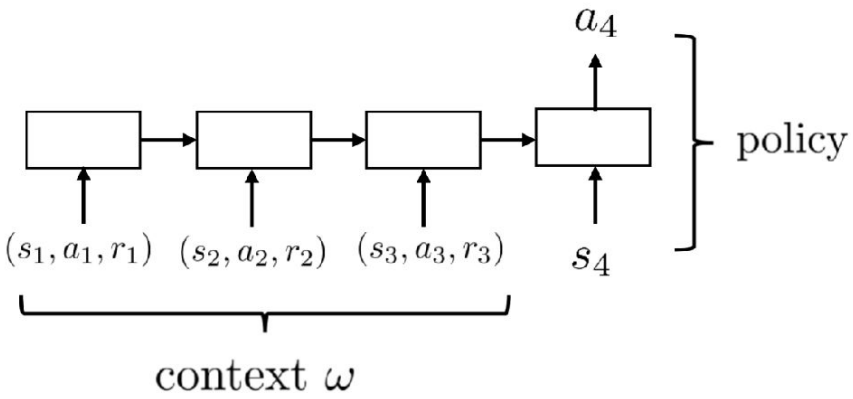
Slide adapted from Chelsea Finn

Relation to goal-conditioned policies

ω: stack location



ω: walking direction



Q: What is an example of a reward function that can't be expressed as a goal state?

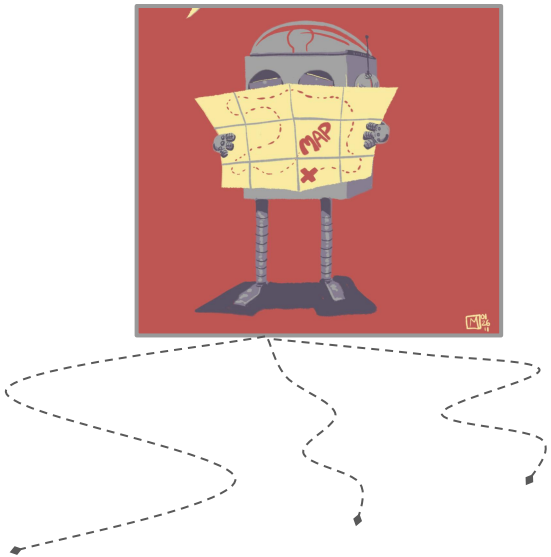
A: E.g., seek while avoiding, action penalties

Slide adapted from Chelsea Finn

Adaptation

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

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



What should the adaptation procedure do?

- **Explore**: Collect the most informative data
- **Adapt**: Use that data to obtain the optimal policy

General meta-RL algorithm outline

while training:

1. sample task i, collect data $\mathcal{D}_i$
2. adapt policy by computing $\phi_i = f(\theta, \mathcal{D}_i)$
3. collect data $\mathcal{D}_i'$ with adapted policy π_{ϕ_i}
4. update θ according to $\mathcal{L}(\mathcal{D}_i', \phi_i)$

Can do more than one round of adaptation

In practice, compute update across a batch of tasks

Different algorithms:

- Choice of function f
- Choice of loss function L

Solution Methods

Solution #1: recurrence

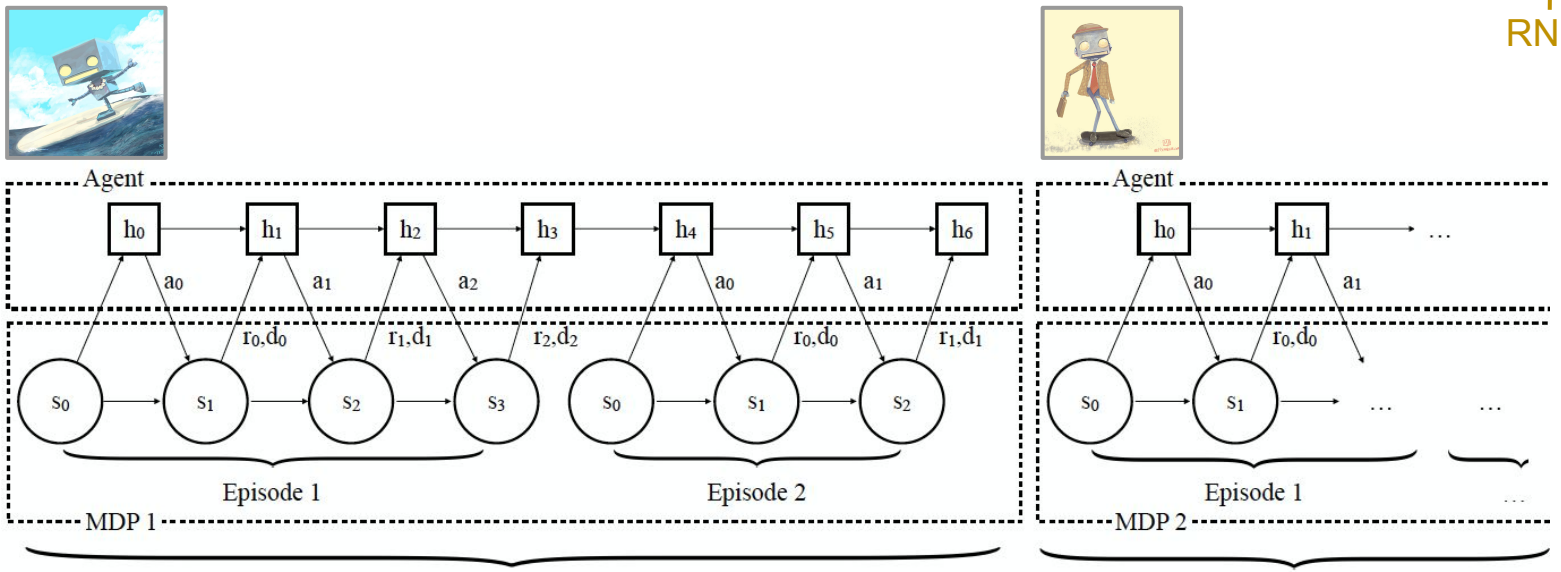
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

PG

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

RNN

Implement the policy as a recurrent network, train across a set of tasks



Persist the hidden state across episode boundaries for continued adaptation!

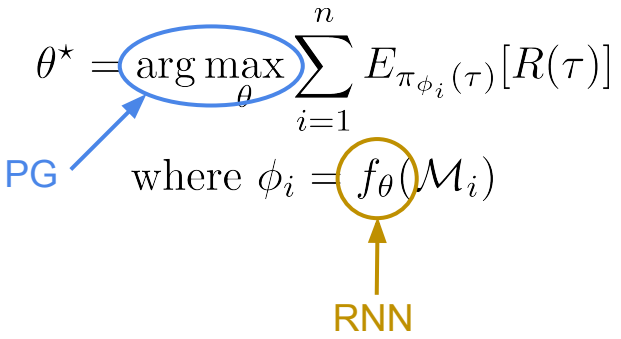
Duan et al. 2016, Wang et al. 2016. Heess et al. 2015. Fig adapted from Duan et al. 2016

Solution #1: recurrence

while training:
  for i in tasks:
    initialize hidden state $\mathbf{h_0} = 0$
    for t in timesteps:
      1. sample 1 transition $\mathcal{D}_i = \mathcal{D}_i \cup \{(s_t, a_t, s_{t+1}, r_t)\}$ from π_{h_t}
      2. update policy hidden state $\mathbf{h_{t+1}} = f_\theta(\mathbf{h_t}, s_t, a_t, s_{t+1}, r_t)$

  update policy parameters $\theta \leftarrow \theta - \nabla_\theta \sum_i \mathcal{L}_i(\mathcal{D}_i, \pi_{\mathbf{h}})$

Solution #1: recurrence



Pro: general, expressive

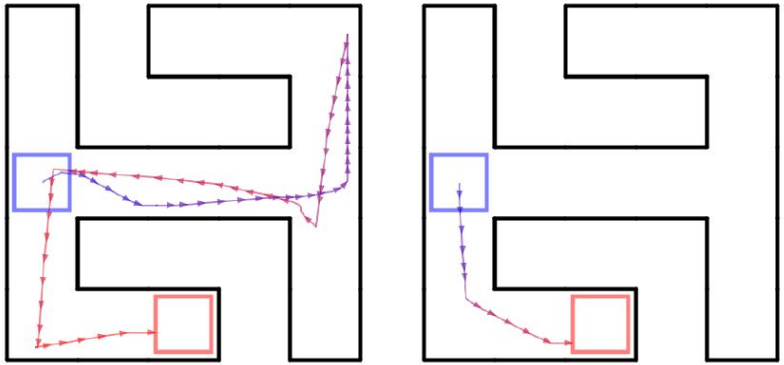
There exists an RNN that can compute any function

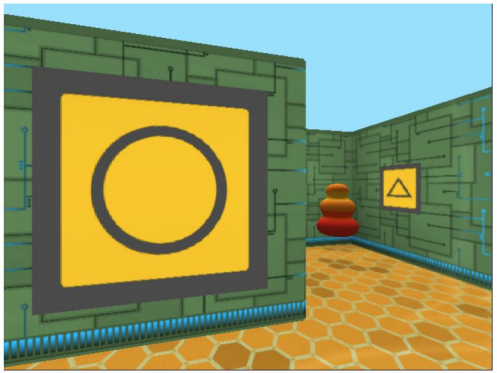
Con: not consistent

What does it mean for adaptation to be “consistent”?

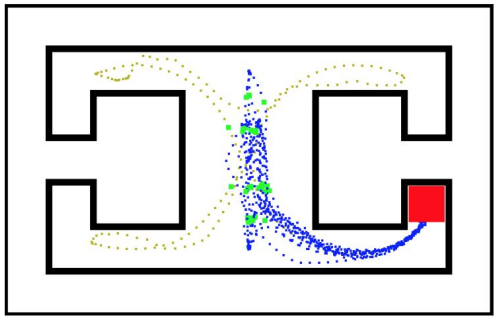
Will converge to the optimal policy given enough data

Solution #1: recurrence





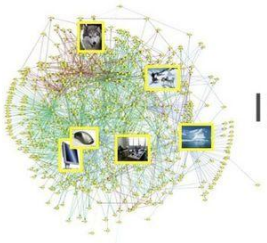
(a) Labryinth I-maze



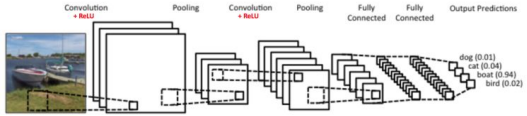
(b) Illustrative Episode

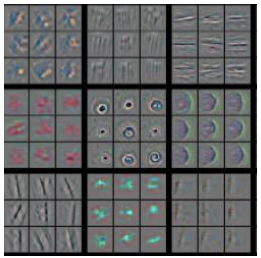
Duan et al 2016, Wang et al. 2016

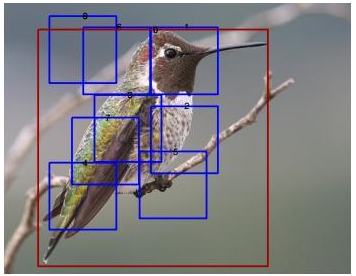
Wait, what if we just fine-tune?



IMAGENET







is pretraining a *type* of meta-learning?

better features = faster learning of new task!

