Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables

Kate Rakelly [1 *] **Aurick Zhou** [1 *] **Deirdre Quillen** [1] **Chelsea Finn** [1] **Sergey Levine** [1]

Abstract

Deep reinforcement learning algorithms require large amounts of experience to learn an individual task. While in principle meta-reinforcement learning (meta-RL) algorithms enable agents to learn new skills from small amounts of experience, several major challenges preclude their practicality. Current methods rely heavily on on-policy experience, limiting their sample efficiency. The also lack mechanisms to reason about task uncertainty when adapting to new tasks, limiting their effectiveness in sparse reward problems. In this paper, we address these challenges by developing an off-policy meta-RL algorithm that disentangles task inference and control. In our approach, we perform online probabilistic filtering of latent task variables to infer how to solve a new task from small amounts of experience. This probabilistic interpretation enables posterior sampling for structured and efficient exploration. We demonstrate how to integrate these task variables with off-policy RL algorithms to achieve both meta-training and adaptation efficiency. Our method outperforms prior algorithms in sample efficiency by 20-100X as well as in asymptotic performance on several meta-RL benchmarks.

1. Introduction

The combination of reinforcement learning (RL) with powerful non-linear function approximators has led to a wide range of advances in sequential decision making problems. However, conventional RL methods learn a separate policy per task, each often requiring millions of interactions with the environment. Learning large repertoires of behaviors with such methods quickly becomes prohibitive. Fortunately, many of the problems we would like our autonomous agents to solve share common structure. For example screwing a cap on a bottle and turning a doorknob both involve grasping an object in the hand and rotating the wrist. Exploiting this structure to learn new tasks more quickly remains an open and pressing topic. Meta-learning methods learn this structure from experience by making use of large quantities of experience collected across a distribution of tasks. Once learned, these methods can adapt quickly to new tasks given a small amount of experience.

While meta-learned policies adapt to new tasks with only a few trials, during training, they require massive amounts of data drawn from a large set of distinct tasks, exacerbating the problem of sample efficiency that plagues RL algorithms. Most current meta-RL methods require on-policy data during both meta-training and adaptation (Finn et al., 2017; Wang et al., 2016; Duan et al., 2016; Mishra et al., 2018; Rothfuss et al., 2018; Houthooft et al., 2018), which makes them exceedingly inefficient during meta-training. However, making use of off-policy data for meta-RL poses new challenges. Meta-learning typically operates on the principle that meta-training time should match meta-test time - for example, an image classification meta-learner tested on classifying images from five examples should be meta-trained to take in sets of five examples and produce accurate predictions (Vinyals et al., 2016). This makes it inherently difficult to meta-train a policy to adapt using off-policy data, which is systematically different from the data the policy would see when it explores (on-policy) in a new task at meta-test time.

In this paper, we tackle the problem of efficient off-policy meta-reinforcement learning. To achieve both meta-training efficiency and rapid adaptation, we propose an approach that integrates online inference of probabilistic context variables with existing off-policy RL algorithms. Rapid adaptation requires reasoning about distributions: when exposed to a new task for the first time, the optimal meta-learned policy must carry out a stochastic exploration procedure to visit potentially rewarding states, as well as adapt to the task at hand (Gupta et al., 2018). During meta-training, we learn a probabilistic encoder that accumulates the necessary statistics from past experience into the context variables that enable the policy to perform the task. At meta-test time, when the agent is faced with an unseen task, the context variables can be sampled and held constant for the duration

[*]Equal contribution [1]EECS Department, UC Berkeley, Berkeley, CA, USA. Correspondence to: Kate Rakelly <rakelly@eecs.berkeley.edu>.

of an episode, enabling temporally-extended exploration. The collected trajectories are used to update the posterior over the context variables, achieving rapid trajectory-level adaptation. In effect, our method adapts by sampling "task hypotheses," attempting those tasks, and then evaluating whether the hypotheses were correct or not. Disentangling task inference from action makes our approach particularly amenable to off-policy meta-learning; the policy can be optimized with off-policy data while the probabilistic encoder is trained with on-policy data to minimize distribution mismatch between meta-train and meta-test.

The primary contribution of our work is an off-policy meta-RL algorithm called probabilistic embeddings for actor-critic RL (PEARL). Our method achieves excellent sample efficiency during meta-training, enables fast adaptation by accumulating experience online, and performs structured exploration by reasoning about uncertainty over tasks. In our experimental evaluation, we demonstrate state-of-the-art results with 20-100X improvement in meta-training sample efficiency and substantial increases in asymptotic performance over prior state-of-the-art on six continuous control meta-learning environments. We further examine how our model conducts structured exploration to adapt rapidly to new tasks in a 2-D navigation environment with sparse rewards. Our open-source implementation of PEARL can be found at `https://github.com/katerakelly/oyster`.

2. Related Work

Meta-learning. Our work builds on the meta-learning framework (Schmidhuber, 1987; Bengio et al., 1990; Thrun & Pratt, 1998) in the context of reinforcement learning. Recently, meta-RL methods have been developed for meta-learning dynamics models (Nagabandi et al., 2019; Sæmundsson et al., 2018) and policies (Finn et al., 2017; Duan et al., 2016; Mishra et al., 2018) that can quickly adapt to new tasks.

Recurrent (Duan et al., 2016; Wang et al., 2016) and recursive (Mishra et al., 2018) meta-RL methods adapt to new tasks by aggregating experience into a latent representation on which the policy is conditioned. These approaches can be categorized into what we will call *context-based* meta-RL methods, since a neural network is trained to take experience as input as a form of task-specific context. Similarly, our approach can also be considered context-based; however, we represent task contexts with probabilistic latent variables, enabling reasoning over task uncertainty. Instead of using recurrence, we leverage the Markov property in our permutation-invariant encoder to aggregate experience, enabling fast optimization especially for long-horizon tasks while mitigating overfitting. While prior work has studied methods that can train recurrent Q-functions with off-policy Q-learning methods, such methods have often been applied to much simpler tasks (Heess et al., 2015), and in discrete environments (Hausknecht & Stone, 2015). Indeed, our own experiments in Section 6.3 demonstrate that straightforward incorporation of recurrent policies with off-policy learning is difficult. Contextual methods have also been applied to imitation learning by conditioning the policy on a learned embedding of a demonstration and optimizing with behavior cloning (Duan et al., 2017; James et al., 2018).

