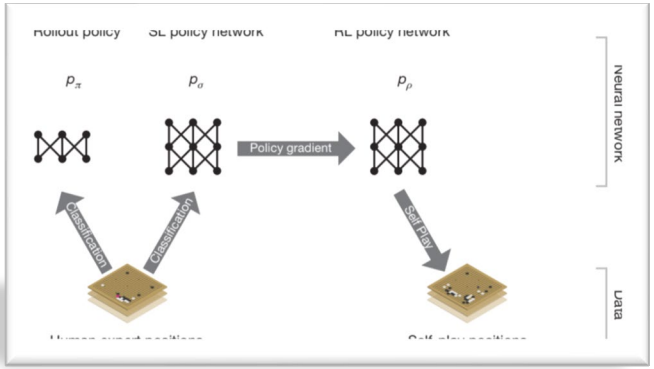
元强化学习
Meta Reinforcement Learning

朱圆恒
中国科学院自动化研究所

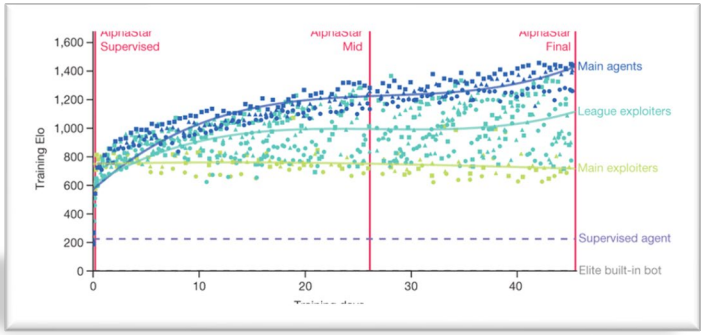
背景

- **机器学习**模型通常需要大量的样本训练
  - ImageNet: 14million，ChatGPT: 45TB的文本数据
- **强化学习**模型同样需要与环境大量交互的样本训练
  - AlphaGo: 3000万样本，AlphaStar: 200年游戏数据量（单个智能体）



Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Table 2.2: Datasets used to train GPT-3. "Weight in training mix" refers to the fraction of examples during training that are drawn from a given dataset, which we intentionally do not make proportional to the size of the dataset. As a result, when we train for 300 billion tokens, some datasets are seen up to 3.4 times during training while other datasets are seen less than once.



背景

- **机器学习**模型通常需要大量的样本训练
  - ImageNet: 14million，ChatGPT: 45TB的文本数据
- **强化学习**模型同样需要与环境大量交互的样本训练
  - AlphaGo: 3000万样本，AlphaStar: 200年游戏数据量（单个智能体）

- 但是**人类学习**过程是非常高效的
  - 儿童仅需几张图片即可区分cats和birds
  - 会骑自行车后只需少量尝试，甚至无需尝试就能骑摩托车

背景

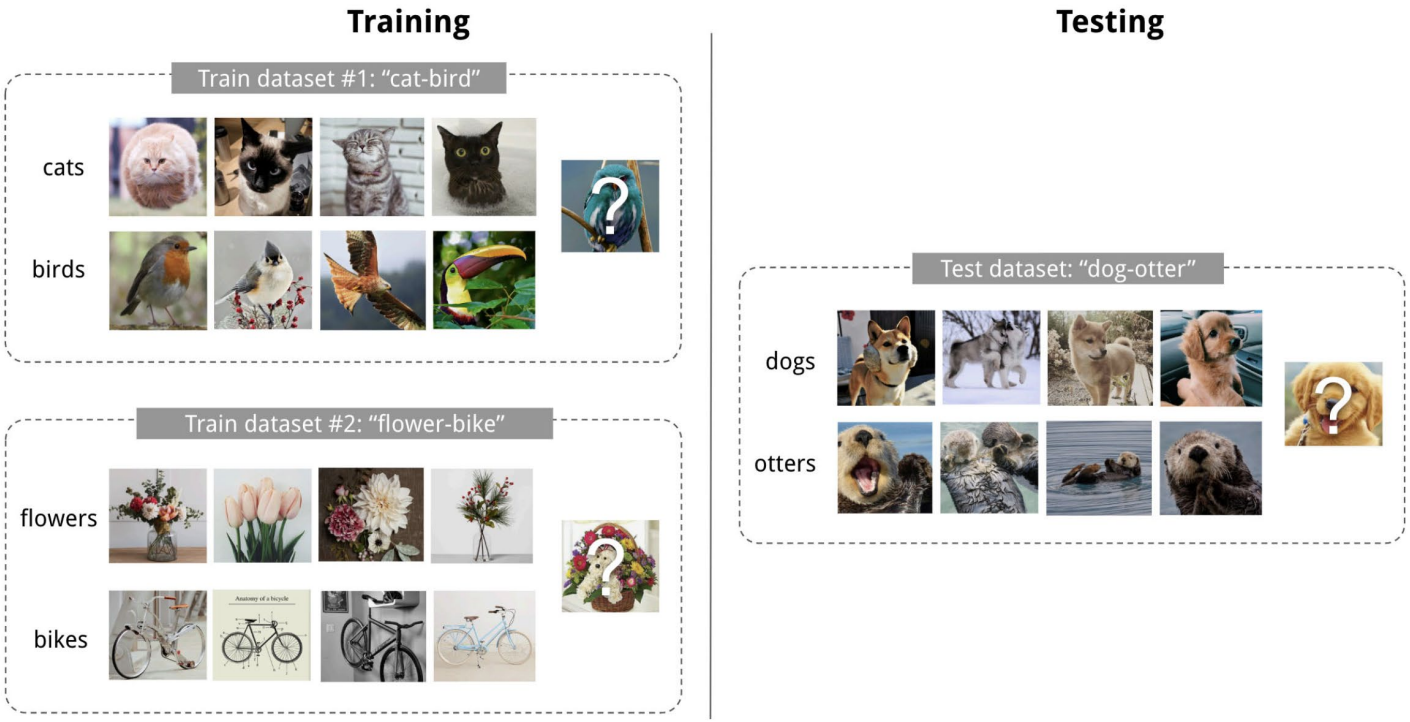
- **机器学习**模型通常需要大量的样本训练
  - ImageNet: 14million，ChatGPT: 45TB的文本数据
- **强化学习**模型同样需要与环境大量交互的样本训练
  - AlphaGo: 3000万样本，AlphaStar: 200年游戏数据量（单个智能体）

- 但是**人类学习**过程是非常高效的
  - 儿童仅需几张图片即可区分cats和birds
  - 会骑自行车后只需少量尝试，甚至无需尝试就能骑摩托车

- 因此，关于机器学习的一个目标：只需少量的训练样本，即可快速学到新任务的内含和技能？
- 元学习！

元学习

- 元学习可以与机器学习中的多种问题相关：监督学习，强化学习，等等
- 举例：
  - 模型在不包含dog/otter的图像上训练后，给它一组（少量）的 dog/otter 图片，能够快速掌握区分两者的能力



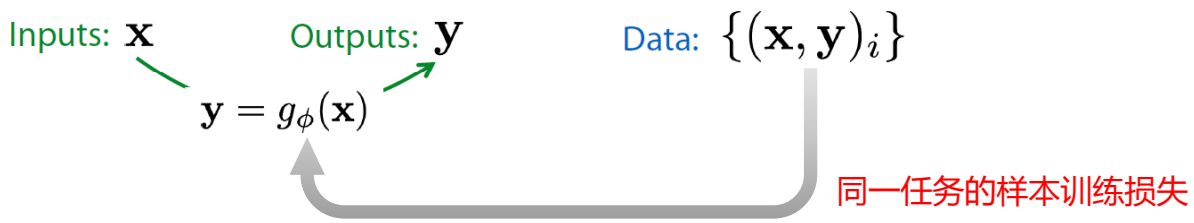
元学习

- 元学习可以与机器学习中的多种问题相关：监督学习，强化学习，等等
- 举例：
  - 模型在不包含dog/otter的图像上训练后，给它一组（少量）的 dog/otter 图片，能够快速掌握区分两者的能力
  - 游戏bot能够快速掌握一个新游戏的玩法
  - 机器人只在水平平面上训练，但是能在一定斜坡的平面上完成指定任务