Sample inefficient, prone to overfitting, and is particularly difficult in RL

Slide adapted from Sergey Levine

Solution #2: optimization

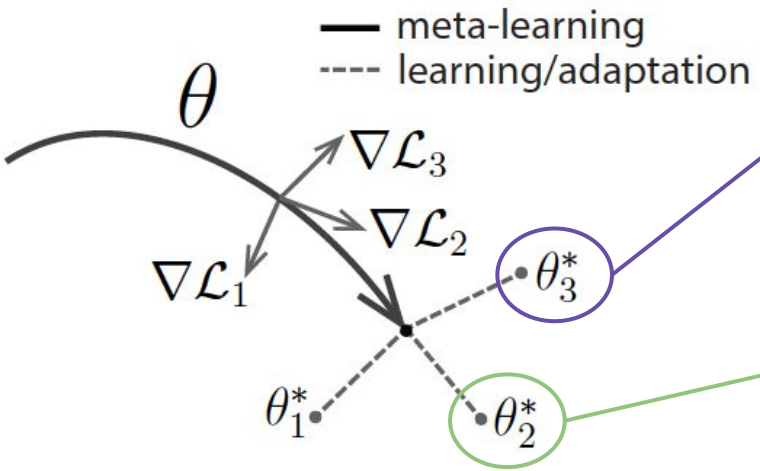
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

PG

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

PG

Learn a parameter initialization from which fine-tuning for a new task works!



Finn et al. 2017. Fig adapted from Finn et al. 2017

Solution #2: optimization

while training:
  for i in tasks:
    1. sample k episodes $\mathcal{D}_i = \{(s, a, s', r)\}_{1:k}$ from π_θ
    2. compute adapted parameters $\phi_i = \theta - \alpha \nabla_\theta \mathcal{L}_i(\pi_\theta, \mathcal{D}_i)$
    3. sample k episodes $\mathcal{D}'_i = \{(s, a, s', r)_{1:k}\}$ from π_ϕ

update policy parameters $\theta \leftarrow \theta - \nabla_\theta \sum_i \mathcal{L}_i(\mathcal{D}'_i, \pi_{\phi_i})$

Requires second order derivatives!

Finn et al. 2017. Fig adapted from Finn et al. 2017

Solution #2: optimization

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

PG

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

PG

How exploration is learned automatically

Pre-update parameters receive credit for producing good exploration trajectories

Causal relationship between pre and post-update trajectories is taken into account

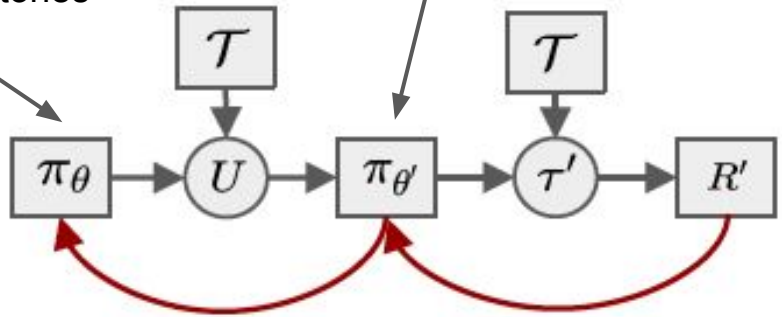


Fig adapted from Rothfuss et al. 2018

Solution #2: optimization

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

PG

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

PG

$$\nabla_\theta J(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\left[\mathbb{E}_{\substack{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta) \\ \boldsymbol{\tau}'\sim P_\mathcal{T}(\boldsymbol{\tau}'|\theta')}}\left[\nabla_\theta J_{\text{post}}(\boldsymbol{\tau},\boldsymbol{\tau}') + \nabla_\theta J_{\text{pre}}(\boldsymbol{\tau},\boldsymbol{\tau}')\right]\right]$$

$$\nabla_\theta J_{\text{post}}(\boldsymbol{\tau},\boldsymbol{\tau}') = \underbrace{\nabla_{\theta'}\log\pi_\theta(\boldsymbol{\tau}')R(\boldsymbol{\tau}')}_{\nabla_{\theta'}J^{\text{outer}}}\underbrace{\left(I + \alpha R(\boldsymbol{\tau})\nabla^2_\theta \log\pi_{\theta'}(\boldsymbol{\tau})\right)}_{\text{transformation from } \theta' \text{ to } \theta}$$

$$\nabla_\theta J^I_{\text{pre}}(\boldsymbol{\tau},\boldsymbol{\tau}') = \alpha\nabla_\theta\log\pi_\theta(\boldsymbol{\tau})\left(\underbrace{(\nabla_\theta\log\pi_\theta(\boldsymbol{\tau})R(\boldsymbol{\tau}))^\top}_{\nabla_\theta J^{\text{inner}}}\underbrace{(\nabla_{\theta'}\log\pi_{\theta'}(\boldsymbol{\tau}')R(\boldsymbol{\tau}'))}_{\nabla_{\theta'}J^{\text{outer}}}\right)$$

View this as a "return" that encourages gradient alignment

Fig adapted from Rothfuss et al. 2018

Solution #2: optimization

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

PG

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

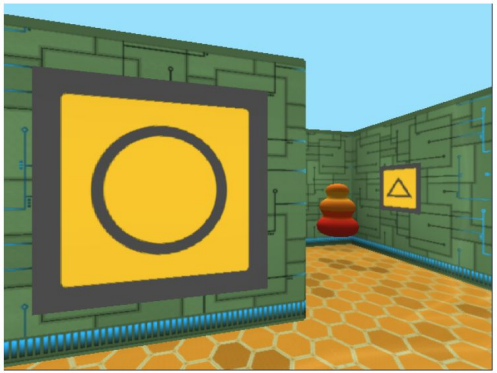
PG

Pro: consistent!

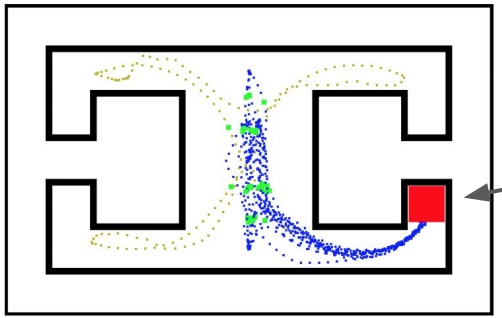
Con: not as expressive

Q: When could the optimization strategy be less expressive than the recurrent strategy?

Example: when no rewards are collected, adaptation will not change the policy, even though this data gives information about which states to avoid



(a) Labryinth I-maze



(b) Illustrative Episode

Suppose reward is given only in this region

Solution #2: optimization

Exploring in a sparse reward setting

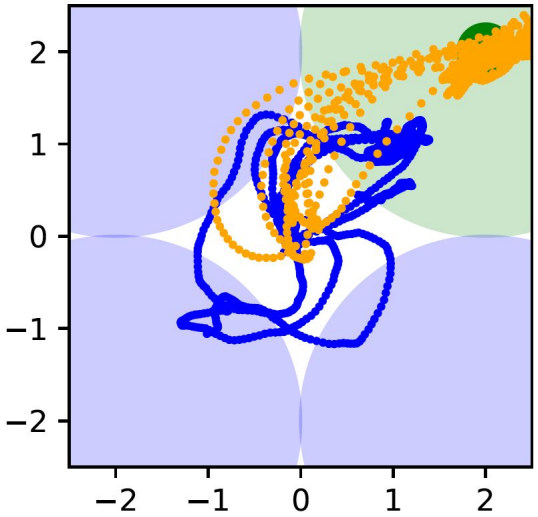


Fig adapted from Rothfuss et al. 2018

Cheetah running forward and back after 1 gradient step

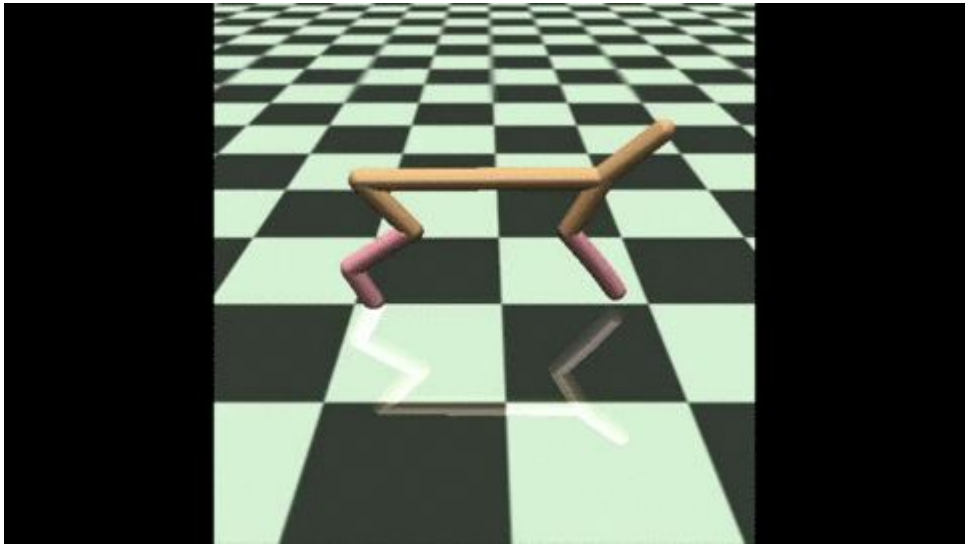
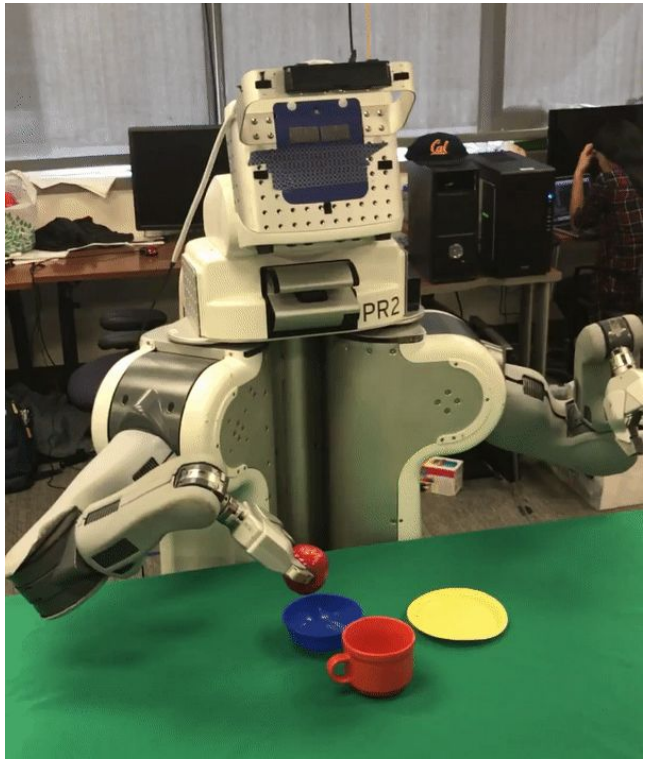