In contrast to context-based methods, *gradient-based* meta-RL methods learn from aggregated experience using policy gradients (Finn et al., 2017; Stadie et al., 2018; Rothfuss et al., 2018; Xu et al., 2018a), meta-learned loss functions (Sung et al., 2017; Houthooft et al., 2018), or hyperparameters (Xu et al., 2018b). These methods focus on on-policy meta-learning. We instead focus on meta-learning from off-policy data, which is non-trivial to do with methods based on policy gradients and evolutionary optimization algorithms. Beyond substantial sample efficiency improvements, we also empirically find that our context-based method is able to reach higher asymptotic performance, in comparison to methods using policy gradients.

Outside of RL, meta-learning methods for few-shot supervised learning problems have explored a wide variety of approaches and architectures (Santoro et al., 2016; Vinyals et al., 2016; Ravi & Larochelle, 2017; Oreshkin et al., 2018). Our permutation-invariant embedding function is inspired by the embedding function of prototypical networks (Snell et al., 2017). While they use a distance metric in a learned, deterministic embedding space to classify new inputs, our embedding is probabilistic and is used to condition the behavior of an RL agent. To our knowledge, no prior work has proposed this particular embedding function for meta-RL.

Probabilistic meta-learning. Prior work has applied probabilistic models to meta-learning in both supervised and reinforcement learning domains. Hierarchical Bayesian models have been used to model few-shot learning (Fei-Fei et al., 2003; Tenenbaum, 1999), including approaches that perform gradient-based adaptation (Grant et al., 2018; Yoon et al., 2018). For supervised learning, Rusu et al. (2019); Gordon et al. (2019); Finn et al. (2018) adapt model predictions using probabilistic latent task variables inferred via amortized approximate inference. We extend this idea to off-policy meta-RL. In the context of RL, Hausman et al. (2018) also conditions the policy on inferred task variables, but the aim is to compose tasks via the embedding space, while we focus on rapid adaptation to new tasks. While we infer task variables and explore via posterior sampling, MAESN (Gupta et al., 2018) adapts by optimizing the task variables with gradient descent and explores by sampling from the prior.

Posterior sampling. In classical RL, posterior sampling (Strens, 2000; Osband et al., 2013) maintains a posterior over possible MDPs and enables temporally extended exploration by acting optimally according to a sampled MDP. Our approach can be interpreted as a meta-learned variant of this method; probabilistic context captures the current uncertainty over the task, allowing the agent to explore in new tasks in a similarly structured manner.

Partially observed MDPs. Adaptation at test time in meta-RL can be viewed as a special case of RL in a POMDP (Kaelbling et al., 1998) by including the task as the unobserved part of the state. We use a variational approach related to Igl et al. (2018) to estimate belief over the task. While they focus on solving general POMDPs, we leverage the additional structure imposed by the meta-learning problem to simplify inference, and use posterior sampling for exploration in new tasks.

3. Problem Statement

Our approach is motivated by situations in which the agent can leverage varied experiences from previous tasks to adapt quickly to the new task at hand. Sample efficiency is central to our problem statement, both in terms of the number of samples from previous experience (meta-training efficiency), and in the amount of experience required in the new task (adaptation efficiency). To achieve meta-training efficiency, we leverage off-policy RL in our approach. Adaptation efficiency requires the agent to reason about its uncertainty over tasks, particularly in sparse reward settings. To capture uncertainty in our belief over the task, we learn a probabilistic latent representation of prior experience. We formalize the problem statement in this section, formulate our approach to adaptation as probabilistic inference in Section 4, and explain how our approach can be integrated with off-policy RL algorithms in Section 5.

Similar to previous meta-RL formulations, we assume a distribution of tasks $p(\mathcal{T})$, where each task is a Markov decision process (MDP), consisting of a set of states, actions, a transition function, and a bounded reward function. We assume that the transition and reward functions are unknown, but can be sampled by taking actions in the environment. Formally, a task $\mathcal{T} = \{p(\mathbf{s}_0), p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t), r(\mathbf{s}_t, \mathbf{a}_t)\}$ consists of an initial state distribution $p(\mathbf{s}_0)$, transition distribution $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, and reward function $r(\mathbf{s}_t, \mathbf{a}_t)$. Note that this problem definition encompasses task distributions with varying transition functions (e.g., robots with different dynamics) and varying reward functions (e.g., navigating to different locations). Given a set of training tasks sampled from $p(\mathcal{T})$, the meta-training process learns a policy that adapts to the task at hand by conditioning on the history of past transitions, which we refer to as *context* $\mathbf{c}$. Let $\mathbf{c}^{\mathcal{T}}_n = (\mathbf{s}_n, \mathbf{a}_n, r_n, \mathbf{s}'_n)$ be one transition in the task $\mathcal{T}$ so that $\mathbf{c}^{\mathcal{T}}_{1:N}$ comprises the experience collected so far. At test-time, the policy must adapt to a new task drawn from $p(\mathcal{T})$.

4. Probabilistic Latent Context

We capture knowledge about how the current task should be performed in a latent probabilistic context variable Z, on which we condition the policy as $\pi_\theta(\mathbf{a}|\mathbf{s}, \mathbf{z})$ in order to adapt its behavior to the task. Meta-training consists of leveraging data from a variety of training tasks to learn to infer the value of Z from a recent history of experience in the new task, as well as optimizing the policy to solve the task given samples from the posterior over Z. In this section we describe the structure of the meta-trained inference mechanism. We address how meta-training can be performed with off-policy RL algorithms in Section 5.

4.1. Modeling and Learning Latent Contexts

To enable adaptation, the latent context Z must encode salient information about the task. Recall that $\mathbf{c}^{\mathcal{T}}_{1:N}$ comprises experience collected so far; throughout this section we will often write c for simplicity. We adopt an amortized variational inference approach (Kingma & Welling, 2014; Rezende et al., 2014; Alemi et al., 2016) to learn to infer Z. We train an *inference network* $q_\phi(\mathbf{z}|\mathbf{c})$, parameterized 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 a log-likelihood, the resulting variational lower bound is:

$$\mathbb{E}_{\mathcal{T}}[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{c}^{\mathcal{T}})}[R(\mathcal{T}, \mathbf{z}) + \beta D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{c}^{\mathcal{T}})||p(\mathbf{z}))]] \quad (1)$$

where $p(\mathbf{z})$ is a unit Gaussian prior over Z and $R(\mathcal{T}, \mathbf{z})$ could be a variety of objectives, as discussed above. The KL divergence term can also be interpreted as the result of a variational approximation to an information bottleneck (Alemi et al., 2016) that constrains the mutual information between Z and $\mathbf{c}$. Intuitively, this bottleneck constrains $\mathbf{z}$ to contain only information from the context that is necessary to adapt to the task at hand, mitigating overfitting to training tasks. While the parameters of q_ϕ are optimized during meta-training, at meta-test time the latent context for a new task is simply inferred from gathered experience.