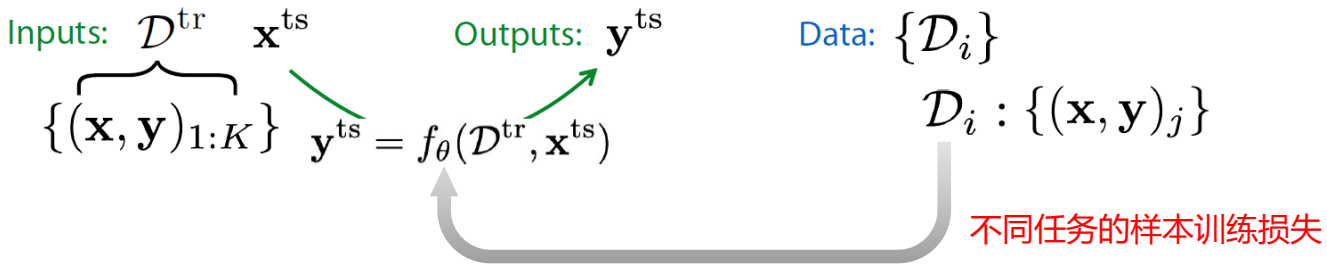
- 理想情况下，如果元学习的智能体足够“smart”，能够解决的unseen tasks足够广泛，包罗万象，我们就达到了通用人工智能的目标：对任何RL问题无需过多人类接入，或人工特征工程，即可快速解决

元学习问题

- 监督学习



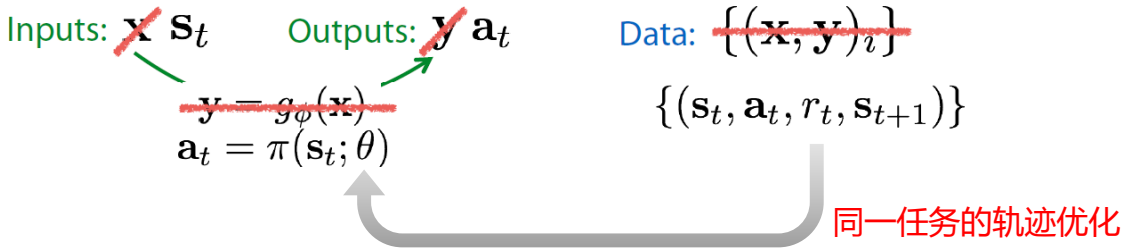
- 元监督学习



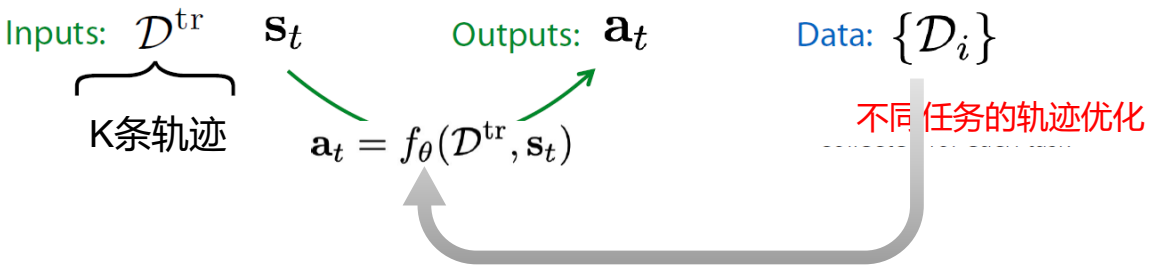
- 在原本**监督学习**的结果 g_ϕ 只考虑单个样本的输入和输出关系，不关心其它样本的分布与当前任务的关系
- **元监督学习**会从其它样本分布中提取出任务信息，在预测下一个样本结果时会有更快的适应提升效果
- 元学习的关键是设计和优化 f 函数

元强化学习问题

- 强化学习



- 元强化学习



- **元强化学习**任务不只是要考虑设计/优化 f，同时要考虑在任务上如何收集数据 （vs 监督学习中 data 是现成的）

元学习与元强化学习

- 学习learning问题

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

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

Learning process/algorithm

- 元学习meta-learning问题

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

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

Parametrized learning process/algorithm

- 强化学习RL问题

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

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

RL process/algorithm

MDP

- 元强化学习 meta RL问题

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

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

Parametrized RL process/algorithm

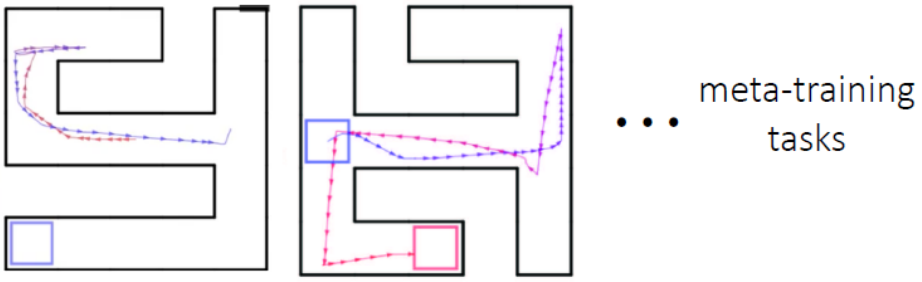
MDP for task i

Meta-RL举例：迷宫导航

- π^{exp} 探索策略，π^{task} 任务策略，两者可以相同也可以不同
- 不同迷宫的状态转移 $p_i(s'|s,a)$ 和奖励函数 $r_i(s,a)$ 不同

Meta-Train Time:

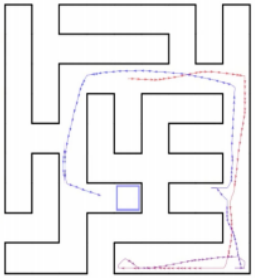
Learn how to efficiently explore & solve many MDPs:



Meta-train $\pi^{\text{exp}}, \pi^{\text{task}}$

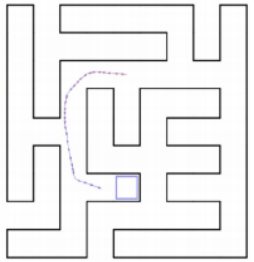
Meta-Test Time:

Collect small amount of experience in new MDP



Collect $\mathcal{D}_{\text{tr}} \sim \pi^{\text{exp}}$

Learn policy that solves that MDP



$\mathcal{D}_{\text{tr}} \rightarrow \pi^{\text{task}}$

- 假设前提：元训练和元测试的MDPs是来自**相同分布**，$\mathcal{M}_i \sim p(\mathcal{M})$

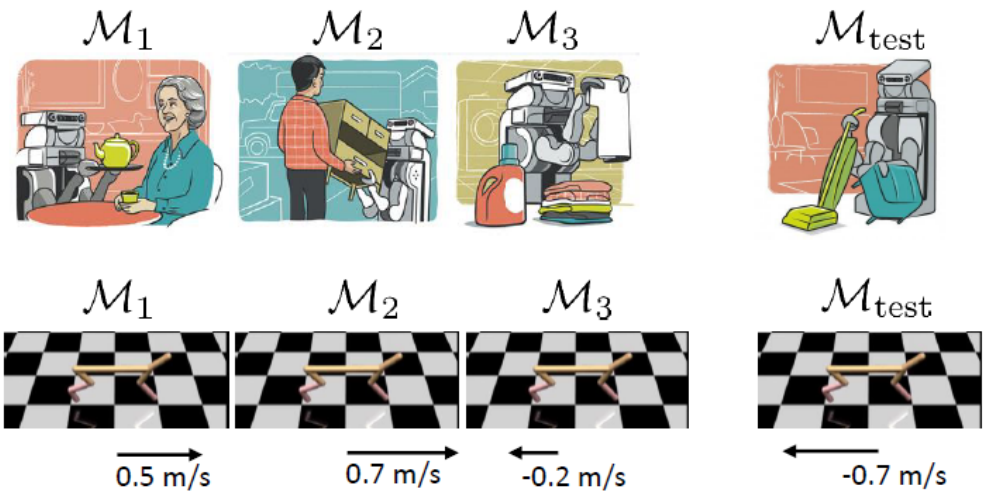
（因而可以实现泛化）

元强化学习问题

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

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

- **假设：** 在元训练meta-train阶段会遇到多个MDP $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$，每个MDP均采样来自于同一分布 $\mathcal{M}_i \sim p(\mathcal{M})$
- 在元测试 meta-test 时，会从同样的分布上采样$\mathcal{M}_{\text{test}} \sim p(\mathcal{M})$，然后期望得到对应的最佳策略 $\phi_i = f_\theta(\mathcal{M}_{\text{test}})$



