Meta-Learning

CS 285

Instructor: Sergey Levine
UC Berkeley

So far…

- Forward transfer: source domain to target domain
  - Diversity is good! The more varied the training, the more likely transfer is to succeed
- Multi-task learning: even more variety
  - No longer training on the same kind of task
  - But more variety = more likely to succeed at transfer
- How do we represent transfer knowledge?
  - Model (as in model-based RL): rules of physics are conserved across tasks
  - Policies – requires finetuning, but closer to what we want to accomplish
  - What about *learning methods*?

What is meta-learning?

- If you've learned 100 tasks already, can you figure out how to *learn* more efficiently?
  - Now having multiple tasks is a huge advantage!
- Meta-learning = *learning to learn*
- In practice, very closely related to multi-task learning
- Many formulations
  - Learning an optimizer
  - Learning an RNN that ingests experience
  - Learning a representation

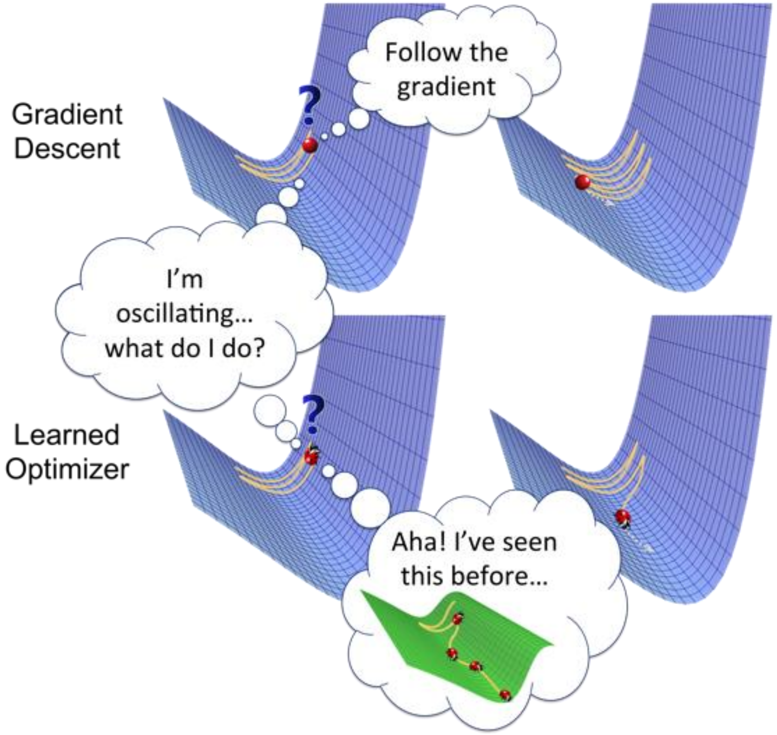

Gradient Descent
Follow the gradient
I'm oscillating... what do I do?
Learned Optimizer
Aha! I've seen this before...

image credit: Ke Li

Why is meta-learning a good idea?

- Deep reinforcement learning, especially model-free, requires a huge number of samples
- If we can *meta-learn* a faster reinforcement learner, we can learn new tasks efficiently!
- What can a *meta-learned* learner do differently?
  - Explore more intelligently
  - Avoid trying actions that are know to be useless
  - Acquire the right features more quickly

Meta-learning with supervised learning

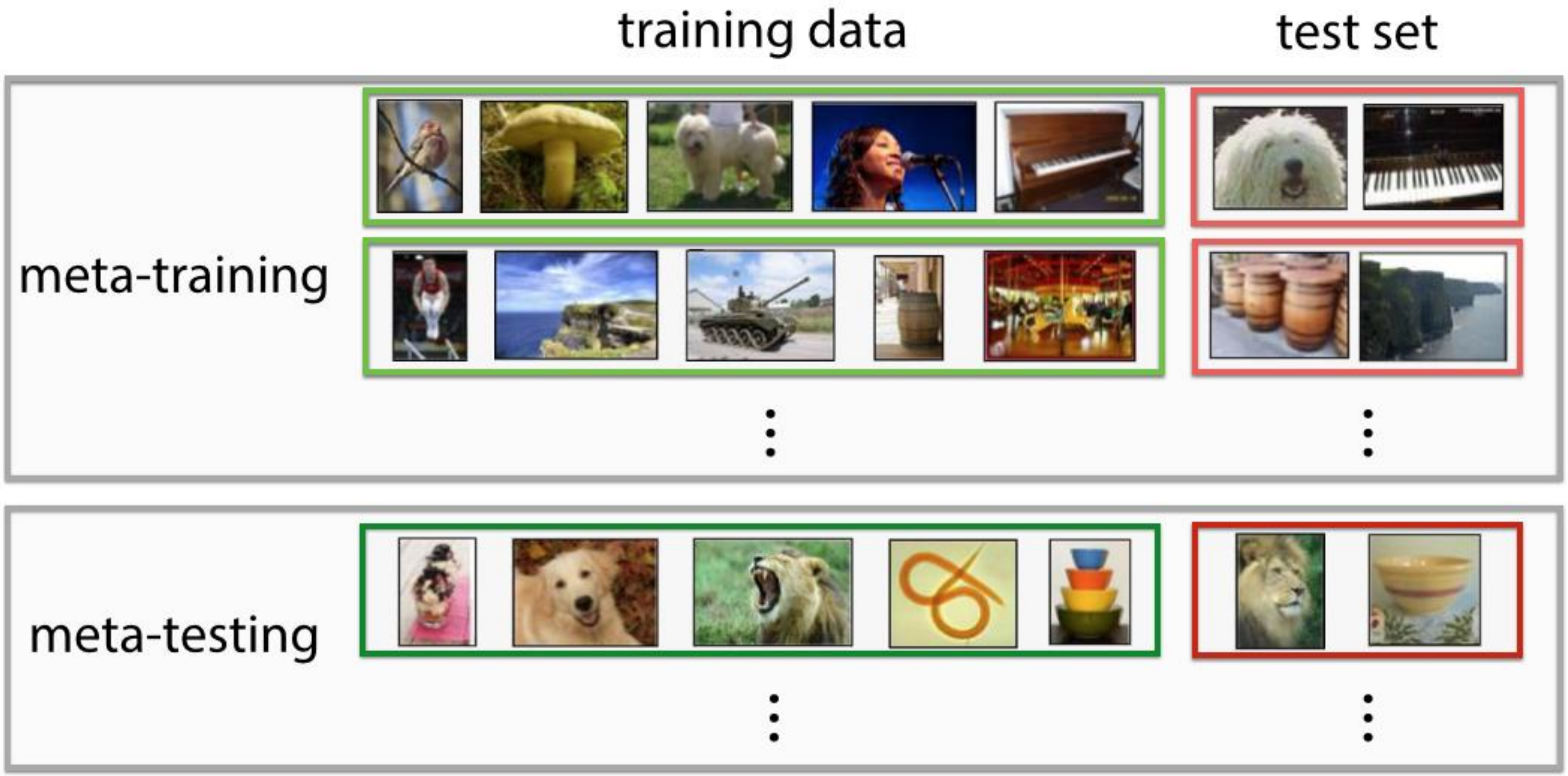
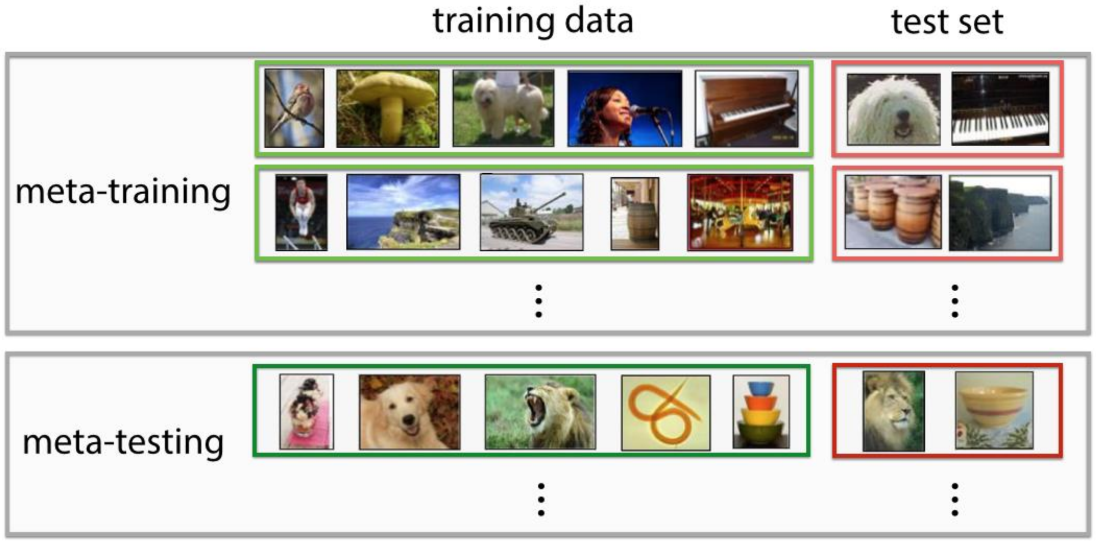
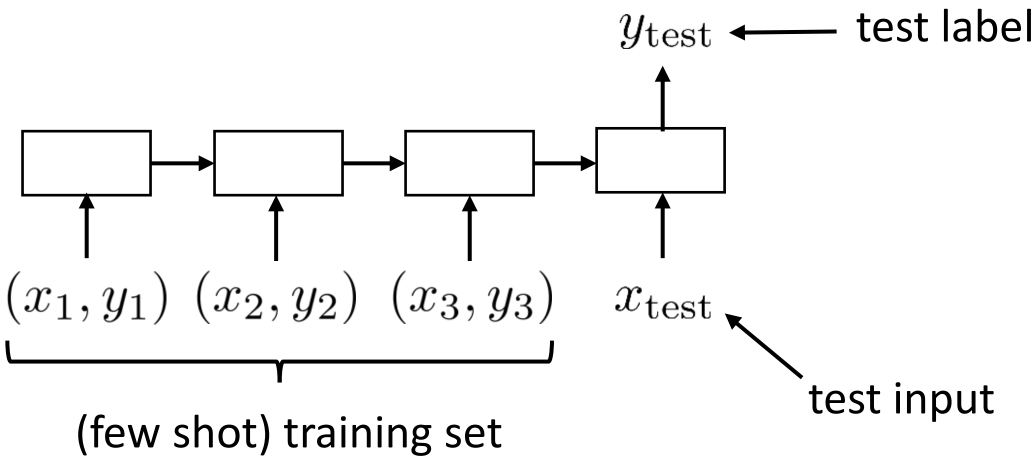


image credit: Ravi & Larochelle ‘17

Meta-learning with supervised learning





supervised learning: $f(x) \rightarrow y$

input (e.g., image)