In designing the architecture of the inference network $q_\phi(\mathbf{z}|\mathbf{c})$, we would like it to be expressive enough to capture minimal sufficient statistics of task-relevant information, without modeling irrelevant dependencies. We note that an encoding of a fully observed MDP should be permutation invariant: if we would like to infer what the task is, identify the MDP model, or train a value function, it is enough to have access to a collection of transitions $\{\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i\}$, without regard for the order in which these transitions were observed. With this observation in mind, we choose a permutation-

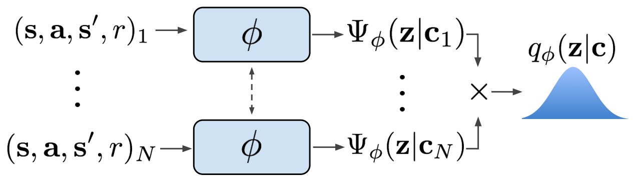
$(\mathbf{s},\mathbf{a},\mathbf{s}',r)_1$
ϕ
$\Psi_\phi(\mathbf{z}|\mathbf{c}_1)$
$(\mathbf{s},\mathbf{a},\mathbf{s}',r)_N$
ϕ
$\Psi_\phi(\mathbf{z}|\mathbf{c}_N)$
$\times$
$q_\phi(\mathbf{z}|\mathbf{c})$

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.

invariant representation for $q_\phi(\mathbf{z}|\mathbf{c}_{1:N})$, modeling it as a product of independent factors

$$q_\phi(\mathbf{z}|\mathbf{c}_{1:N}) \propto \Pi_{n=1}^{N} \Psi_\phi(\mathbf{z}|\mathbf{c}_n) \tag{2}$$

To keep the method tractable, we use Gaussian factors $\Psi_\phi(\mathbf{z}|\mathbf{c}_n) = \mathcal{N}(f_\phi^\mu(\mathbf{c}_n), f_\phi^\sigma(\mathbf{c}_n))$, which result in a Gaussian posterior. The function f_ϕ, represented as a neural network parameterized by ϕ, predicts the mean μ as well as the variance σ as a function of the $\mathbf{c}_n$, is shown in Figure 1.

4.2. Posterior Sampling and Exploration via Latent Contexts

Modeling the latent context as probabilistic allows us to make use of posterior sampling for efficient exploration at meta-test time. In classical RL, posterior sampling (Strens, 2000; Osband et al., 2013) begins with a prior distribution over MDPs, computes a posterior distribution conditioned on the experience it has seen so far, and executes the optimal policy for a sampled MDP for the duration of an episode as an efficient method for exploration. In particular, acting optimally according to a random MDP allows for temporally extended (or deep) exploration, meaning that the agent can act to test hypotheses even when the results of actions are not immediately informative of the task.

In the single-task deep RL setting, posterior sampling and the benefits of deep exploration has been explored by Osband et al. (2016), which maintains an approximate posterior over value functions via bootstraps. In contrast, our method PEARL directly infers a posterior over the latent context Z, which may encode the MDP itself if optimized for reconstruction, optimal behaviors if optimized for the policy, or the value function if optimized for a critic. Our meta-training procedure leverages training tasks to learn a prior over Z that captures the distribution over tasks and also learns to efficiently use experience to infer new tasks. At meta-test time, we initially sample $\mathbf{z}$'s from the prior and execute according to each $\mathbf{z}$ for an episode, thus exploring in a temporally extended and diverse manner. We can then use the collected experience to update our posterior and continue exploring coherently in a manner that acts more and more optimally as our belief narrows, akin to posterior sampling.

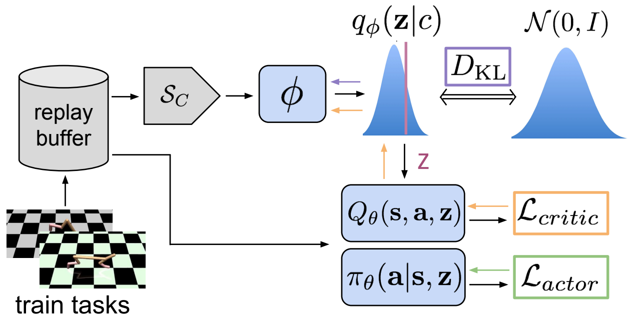

Figure 2. **Meta-training procedure.** The inference network q_ϕ uses context data to infer the posterior over the latent context variable Z, which conditions the actor and critic, and is optimized with gradients from the critic as well as from an information bottleneck on Z. De-coupling the data sampling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

5. Off-Policy Meta-Reinforcement Learning

While our probabilistic context model is straightforward to combine with on-policy policy gradient methods, a primary goal of our work is to enable efficient off-policy meta-reinforcement learning, where the number of samples for *both* meta-training and fast adaptation is minimal. The efficiency of the meta-training process is largely disregarded in prior work, which make use of stable but relatively inefficient on-policy algorithms (Duan et al., 2016; Finn et al., 2017; Gupta et al., 2018; Mishra et al., 2018). However, designing off-policy meta-RL algorithms is non-trivial partly because modern meta-learning is predicated on the assumption that the distribution of data used for adaptation will match across meta-training and meta-test. In RL, this implies that since at meta-test time on-policy data will be used to adapt, on-policy data should be used during meta-training as well. Furthermore, meta-RL requires the policy to reason about *distributions*, so as to learn effective stochastic exploration strategies. This problem inherently cannot be solved by off-policy RL methods that minimize temporal-difference error, as they do not have the ability to directly optimize for distributions of states visited. In contrast, policy gradient methods have direct control over the actions taken by the policy. Given these two challenges, a naive combination of meta-learning and value-based RL could be ineffective. In practice, we were unable to optimize such a method.

Our main insight in designing an off-policy meta-RL method with the probabilistic context in Section 4 is that the data used to train the encoder need not be the same as the data used to train the policy. The policy can treat the context $\mathbf{z}$ as part of the state in an off-policy RL loop, while the stochasticity of the exploration process is provided by the uncertainty in the encoder $q(\mathbf{z}|\mathbf{c})$. The actor and critic are always trained with off-policy data sampled from the entire replay buffer $\mathcal{B}$. We define a sampler $\mathcal{S}_\mathbf{c}$ to sample