Fig adapted from Finn et al. 2017

Meta-RL on robotic systems

Meta-imitation learning

Demonstration



1-shot imitation

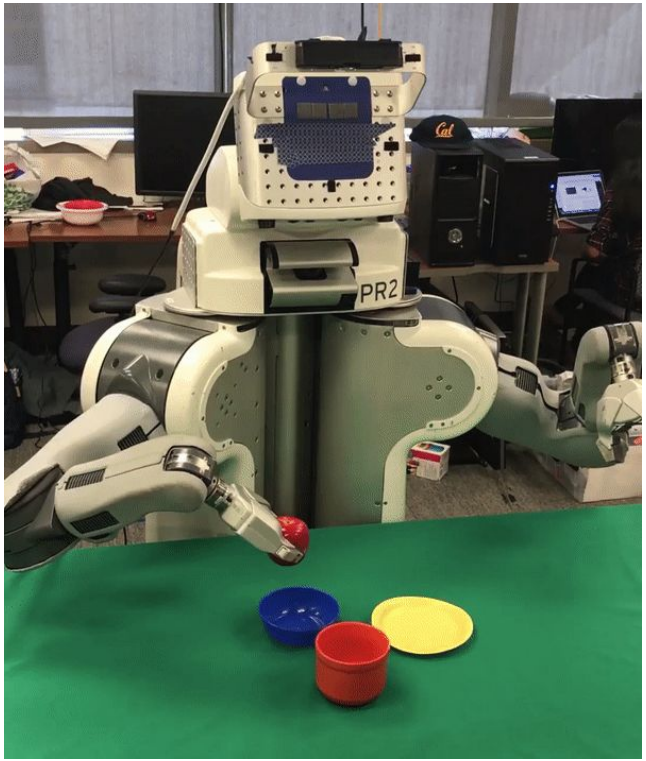


Figure adapted from BAIR Blog Post: One-Shot Imitation from Watching Videos

Meta-imitation learning

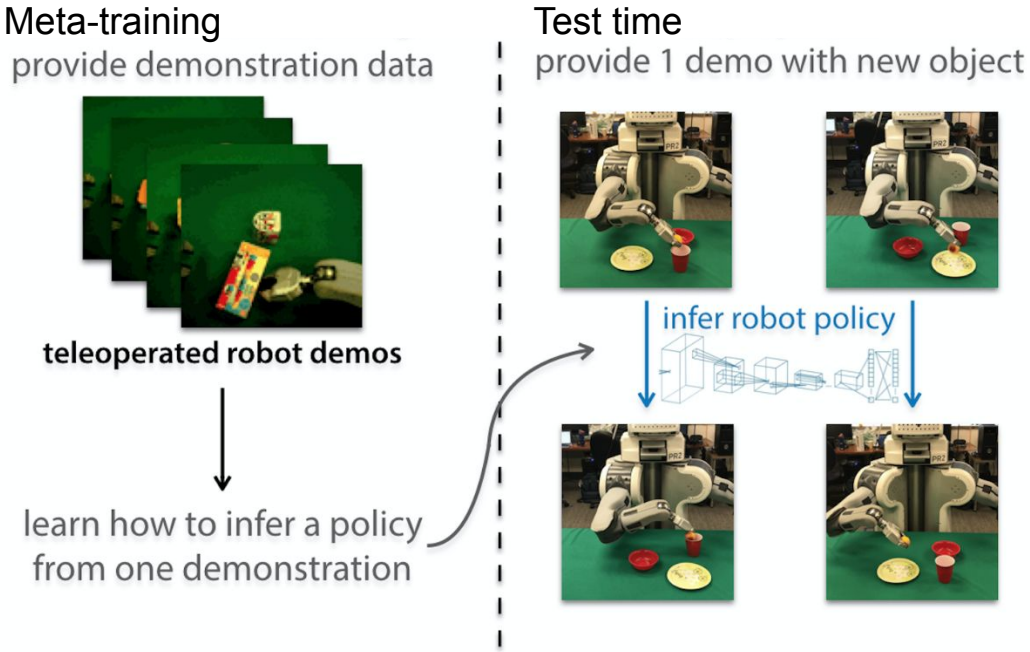
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

PG

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

Behavior cloning

Test: perform task given single **robot demo**
Training: run **behavior cloning** for adaptation

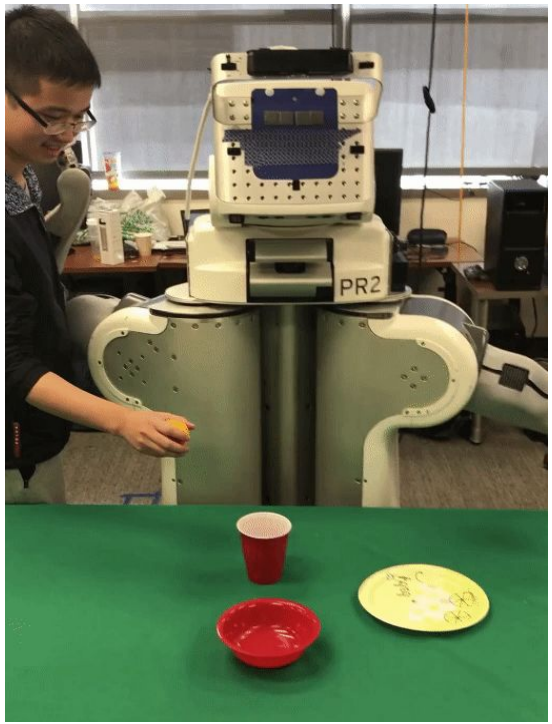


$$\phi_i = \theta - \alpha \nabla_\theta \sum_t ||\pi_\theta(o_t) - a_t^*||^2$$

Yu et al. 2017

Meta-imitation learning from **human** demos

demonstration



1-shot imitation

Figure adapted from BAIR Blog Post: One-Shot Imitation from Watching Videos

Meta-imitation learning from **humans**

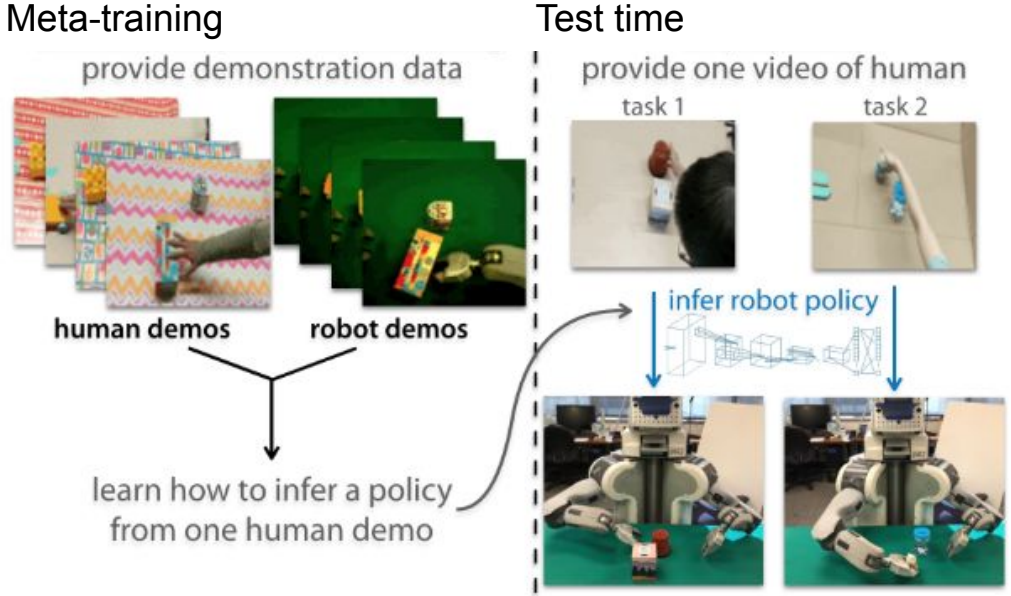
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

PG

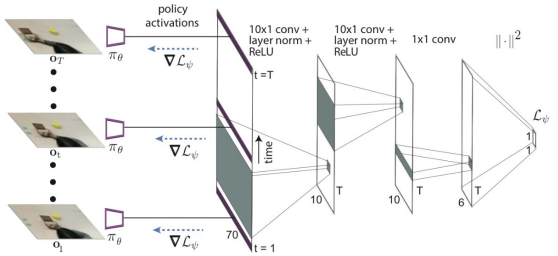
$$\text{where } \phi_i = f_\theta(\mathcal{M}_i)$$

Learned loss

Test: perform task given single **human demo**
Training: **learn a loss function** that adapts policy



$$\phi = \theta - \alpha \nabla_\theta \mathcal{L}_\psi(\theta, \mathbf{d}^h)$$

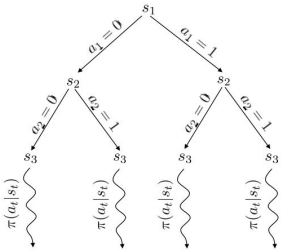


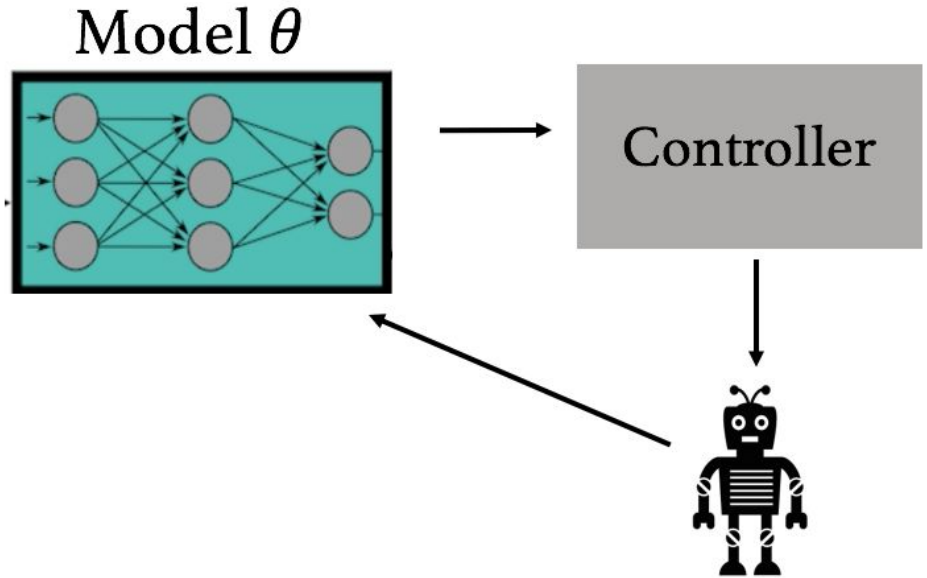
Supervised by **paired robot-human demos** only during meta-training!

Yu et al. 2018

Model-Based meta-RL

1. run base policy $\pi_0(\mathbf{a}_t|\mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. plan through $f(\mathbf{s}, \mathbf{a})$ to choose actions





What if the system dynamics change?