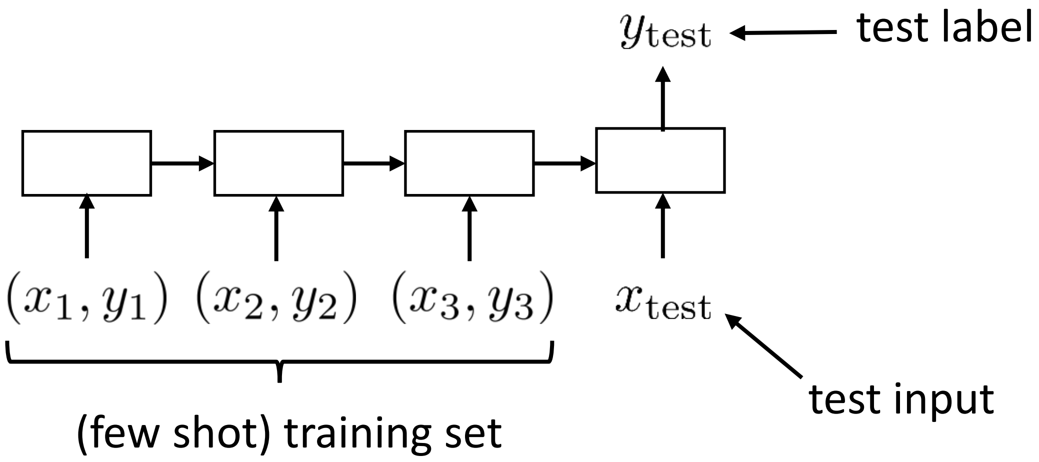
output (e.g., label)

supervised meta-learning: $f(\mathcal{D}^{\text{tr}}, x) \rightarrow y$

training set

- How to read in training set?
  - Many options, RNNs can work
  - More on this later

What is being “learned”?



supervised meta-learning: $f(\mathcal{D}^{\text{tr}}, x) \to y$

“Generic” learning:

$$\theta^\star = \arg\min_\theta \mathcal{L}(\theta, \mathcal{D}^{\text{tr}})$$

$$= f_{\text{learn}}(\mathcal{D}^{\text{tr}})$$

“Generic” meta-learning:

$$\theta^\star = \arg\min_\theta \sum_{i=1}^{n} \mathcal{L}(\phi_i, \mathcal{D}_i^{\text{ts}})$$

$$\text{where } \phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$$

What is being “learned”?

“Generic” learning:

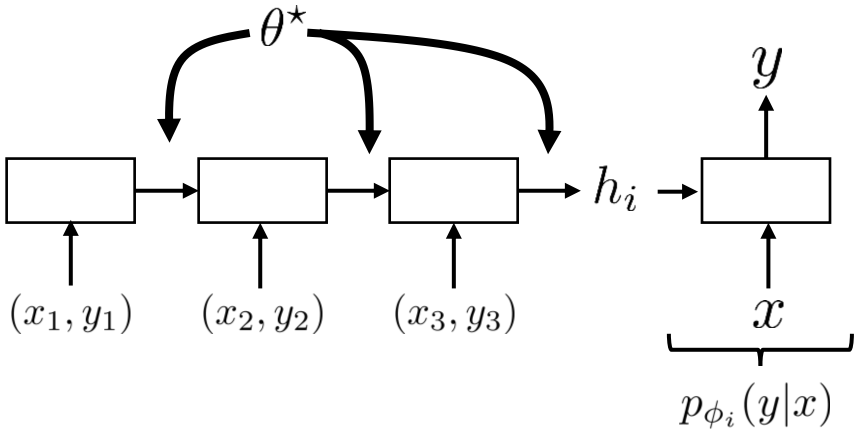
$$\theta^\star = \arg\min_\theta \mathcal{L}(\theta, \mathcal{D}^{\text{tr}})$$

$$= f_{\text{learn}}(\mathcal{D}^{\text{tr}})$$

“Generic” meta-learning:

$$\theta^\star = \arg\min_\theta \sum_{i=1}^{n} \mathcal{L}(\phi_i, \mathcal{D}_i^{\text{ts}})$$

$$\text{where } \phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$$



RNN hidden state

meta-learned weights

$$\phi_i = [h_i, \theta_p]$$

Meta **Reinforcement** Learning

The meta reinforcement learning problem

“Generic” learning:

$$\theta^\star = \arg\min_\theta \mathcal{L}(\theta, \mathcal{D}^{\text{tr}})$$

$$= f_{\text{learn}}(\mathcal{D}^{\text{tr}})$$

Reinforcement learning:

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

$$= f_{\text{RL}}(\underset{\substack{\uparrow \\ \text{MDP}}}{\mathcal{M}}) \qquad \mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$$

“Generic” meta-learning:

$$\theta^\star = \arg\min_\theta \sum_{i=1}^{n} \mathcal{L}(\phi_i, \mathcal{D}_i^{\text{ts}})$$

$$\text{where } \phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$$

Meta-reinforcement learning:

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

$$\text{where } \phi_i = f_\theta(\underset{\substack{\uparrow \\ \text{MDP for task } i}}{\mathcal{M}_i})$$

The meta reinforcement learning problem

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

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

assumption: $\mathcal{M}_i \sim p(\mathcal{M})$

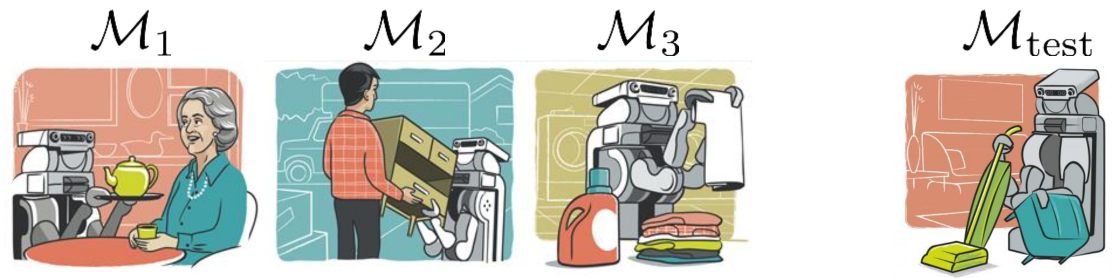
meta test-time:

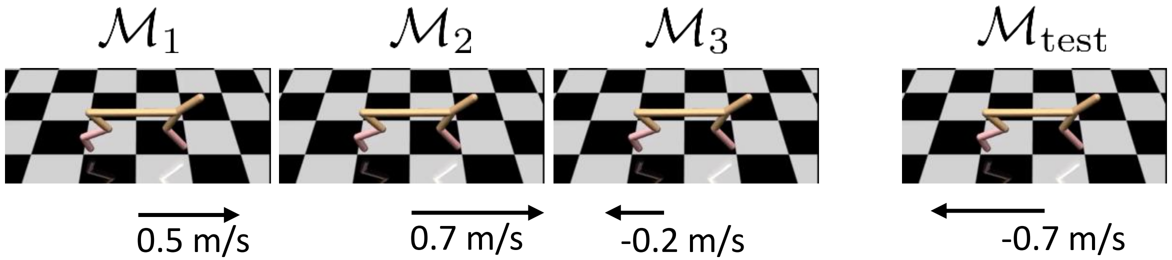
sample $\mathcal{M}_{\text{test}} \sim p(\mathcal{M})$, get $\phi_i = f_\theta(\mathcal{M}_{\text{test}})$

$\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$

meta-training MDPs

Some examples:





Contextual policies and meta-learning

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

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

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

$$\pi_\theta(a_t|s_t, \underbrace{s_1, a_1, r_1, \dots, s_{t-1}, a_{t-1}, r_{t-1}}_{})$$

context used to infer whatever we need to solve $\mathcal{M}_i$
i.e., z_t or ϕ_i (which are really the same thing)

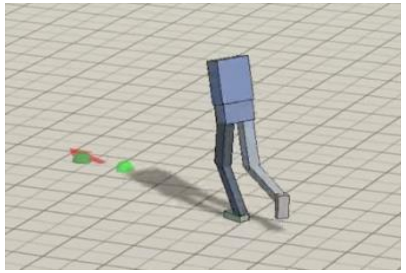
in meta-RL, the *context* is inferred from experience from $\mathcal{M}_i$

in multi-task RL, the context is typically given

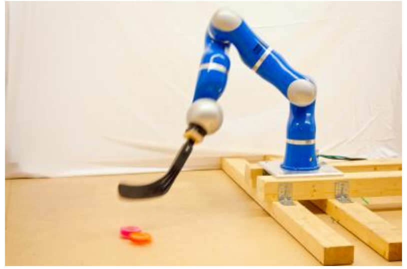
$$\pi_\theta(a_t|s_t, \phi_i)$$

“context”

ϕ: stack location



ϕ: walking direction

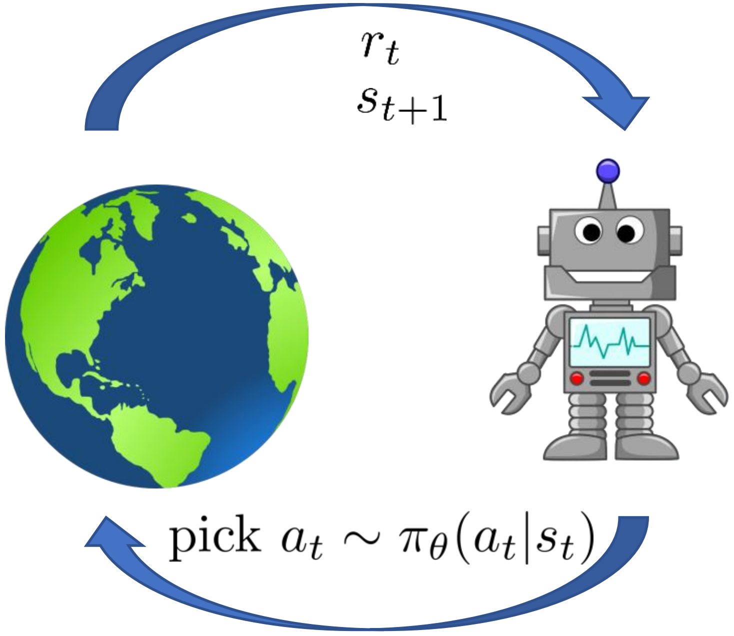


ϕ: where to hit puck

Meta-RL with recurrent policies

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

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

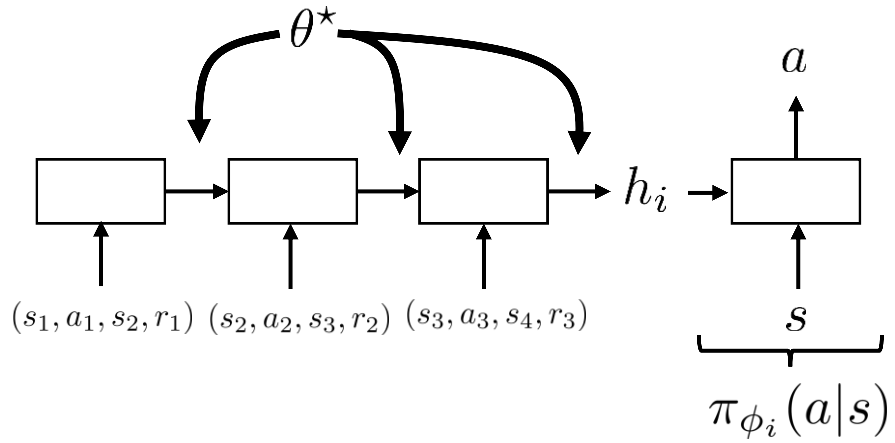


use (s_t, a_t, s_{t+1}, r_t) to improve π_θ

main question: how to implement $f_\theta(\mathcal{M}_i)$?

what should $f_\theta(\mathcal{M}_i)$ *do*?