Algorithm 1 PEARL Meta-training

```
Require: Batch of training tasks $\{\mathcal{T}_i\}_{i=1 \ldots T}$ from $p(\mathcal{T})$,
    learning rates $\alpha_1, \alpha_2, \alpha_3$
 1: Initialize replay buffers $\mathcal{B}^i$ for each training task
 2: while not done do
 3:   for each $\mathcal{T}_i$ do
 4:     Initialize context $\mathbf{c}^i = \{\}$
 5:     for $k = 1, \ldots, K$ do
 6:       Sample $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{c}^i)$
 7:       Gather data from $\pi_\theta(\mathbf{a}|\mathbf{s}, \mathbf{z})$ and add to $\mathcal{B}^i$
 8:       Update $\mathbf{c}^i = \{(\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j, r_j)\}_{j:1 \ldots N} \sim \mathcal{B}^i$
 9:     end for
10:   end for
11:   for step in training steps do
12:     for each $\mathcal{T}_i$ do
13:       Sample context $\mathbf{c}^i \sim \mathcal{S}_c(\mathcal{B}^i)$ and RL batch $b^i \sim$
          $\mathcal{B}^i$
14:       Sample $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{c}^i)$
15:       $\mathcal{L}^i_{actor} = \mathcal{L}_{actor}(b^i, \mathbf{z})$
16:       $\mathcal{L}^i_{critic} = \mathcal{L}_{critic}(b^i, \mathbf{z})$
17:       $\mathcal{L}^i_{KL} = \beta D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{c}^i)||r(\mathbf{z}))$
18:     end for
19:     $\phi \leftarrow \phi - \alpha_1 \nabla_\phi \sum_i \left(\mathcal{L}^i_{critic} + \mathcal{L}^i_{KL}\right)$
20:     $\theta_\pi \leftarrow \theta_\pi - \alpha_2 \nabla_\theta \sum_i \mathcal{L}^i_{actor}$
21:     $\theta_Q \leftarrow \theta_Q - \alpha_3 \nabla_\theta \sum_i \mathcal{L}^i_{critic}$
22:   end for
23: end while
```

context batches for training the encoder. Allowing $\mathcal{S}_\mathbf{c}$ to sample from the entire buffer presents too extreme of a distribution mismatch with on-policy test data. However, the context does not need to be strictly on-policy; we find that an in-between strategy of sampling from a replay buffer of recently collected data retains on-policy performance with better efficiency. We summarize our training procedure in Figure 2 and Algorithm 1. Meta-testing is described in Algorithm 2.

5.1. Implementation

We build our algorithm on top of the soft actor-critic algorithm (SAC) (Haarnoja et al., 2018), an off-policy actor-critic method based on the maximum entropy RL objective which augments the traditional sum of discounted returns with the entropy of the policy.

SAC exhibits good sample efficiency and stability, and further has a probabilistic interpretation which integrates well with probabilistic latent contexts. We optimize the parameters of the inference network $q(\mathbf{z}|\mathbf{c})$ jointly with the parameters of the actor $\pi_\theta(\mathbf{a}|\mathbf{s}, \mathbf{z})$ and critic $Q_\theta(\mathbf{s}, \mathbf{a}, \mathbf{z})$, using the reparameterization trick (Kingma & Welling, 2014) to compute gradients for parameters of $q_\phi(\mathbf{z}|\mathbf{c})$ through sampled $\mathbf{z}$'s. We train the inference network using gradients from

Algorithm 2 PEARL Meta-testing

```
Require: test task $\mathcal{T} \sim p(\mathcal{T})$
 1: Initialize context $\mathbf{c}^\mathcal{T} = \{\}$
 2: for $k = 1, \ldots, K$ do
 3:   Sample $z \sim q_\phi(\mathbf{z}|c^\mathcal{T})$
 4:   Roll out policy $\pi_\theta(\mathbf{a}|\mathbf{s}, \mathbf{z})$ to collect data $D^\mathcal{T}_k =$
      $\{(\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j, r_j)\}_{j:1 \ldots N}$
 5:   Accumulate context $\mathbf{c}^\mathcal{T} = \mathbf{c}^\mathcal{T} \cup D^\mathcal{T}_k$
 6: end for
```

the Bellman update for the critic. We found empirically that training the encoder to recover the state-action value function outperforms optimizing it to maximize actor returns, or reconstruct states and rewards. The critic loss can then be written as,

$$\mathcal{L}_{critic} = \mathbb{E}_{\substack{(\mathbf{s},\mathbf{a},r,\mathbf{s}') \sim \mathcal{B} \\ \mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{c})}} [Q_\theta(\mathbf{s}, \mathbf{a}, \mathbf{z}) - (r + \bar{V}(\mathbf{s}', \bar{\mathbf{z}}))]^2 \quad (3)$$

where $\bar{V}$ is a target network and $\bar{\mathbf{z}}$ indicates that gradients are not being computed through it. The actor loss is nearly identical to SAC, with the additional dependence on $\mathbf{z}$ as a policy input.

$$\mathcal{L}_{actor} = \mathbb{E}_{\substack{\mathbf{s} \sim \mathcal{B}, \mathbf{a} \sim \pi_\theta \\ \mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{c})}} \left[D_{\mathrm{KL}} \left(\pi_\theta(\mathbf{a}|\mathbf{s}, \bar{\mathbf{z}}) \middle\| \frac{\exp(Q_\theta(\mathbf{s}, \mathbf{a}, \bar{\mathbf{z}}))}{\mathcal{Z}_\theta(\mathbf{s})} \right) \right] \quad (4)$$

Note that the context used to infer $q_\phi(\mathbf{z}|\mathbf{c})$ is distinct from the data used to construct the critic loss. As described in Section 5, during meta-training we sample context batches separately from RL batches. Concretely,the context data sampler $\mathcal{S}_\mathbf{c}$ samples uniformly from the most recently collected batch of data, recollected every 1000 meta-training optimization steps. The actor and critic are trained with batches of transitions drawn uniformly from the entire replay buffer.

6. Experiments

In our experiments, we assess the performance of our method and analyze its properties. We first evaluate how our approach compares to prior meta-RL methods, especially in terms of sample efficiency, on several benchmark meta-RL problems in Section 6.1. We examine how probabilistic context and posterior sampling enable rapid adaptation via structured exploration strategies in sparse reward settings in Section 6.2. Finally, in Section 6.3, we evaluate the specific design choices in our algorithm through ablations.

