The Reflective Explorer: Online Meta-Exploration from Offline Data in Realistic Robotic Tasks

Anonymous Author(s)

Abstract

Reinforcement learning is difficult to apply to real world problems due to high sample complexity, the need to adapt to frequent distribution shifts and the complexities of learning from high-dimensional inputs, such as images. Over the last several years, meta-learning has emerged as a promising approach to tackle these problems by explicitly training an agent to quickly adapt to new tasks. However, such methods still require huge amounts of data during training and are difficult to optimize in high-dimensional domains. One potential solution is to consider offline or batch meta-reinforcement learning (RL) - learning from existing datasets without additional environment interactions during training. In this work we develop the first offline model-based meta-RL algorithm that operates from images in tasks with sparse rewards. Our approach has three main components: a novel strategy to construct meta-exploration trajectories from offline data, which allows agents to learn meaningful meta-test time task inference strategy; representation learning via variational filtering and latent conservative model-free policy optimization. We show that our method completely solves a realistic meta-learning task involving robot manipulation, while naive combinations of previous approaches fail.

1 Introduction

There are several difficulties in applying reinforcement learning (RL) to the real world: 1) sample complexity tends to be prohibitively high, with even simple low-dimensional tasks requiring millions of interactions, 2) in some setups online exploration and data collection could be difficult, costly and unsafe for humans or the hardware, 3). the agents need to be robust to potential distributional shifts that regularly happen in the real world, 4). the agents need to operate from high-dimensional observations, such as camera images and 5). reward specification can be difficult in realistic domains. Our goal in this work is to develop an agent that can learn from previously collected datasets from realistic robotic tasks, without additional rollouts that can successfully adapt to new tasks and distribution shifts at test time.

Although such an approach has clear utility, as it would allow us to reuse previously collected data and train robot agents that can efficiently generalize to new domains, it has so far been understudied. Prior works in offline meta-reinforcement learning [19], [3] consider only simple tasks in low-dimensional domains and are hard to scale to longer range problems with high-dimensional observations and sparse rewards. More realistic robot tasks were considered before in online settings [28], however naive application of such inference models in the offline domain fails.

We identify three key challenges in this setting. Spurious correlations between tasks and the specific task data prevent the agent from learning inference models that are robust to test-time distribution shifts and new task specifications. If the induced state-action distributions in each task buffer are sufficiently disjoint across tasks, then an inference-based meta-RL agent can easily pinpoint the task simply by memorizing the action distributions of the offline data during training, which yields poor policy performance at test time. Furthermore, high-dimensional observations and sparse rewards

Submitted to 35th Conference on Neural Information Processing Systems Deep Reinforcement Learning Workshop.

present significant difficulty for reinforcement learning with task uncertainty, and finally classical reinforcement learning algorithms are not robust to test time distribution shifts when trained with limited data.

This work has two main contributions: First, we propose a new method to construct synthetic exploration strategies from fixed datasets of task-specific interactions. We also develop a method explicitly targeted to offline meta-learning from high-dimensional observations and sparse rewards, which is the first such approach to our knowledge.

2 Related Work

In the last several years meta-reinforcement learning [4, 5, 22] has emerged as a promising approach to build broadly generalizing intelligent agents that are capable of quickly adapting to novel tasks with small amounts of additional data. However, such methods still require huge amounts of data at *meta-train* time, making training them impractical in realistic domains. Meta-reinforcement learning methods fall in two broad-categories: optimization-based models such as MAML [5] use on-policy training in combination with a bi-level optimization scheme that aims to learn initial policy parameters, such that with only a few additional gradient steps the agent can achiever high rewards on novel tasks. These are inherently difficult to apply to the offline domain due to the need for on-policy data and the difficult optimization procedure. Previous attempts [19] have only focused on tasks with low-dimensional observation spaces and continuous smooth rewards.