1. improve policy with experience from $\mathcal{M}_i$
   $\{(s_1, a_1, s_2, r_1), \ldots, (s_T, a_T, s_{T+1}, r_T)\}$
2. (new in RL): choose how to interact, i.e. choose a_t
   meta-RL must also *choose* how to *explore*!



RNN hidden state

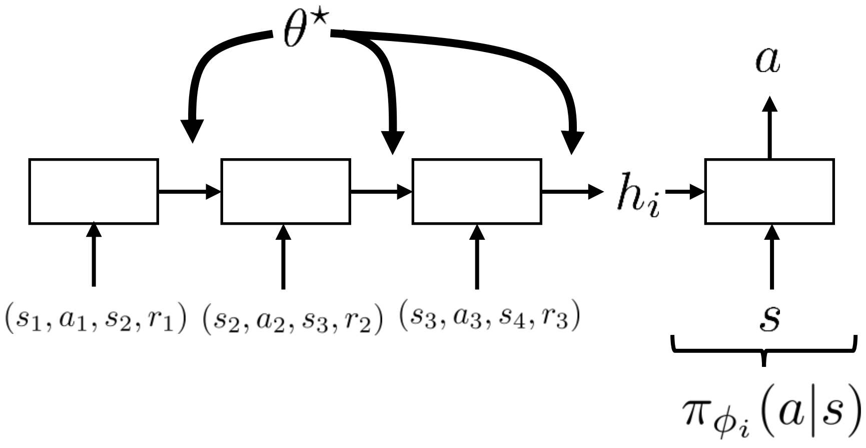
meta-learned weights

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

Meta-RL with recurrent policies

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

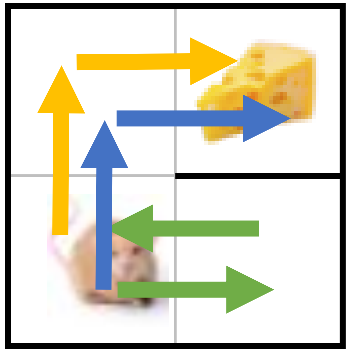
$$\text{where } \phi_i = f_\theta(\mathcal{M}_i)$$

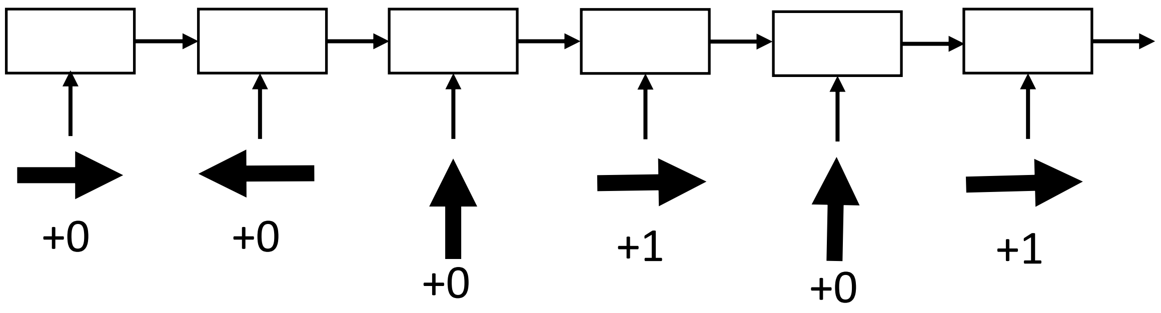


so... we just train an RNN policy?

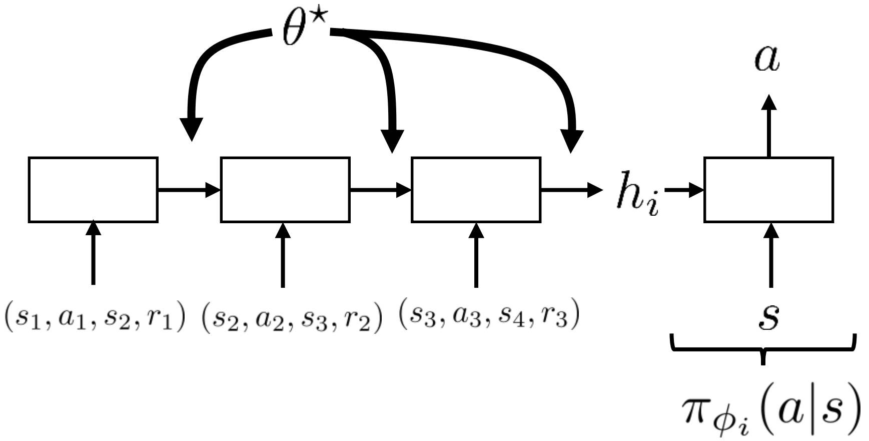
yes!

crucially, RNN hidden state is **not** reset between episodes!





Why recurrent policies *learn to explore*



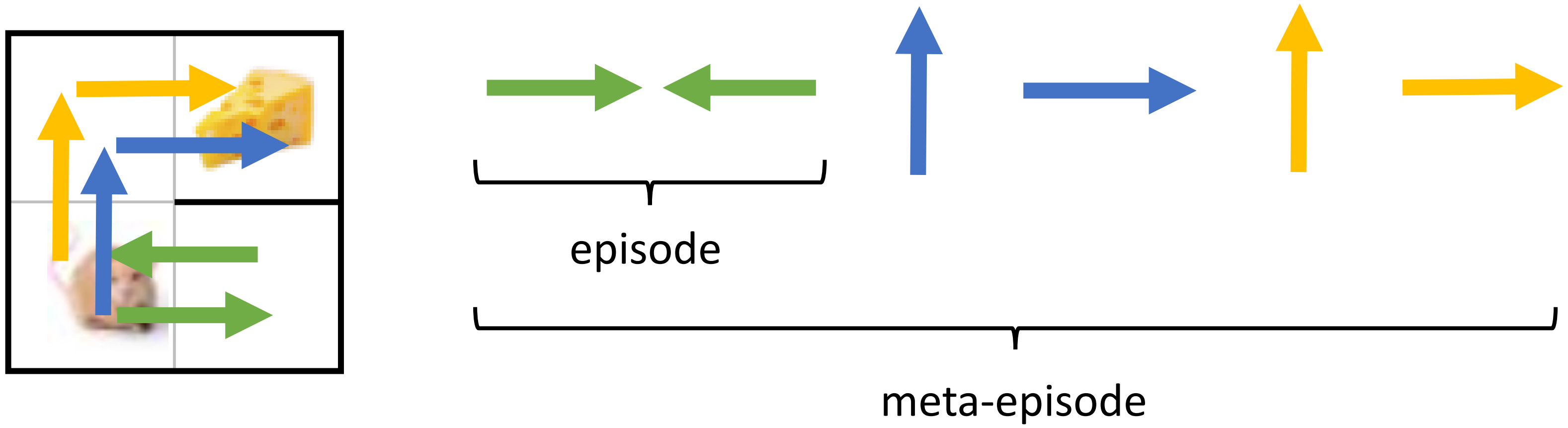
1. improve policy with experience from $\mathcal{M}_i$

   $\{(s_1, a_1, s_2, r_1), \ldots, (s_T, a_T, s_{T+1}, r_T)\}$

2. (new in RL): choose how to interact, i.e. choose a_t

   meta-RL must also *choose* how to *explore*!

$$\theta^\star = \arg\max_\theta E_{\pi_\theta}\left[\sum_{t=0}^{T} r(s_t, a_t)\right]$$

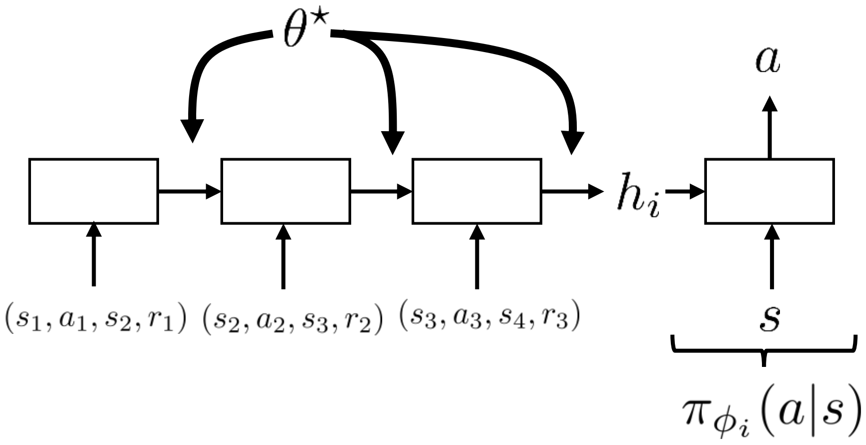


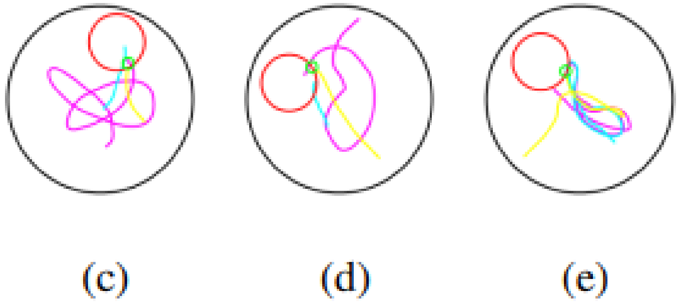
optimizing total reward over the entire **meta**-episode with RNN policy **automatically** learns to explore!