6.1. Sample Efficiency and Performance

Experimental setup. We evaluate PEARL on six continuous control environments focused around robotic locomotion, simulated via the MuJoCo simulator (Todorov et al.,

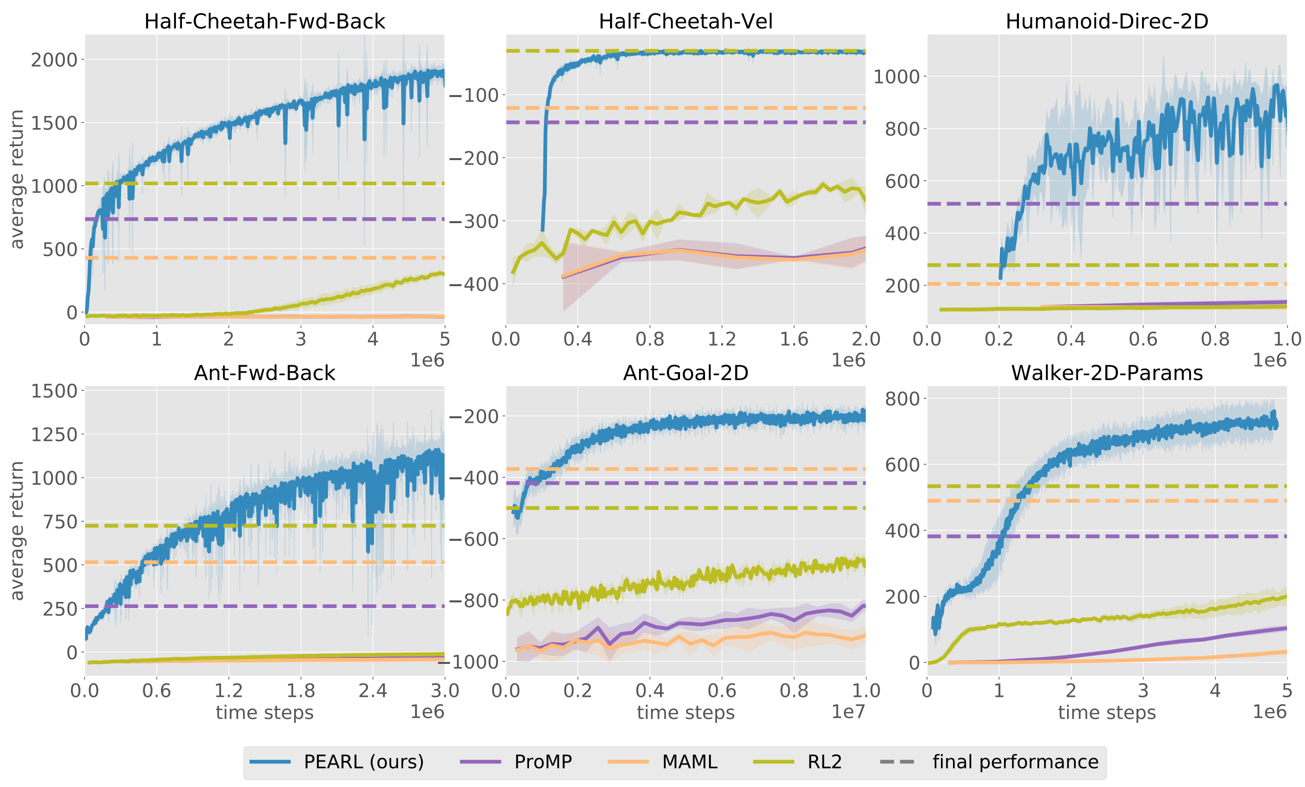

Figure 3. **Meta-learning continuous control**. Test-task performance vs. samples collected during *meta-training*. Our approach PEARL outperforms previous meta-RL methods both in terms of asymptotic performance and meta-training sample efficiency across six benchmark tasks. Dashed lines correspond to 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.

2012). These locomotion task families require adaptation across reward functions (walking direction for Half-Cheetah-Fwd-Back, Ant-Fwd-Back, Humanoid-Direc-2D, target velocity for Half-Cheetah-Vel, and goal location for Ant-Goal-2D) or across dynamics (random system parameters for Walker-2D-Params). These meta-RL benchmarks were previously introduced by Finn et al. (2017) and Rothfuss et al. (2018). All tasks have horizon length 200. We compare to existing policy gradient meta-RL methods ProMP (Rothfuss et al., 2018) and MAML-TRPO (Finn et al., 2017) using publicly available code. We also re-implement the recurrence-based policy gradient RL^2 method (Duan et al., 2016) with PPO (Schulman et al., 2017). The results of each algorithm are averaged across three random seeds. We attempted to adapt recurrent DDPG (Heess et al., 2015) to our setting, but were unable to obtain reasonable results with this method. We hypothesize that this is due to a combination of factors including the distribution mismatch in the adaptation data discussed in Section 5 and the difficulty of training with trajectories rather than decorrelated transitions. This approach does not explicitly infer a belief over the task as we do, instead leaving the burden of both task inference and optimal behavior to the RNN. In PEARL, decoupling task inference from the policy allows us the freedom to choose the encoder data and objective that work best with off-policy learning. We experiment with recurrent architectures in the context of our own method in Section 6.3.

Results. To evaluate on the meta-testing tasks, we perform adaptation at the trajectory level, where the first trajectory is collected with context variable $\mathbf{z}$ sampled from the prior $r(\mathbf{z})$. Subsequent trajectories are collected with $\mathbf{z} \sim q(\mathbf{z}|\mathbf{c})$ where the context is aggregated over all trajectories collected. To compute final test-time performance, we report the average returns of trajectories collected after two trajectories have been aggregated into the context. Notably, we find RL^2 to perform much better on these benchmarks than previously reported, possibly due to using PPO for optimization and selecting better hyper-parameters. We observe that PEARL significantly outperforms prior meta-RL methods across all domains in terms of both asymptotic performance and sample efficiency, as shown in Figure 3. Here we truncate the x-axis at the number of timesteps re-

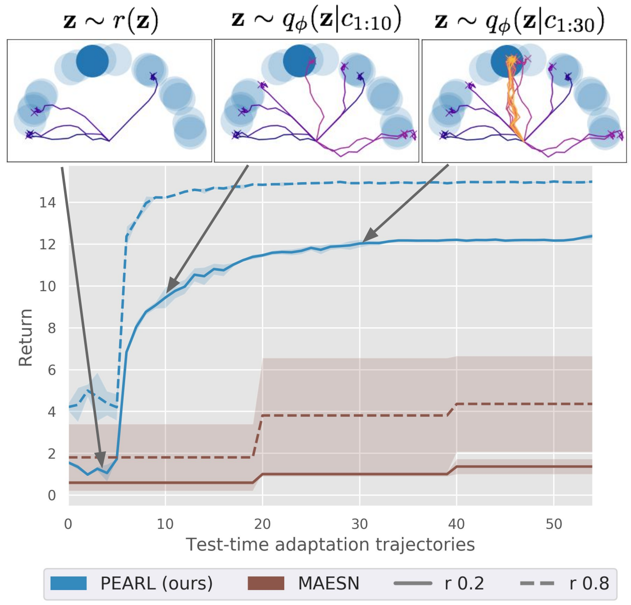

Figure 4. **Sparse 2D navigation**. The agent must navigate to a previously unseen goal (dark blue, other test goals in light blue) with reward given only when inside the goal radius – radius of 0.2 (illustrated) and 0.8 are tested here. The agent is trained to navigate to a training set of goals, then tested on a distinct set of unseen test goals. By using posterior sampling to explore efficiently, PEARL is able to start adapting to the task after collecting on average only 5 trajectories, outperforming MAESN (Gupta et al., 2018).

quired for PEARL to converge; see Appendix A for the full timescale version of this plot. We find that PEARL uses 20-100x fewer samples during meta-training than previous meta-RL approaches while improving final asymptotic performance by 50-100% in five of the six domains.