- Low battery
- Malfunction
- Different terrain

Re-train model? :(

Figure adapted from Anusha Nagabandi

Model-Based meta-RL

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

MPC

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

Supervised model learning

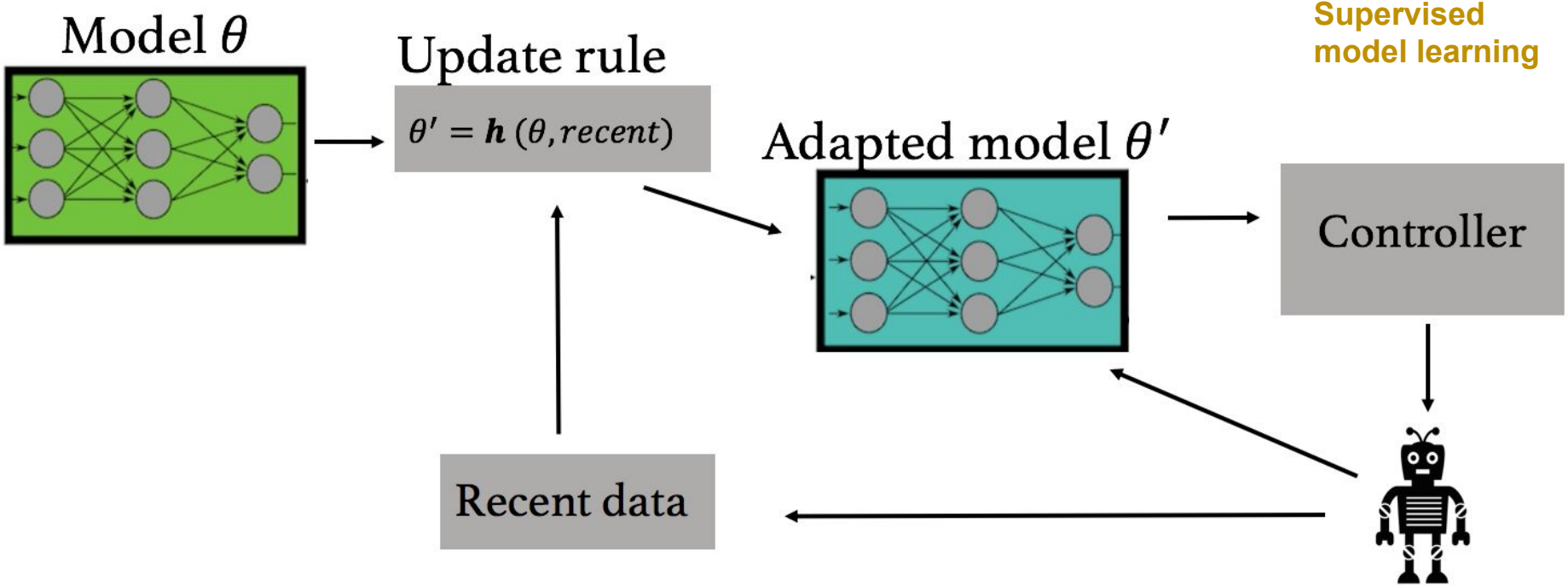
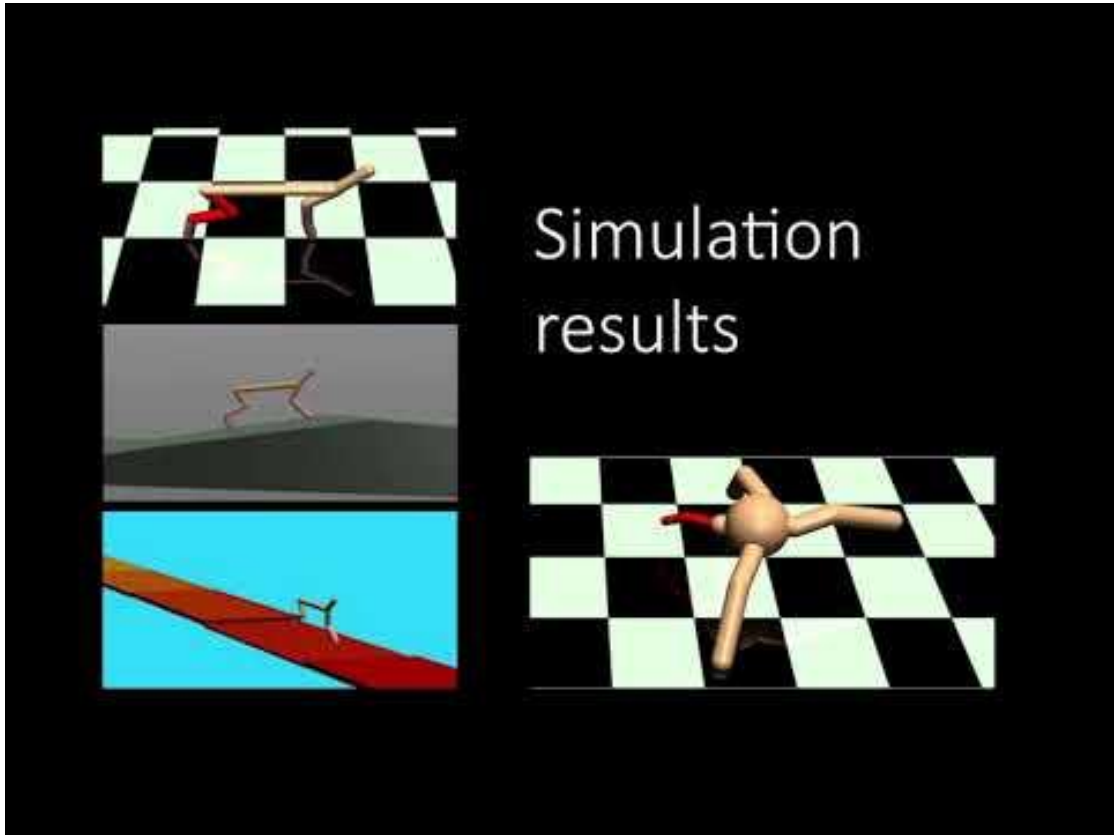


Figure adapted from Anusha Nagabandi

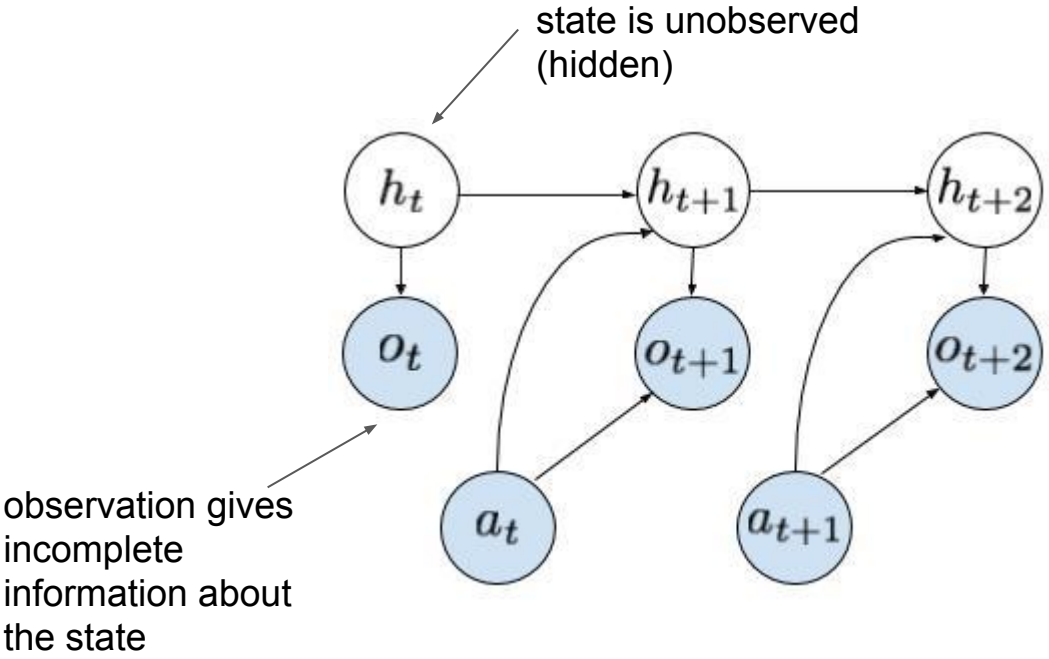
Model-Based meta-RL



Video from Nagabandi et al. 2019

Break

Aside: POMDPs

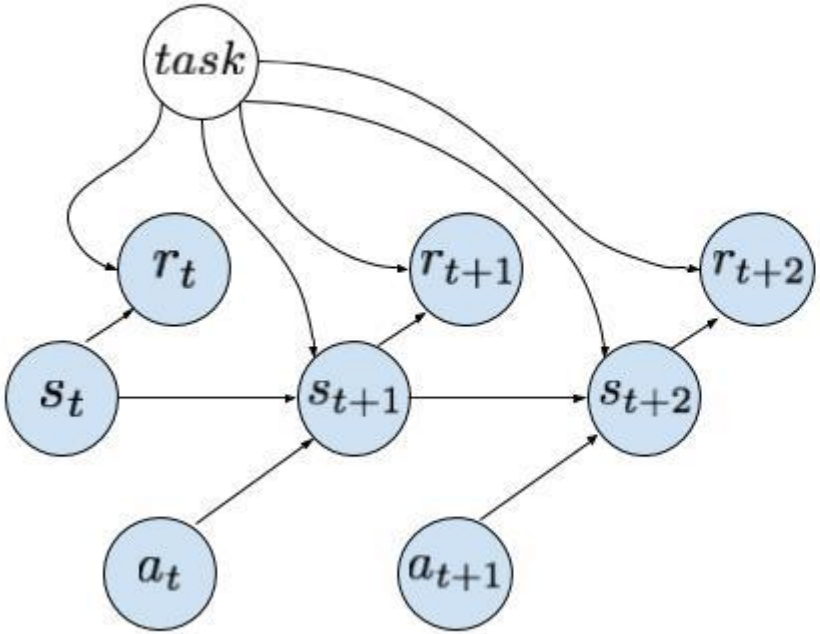


Example: incomplete sensor data

"That Way We Go" by Matt Spangler

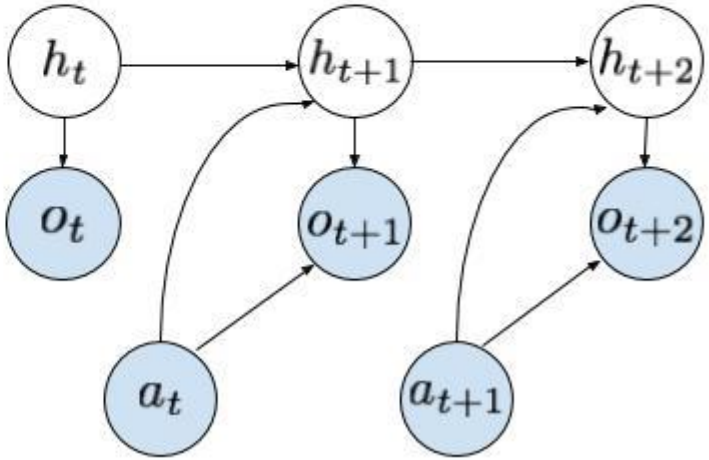
The POMDP view of meta-RL

meta-RL...