Meta-RL with recurrent policies

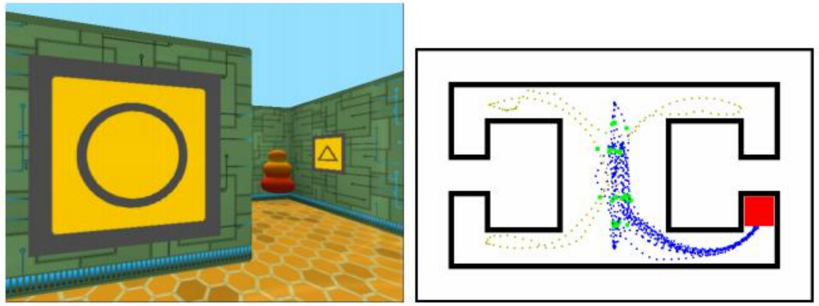
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

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



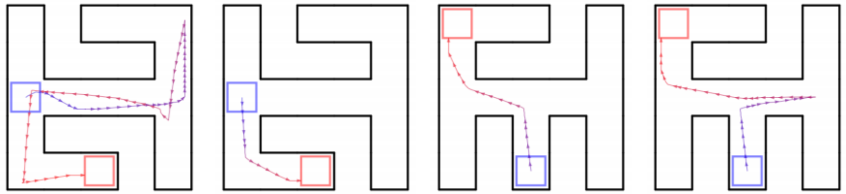

(c) (d) (e)

Heess, Hunt, Lillicrap, Silver. **Memory-based control with recurrent neural networks.** 2015.


(a) Labryinth I-maze

(b) Illustrative Episode

Wang, Kurth-Nelson, Tirumala, Soyer, Leibo, Munos, Blundell, Kumaran, Botvinick. **Learning to Reinforcement Learning.** 2016.


(a) Good behavior, 1st episode

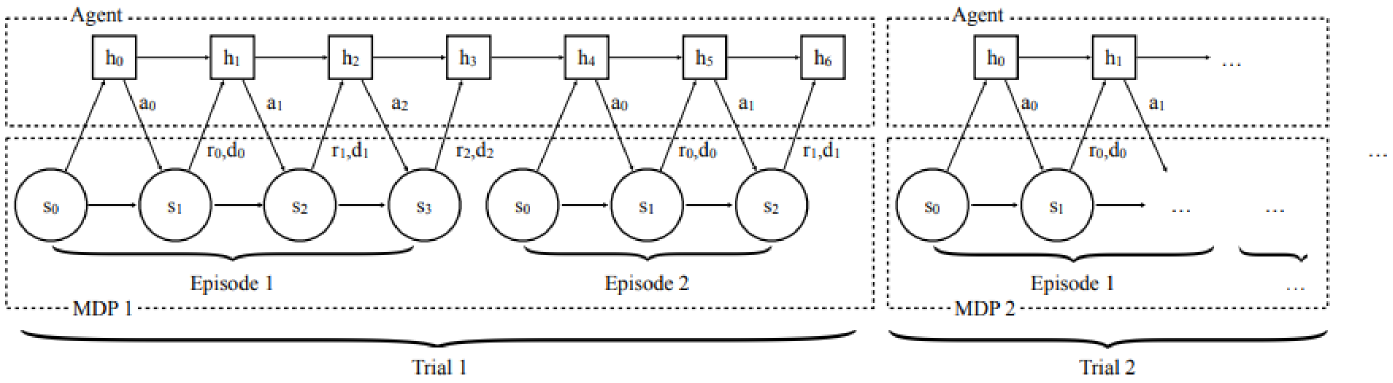
(b) Good behavior, 2nd episode

(c) Bad behavior, 1st episode

(d) Bad behavior, 2nd episode

Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel. **RL2: Fast Reinforcement Learning via Slow Reinforcement Learning.** 2016.

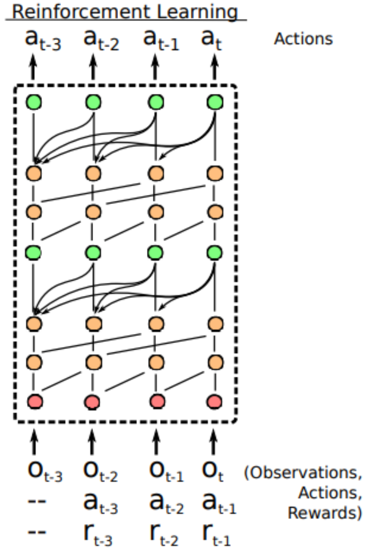
Architectures for meta-RL



Agent
h_0 h_1 h_2 h_3 h_4 h_5 h_6
a_0 a_1 a_2 a_0 a_1
r_0,d_0 r_1,d_1 r_2,d_2 r_0,d_0 r_1,d_1
s_0 s_1 s_2 s_3 s_0 s_1 s_2
Episode 1
Episode 2
MDP 1
Trial 1
Agent
h_0 h_1 …
a_0 a_1
r_0,d_0
s_0 s_1 … …
Episode 1
…
MDP 2
Trial 2
…

standard RNN (LSTM) architecture

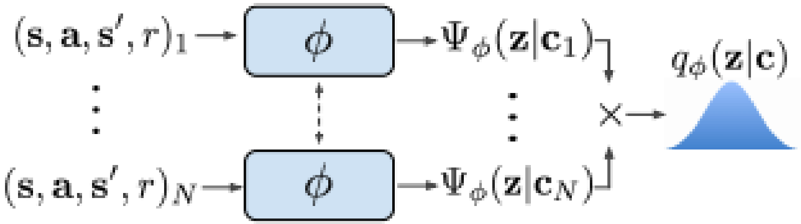
Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel. **RL2: Fast Reinforcement Learning via Slow Reinforcement Learning.** 2016.



Reinforcement Learning
a_{t-3} a_{t-2} a_{t-1} a_t
Actions
o_{t-3} o_{t-2} o_{t-1} o_t
-- a_{t-3} a_{t-2} a_{t-1}
-- r_{t-3} r_{t-2} r_{t-1}
(Observations, Actions, Rewards)

attention + temporal convolution

Mishra, Rohaninejad, Chen, Abbeel. **A Simple Neural Attentive Meta-Learner.**



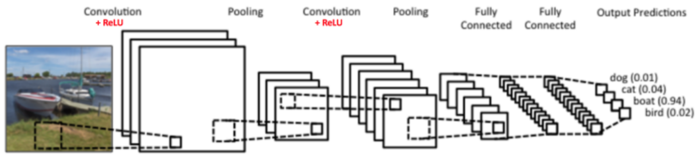
$(\mathbf{s},\mathbf{a},\mathbf{s}',r)_1$
$(\mathbf{s},\mathbf{a},\mathbf{s}',r)_N$
ϕ
ϕ
$\Psi_\phi(\mathbf{z}|\mathbf{c}_1)$
$\Psi_\phi(\mathbf{z}|\mathbf{c}_N)$
×
$q_\phi(\mathbf{z}|\mathbf{c})$

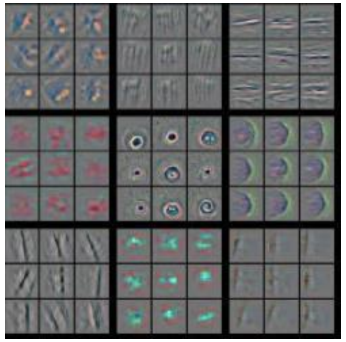
parallel permutation-invariant context encoder

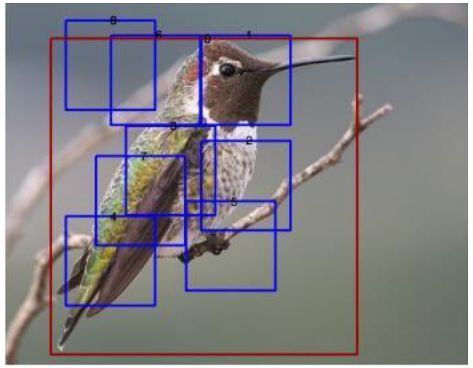
Rakelly*, Zhou*, Quillen, Finn, Levine. **Efficient Off-Policy Meta-Reinforcement learning via Probabilistic Context Variables.**

Gradient-Based Meta-Learning

Back to representations…







is pretraining a *type* of meta-learning?

better features = faster learning of new task!

Meta-RL as an optimization problem

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

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

1. improve policy with experience from $\mathcal{M}_i$

$$\{(s_1, a_1, s_2, r_1), \ldots, (s_T, a_T, s_{T+1}, r_T)\}$$

what if $f_\theta(\mathcal{M}_i)$ is *itself* an RL algorithm?

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

requires interacting with $\mathcal{M}_i$
to estimate $\nabla_\theta E_{\pi_\theta}[R(\tau)]$

standard RL:

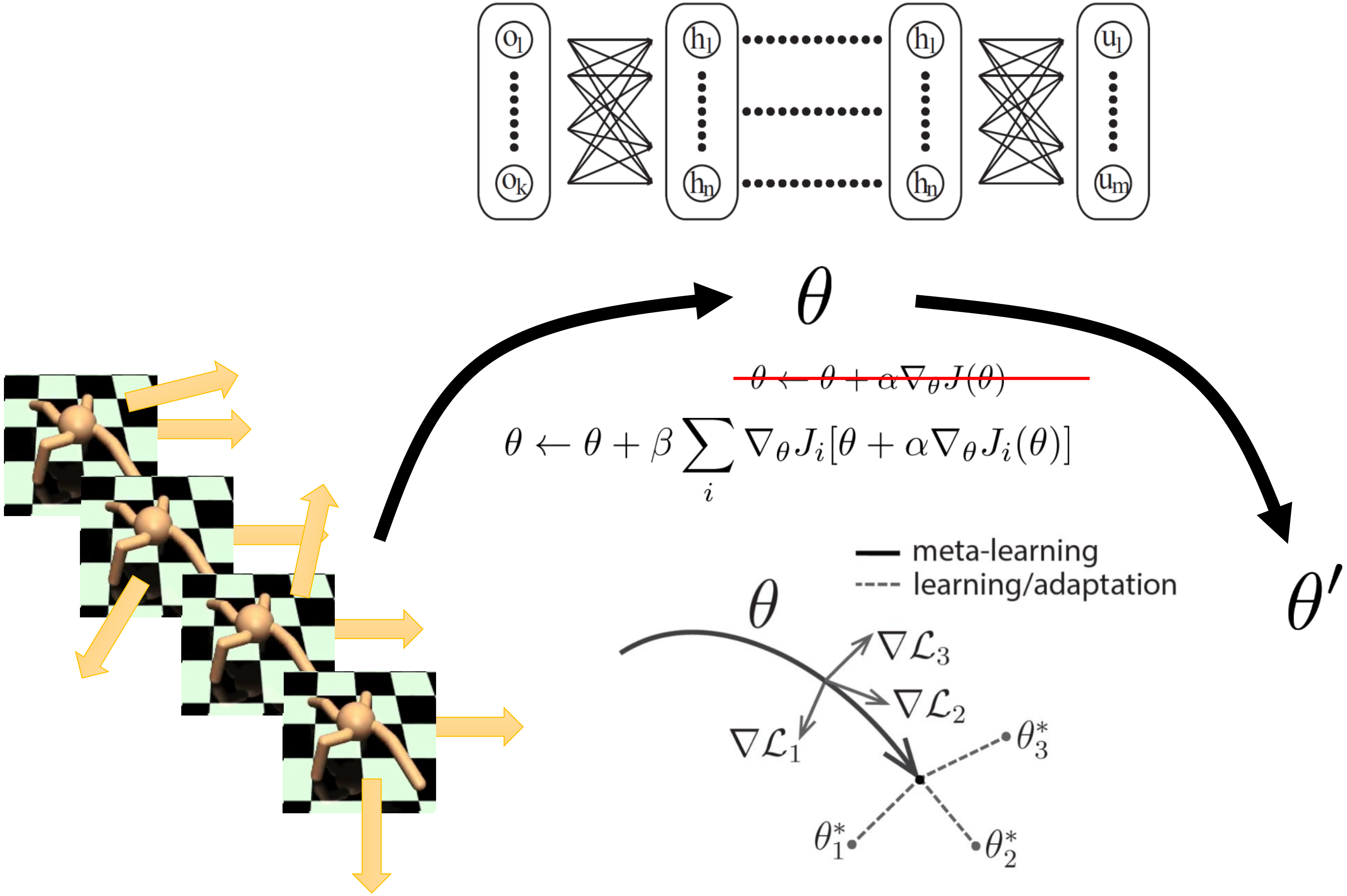
$$\theta^\star = \arg\max_\theta \underbrace{E_{\pi_\theta(\tau)}[R(\tau)]}_{J(\theta)}$$