6.2. Posterior Sampling For Exploration

In this section we evaluate whether posterior sampling in our model enables effective exploration strategies in sparse reward MDPs. Intuitively, by sampling from the prior context distribution $r(\mathbf{z})$, the agent samples a hypothesis according to the distribution of training tasks it has seen before. As the agent acts in the environment, the context posterior $p(\mathbf{z}|\mathbf{c})$ is updated, allowing it to reason over multiple hypotheses to determine the task. We demonstrate this behavior with a 2-D navigation task in which a point robot must navigate to different goal locations on edge of a semi-circle. We sample training and testing sets of tasks, each consisting of 100 randomly sampled goals. A reward is given only when the agent is within a certain radius of the goal. We experiment with radius 0.2 and 0.8. While our aim is to adapt to new tasks with sparse rewards, meta-training with sparse rewards is extremely difficult as it amounts to solving many sparse reward tasks from scratch. For simplicity we therefore assume access to the dense reward during meta-training, as done by Gupta et al. (2018), but this burden could also be mitigated with task-agnostic exploration strategies.

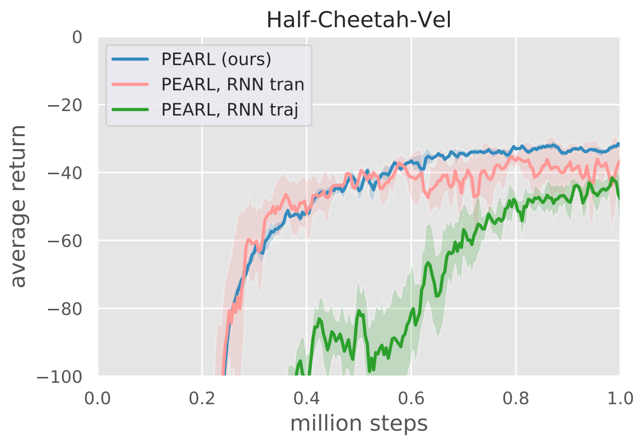

Figure 5. **Recurrent encoder ablation**. We compare our encoder architecture to a recurrent network. We sample context as trajectories rather than unordered transitions. Sampling the RL batch as de-correlated transitions ("RNN tran") fares much better than sampling trajectories ("RNN traj").

In this setting, we compare to MAESN (Gupta et al., 2018), a prior method that also models probabilistic task variables and performs on-policy gradient-based meta-learning. We demonstrate we are able to adapt to the new sparse goal in fewer trajectories. Even with fewer samples, PEARL also outperforms MAESN in terms of final performance. In Figure 4 we compare adaptation performance on test tasks. In addition to achieving higher returns and adapting faster, PEARL is also more efficient during meta-training. Our results were achieved with $\sim 1e6$ timesteps while MAESN uses $\sim 1e8$ timesteps.

6.3. Ablations

In this section we ablate the features of our approach to better understand the salient features of our method.

Inference network architecture. We examine our choice of permutation-invariant encoder for the latent context Z by comparing it to a conventional choice for encoding MDPs, a recurrent network (Duan et al., 2016; Heess et al., 2015). Note that while in Section 6.1 we considered a recurrent-based baseline similar to recurrent DDPG (Heess et al., 2015), here we retain all other features of our method and ablate only the encoder structure. We backprop through the RNN to 100 timesteps. We sample the context as full trajectories rather than unordered transitions as in PEARL. We experiment with two options for sampling the RL batch:

- unordered transitions as in PEARL ("RNN tran")
- sets of trajectories ("RNN traj")

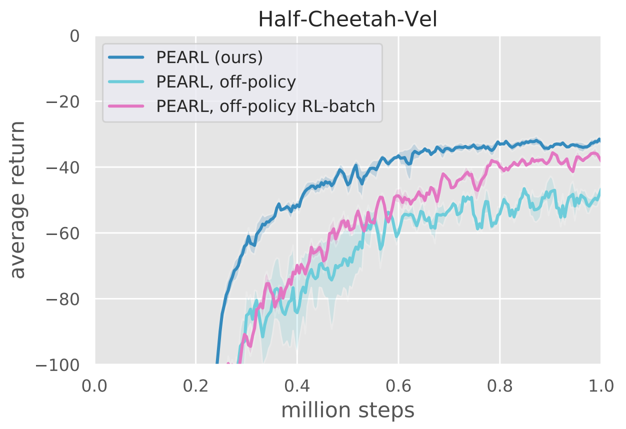

Figure 6. **Context sampling ablation**. PEARL samples context batches of recently collected transitions de-correlated with the batches sampled for RL. We compare to sampling context from the entire history ("off-policy"), as well as using the same sampled batch for the context and the RL batch ("off-policy RL").

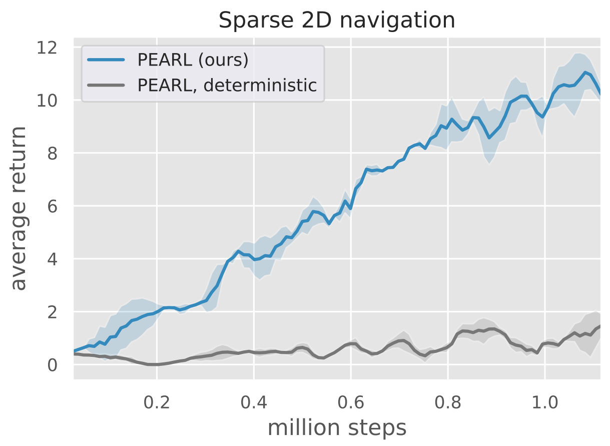

Figure 7. **Deterministic latent context**. We compare PEARL to a variant with deterministic latent context on the sparse reward 2D navigation domain. As expected, without a mechanism for reasoning about uncertainty over tasks, this approach is unable to explore effectively and performs poorly.

In Figure 5, we compare the test task performance in the Half-Cheetah-Vel domain as a function of the number of meta-training samples. Replacing our encoder with an RNN results in comparable performance to PEARL, at the cost of slower optimization. However, sampling trajectories for the RL batch results in a steep drop in performance. This result demonstrates the importance of decorrelating the samples used for the RL objective.