...as a POMDP

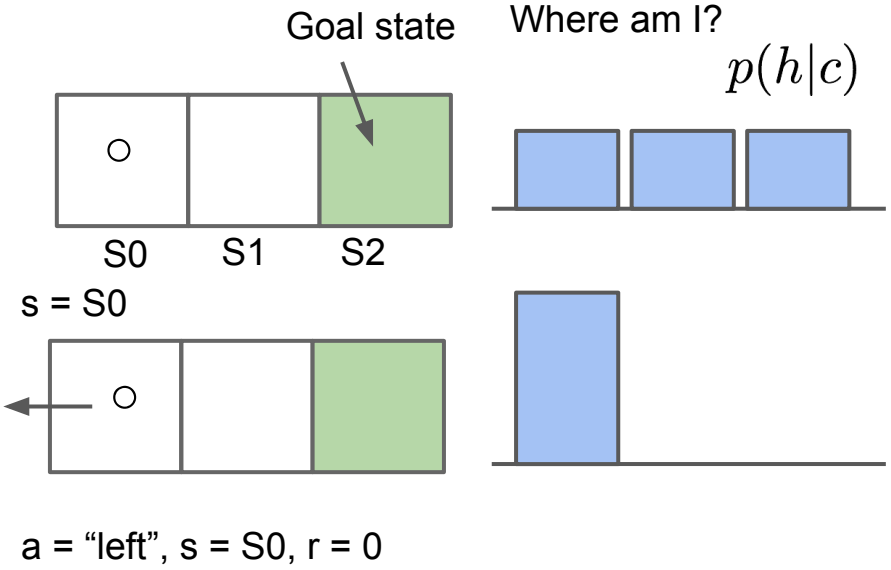
$h_t = (s_t, task)$ $o_t = (s_t, r_t)$



Two approaches to solve: 1) policy with memory (RNN) | 2) explicit state estimation

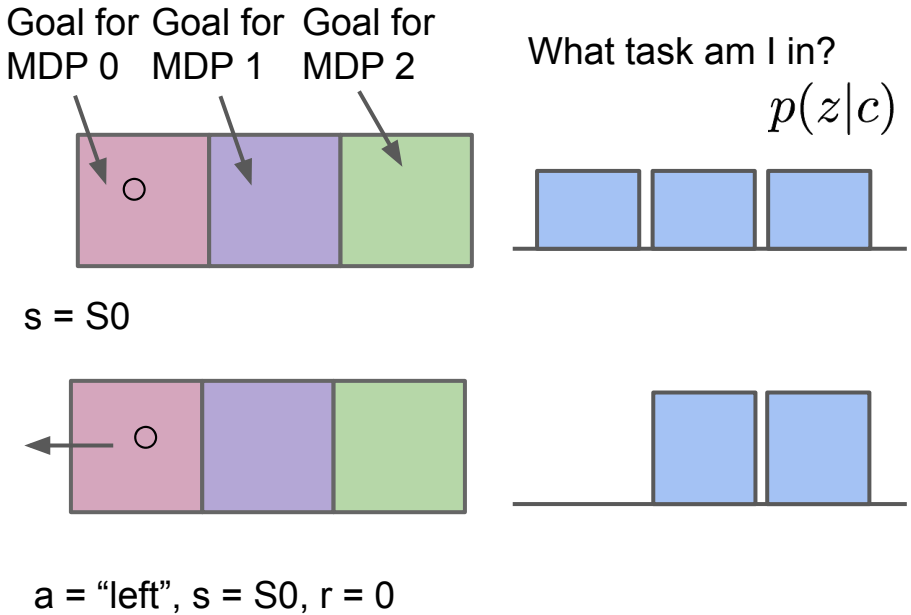
Model belief over latent task variables

POMDP for unobserved state



a = “left”, s = S0, r = 0

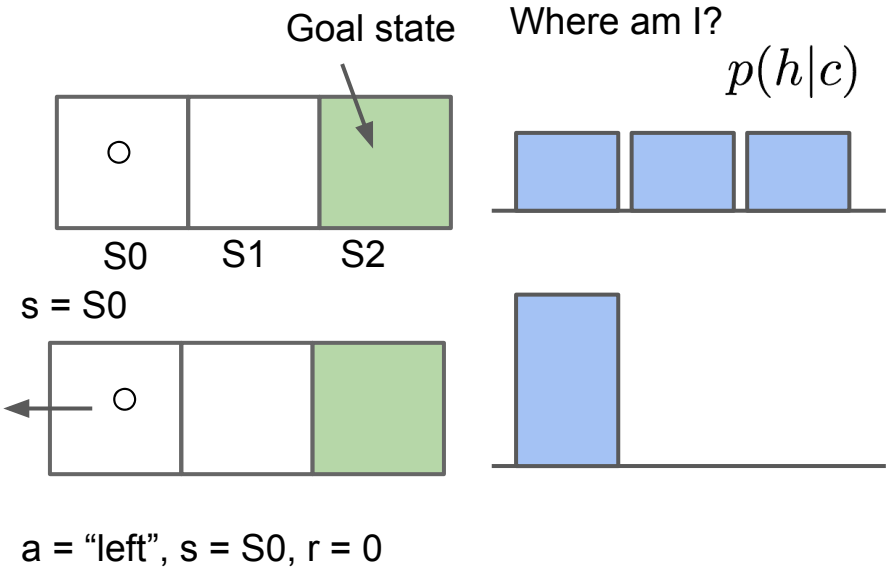
POMDP for unobserved task



a = “left”, s = S0, r = 0

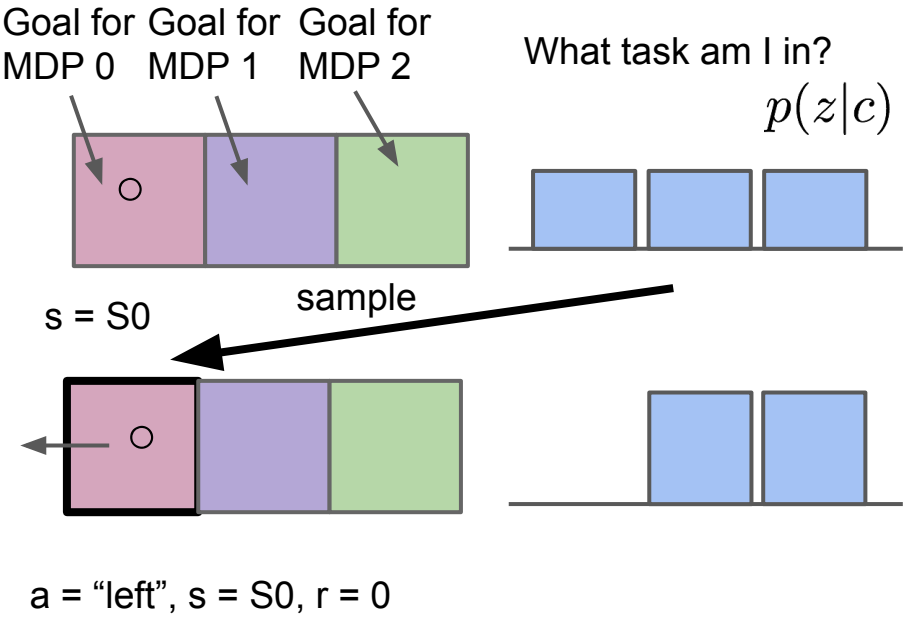
Model belief over latent task variables

POMDP for unobserved state



a = “left”, s = S0, r = 0

POMDP for unobserved task



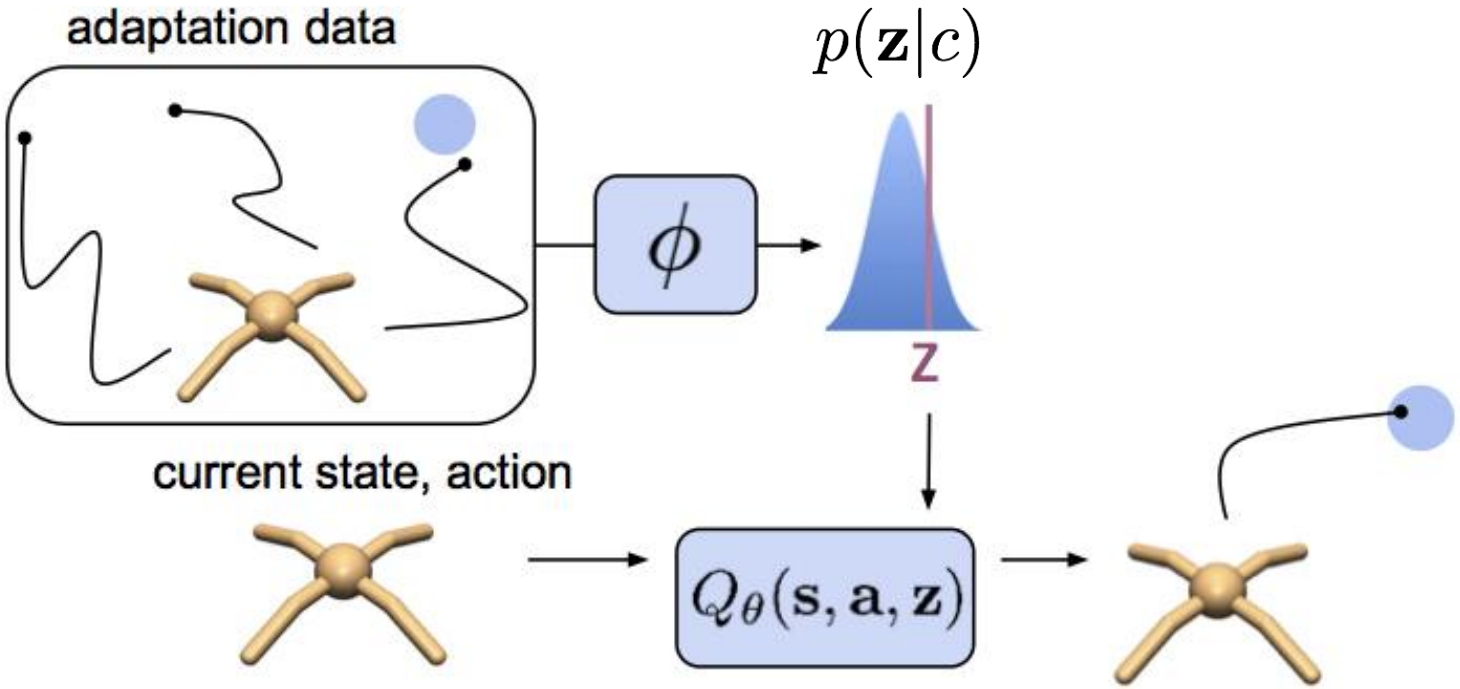
a = “left”, s = S0, r = 0

Solution #3: task-belief states

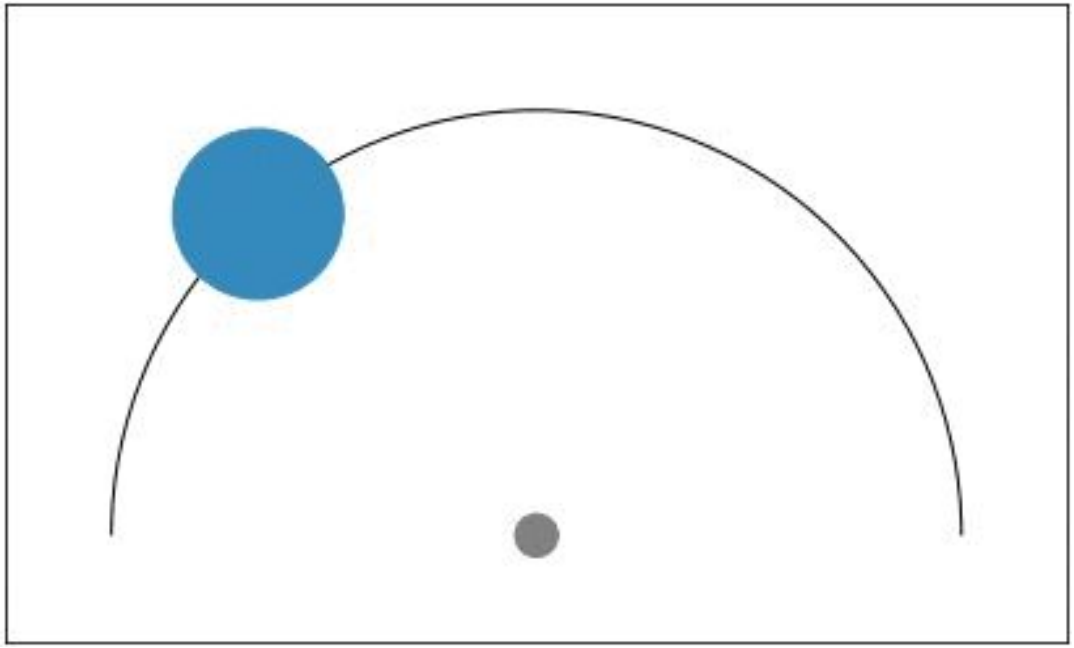
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