$$\theta^{k+1} \leftarrow \theta_k + \alpha \nabla_{\theta^k} J(\theta^k)$$

this is model-agnostic meta-learning (MAML) for RL!

Finn, Abbeel, Levine. **Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks.**

MAML for RL in pictures



θ

~~$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$~~

$$\theta \leftarrow \theta + \beta \sum_i \nabla_\theta J_i[\theta + \alpha \nabla_\theta J_i(\theta)]$$

θ'

What did we just do??

supervised learning: $f(x) \to y$

supervised meta-learning: $f(\mathcal{D}^{\text{tr}}, x) \to y$

model-agnostic meta-learning: $f_{\text{MAML}}(\mathcal{D}^{\text{tr}}, x) \to y$

$$f_{\text{MAML}}(\mathcal{D}^{\text{tr}}, x) = f_{\theta'}(x)$$

$$\theta' = \theta - \alpha \sum_{(x,y) \in \mathcal{D}^{\text{tr}}} \nabla_\theta \mathcal{L}(f_\theta(x), y)$$

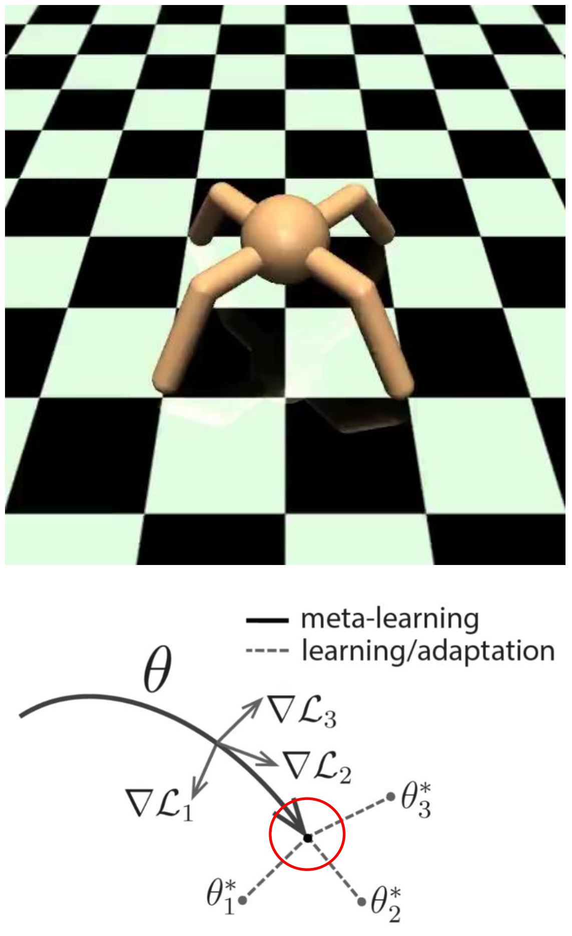
Just another computation graph…

Can implement with any autodiff package (e.g., TensorFlow)

But has favorable inductive bias…

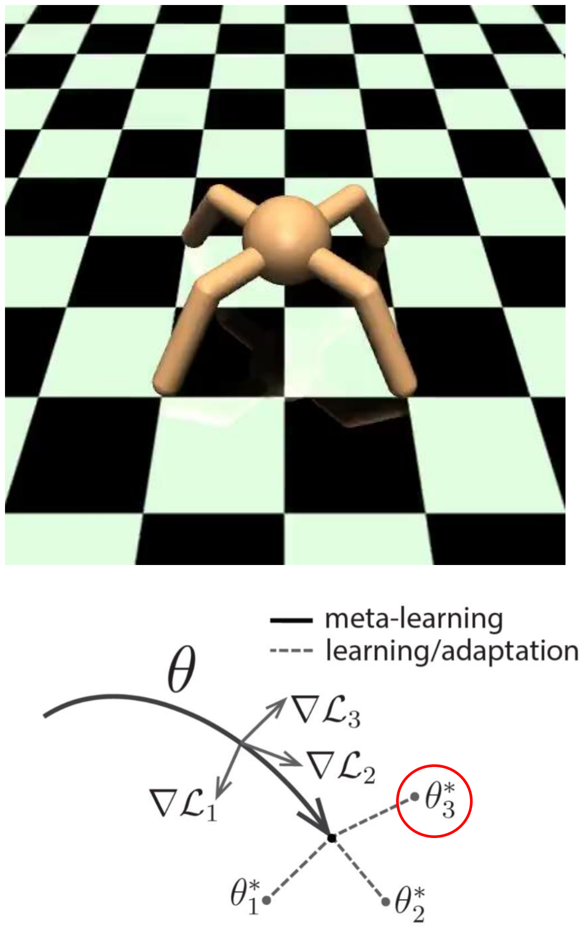
MAML for RL in videos

after MAML training



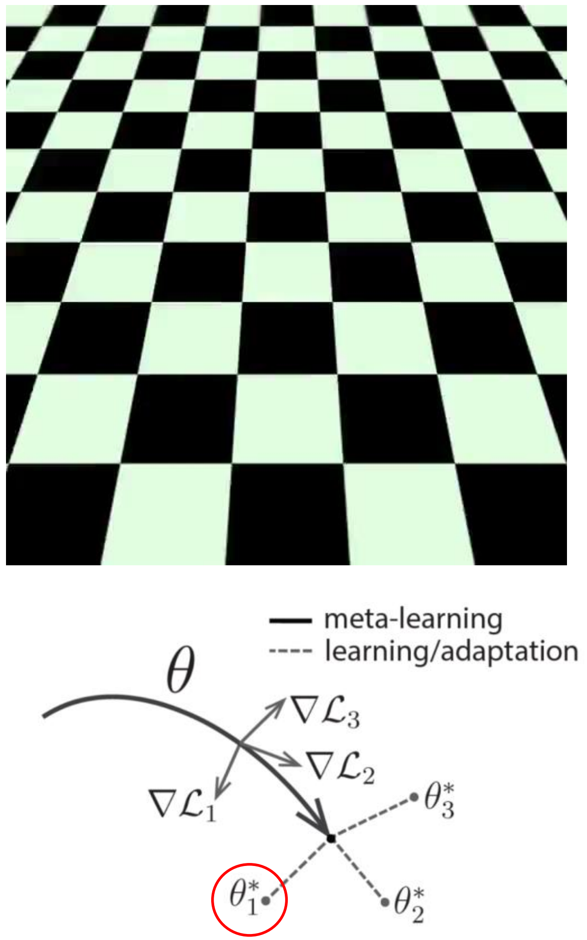
after 1 gradient step

(forward reward)



after 1 gradient step

(backward reward)



More on MAML/gradient-based meta-learning for RL

MAML meta-policy gradient estimators:

- Finn, Abbeel, Levine. **Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks.**
- Foerster, Farquhar, Al-Shedivat, Rocktaschel, Xing, Whiteson. **DiCE: The Infinitely Differentiable Monte Carlo Estimator.**
- Rothfuss, Lee, Clavera, Asfour, Abbeel. **ProMP: Proximal Meta-Policy Search.**

Improving exploration:

- Gupta, Mendonca, Liu, Abbeel, Levine. **Meta-Reinforcement Learning of Structured Exploration Strategies.**
- Stadie*, Yang*, Houthooft, Chen, Duan, Wu, Abbeel, Sutskever. **Some Considerations on Learning to Explore via Meta-Reinforcement Learning.**

Hybrid algorithms (not necessarily gradient-based):

- Houthooft, Chen, Isola, Stadie, Wolski, Ho, Abbeel. **Evolved Policy Gradients.**
- Fernando, Sygnowski, Osindero, Wang, Schaul, Teplyashin, Sprechmann, Pirtzel, Rusu. **Meta-Learning by the Baldwin Effect.**

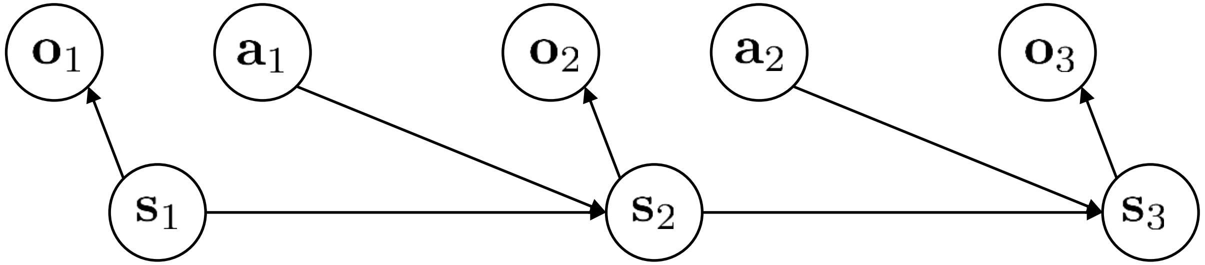
Meta-RL as a POMDP

Meta-RL as… partially observed RL?

$\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{E}, r\}$

$\mathcal{O}$ – observation space $\qquad$ observations $o \in \mathcal{O}$ (discrete or continuous)

$\mathcal{E}$ – emission probability $p(o_t|s_t)$



policy must act on observations o_t!