元学习 vs 基于上下文的策略

- 元学习：利用在 $\mathcal{M}_i$ 上交互的经验，尽快找到最佳策略

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

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

- 可以看作是基于上下文的策略 contextual policies
  - 利用上下文提升策略性能，等价于根据与当前MDP的交互经验，实现最佳性能

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

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

上下文 context

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

元学习 vs 多任务学习

- 在 meta-RL 中，上下文是由从 $\mathcal{M}_i$ 交互的经验中推断出来

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

看作是多任务策略，但是以轨迹历史作为task identifier

Encoder

$s_1, a_1, r_1, \ldots, s_{t-1}, a_{t-1}, r_{t-1}$

在新任务上的**适应（k-shot）**

- 在 multi-task RL中，上下文通常是给定的

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

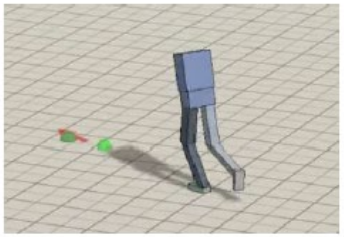
or

$\pi_{\theta_{\text{ID}}}(a_t|s_t)$

事先已知

在不同/新任务的**泛化 (0-shot)**

ϕ: stack location



ϕ: walking direction

ϕ: where to hit puck

多任务学习 vs 迁移学习 vs 元学习

- 多任务学习 multi-task learning
  - 同一套模型和参数解决多个任务 $\mathcal{T}_1, \ldots, \mathcal{T}_T$

$$\min_{\theta} \sum_{i=1}^{T} \mathcal{L}_i(\theta, \mathcal{D}_i)$$

- 迁移学习 transfer learning
  - 将在源任务 $\mathcal{T}_a$ 上学到的知识迁移到目标任务 $\mathcal{T}_b$

- 元学习 meta-learning
  - 给定任务 $\mathcal{T}_1, \ldots, \mathcal{T}_T$ 的数据后，快速解决新任务 $\mathcal{T}_{test}$

- 前提：任务共享相同或相似的结构
  - 例如 强化学习任务

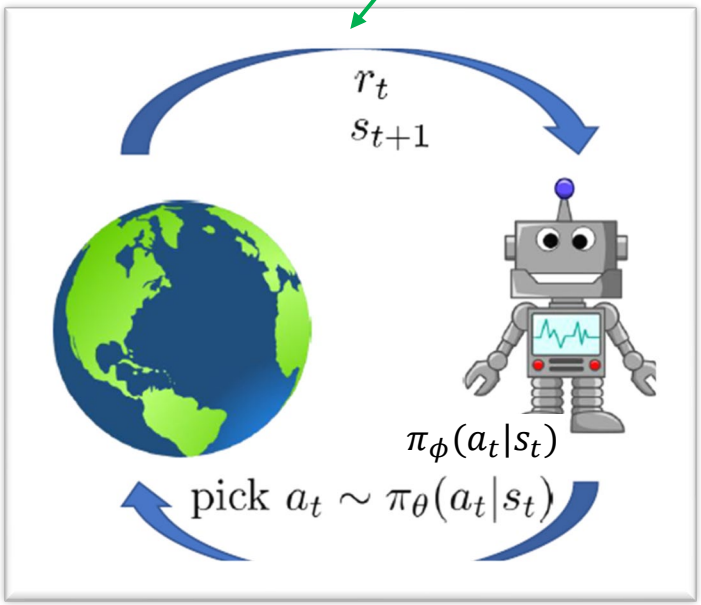
$$\mathcal{T}_i = \{S_i, A_i, \rho_i(s), p_i(s'|s,a), r_i(s,a)\}$$

  - 不同的任务状态空间、动作空间、初始状态、转移函数相同，奖励函数不同
    （half-cheetah 不同奔跑速度，迷宫中不同的目标点）

元强化学习

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

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



利用 (s_t, a_t, s_{t+1}, r_t) 提升 π_ϕ

- 关键问题：$f_\theta(\mathcal{M}_i)$ 要做什么？
  1. 利用 $\mathcal{M}_i$ 产生的经验，学习提升策略

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

  2. 除此之外，还要选择动作 a_t，决定如何与 $\mathcal{M}_i$ 交互，即 meta-RL需要决定如何探索

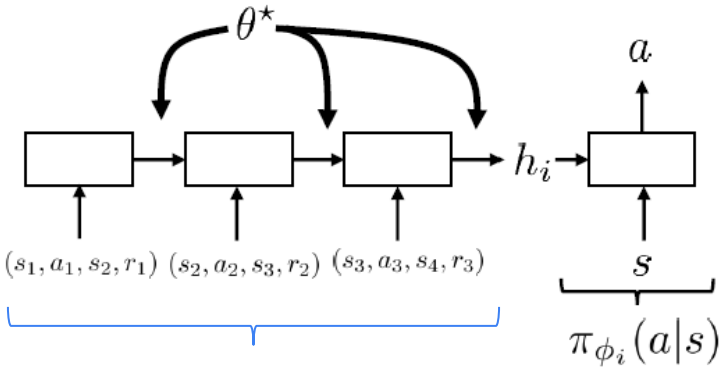
基于递归的元强化学习

Recurrent Based Meta Learning

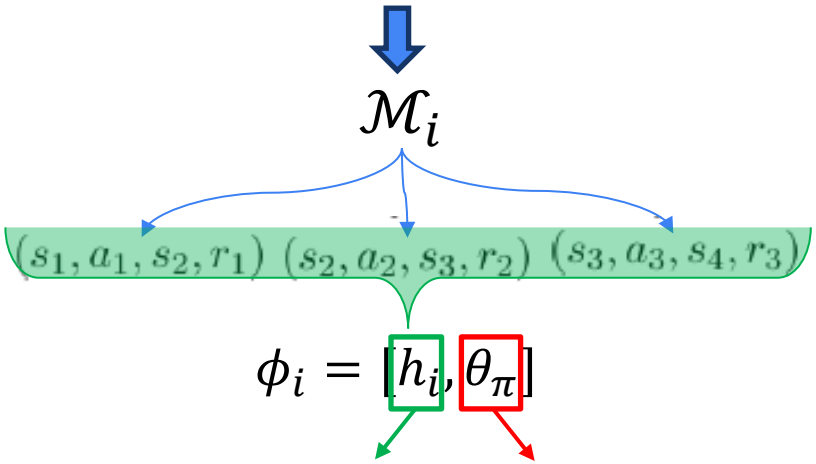
递归策略

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

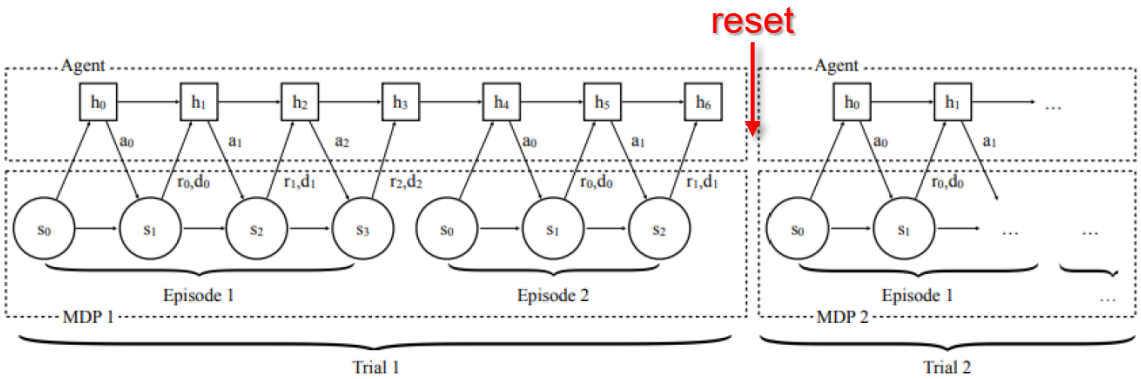
where $\phi_i = f_\theta(\mathcal{M}_i)$ 从MDP中找到策略



递归网络
RNN, LSTM, GRU, etc



- 训练一个递归策略（输入包括奖励，对多个MDPs的训练实现泛化）
- 关键在于 RNN 的隐含状态在（同一MDP）不同 episodes之间是不会被重置



递归策略

- 为什么递归策略能够学习如何探索？

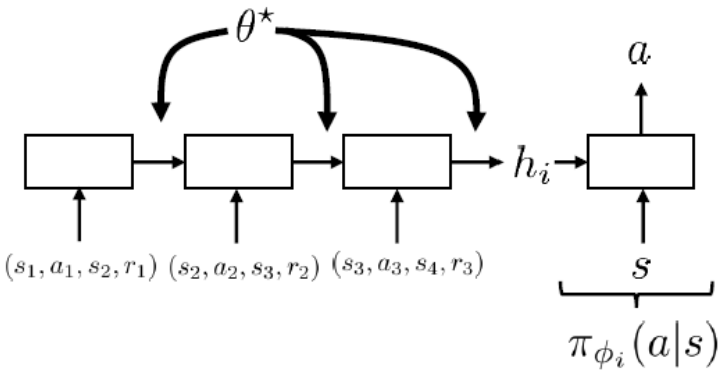
$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} E_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