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

Stochastic encoder



Solution #3: posterior sampling in action

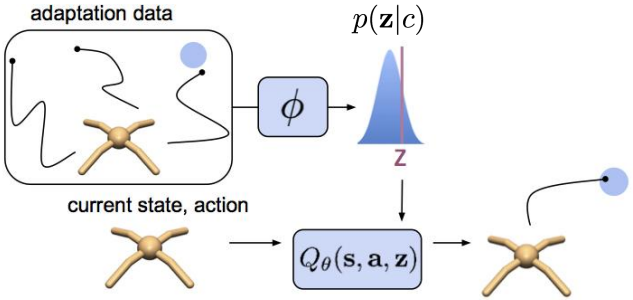


Solution #3: belief training objective

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

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

Stochastic encoder



"Likelihood" term (Bellman error)

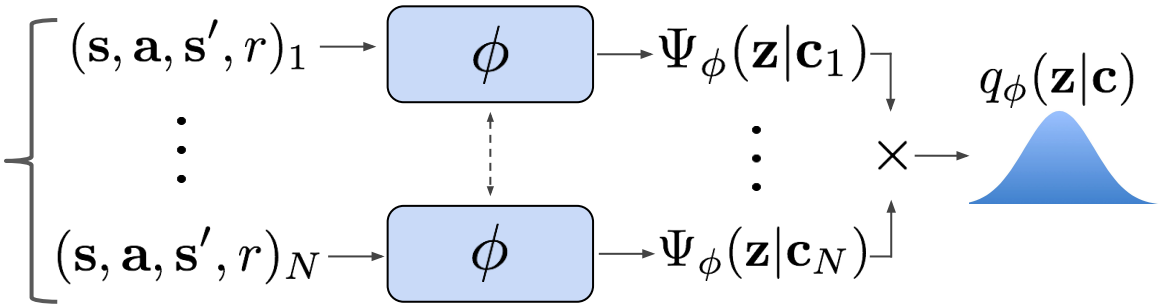
Variational approximations to posterior and prior

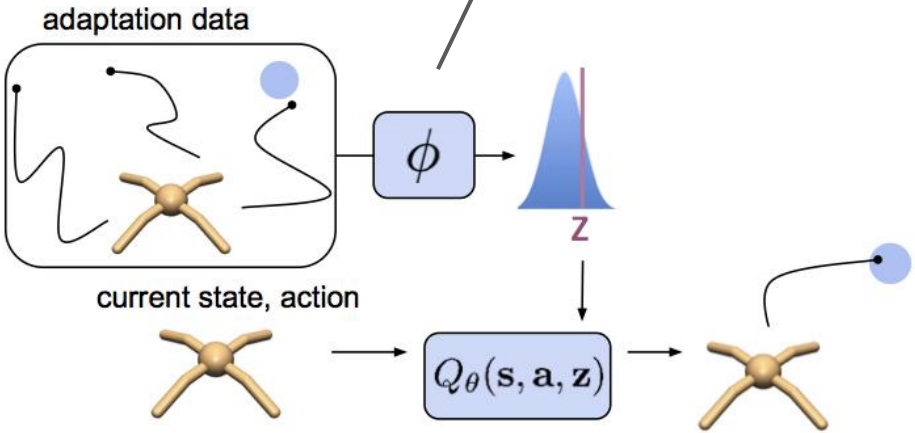
$$\mathbb{E}_{\mathcal{T}}[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{c}^\mathcal{T})}[R(\mathcal{T}, \mathbf{z}) + \beta D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{c}^\mathcal{T})||p(\mathbf{z}))]]$$

"Regularization" term / information bottleneck

See Control as Inference (Levine 2018) for justification of thinking of Q as a pseudo-likelihood

Solution #3: encoder design





Don't need to know the order of transitions in order to identify the MDP (Markov property)

Use a **permutation-invariant encoder** for simplicity and speed

Aside: Soft Actor-Critic (SAC)

"Soft": Maximize rewards *and* entropy of the policy (higher entropy policies explore better)

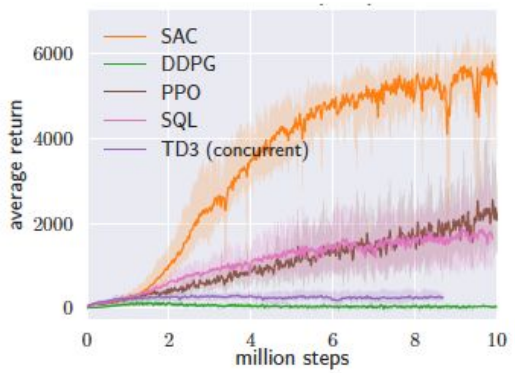
$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \rho_\pi} \left[r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi(\cdot|\mathbf{s}_t)) \right]$$

"Actor-Critic": Model *both* the actor (aka the policy) and the critic (aka the Q-function)

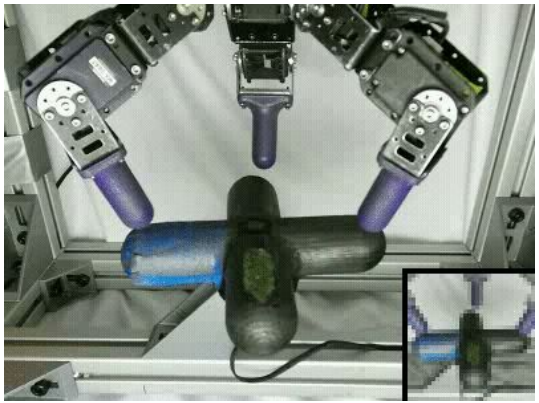
$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \mathcal{D}} \left[\frac{1}{2} \left(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \hat{Q}(\mathbf{s}_t, \mathbf{a}_t) \right)^2 \right]$$

$$J_\pi(\phi) = \mathbb{E}_{s_t, a_t} [Q_\theta(s_t, a_t) + \alpha \mathcal{H}(\pi_\phi(\cdot|s_t))]$$

Much more **sample efficient** than on-policy algs.



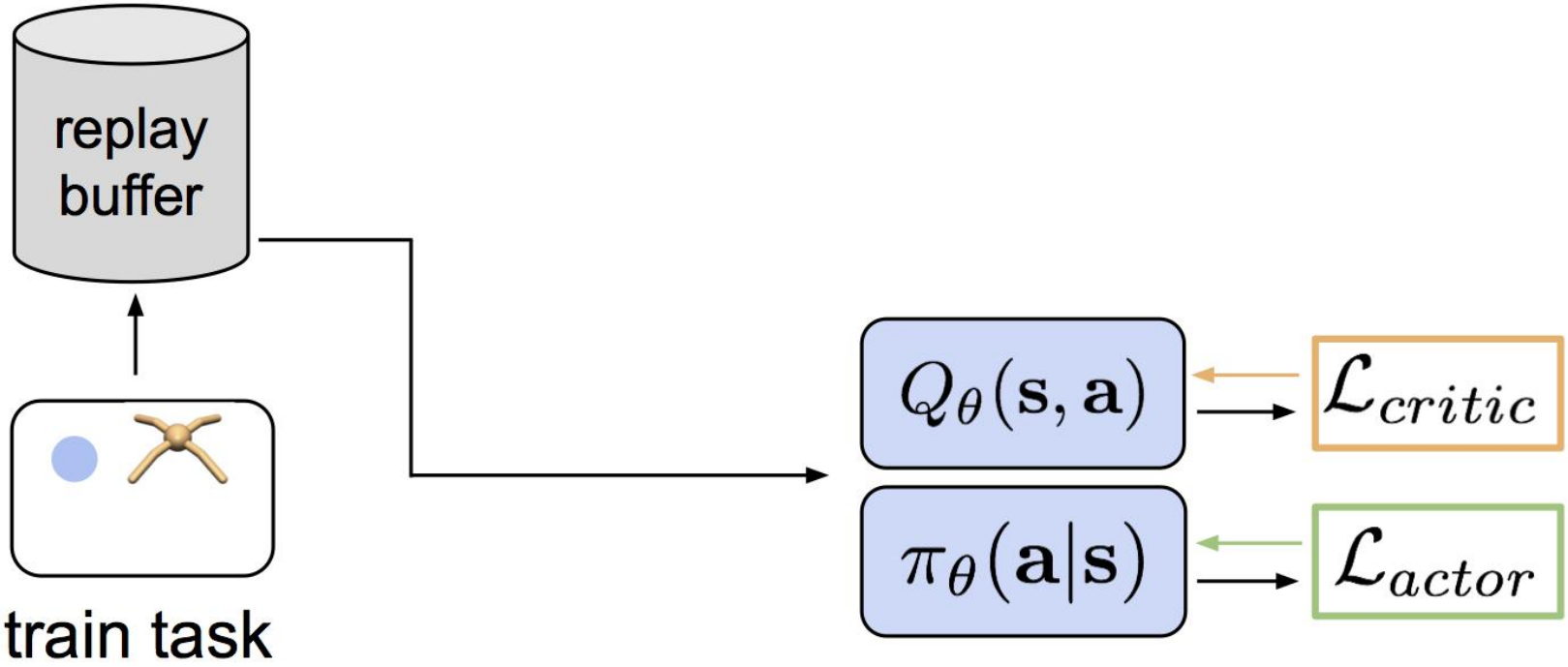
(f) Humanoid (rllab)



Dclaw robot turns valve from pixels

SAC Haarnoja et al. 2018, Control as Inference Tutorial. Levine 2018, SAC BAIR Blog Post 2019