The second broad category casts meta-RL as learning in a POMDP setup, in which the task itself is a hidden variable sampled from some task distribution [10]. Successful methods in this framework use recurrent policies [4] or specific task-embedding schemes [22, 29] in a combination with a task-conditioned policy. One significant benefit of the second class of approaches is that it's amenable to off-policy training. Previous works [3] have successfully adapted such methods to the offline setting, however they also only focus on relatively simple tasks. Due to their complex training regimes both optimization-based and inference-based approaches are difficult to train from high-dimensional observations, such as images. To tackle this issue [28] adopts the task inference view of meta-learning and trains a deep variational filter with image and reward reconstruction, which jointly models uncertainty over the agent's environment state and task. Although the authors report state of the art performance, this method still requires millions of environment steps to achieve expert-level performance in the online setting. On the other hand, recently batch or offline reinforcement learning has emerged as alternative in addressing sample complexity issues with online reinforcement learning [18]. In this setting the agent has access to a fixed dataset of environment interactions and does not carry out additional exploration. The goal of this framework is to reuse previously collected datasets, which could come from a variety of distributions, such as unsupervised exploration, previous runs of agent training and expert level demonstrations. These approaches are especially relevant to real-world scenarios, such as autonomous driving, medical applications and financial systems. There have been efforts to scale these methods to more realistic visual domains [21, 27, 15], but these works only focus on a single task approach and do not consider transfer to new tasks or distribution shifts.

3 Preliminaries

We begin by reviewing necessary preliminaries from meta-learning and offline reinforcement learning.

3.1 Meta Reinforcement Learning as Task Inference

A Markov Decision Process (MDP) is a tuple $\mathcal{T} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma)$, where $s \in \mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(s'|s, a)$ is the transition model, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function and γ is a discount factor. The meta-RL problem assumes a distribution over MDPs - $p(\mathcal{T})$ with samples from the distribution referred to as tasks. At train time we can sample training tasks from the distribution and the goal is to learn an agent that can quickly adapt to a new task sampled from the same distribution. Several works [10, 22, 29] cast this problem in the frame of partially observed MDPs (POMDP). A POMDP is a 7-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \Omega, \mathcal{O}, \gamma)$, where in addition to the components of an MDP, we have the observation space $o \in \Omega$ and the observation model $\mathcal{O}(o|s)$. We can then consider meta-learning as learning in the POMDP, with state (s, w), where w are some task parameters and the observation model is $\mathcal{O}(s, w) = s$. Several methods have deployed techniques

from POMDP training to the meta-learning setting, such as recurrent policies, which integrate agent observations and build an internal task representation [4] or explicitly construct a task inference network that produces a task embedding in combination with a task-conditioned feedforward policy [22, 29]. State of-the art inference-based algorithms have so-far outperformed optimization based ones, both in sample efficiency and asymptotic performance.

3.2 Variational Model Training

A separate line of work considers sample-efficient reinforcement learning from images using variational models. The main idea is to cast image-based RL in the POMDP setup and learn a low-dimensional state-based representation, which can then efficiently be used for model-free learning [17]. This is generally cast as a deep Bayesian filter and with standard variational methods [12, 28], we can introduce the variational network $q(\mathbf{s}_t|\mathbf{o}_{\leq t}, a_{<t})$ and obtain the evidence lower bound (ELBO):

$$\log P(\mathbf{o}_{1:T}|\mathbf{a}_{1:T}) \geq \widehat{\mathbb{E}}_{q_\theta}\Big[\sum_{t=1}^{T} \underbrace{\log \widehat{\mathcal{O}}_\theta(\mathbf{o}_t|\mathbf{s}_t)}_{\text{reconstruction}} - \underbrace{\mathbb{D}_{KL}(q_\theta(\mathbf{s}_t|\mathbf{o}_t, \mathbf{s}_{t-1}, \mathbf{a}_{t-1})||\widehat{\mathcal{P}}_\theta(\mathbf{s}_t|\mathbf{s}_{t-1}, \mathbf{a}_{t-1}))}_{\text{forward model}}\Big]. \tag{1}$$

In practice, we can iterate between model training, off-policy agent training and sample collection, which is the approach taken by [17]. Alternatively, other works [9, 8], have used the forward prediction model for model-based policy training. However, since these models depend on posterior sampling, which could make learning exploration strategies sub-optimal [29], and since we do not concern ourselves with sample efficiency (we're working in the fully offline case) we only use the learned model for representation purposes.