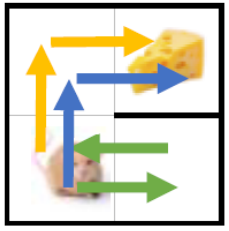
where $\phi_i = f_\theta(\mathcal{M}_i)$

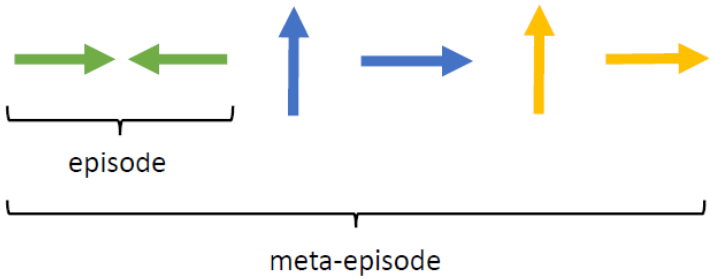
- 关键问题：$f_\theta(\mathcal{M}_i)$ 要做什么？
  1. 利用 $\mathcal{M}_i$ 产生的经验，学习提升策略

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

  2. 除此之外，还要选择动作 a_t，决定如何与 $\mathcal{M}_i$ 交互
     即 meta-RL需要决定如何探索

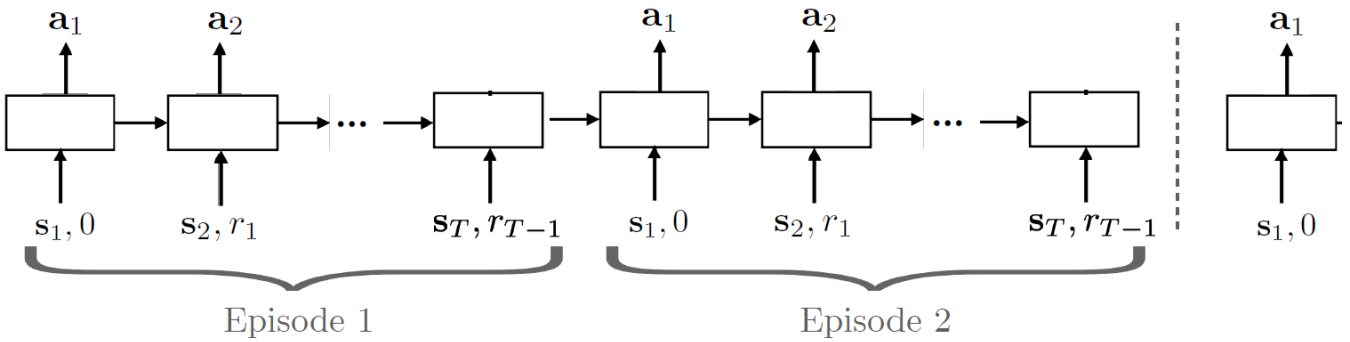






通过在整个 meta-episode 上对递归策略优化所有的奖励，自动会学习如何探索！

算法流程



- 元训练 Meta-Training

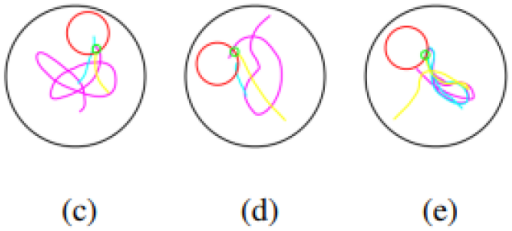
1. 采样 MDP $\mathcal{M}_i$ (p_i, r_i)
2. 运行策略 $\pi_{\phi_i}(a|s)$ 生成 N 条轨迹
3. 将$\mathcal{M}_i$的轨迹序列存储在经验池中
4. 在所有任务上更新策略参数，实现回报的最大化

- 元测试 Meta-Test

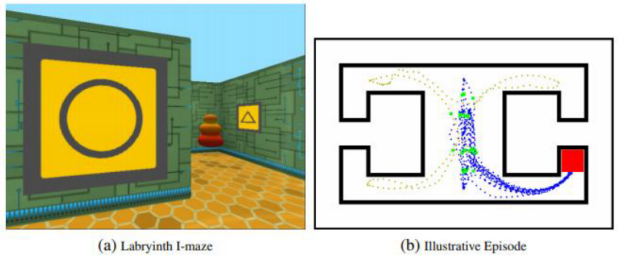
1. 采样新的 MDP $\mathcal{M}_j$
2. 运行策略 $\pi_{\phi_j}(a|s)$ 生成 N 条轨迹
3. 统计评估策略性能

基于递归策略的元强化学习

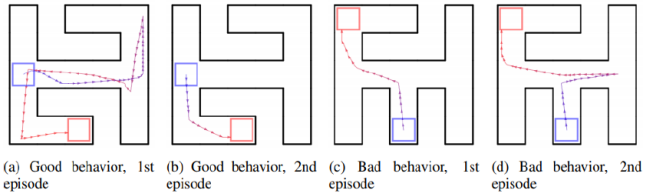
- 典型工作



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



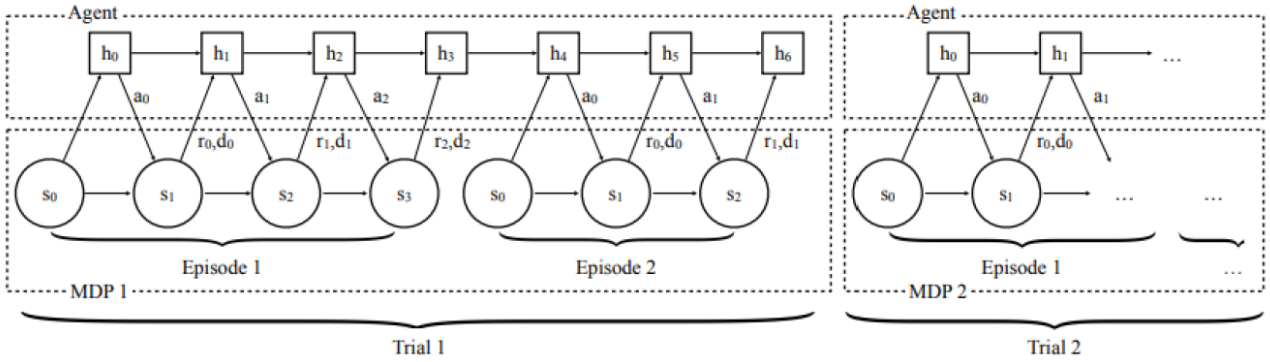
Wang, Kurth-Nelson, Tirumala, Soyer, Leibo, Munos, Blundell, Kumaran, Botvinick. **Learning to Reinforcement Learning.** 2016.



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

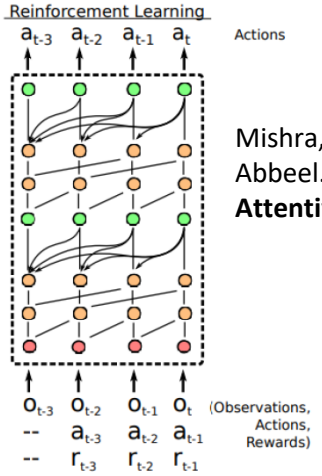
基于递归策略的元强化学习

- 常见的结构
  - 标准的RNN (LSTM) 结构



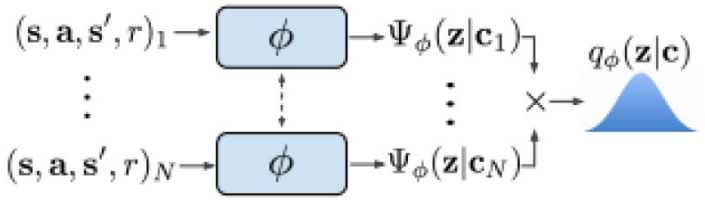
Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel. **RL2: Fast Reinforcement Learning via Slow Reinforcement Learning.** 2016.