$\pi_\theta(a|o)$

typically requires *either*:

explicit state estimation, i.e. to estimate $p(s_t|o_{1:t})$

policies with memory

Meta-RL as... partially observed RL?

$$\pi_\theta(a|\overbrace{s, z}^{\tilde{s}})$$

encapsulates information policy needs to solve current task

learning a task $=$ inferring z

from *context* $(s_1, a_1, s_2, r_1), (s_2, a_2, s_3, r_2), \ldots$

this is just a POMDP!

before: $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$

now: $\tilde{\mathcal{M}} = \{\tilde{\mathcal{S}}, \mathcal{A}, \tilde{\mathcal{O}}, \tilde{\mathcal{P}}, \mathcal{E}, r\}$

$\tilde{\mathcal{S}} = \mathcal{S} \times \mathcal{Z}$ $\quad$ $\tilde{s} = (s, z)$

$\tilde{\mathcal{O}} = \mathcal{S}$ $\quad$ $\tilde{o} = s$

key idea: solving the POMDP $\tilde{\mathcal{M}}$ is equivalent to meta-learning!

Meta-RL as… partially observed RL?

$\pi_\theta(a|s, z)$

encapsulates information policy needs to solve current task

learning a task $=$ inferring z

from *context* $(s_1, a_1, s_2, r_1), (s_2, a_2, s_3, r_2), \ldots$

this is just a POMDP!

typically requires *either*:

explicit state estimation, i.e. to estimate $p(s_t|o_{1:t})$

policies with memory

need to estimate $p(z_t|s_{1:t}, a_{1:t}, r_{1:t})$

exploring via posterior sampling with latent context

1. sample $z \sim \hat{p}(z_t|s_{1:t}, a_{1:t}, r_{1:t})$
2. act according to $\pi_\theta(a|s, z)$ to collect more data

some approximate posterior (e.g., variational)

act *as though z was correct!*

this is *not* optimal! why?

but it's pretty good, both in theory and in practice!

See, e.g. Russo, Roy. **Learning to Optimize via Posterior Sampling.**

Variational inference for meta-RL

policy: $\pi_\theta(a_t|s_t, z_t)$

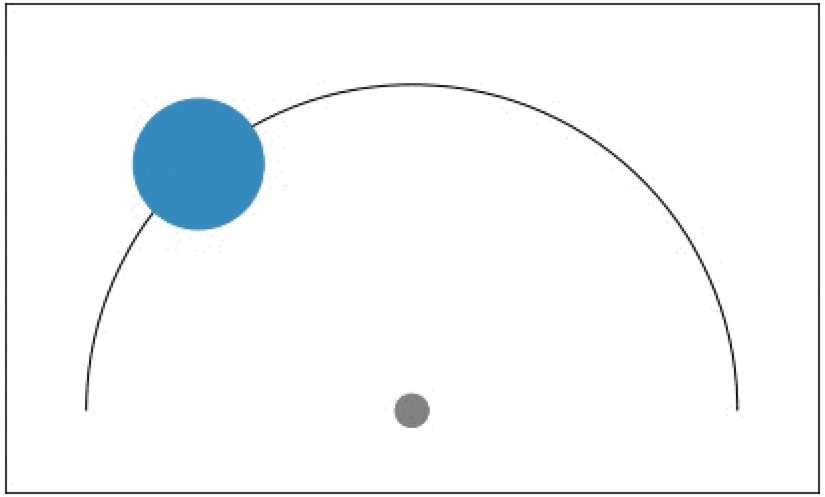
inference network: $q_\phi(z_t|s_1, a_1, r_1, \dots, s_t, a_t, r_t)$

$$(\theta, \phi) = \arg\max_{\theta,\phi} \frac{1}{N} \sum_{i=1}^{n} E_{z\sim q_\phi, \tau\sim\pi_\theta}[R_i(\tau) - D_{\text{KL}}(q(z|\dots)\|p(z))]$$

maximize *post-update* reward (same as standard meta-RL)

stay close to prior

$z_t \sim q_\phi(z_t|s_1, a_1, r_1, \dots, s_t, a_t, r_t)$



conceptually *very* similar to RNN meta-RL, but with stochastic z

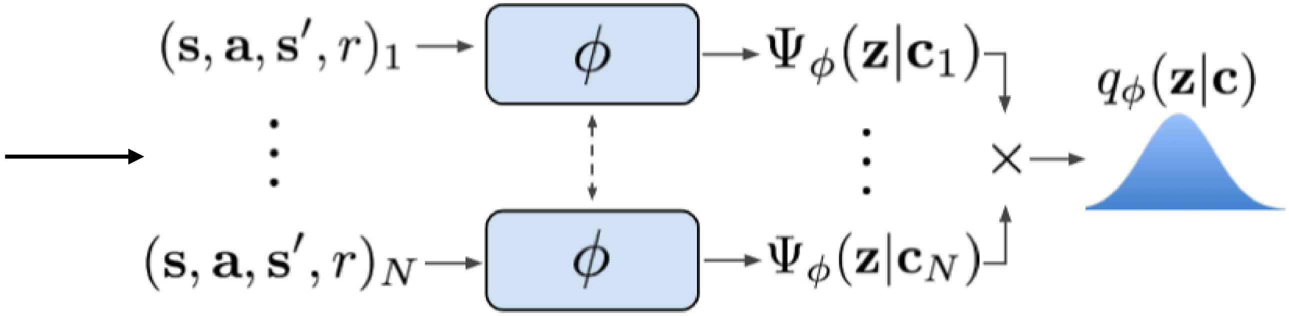
stochastic z enables exploration via *posterior sampling*

Rakelly*, Zhou*, Quillen, Finn, Levine. **Efficient Off-Policy Meta-Reinforcement learning via Probabilistic Context Variables.** ICML 2019.

Specific instantiation: PEARL

policy: $\pi_\theta(a_t|s_t, z_t)$

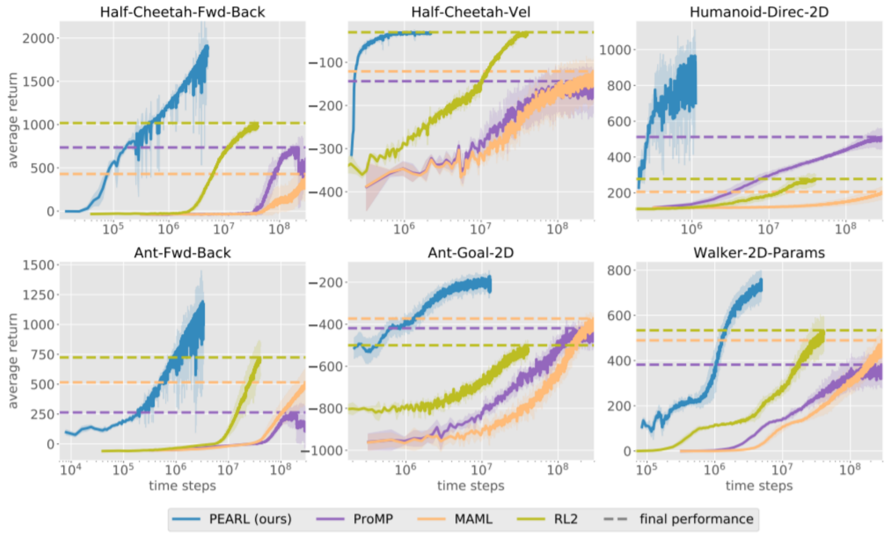
inference network: $q_\phi(z_t|s_1, a_1, r_1, \ldots, s_t, a_t, r_t)$



$(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)_1$
$(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)_N$
ϕ
ϕ
$\Psi_\phi(\mathbf{z}|\mathbf{c}_1)$
$\Psi_\phi(\mathbf{z}|\mathbf{c}_N)$
×
$q_\phi(\mathbf{z}|\mathbf{c})$

$$(\theta, \phi) = \arg\max_{\theta,\phi} \frac{1}{N} \sum_{i=1}^{n} E_{z\sim q_\phi, \tau\sim\pi_\theta}[R_i(\tau) - D_{\mathrm{KL}}(q(z|\ldots)\|p(z))]$$

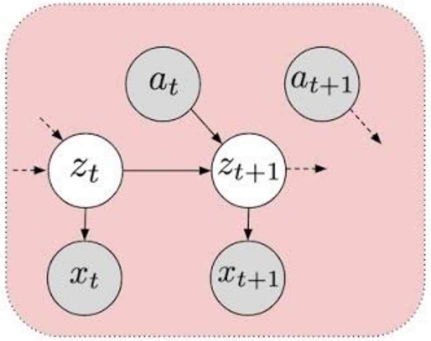
perform maximization using soft actor-critic (SAC), state-of-the-art off-policy RL algorithm



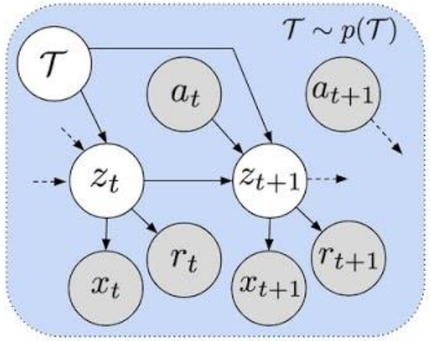
Rakelly*, Zhou*, Quillen, Finn, Levine. **Efficient Off-Policy Meta-Reinforcement learning via Probabilistic Context Variables.** ICML 2019.

MELD: Model-Based Meta-RL with Images

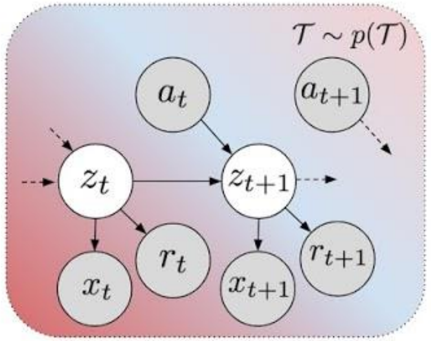
meta-learning can be viewed as a (kind of) POMDP



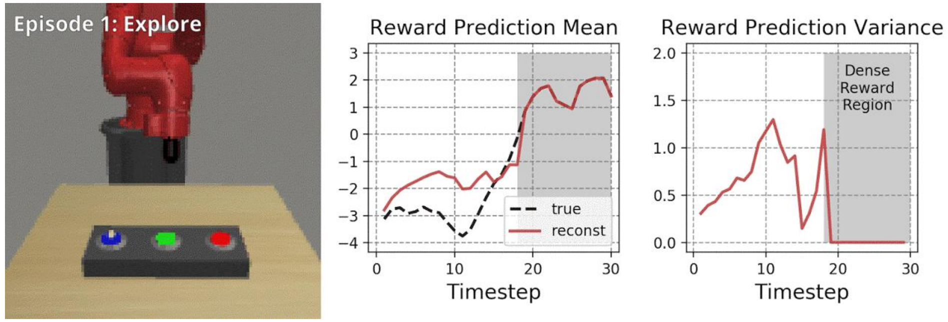
regular POMDP



meta-RL



MELD



Using this latent variable model generalizes meta-learning **and** POMDPs

Turns out to work very well as a meta-learning algorithm!