Data sampling strategies. In our next experiment, we ablate the context sampling strategy used during training. With sampler $\mathcal{S}_\mathbf{c}$, PEARL samples batches of unordered transitions that are **(1)** restricted to samples recently collected by the policy, and **(2)** distinct from the set of transitions collected by the RL mini-batch sampler. We consider two other options for $\mathcal{S}_\mathbf{c}$:

- sample fully off-policy data from the entire replay buffer, but distinct from the RL batch ("off-policy")
- use the same off-policy RL batch as the context ("off-policy RL-batch")

Results are shown in Figure 6. Sampling context off-policy significantly hurts performance. Using the same batch for RL and context in this case helps, perhaps because the correlation makes learning easier. Overall these results demonstrate the importance of careful data sampling in off-policy meta-RL.

Deterministic context. Finally, we examine the importance of modeling the latent context as probabilistic. As discussed in Section 4, we hypothesize that a probabilistic context is particularly important in sparse reward settings because it allows the agent to model a distribution over tasks and conduct exploration via posterior sampling. To test this empirically, we train a deterministic version of PEARL by reducing the distribution $q_\phi(\mathbf{z}|\mathbf{c})$ to a point estimate. We compare probabilistic and deterministic context on the sparse navigation domain in Figure 7. With no stochasticity in the latent context variable, the only stochasticity comes from the policy and is thus time-invariant, hindering temporally extended exploration. As a result this approach is unable to solve a sparse reward navigation task.

7. Conclusion

In this paper, we propose a novel meta-RL algorithm, PEARL, which adapts by performing inference over a latent context variable on which the policy is conditioned. Our approach is particularly amenable to off-policy RL algorithms as it decouples the problems of inferring the task and solving it, allowing for off-policy meta-training while minimizing mismatch between train and test context distributions. Modeling the context as probabilistic enables posterior sampling for exploration at test time, resulting in temporally extended exploration behaviors that enhance adaptation efficiency. Our approach obtains superior results compared to prior meta-RL algorithms while requiring far less experience on a diverse set of continuous control meta-RL domains.

Acknowledgements We gratefully acknowledge Ignasi Clavera for running the MAML, ProMP, and RL^2 baselines. We thank Ignasi Clavera, Abhishek Gupta, and Rowan McAllister for insightful discussions, and Coline Devin, Kelvin Xu, Vitchyr Pong, and Karol Hausman for feedback on early drafts. This research was supported by NSF IIS-1651843 and IIS-1614653, Berkeley DeepDrive, the Office of Naval Research, ARL DCIST CRA W911NF-17-2-0181, Amazon, Google, and NVIDIA.

References

Alemi, A. A., Fischer, I., Dillon, J. V., and Murphy, K. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

Bengio, Y., Bengio, S., and Cloutier, J. *Learning a synaptic learning rule*. Université de Montréal, 1990.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. Rl ^{2}: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Duan, Y., Andrychowicz, M., Stadie, B., Ho, O. J., Schneider, J., Sutskever, I., Abbeel, P., and Zaremba, W. One-shot imitation learning. In *Advances in neural information processing systems*, 2017.

Fei-Fei, L. et al. A bayesian approach to unsupervised one-shot learning of object categories. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 1134–1141. IEEE, 2003.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

Finn, C., Xu, K., and Levine, S. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pp. 9537–9548, 2018.

Gordon, J., Bronskill, J., Bauer, M., Nowozin, S., and Turner, R. Meta-learning probabilistic inference for prediction. In *International Conference on Learning Representations*, 2019.

Grant, E., Finn, C., Levine, S., Darrell, T., and Griffiths, T. Recasting gradient-based meta-learning as hierarchical bayes. In *International Conference on Learning Representations*, 2018.

Gupta, A., Mendonca, R., Liu, Y., Abbeel, P., and Levine, S. Meta-reinforcement learning of structured exploration strategies. In *Advances in Neural Information Processing Systems*, 2018.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, 2018.

Hausknecht, M. and Stone, P. Deep recurrent q-learning for partially observable mdps. In *2015 AAAI Fall Symposium Series*, 2015.

Hausman, K., Springenberg, J. T., Wang, Z., Heess, N., and Riedmiller, M. Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations*, 2018.

Heess, N., Hunt, J. J., Lillicrap, T. P., and Silver, D. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.

Houthooft, R., Chen, R. Y., Isola, P., Stadie, B. C., Wolski, F., Ho, J., and Abbeel, P. Evolved policy gradients. In *Neural Information Processing Systems (NIPS)*, 2018.

Igl, M., Zintgraf, L., Le, T. A., Wood, F., and Whiteson, S. Deep variational reinforcement learning for pomdps. In *International Conference on Machine Learning*, 2018.

James, S., Bloesch, M., and Davison, A. J. Task-embedded control networks for few-shot imitation learning. In *Conference on Robot Learning*, 2018.

Kaelbling, L. P., Littman, M., and Cassandra, A. Planning and acting in partially observable stochastic domains. volume 101, pp. 99–134, 1998.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018.

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., and Finn, C. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2019.

Oreshkin, B. N., Lacoste, A., and Rodriguez, P. Tadam: Task dependent adaptive metric for improved few-shot learning. 2018.

Osband, I., Van Roy, B., and Russo, D. (more) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*, 2013.

Osband, I., Blundell, C., Pritzel, A., and Van Roy, B. Deep exploration via bootstrapped dqn. In *Advances in Neural Information Processing Systems*, 2016.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

Rothfuss, J., Lee, D., Clavera, I., Asfour, T., and Abbeel, P. Promp: Proximal meta-policy search. In *International Conference on Learning Representations*, 2018.

Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., and Hadsell, R. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*, 2019.

Sæmundsson, S., Hofmann, K., and Deisenroth, M. P. Meta reinforcement learning with latent variable gaussian processes. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning*, 2016.

Schmidhuber, J. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

Schulman, J., Wolski, F., Dhariwal, P. D., Radford, A. R., and Klimov, O. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

Stadie, B. C., Yang, G., Houthooft, R., Chen, X., Duan, Y., Wu, Y., Abbeel, P., and Sutskever, I. Some considerations on learning to explore via meta-reinforcement learning. *arXiv preprint arXiv:1803.01118*, 2018.

Strens, M. A bayesian framework for reinforcement learning. In *International Conference on Machine Learning*, 2000.

Sung, F., Zhang, L., Xiang, T., Hospedales, T., and Yang, Y. Learning to learn: Meta-critic networks for sample efficient learning. *arXiv preprint arXiv:1706.09529*, 2017.

Tenenbaum, J. B. *A Bayesian framework for concept learning*. PhD thesis, Massachusetts Institute of Technology, 1999.