  - 注意力+时序卷积
    attention + temporal covolution



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

  - 并行枚举不变的上下文编码器
    parallel permutation-invariant context encoder

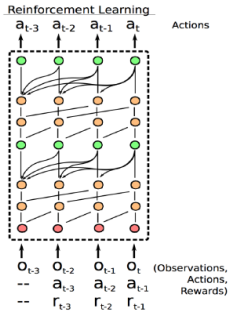


Rakelly*, Zhou*, Quillen, Finn, Levine. **Efficient Off-Policy Meta-Reinforcement learning via Probabilistic Context Variables.**

Meta-RL Example

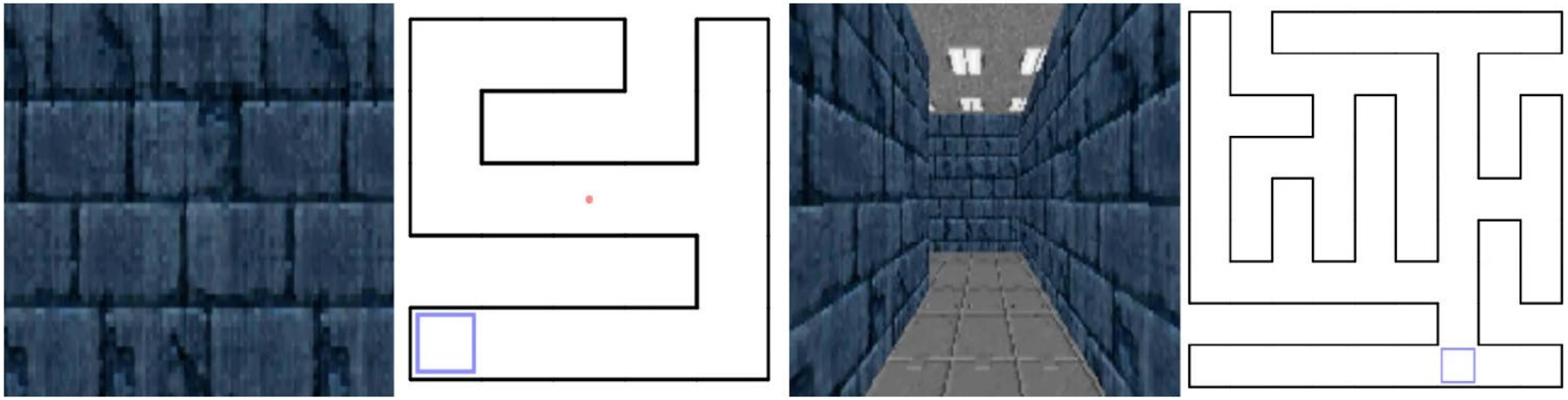
Attention + 1D conv

TRPO (on-policy)



Mishra, Rohaninejad, Chen, Abbeel. *A Simple Neural Attentive Meta-Learner*. ICLR 2018

- Mishra, Rohaninejad, Chen, Abbeel. *A Simple Neural Attentive Meta-Learner*. ICLR 2018



  - 实验：学习基于视觉在迷宫中进行导航
    - 在1000个小迷宫上训练
    - 在未见过的小地图以及大地图上测试

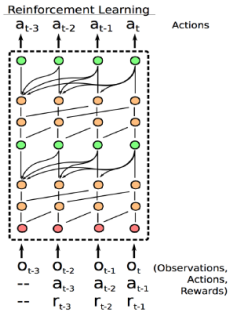
https://sites.google.com/view/snail-iclr-2018/

Meta-RL Example

Attention + 1D conv

TRPO (on-policy)

Mishra, Rohaninejad, Chen, Abbeel. *A Simple Neural Attentive Meta-Learner*. ICLR 2018



- Mishra, Rohaninejad, Chen, Abbeel. *A Simple Neural Attentive Meta-Learner*. ICLR 2018

Method	Small Maze		Large Maze	
	Episode 1	Episode 2	Episode 1	Episode 2
Random	188.6 ± 3.5	187.7 ± 3.5	420.2 ± 1.2	420.8 ± 1.2
LSTM	52.4 ± 1.3	39.1 ± 0.9	180.1 ± 6.0	150.6 ± 5.9
SNAIL (ours)	$\mathbf{50.3 \pm 0.3}$	$\mathbf{34.8 \pm 0.2}$	$\mathbf{140.5 \pm 4.2}$	$\mathbf{105.9 \pm 2.4}$

Table 5: Average time to find the goal on each episode

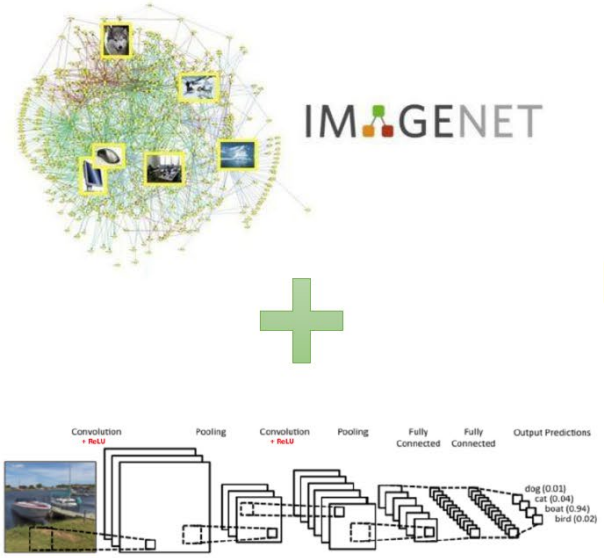
  - 实验：学习基于视觉在迷宫中进行导航
    - 在1000个小迷宫上训练
    - 在未见过的小地图以及大地图上测试

https://sites.google.com/view/snail-iclr-2018/

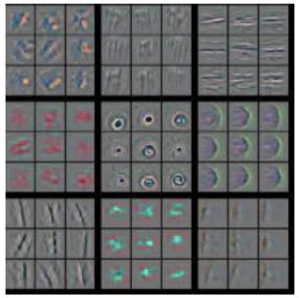
基于梯度的元强化学习

Gradient Based Meta Learning

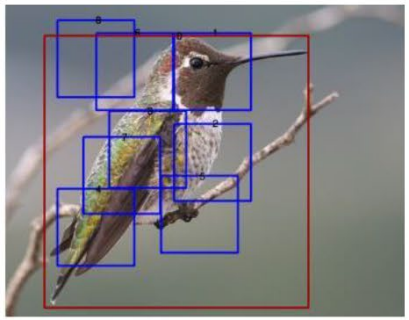
图像表征



海量数据预训练



图像特征表征



下游微调

- 预训练 pretraining 是一种元学习 meta-learning ？
- 更好的特征 → 在新任务上更快的学习！

元强化学习作为优化问题

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

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

1. 利用 $\mathcal{M}_i$ 产生的经验，学习提升策略

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

- 如果 $f_\theta(\mathcal{M}_i)$ 本身代表了一个RL算法：

一个标准的RL过程：

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

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

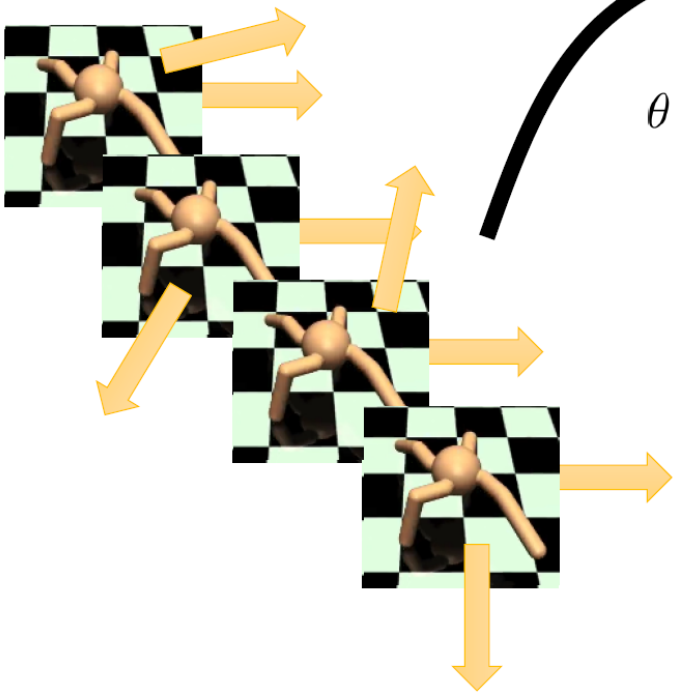
与 $\mathcal{M}_i$ 交互产生数据，评估策略梯度

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

找到一个好的初始策略参数，在少量RL优化下即可快速达到MDP的最优解

Model-Agnostic Meta-Learning (MAML)