Zhao, Nagabandi, Rakelly, Finn, Levine. **MELD: Meta-Reinforcement Learning from Images via Latent State Models.** ‘20

References on meta-RL, inference, and POMDPs

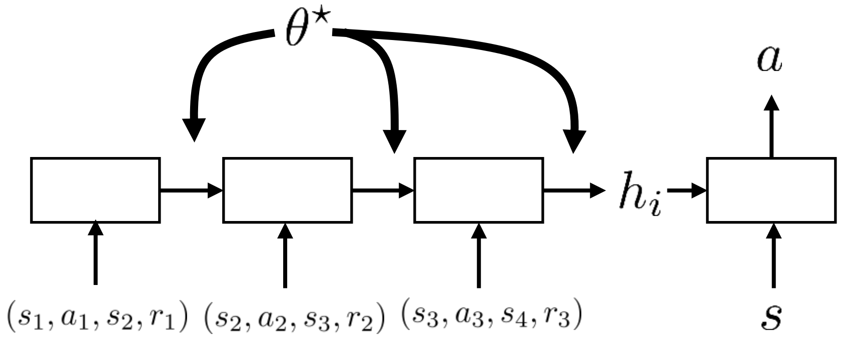
- Rakelly*, Zhou*, Quillen, Finn, Levine. **Efficient Off-Policy Meta-Reinforcement learning via Probabilistic Context Variables.** ICML 2019.

- Zintgraf, Igl, Shiarlis, Mahajan, Hofmann, Whiteson. **Variational Task Embeddings for Fast Adaptation in Deep Reinforcement Learning.**

- Humplik, Galashov, Hasenclever, Ortega, Teh, Heess. **Meta reinforcement learning as task inference.**

The three perspectives on meta-RL

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

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

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

what should $f_\theta(\mathcal{M}_i)$ *do*?

1. improve policy with experience from $\mathcal{M}_i$

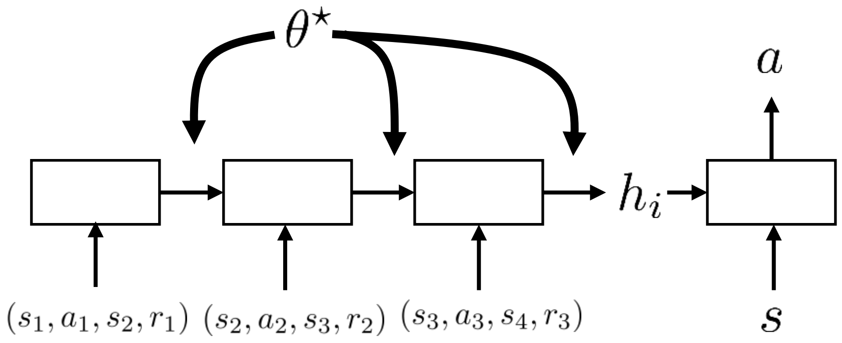
   $\{(s_1, a_1, s_2, r_1), \ldots, (s_T, a_T, s_{T+1}, r_T)\}$

2. (new in RL): choose how to interact, i.e. choose a_t

   meta-RL must also *choose* how to *explore*!

The three perspectives on meta-RL

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

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

+ simple, effective exploration via posterior sampling

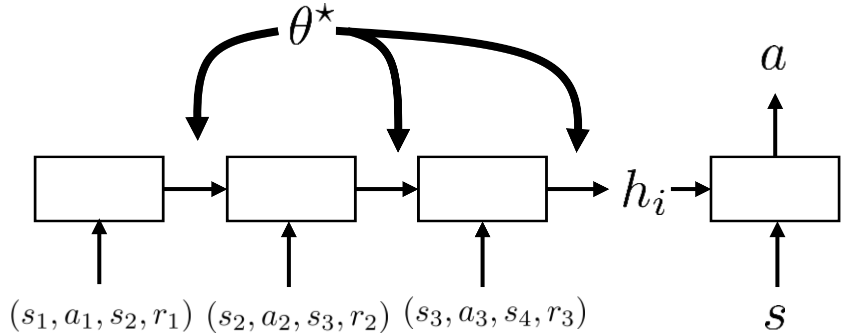
+ elegant reduction to solving a special POMDP

- vulnerable to *meta-overfitting*

- challenging to optimize in practice

But they’re not that different!

Perspective 1: just RNN it



just perspective 1, but with stochastic hidden variables!

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

Perspective 2: bi-level optimization

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

MAML for RL

just a particular architecture choice for these

Perspective 3: it’s an inference problem!

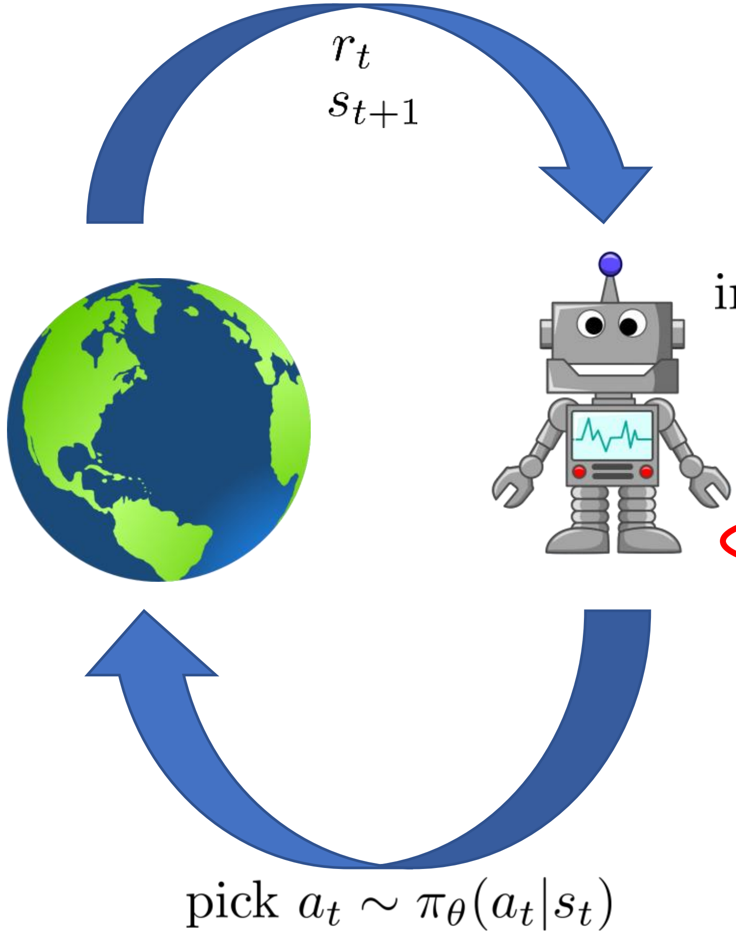
$$\pi_\theta(a|s, z) \qquad z_t \sim p(z_t|s_{1:t}, a_{1:t}, r_{1:t})$$

everything needed to solve task

Model-Based Meta-RL

Model-based meta-RL

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



improve π_θ...

...directly, via policy gradients

...via value function or Q-function

...implicitly, via model $\hat{p}(s_{t+1}|s_t, a_t)$

short sketch of model-based RL:

1. collect data $\mathcal{B}$
2. use $\mathcal{B}$ to get $\hat{p}(s_{t+1}|s_t, a_t)$
3. use $\hat{p}(s_{t+1}|s_t, a_t)$ to *plan* a

why?

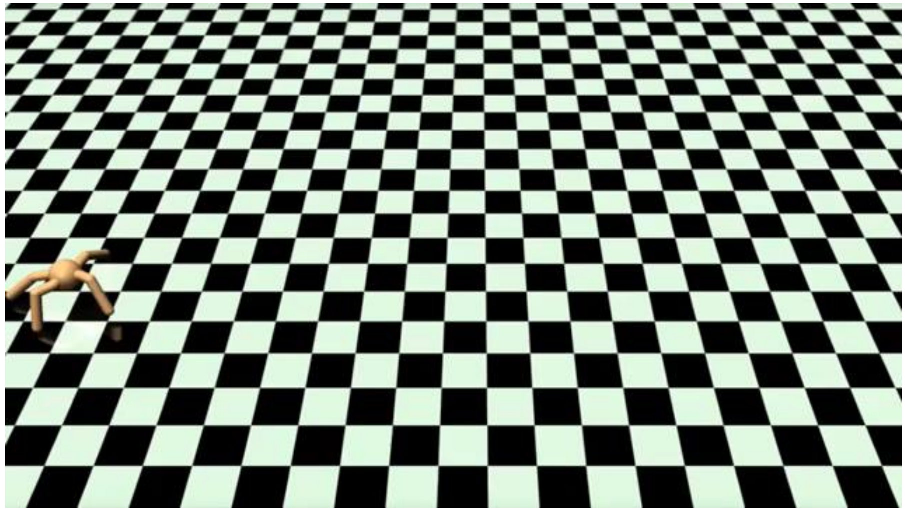
+ requires much less data vs model-free

+ a bit different due to model

+ can adapt extremely quickly!

Model-based meta-RL

example task: ant with broken leg



non-adaptive method:

a few episodes

1. collect data $\mathcal{B} = \{s_i, a_i, s'_i\}$
2. train $d_\theta(s, a) \rightarrow s'$ on $\mathcal{B}$
3. use d_θ to optimize actions

$$a_t, \dots, a_{t+k} = \arg \max_{a_t, \dots, a_{t+k}} \sum_{\tau=t}^{t+k} r(s_\tau, a_\tau)$$

$$\text{s.t. } s_{t+1} = d_\theta(s_t, a_t)$$

adaptive method:

1. take *one* step, get $\{s, a, s'\}$
2. $\theta \leftarrow \theta - \alpha \nabla_\theta \| d_\theta(s, a) - s' \|^2$
3. use d_θ to optimize $a_t, \dots, a_{t+k}$, take a_t

nice idea, but how much can we really adapt in just *one* (or a few) step(s)?