Thrun, S. and Pratt, L. *Learning to learn*. Springer Science & Business Media, 1998.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033, 2012.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, 2016.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

Xu, T., Liu, Q., Zhao, L., and Peng, J. Learning to explore via meta-policy gradient. In *International Conference on Machine Learning*, pp. 5459–5468, 2018a.

Xu, Z., van Hasselt, H., and Silver, D. Meta-gradient reinforcement learning. *arXiv:1805.09801*, 2018b.

Yoon, J., Kim, T., Dia, O., Kim, S., Bengio, Y., and Ahn, S. Bayesian model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pp. 7343–7353, 2018.

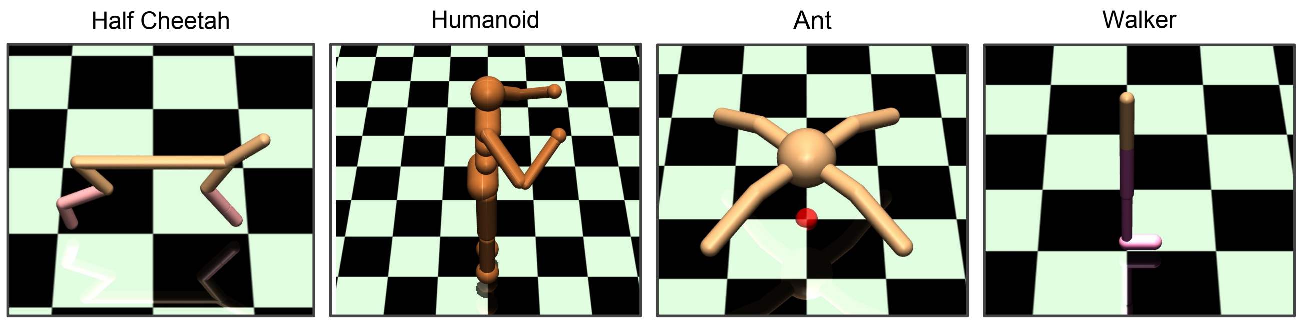

Figure 8. **Continuous control tasks**: left-to-right: the half-cheetah, humanoid, ant, and walker robots used in our evaluation.

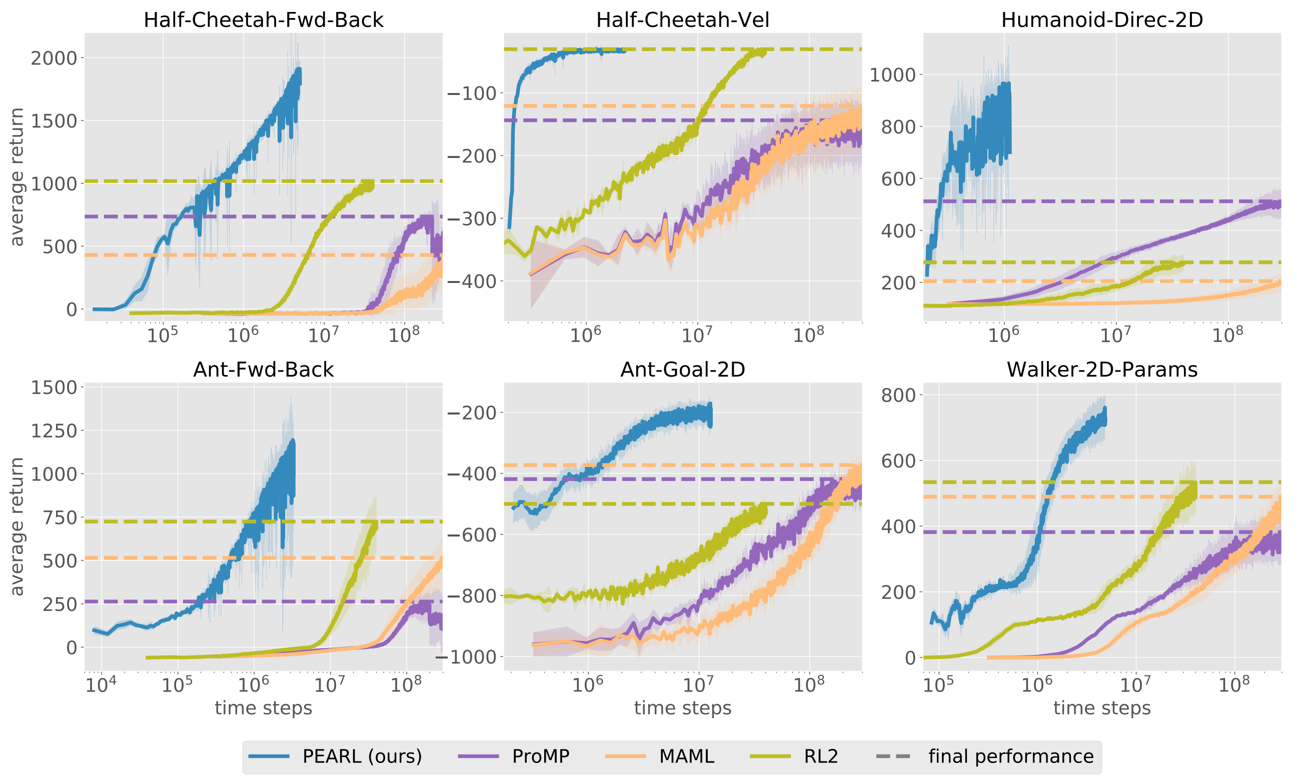

Figure 9. **Meta-learning continuous control**. Test task performance vs. samples collected during *meta-training*. While in the main paper we truncate the x-axis to better illustrate the performance of PEARL, here we plot PEARL against the on-policy methods run for the full number of time steps ($1e8$). PEARL is 20-100 times more sample efficient. Note that the x-axis is in **log scale**.

A. Experimental Details

The on-policy baseline approaches require many more samples to learn the benchmark tasks. Here we plot the same data as in Figure 3 for the full number of time steps used by the baselines, in Figure 9. The agents used in these continuous control domains are visualized in Figure 8. Here we describe each meta-learning domain.

- Half-Cheetah-Dir: move forward and backward (2 tasks)
- Half-Cheetah-Vel: achieve a target velocity running forward (100 train tasks, 30 test tasks)
- Humanoid-Dir-2D: run in a target direction on 2D grid (100 train tasks, 30 test tasks)
- Ant-Fwd-Back: move forward and backward (2 tasks)
- Ant-Goal-2D: navigate to a target goal location on 2D grid (100 train tasks, 30 test tasks)
- Walker-2D-Params: agent is initialized with some system dynamics parameters randomized and must move forward (40 train tasks, 10 test tasks)