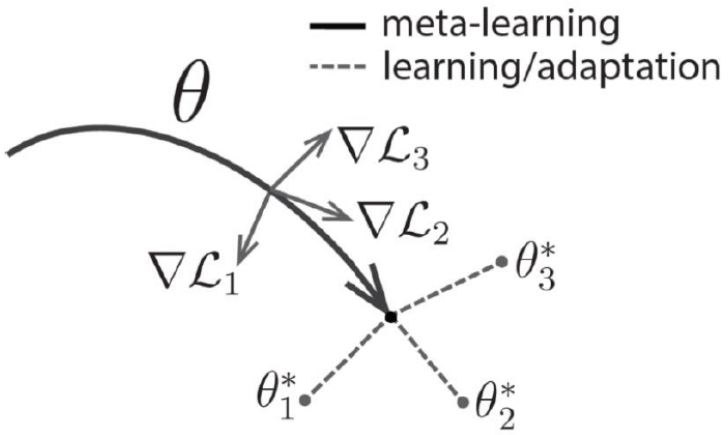
Finn, Abbeel, Levine. **Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks.**



θ

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

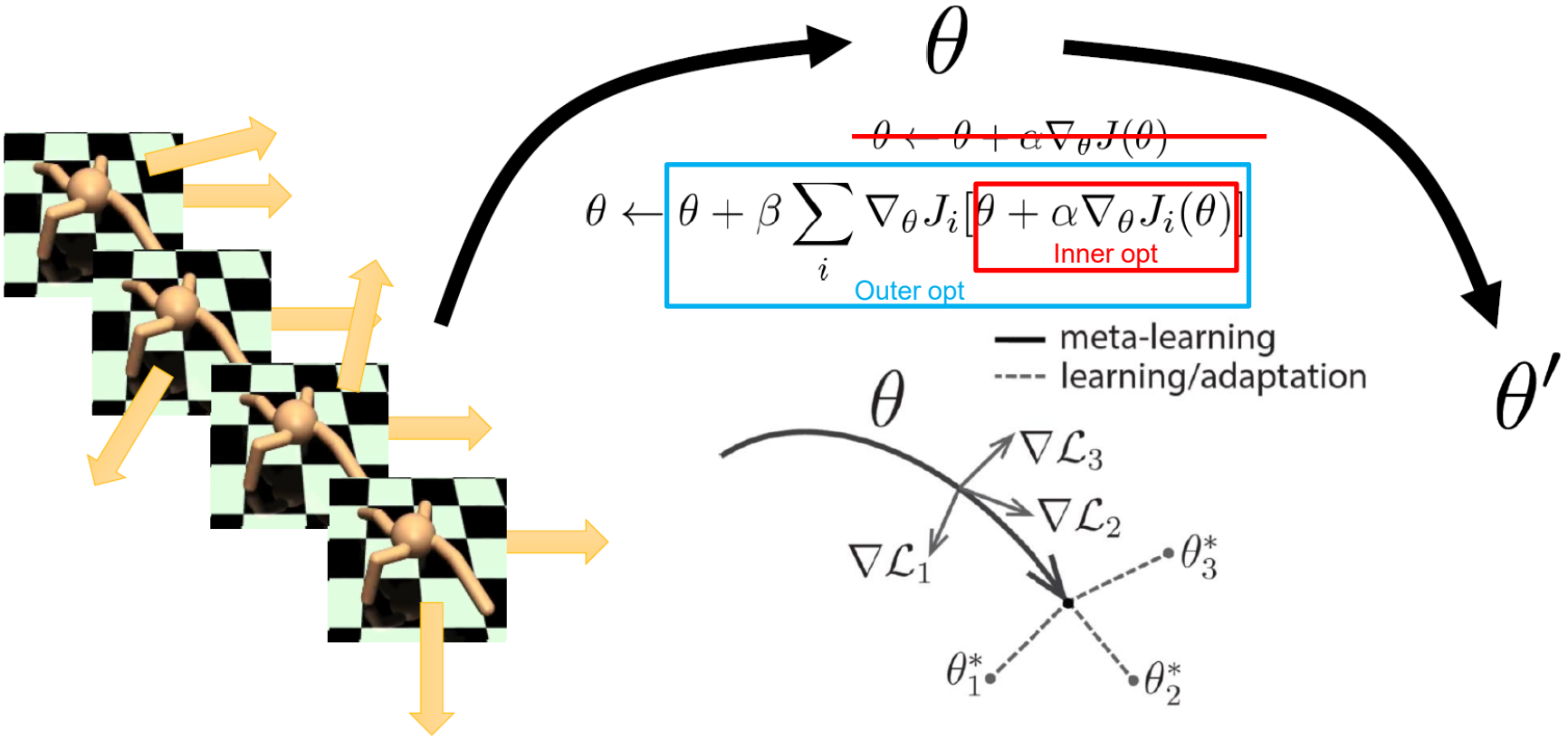
$$\theta \leftarrow \theta + \beta \sum_i \nabla_\theta J_i[\theta + \alpha \nabla_\theta J_i(\theta)]$$



θ'

Model-Agnostic Meta-Learning (MAML)

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



MAML+policy gradients

MAML: $\min_{\theta} \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}_i^{\text{tr}}), \mathcal{D}_i^{\text{ts}})$

Policy Gradient: $\nabla_{\theta} J_i(\theta) = E_{\tau \sim \pi_{\theta}, \mathcal{T}_i} \left[\left(\sum_t \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t) \right) \left(\sum_t r_i(\mathbf{s}_t, \mathbf{a}_t) \right) \right]$

- Meta-Training

1. 采样 MDP $\mathcal{M}_i$
2. 运行策略 π_{θ} 收集数据 $\mathcal{D}_i^{\text{tr}}$
3. 内部优化更新： $\phi_i = \theta + \alpha \nabla_{\theta} J_i(\theta)$
4. 运行策略 π_{ϕ_i} 收集数据 $\mathcal{D}_i^{ts}$
5. 外部优化更新： $\theta \leftarrow \theta + \beta \sum_{\text{task } i} \nabla_{\theta} J_i(\phi_i)$

- Meta-Testing

1. 采样新的 MDP $\mathcal{M}_j$
2. 运行策略 π_{θ} 收集数据 $\mathcal{D}_i^{\text{tr}}$
3. 优化更新策略： $\phi_j = \theta + \alpha \nabla_{\theta} J_j(\theta)$
4. 测试评估 π_{ϕ_j}

Model-Agnostic Meta-Learning (MAML)

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

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

- 1-step inner loop:

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

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

- 2-step inner loop:

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

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

- 3-step inner loop
- …

MAML for supervised learning

- Supervised learning: $f(x) \rightarrow y$
- Supervised meta-learning: $f(\mathcal{D}^{\text{tr}}, x) \rightarrow y$
- Model-agnostic meta-learning: $f_{\text{MAML}}(\mathcal{D}^{\text{tr}}, x) \rightarrow y$

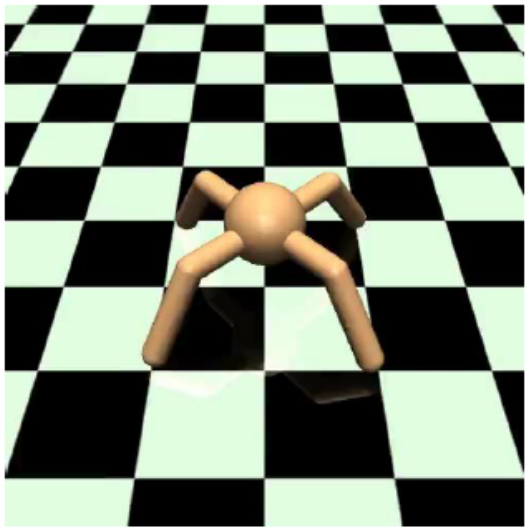
$$f_{\text{MAML}}(\mathcal{D}^{\text{tr}}, x) = f_{\theta'}(x)$$

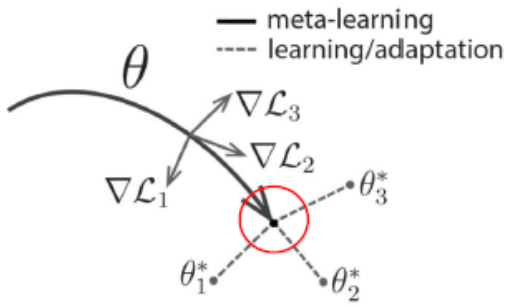
$$\theta' = \theta - \alpha \sum_{(x,y) \in \mathcal{D}^{\text{tr}}} \nabla_\theta \mathcal{L}(f_\theta(x), y)$$