Soft Actor-Critic



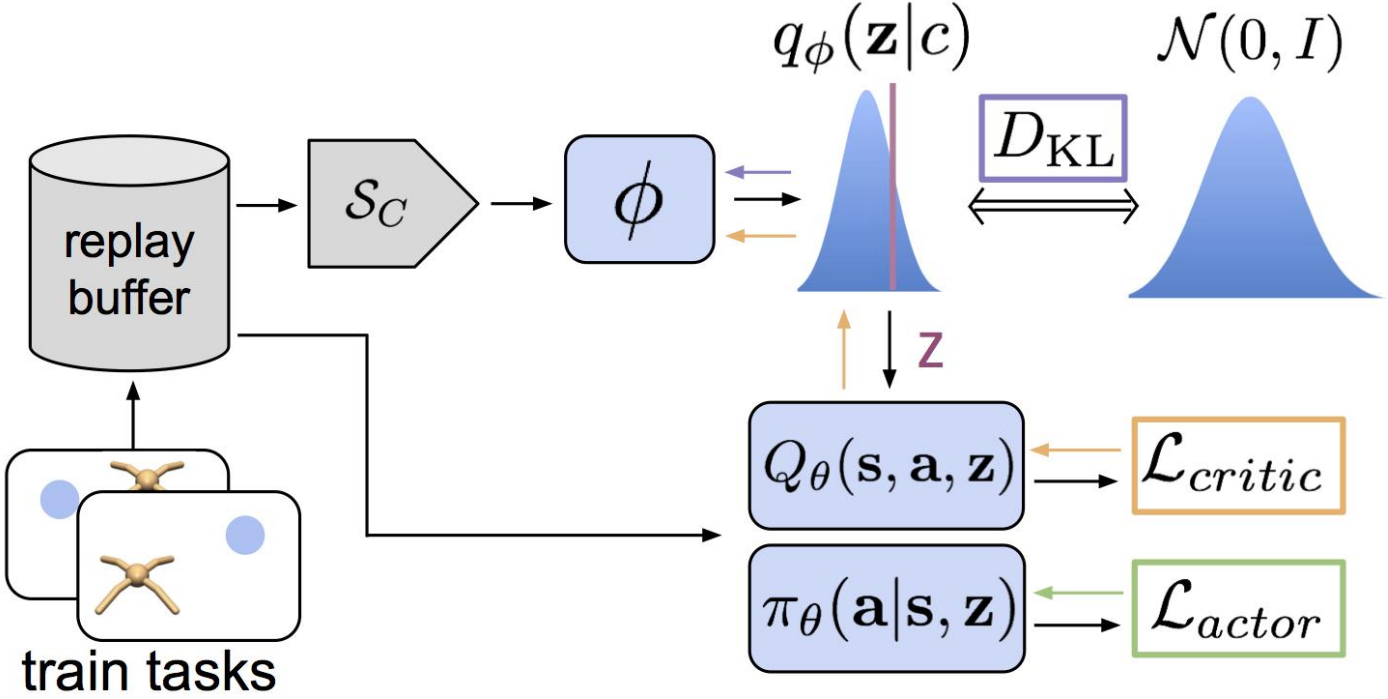
Solution #3: task-belief + SAC

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

SAC

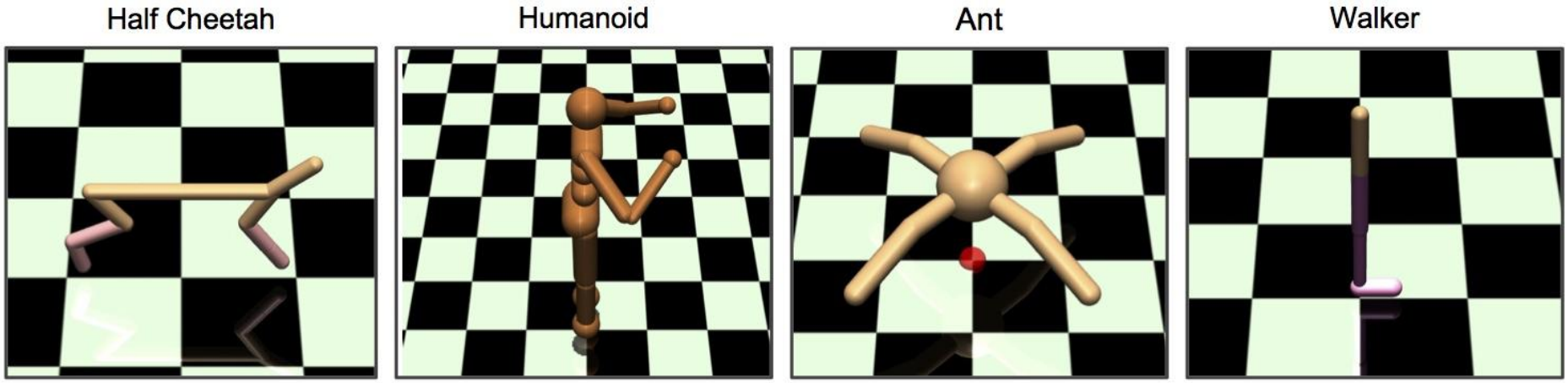
$$\text{where } \phi_i = f_\theta(\mathcal{M}_i)$$

Stochastic encoder



Rakelly & Zhou et al. 2019

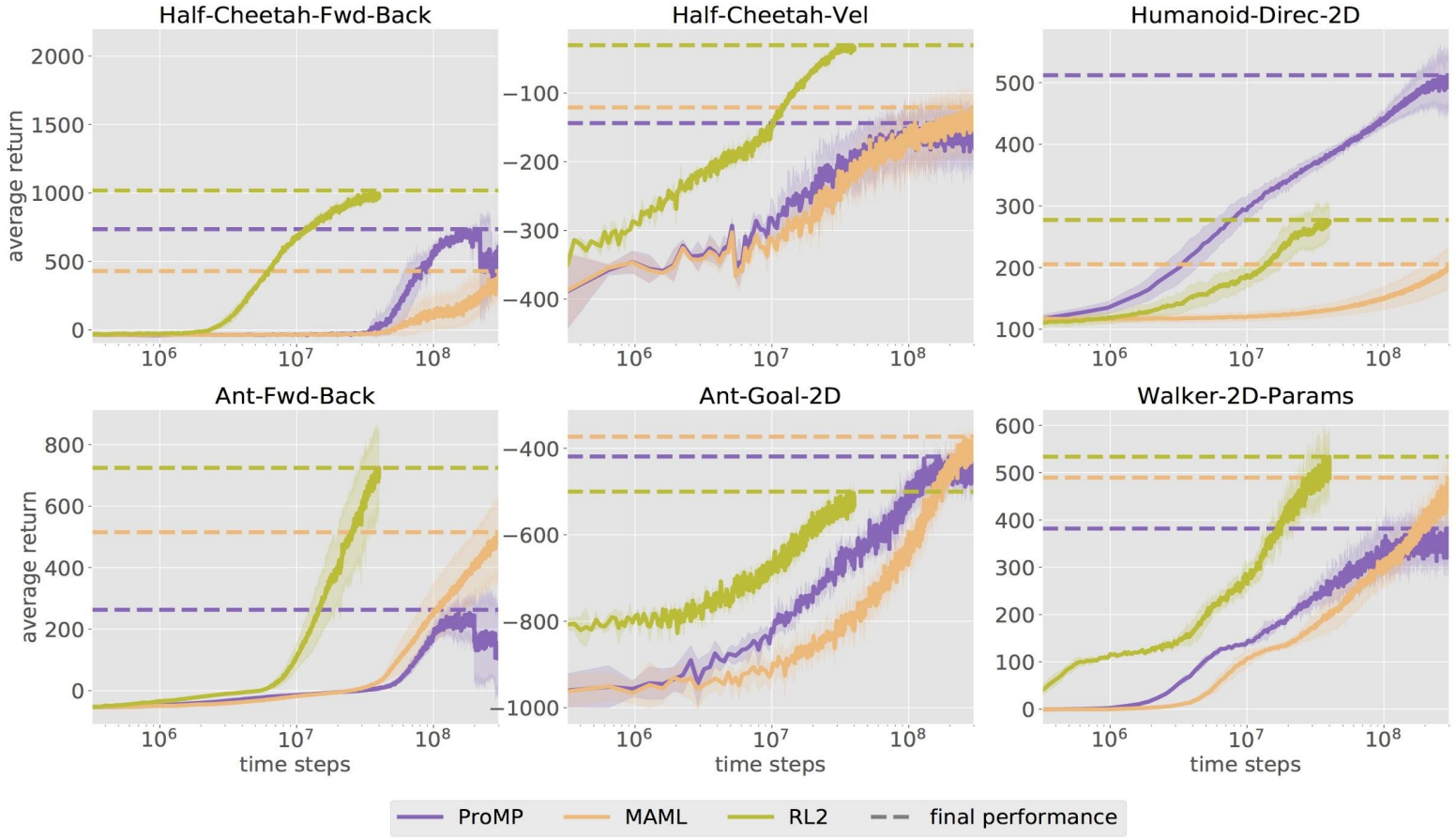
Meta-RL experimental domains



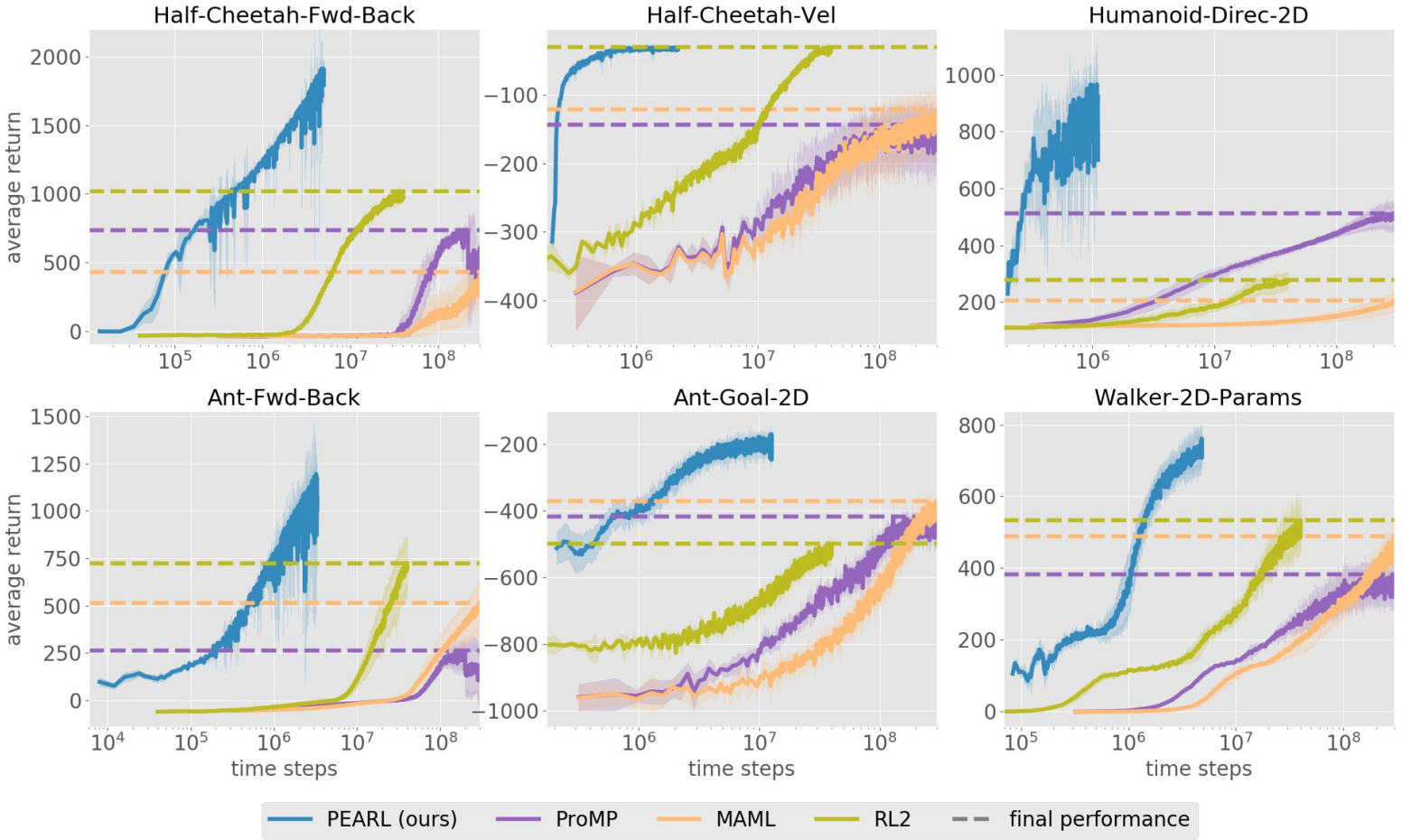
variable reward function
(locomotion direction, velocity, or goal)

variable dynamics
(joint parameters)

Simulated via MuJoCo (Todorov et al. 2012), tasks proposed by (Finn et al. 2017, Rothfuss et al. 2019)



ProMP (Rothfuss et al. 2019), MAML (Finn et al. 2017), RL2 (Duan et al. 2016)



20-100X more sample efficient!