Model-based meta-RL

meta-training time

$$\mathcal{D}_{\text{meta-train}} = \{(\mathcal{D}_1^{\text{tr}}, \mathcal{D}_1^{\text{ts}}), \dots, (\mathcal{D}_n^{\text{tr}}, \mathcal{D}_n^{\text{ts}})\}$$

$$\mathcal{D}_i^{\text{tr}} = \{(x_1^i, y_1^i), \dots, (x_k^i, y_k^i)\}$$

$$\mathcal{D}_i^{\text{ts}} = \{(x_1^i, y_1^i), \dots, (x_l^i, y_l^i)\}$$

$$x \leftarrow (s, a) \qquad y \leftarrow s'$$

generate each $\mathcal{D}_i^{\text{tr}}, \mathcal{D}_i^{\text{ts}}$:

sample subsequence $s_t, a_t, \dots, s_{t+k}, a_{t+k}, s_{t+k+1}$ from past experience

$$\mathcal{D}_i^{\text{tr}} \leftarrow \{(s_t, a_t, s_{t+1}), \dots, (s_{t+k-1}, a_{t+k-1}, s_{t+k})\}$$

$$\mathcal{D}_i^{\text{ts}} \leftarrow \{(s_{t+k}, a_{t+k}, s_{t+k+1})\}$$

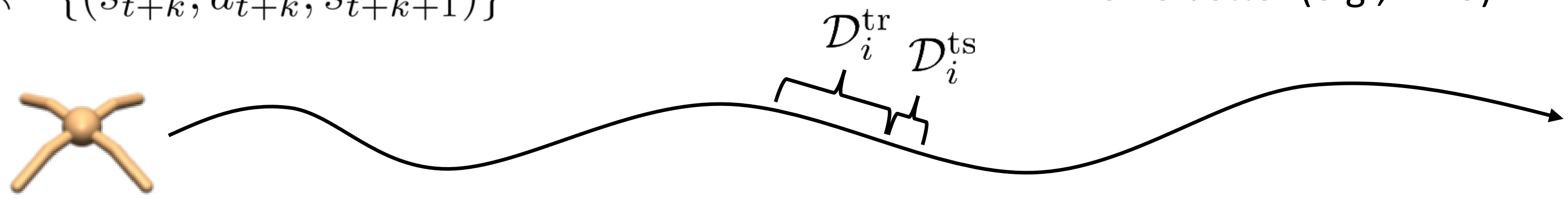
meta-test time

adaptive method:

1. take *one* step, get $\{s, a, s'\}$
2. $\theta \leftarrow \theta - \alpha \nabla_\theta \| d_\theta(s, a) - s' \|^2$
3. use d_θ to optimize $a_t, \dots, a_{t+k}$, take a_t

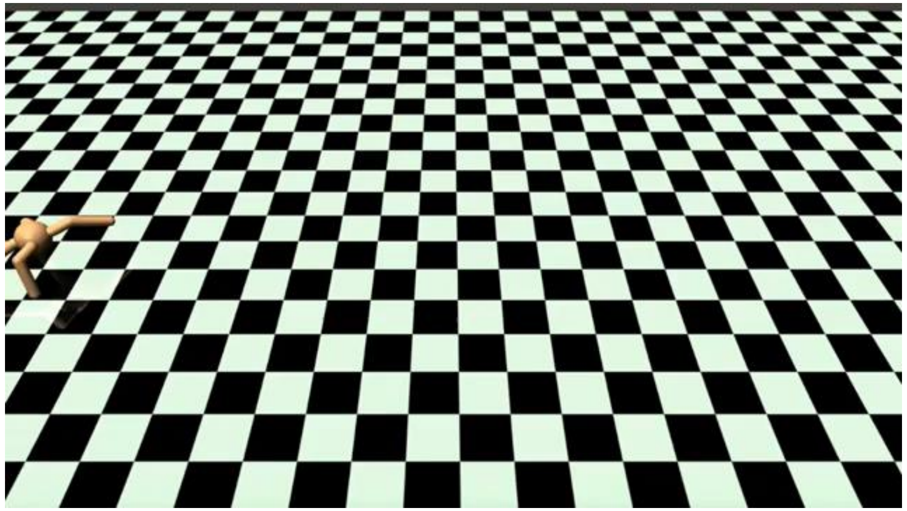
assumes past experience has many different dynamics

could choose k = 1, but k > 1 works better (e.g., k = 5)



Model-based meta-RL

example task: ant with broken leg



meta-test time

adaptive method:

1. take *one* step, get $\{s, a, s'\}$
2. $\theta \leftarrow \theta - \alpha \nabla_\theta \| d_\theta(s, a) - s' \|^2$
3. use d_θ to optimize $a_t, \ldots, a_{t+k}$, take a_t

Real-world results

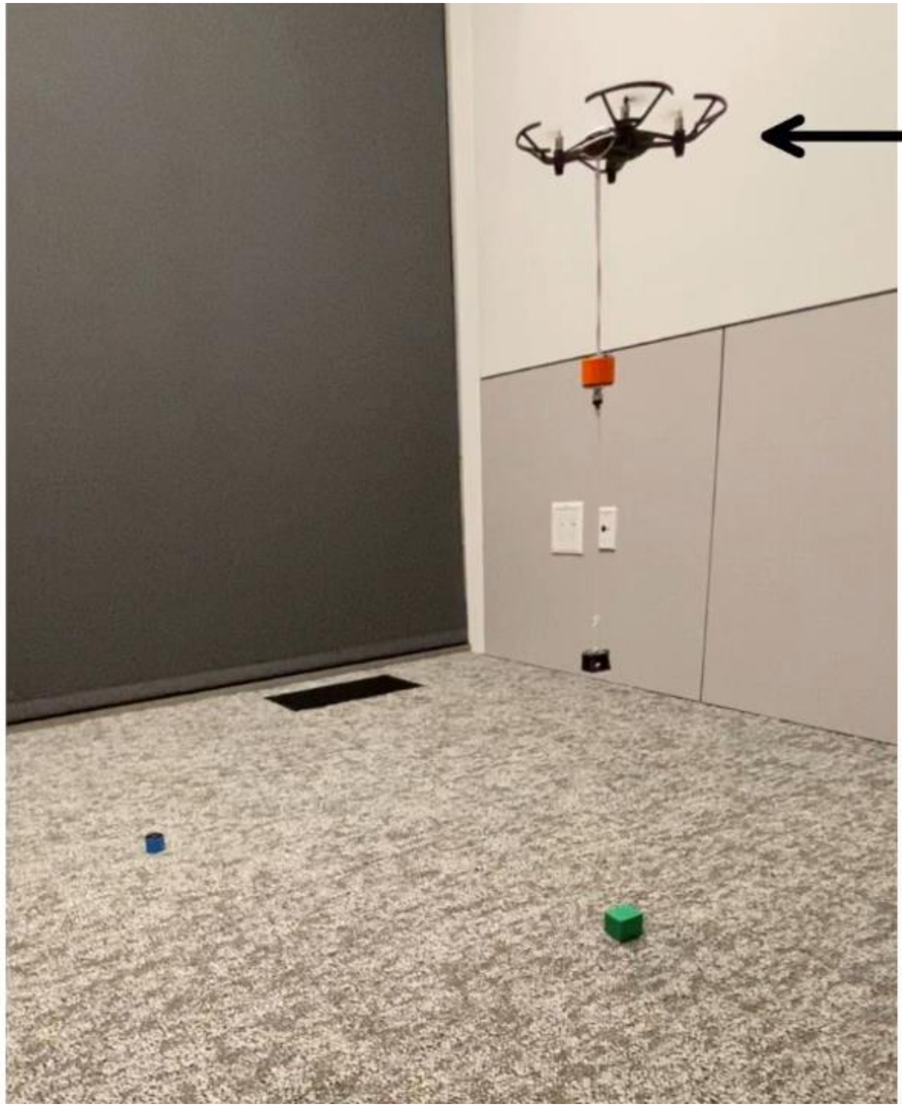
See also:

Saemundsson, Hofmann, Deisenroth. **Meta-Reinforcement Learning with Latent Variable Gaussian Processes.**

Nagabandi, Finn, Levine. **Deep Online Learning via Meta-Learning: Continual Adaptation for Model-Based RL.**

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. **Learning to Adapt in Dynamic, Real-World Environments Through Meta-Reinforcement Learning.** ICLR 2019.

Model-Based Meta-RL for Quadrotor Control

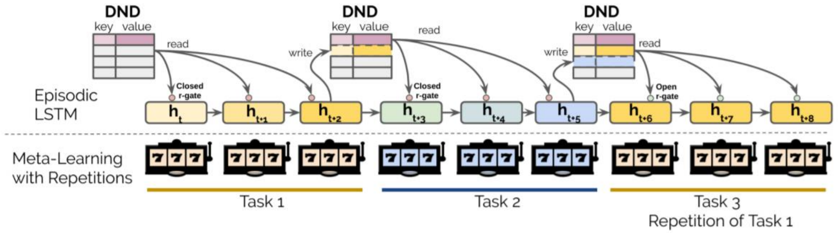


Quadcopter

Belkhale, Li, Kahn, McAllister, Calandra, Levine. **Model-Based Meta-Reinforcement Learning for Flight with Suspended Payloads.** ‘20

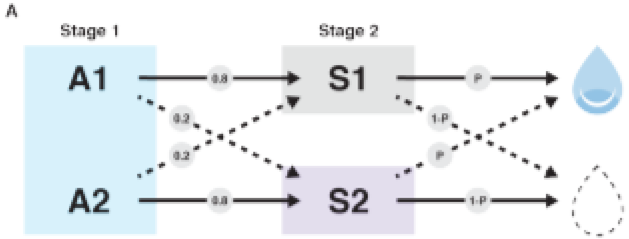
Meta-RL and emergent phenomena

meta-RL gives rise to episodic learning



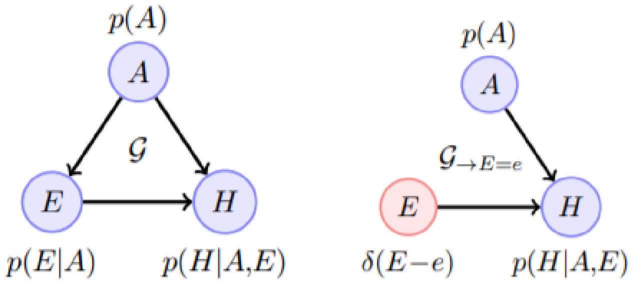
Ritter, Wang, Kurth-Nelson, Jayakumar, Blundell, Pascanu, Botvinick. **Been There, Done That: Meta-Learning with Episodic Recall.**

model-free meta-RL gives rise to model-based adaptation



Wang, Kurth-Nelson, Kumaran, Tirumala, Soyer, Leibo, Hassabis, Botvinick. **Prefrontal Cortex as a Meta-Reinforcement Learning System.**

meta-RL gives rise to causal reasoning (!)



Dasgupta, Wang, Chiappa, Mitrovic, Ortega, Raposo, Hughes, Battaglia, Botvinick, Kurth-Nelson. **Causal Reasoning from Meta-Reinforcement Learning.**

Humans and animals *seemingly* learn behaviors in a variety of ways:

- Highly efficient but (apparently) model-free RL
- Episodic recall
- Model-based RL
- Causal inference
- etc.

Perhaps each of these is a separate “algorithm” in the brain

But maybe these are all emergent phenomena resulting from meta-RL?