- 利用如Tensorflow等autodiff库，建立 computation graph，自动计算相关梯度进行优化

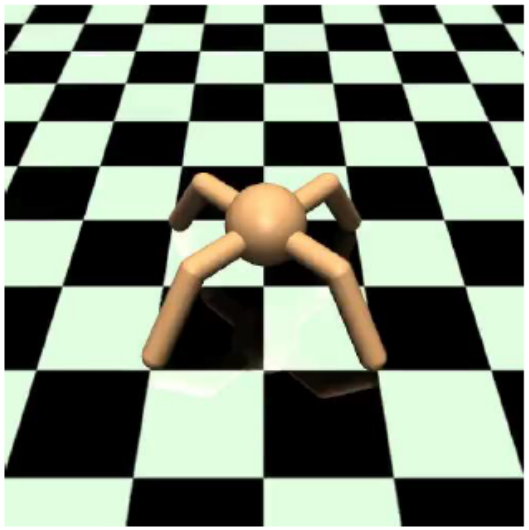
MAML for RL

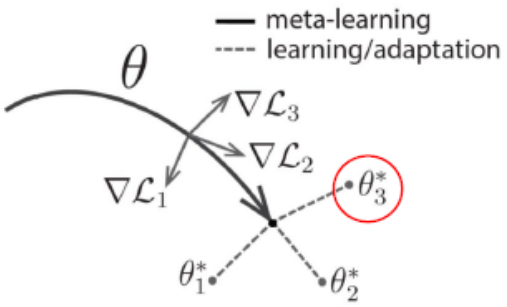
After MAML training



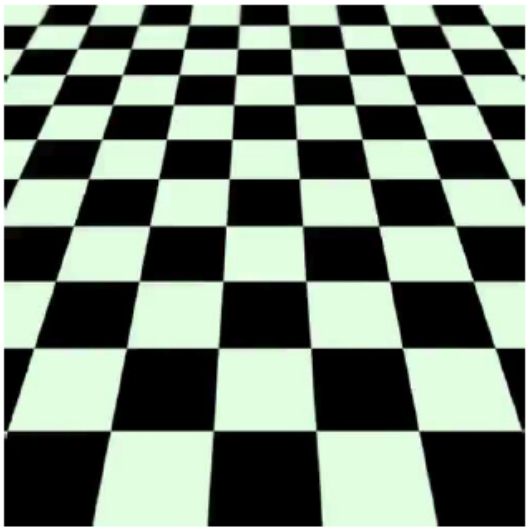


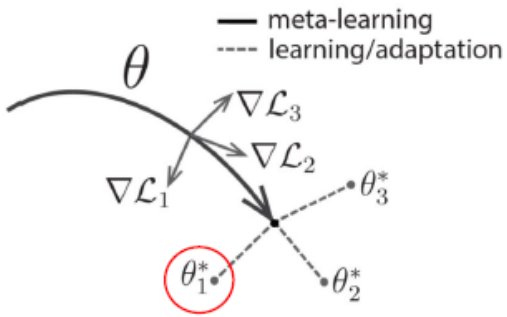
After 1 gradient step
(前向移动任务)





After 1 gradient step
(后向移动任务)

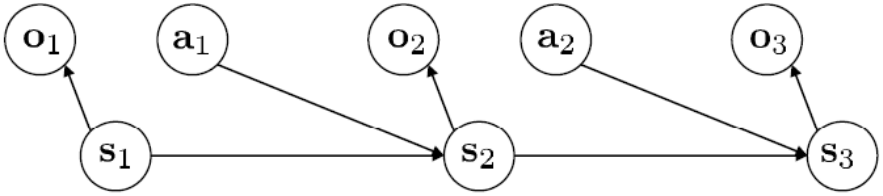




Meta-RL as a POMDP

部分可观测马尔可夫决策过程 POMDP

- MDP: $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$
- POMDP: $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{E}, r\}$



- $\mathcal{O}$ — 观测空间， 观测 $o \in \mathcal{O}$ （离散或连续）
- $\mathcal{E}$ — emission probability $p(o_t|s_t)$ 在s_t 状态下智能体观测到的 o_t 概率
- 智能体策略只基于观测决定动作：$\pi_\theta(a|o)$
- 通常是使用两种形式：
  - 根据观测估计状态，$p(\hat{s}_t|o_{1:t}) \rightarrow \pi(a|\hat{s})$
  - 策略依赖记忆，$\pi(a_t|o_0, a_0, r_0, o_1, a_1, r_1, \ldots, o_t)$

Meta-RL看作POMDP问题

- 回顾：Meta-RL 使用 基于上下文的策略

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

看作global state $\tilde{s}$

  - z 包含策略为解决当前任务所需的信息
    - 迷宫中的目标出口位置 p_{exit}，机器人的前进方向与速度 v_{desire}
  - 对某一任务的学习 == 从上下文 $(s_1,a_1,s_2,r_1),(s_2,a_2,s_3,r_2),\ldots$ 中推断 z

- 将meta-RL构造为一个POMDP问题

原始MDP

$$\mathcal{M}_i=\{\mathcal{S},\mathcal{A},\mathcal{P}(z),r(z)\}$$

Task information

新的POMDP

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

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

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

- 关键：通过对POMDP $\tilde{\mathcal{M}}$ 的求解，实现元强化学习

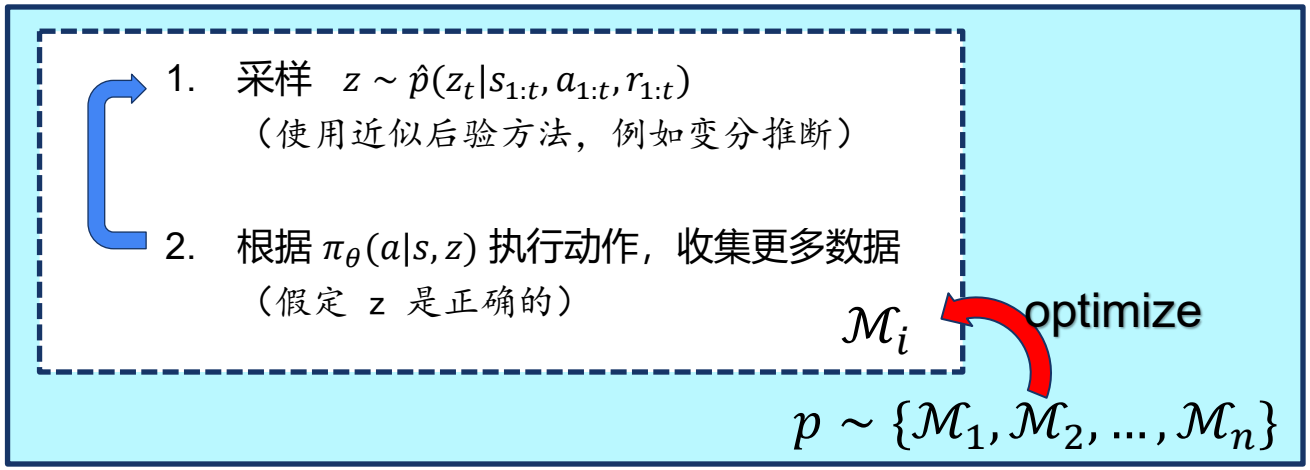
Meta-RL看作POMDP问题

- 回顾：Meta-RL 使用 基于上下文的策略

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

  - z 包含策略为解决当前任务所需的信息
    - 迷宫中的目标出口位置 p_{exit}，机器人的前进方向与速度 v_{desire}
  - 对某一任务的学习 == 从上下文 $(s_1,a_1,s_2,r_1),(s_2,a_2,s_3,r_2),\ldots$ 中推断 z

- 从观测中推断出隐含的上下文 z: $p(z_t|s_{1:t},a_{1:t},r_{1:t})$



基于变分推断的Meta-RL

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

by ϕ, that estimates the posterior $p(\mathbf{z}|\mathbf{c})$. In a generative approach, this can be achieved by optimizing $q_\phi(\mathbf{z}|\mathbf{c})$ to reconstruct the MDP by learning a predictive models of reward and dynamics. Alternatively, $q_\phi(\mathbf{z}|\mathbf{c})$ can be optimized in a model-free manner to model the state-action value functions or to maximize returns through the policy over the distribution of tasks. Assuming this objective to be

- 策略：$\pi_\theta(a_t|s_t, z_t)$
- 推断网络：$q_\phi(z_t|s_1, a_1, r_1, \ldots, s_t, a_t, r_t)$
- 优化目标：

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

最大化回报/价值目标　　逼近先验分布

- 与基于RNN的meta-RL相似，但是RNN使用的是**确定性的**隐含变量，这里使用**随机变量** z
- 通过后验采样的随机 z 有助于对任务的探索

典型示例：PEARL

- 策略：$\pi_\theta(a_t|s_t, z_t)$
- 推断网络：$q_\phi(z_t|s_1, a_1, r_1, \ldots, s_t, a_t, r_t)$
  - 关于transition样本具有枚举不变 permutation invariant 的推断函数

$$q_\phi(\mathbf{z}|\mathbf{c}_{1:N}) \propto \Pi_{n=1}^{N} \Psi_\phi(\mathbf{z}|\mathbf{c}_n)$$