3.3 Offline Reinforcement Learning

Offline RL is the setting where we have access only to a fixed dataset $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{r}, \mathbf{s}')\}$, which consists of transition tuples from trajectories collected using a behavior policy . In other words, the dataset $\mathcal{D}$ is sampled from $d^(\mathbf{s}, \mathbf{a})d^(\mathbf{s})(\mathbf{a}|\mathbf{s})$. In the offline setting, even in the limit of an infinite size dataset, it may not be possible to find the optimal policy for the underlying MDP [2, 11]. Thus, we typically forgo the goal of finding the optimal policy, and instead aim to find the best possible policy using the fixed offline dataset.

One popular class of approaches for solving MDPs involves the use of dynamic programming and actor-critic schemes [24, 1], which do not explicitly require the learning of a dynamics model. To capture the long term behavior of a policy without a model, we define the action value function as

$$Q^\pi(\mathbf{s}, \mathbf{a}) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \, r(\mathbf{s}_t, \mathbf{a}_t) \mid \mathbf{s}_0 = \mathbf{s}, \mathbf{a}_0 = \mathbf{a}\right], \tag{2}$$

where future actions are sampled from $\pi(\cdot|)$ and state transitions happen according to the MDP dynamics. Consider the following Bellman operator:

$$\mathcal{B}^\pi Q(\mathbf{s}, \mathbf{a}) := r(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}' \sim \mathcal{P}(\cdot|\mathbf{s},\mathbf{a}), \mathbf{a}' \sim \pi(\cdot|\mathbf{s}')}\left[Q(\mathbf{s}', \mathbf{a}')\right],$$

and its sample based counterpart:

$$\widehat{\mathcal{B}}^\pi Q(\mathbf{s}, \mathbf{a}) := r(\mathbf{s}, \mathbf{a}) + \gamma Q(\mathbf{s}', \mathbf{a}'),$$

associated with a single transition $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ and $\mathbf{s}' \sim \pi(\cdot|\mathbf{s}')$. The action-value function satisfies the Bellman consistency criterion given by $\mathcal{B}^\pi Q^\pi(\mathbf{s}, \mathbf{a}) = Q^\pi(\mathbf{s}, \mathbf{a}) \; \forall(\mathbf{s}, \mathbf{a})$. When given an offline dataset $\mathcal{D}$, standard approximate dynamic programming (ADP) and actor-critic methods use this criterion to learn by alternating between the following steps [20].

1. **Policy Evaluation:** The Q function associated with the current policy π is approximated by repeatedly applying the Bellman operator:

$$Q^{k+1} \leftarrow \arg\min_Q \; \mathbb{E}_{\mathcal{D}} \left[\Big(Q(\mathbf{s}, \mathbf{a}) - \widehat{\mathcal{B}}^\pi Q^k(\mathbf{s}, \mathbf{a})\Big)^2\right].$$

The approximate value $\hat{Q}^\pi$ is obtained after several steps of the above recursion.

2. **Policy Improvement:** After approximating the Q function as $\hat{Q}^\pi$, the policy is improved as:

$$\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi'(\cdot|\mathbf{s})}\left[\hat{Q}^\pi(\mathbf{s},\mathbf{a})\right]$$

Actor-critic schemes with parameterized policies and Q functions approximate the $\arg\max$ and $\arg\min$ in above equations with a few steps of gradient descent.