ProMP (Rothfuss et al. 2019), MAML (Finn et al. 2017), RL2 (Duan et al. 2016)

two views of meta-RL

Mechanistic view

- Deep neural network model that can read in an entire dataset and make predictions for new datapoints
- Training this network uses a meta-dataset, which itself consists of many datasets, each for a different task

Probabilistic view

- Extract prior information from a set of (meta-training) tasks that allows efficient learning of new tasks
- Learning a new task uses this prior and (small) training set to infer most likely posterior parameters

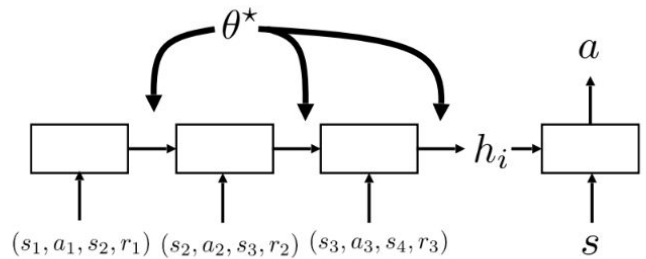
Slide adapted from Sergey Levine and Chelsea Finn

Summary

just perspective 1, but with stochastic hidden variables!

i.e., $\phi = \mathbf{z}$

Perspective 1: just RNN it



Perspective 2: bi-level optimization

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

MAML for RL

Perspective 3: it's an inference problem!

$\pi_\theta(a|s, z)$ $\qquad z_t \sim p(z_t|s_{1:t}, a_{1:t}, r_{1:t})$

everything needed to solve task

just a particular architecture choice for these

Slide adapted from Sergey Levine and Chelsea Finn

Frontiers

Where do tasks come from?

Idea: generate self-supervised tasks and use them during meta-training

$$\max \; \mathcal{H}[Z] - \mathcal{H}[Z \mid S] + \mathcal{H}[A \mid S, Z]$$

Separate skills visit different states

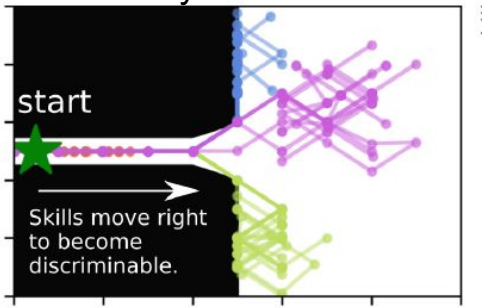
Skills should be high entropy

Limitations

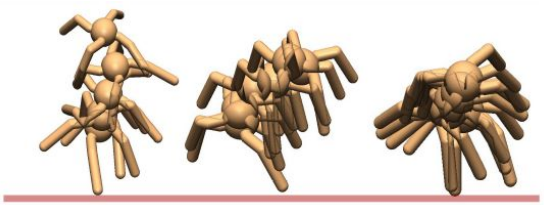
Assumption that skills shouldn't depend on action not always valid

Distribution shift meta-train -> meta-test

Point robot learns to explore different areas after the hallway



Ant learns to run in different directions, jump, and flip

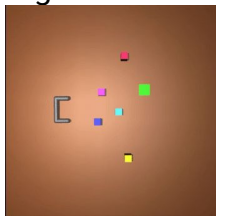


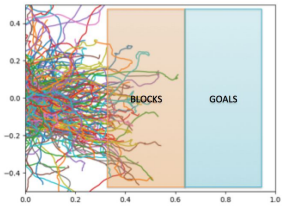
Eysenbach et al. 2018, Gupta et al. 2018

How to explore efficiently in a new task?

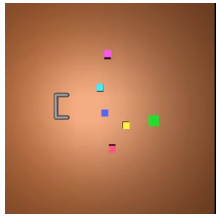
Learn exploration strategies better...

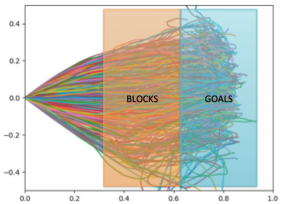
Plain gradient meta-RL

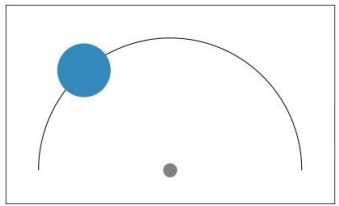




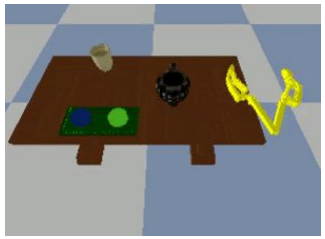
Latent-variable model



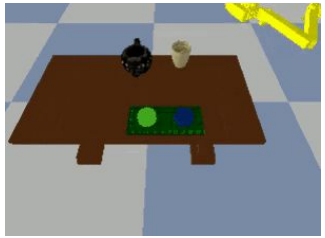




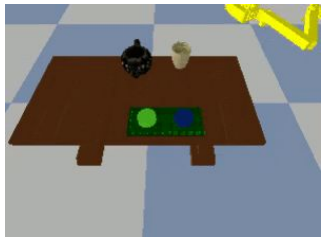
Bias exploration with extra information…



human -provided demo



Robot attempt #1, w/ only demo info

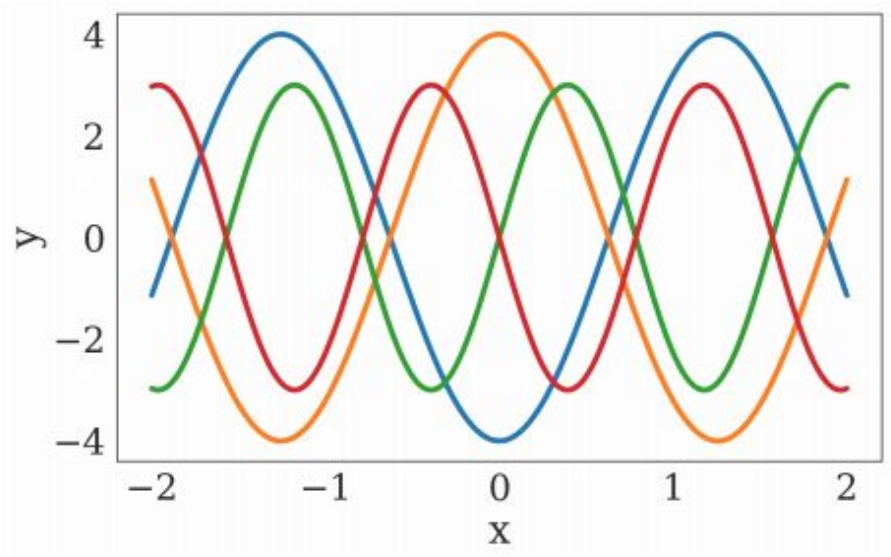


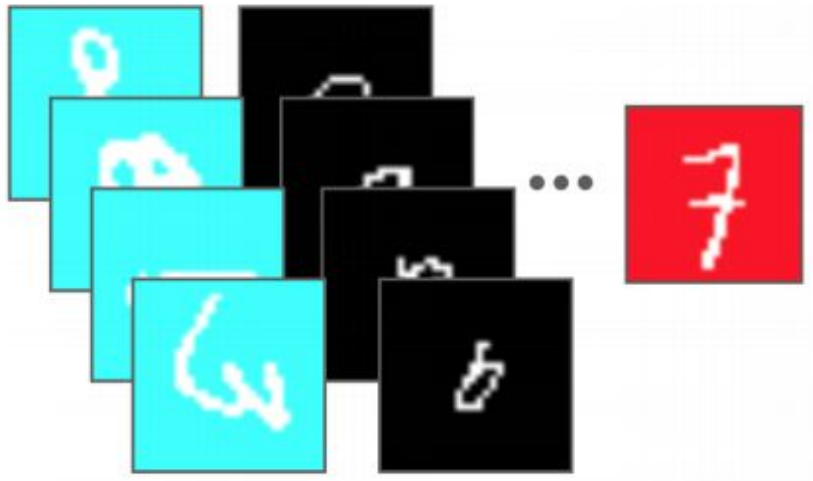
Robot attempt #2, w/ demo + reward info

Gupta et al. 2018, Rakelly et al. 2019, Zhou et al. 2019

Online meta-learning

Meta-training tasks are presented in a sequence rather than a batch





Finn et al. 2019

Summary

Meta-RL finds an adaptation procedure that can quickly adapt the policy to a new task

Three main solution classes: RNN, optimization, task-belief and several learning paradigms: model-free (on and off policy), model-based, imitation learning

Connection to goal-conditioned RL and POMDPs

Some open problems (there are more!): better exploration, defining task distributions, meta-learning online

References

Recurrent meta-RL
Learning to Reinforcement Learn, Wang et al. 2016
Fast Reinforcement Learning by Slow Reinforcement Learning, Duan et al. 2016
Memory-Based Control with Recurrent Neural Networks, Heess et al. 2015

Optimization-based meta-RL
Model-Agnostic Meta-Learning, Finn et al. 2017
Proximal Meta-Policy Search, Rothfuss et al. 2018

Optimization-based meta-RL + imitation learning
One-Shot Visual Imitation Learning via Meta-Learning, Yu et al. 2017
One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning, Yu et al. 2018

Model-based meta-RL
Learning to Adapt in Dynamic, Real-World Environments through Meta-Reinforcement Learning, Nagabandi et al. 2019

Off-policy meta-RL
Soft Actor-Critic, Haarnoja et al. 2018
Control as Inference, Levine 2018.
Efficient Off-Policy Meta-RL via Probabilistic Context Variables, Rakelly et al. 2019

References

Open Problems
Diversity is All You Need: Learning Skills without a Reward Function, Eysenbach et al. 2018
Unsupervised Meta-learning for RL, Gupta et al. 2018
Meta-Reinforcement Learning of Structured Exploration Strategies, Gupta et al. 2018
Watch, Try, Learn, Meta-Learning from Demonstrations and Reward, Zhou et al. 2019
Online Meta-Learning, Finn et al. 2019

Slides and Figures
Some slides adapted from Meta-Learning Tutorial at ICML 2019, Finn and Levine
Robot illustrations by Matt Spangler, mattspangler.com