  - 每一时刻的样本使用高斯分布

$$\Psi_\phi(\mathbf{z}|\mathbf{c}_n) = \mathcal{N}(f_\phi^\mu(\mathbf{c}_n), f_\phi^\sigma(\mathbf{c}_n)).$$

- 优化目标：

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

  - 使用 SAC 优化目标函数

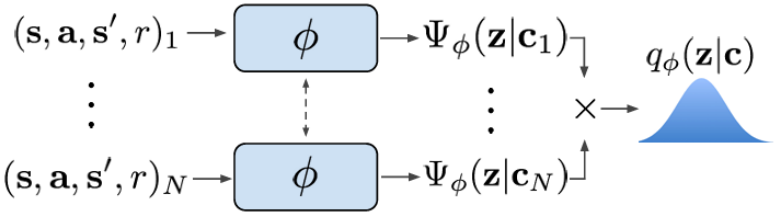


Figure 1. **Inference network architecture**. The amortized inference network predicts the posterior over the latent context variables $q_\phi(\mathbf{z}|\mathbf{c})$ as a permutation-invariant function of prior experience.

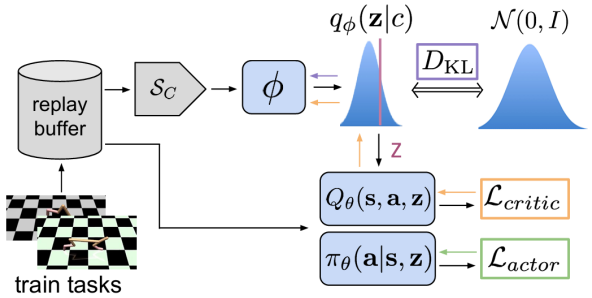
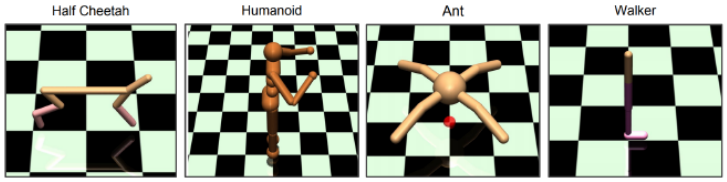


Figure 2. **Meta-training procedure.** The inference network q_ϕ uses context data to infer the posterior over the latent context variable Z, which conditions the actor and critic, and is optimized with gradients from the critic as well as from an information bottleneck on Z. De-coupling the data sampling strategies for context ($\mathcal{S}_C$) and RL batches is important for off-policy learning.

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

典型示例：PEARL

- 实验：连续控制问题



  - 不同方向、速度
  - 不同的动力学

- Meta-RL在新任务上表现高效

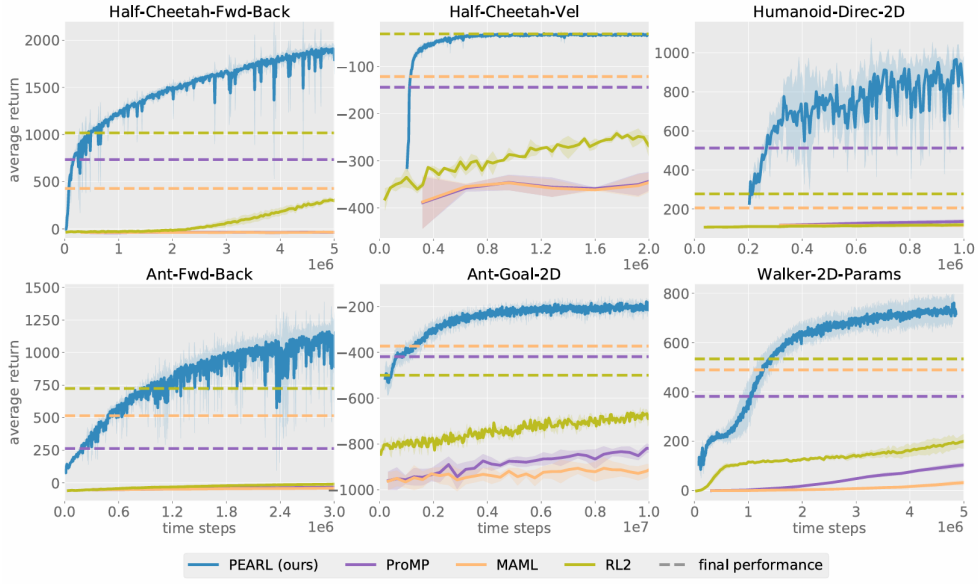


Figure 3. **Meta-learning continuous control**. Test-task performance vs. samples collected during *meta-training*. Our approach PEARL outperforms previous meta-RL methods both in terms of asymptotic performance and meta-training sample efficiency across six benchmark tasks. Dashed lines correspond to the maximum return achieved by each baseline after 1e8 steps. By leveraging off-policy data during meta-training, PEARL is $20-100$x more sample efficient than the baselines, and achieves consistently better or equal final performance compared to the best performing prior method in each environment. See Appendix A for the full timescale version of this plot.

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

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

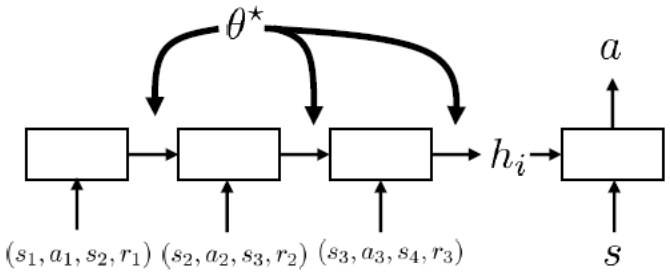
$$\text{where } \phi_i = f_\theta(\mathcal{M}_i)$$

$f_\theta(\mathcal{M}_i)$ 需要：

1. 利用 $\mathcal{M}_i$ 产生的经验，学习提升策略
2. 选择动作 a_t，决定如何与 $\mathcal{M}_i$ 交互/探索

Meta-RL方法总结

- 方法一：基于RNN



- 方法二：双层 (bi-level) 优化
  - MAML for RL

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

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

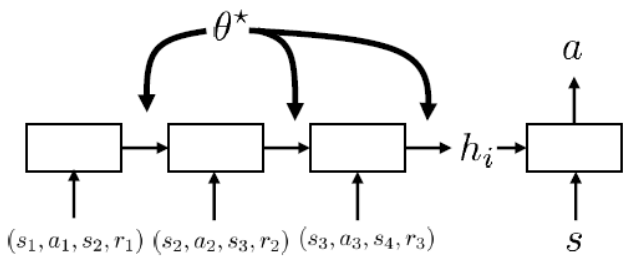
- 方法三：POMDP的推断问题

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

everything needed to solve task

Meta-RL方法总结

- 方法一：基于RNN



+ 概念上简单
+ 相对容易实现
- 易受 meta-overfitting 影响
- 实际场景难以优化

- 方法二：双层 (bi-level) 优化
  - MAML for RL

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

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

+ 良好的外推能力 (“一致性”)
+ 概念上精巧
- 复杂，需要大量样本

- 方法三：POMDP的推断问题

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

everything needed to solve task

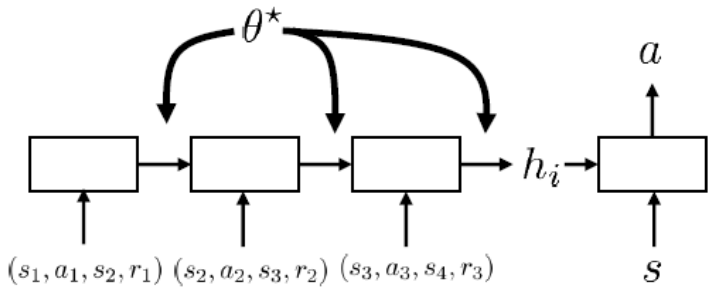
+ 结构简单，通过后验采样能够高效探索
+ 问题简化为解决特殊的POMDP问题
- 易受 meta-overfitting 影响
- 实际场景难以优化

Meta-RL方法总结

- 三种方法存在相通的地方：

等同于方法一，但是使用随机的隐含变量，即 $\phi = z$

- 方法一：基于RNN



- 方法二：双层 (bi-level) 优化

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

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

- 方法三：POMDP的推断问题

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

everything needed to solve task

采用特殊的结构