A number of prior works have observed that such a direct extension of ADP and actor-critic schemes to offline RL leads to poor results due to distribution shift over the course of learning and over-estimation bias in the Q function [7, 13, 25]. To address these drawbacks, prior works have proposed a number of modifications aimed towards regularizing the policy or value function. In this work, we primarily focus on CQL [16], which modifies the action-value learning step as:

$$\begin{aligned} Q^{k+1} \leftarrow & \arg\min_{Q} \beta\left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\cdot|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}\left[Q(\mathbf{s},\mathbf{a})\right]\right) \\ & + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \widehat{\mathcal{B}}^\pi Q^k(\mathbf{s},\mathbf{a})\right)\right)^2\right], \end{aligned} \tag{3}$$

where $\mu(\cdot|s)$ is a wide sampling distribution such as the uniform distribution over action bounds. CQL effectively penalizes the Q function at states in the dataset for actions not observed in the dataset. This enables a conservative estimation of the value function for any policy [16], mitigating the challenges of over-estimation bias and distribution shift.

3.4 Offline Meta-Reinforcement Learning

Similarly to [3, 19], we consider the standard meta-RL problem in an offline setup. We have a set of N fixed training tasks $\mathcal{T}_i, i = 1 : N$. Each training task comes with an offline buffer of visual data $\mathcal{B}_i$ in the form of trajectories, generated by a behavior policy π_i^β and we do not have access to additional data collection in the *meta-training* environments. Here we consider the case where each behavior policy π_i^β is task-specific, i.e. operates in the MDP of $\mathcal{T}_i$, rather than the POMDP induced by the meta-learning problem.

4 The Reflective Explorer: Offline Visual Meta-RL

One major challenge of such offline meta-RL set-up is that the agent needs to learn coherent exploration policies at test time. Inference-based meta-RL approaches allow off-policy learning, but suffers poor performance in our particular setting: if the agent can easily infer the tasks from offline replay buffers based on spurious correlations, then it would fail to learn a coherent meta-test exploration strategy. This is especially prevalent in the case when individual replay buffers hold more narrow, task-specific data, which can be very disjoint. In this work, we tackle the spurious memorization issue using a strategy for generating synthetic exploration rollouts from offline data only. Furthermore, we use a recurrent deep variational filter method, which jointly infers environment and task state from sequences of images, actions and rewards from the synthetic exploration distribution and train an offline RL agent using conservative Q-learning [16] on top of the learned latent space. We describe each step in detail in the following three subsections.

4.1 Synthetic Exploration

If each offline replay buffer $\mathcal{B}_i$ contains only task-specific data, then a naive combination of meta-learning and offline learning methods would fail to learn a coherent task-inference method and would thus fail at test time. In order to avoid this phenomenon we create extended synthetic exploration rollouts by mixing offline rollouts from different tasks. In particular, we create a K-extended rollout by randomly sampling K indices (with replacement) $i_1, \ldots, i_K$ from $\{1, \ldots, N\}$. We then sample random trajectories from the corresponding replay buffers: $\tau^{i_j} = [o_1^{i_j}, \ldots, o_T^{i_j}, a_1^{i_j}, \ldots, a_T^{i_j}, r_1^{i_j}, \ldots, r_T^{i_j}] \sim \mathcal{B}_{i_j}, j = 1, \ldots, K$ and an additional target task index $i^* \sim [N]$. We then create synthetic rewards in rollout τ^{i_j} using $[\hat{r}_1^{i_j}, \ldots, \hat{r}_T^{i_j}] = [I\{i_j = i^*\}r_1^{i_j}, \ldots, I\{i_j = i^*\}r_T^{i_j}]$. We then concatenate the trajectories together to obtain a single meta-exploration rollout $\tau = [o_1, \ldots, o_{KT}, a_1, \ldots, a_{KT}, \hat{r}_1, \ldots, \hat{r}_{KT}]$. We denote the resulting meta-rollout distribution as $\mathcal{M}^K(\mathcal{B}_1, \ldots, \mathcal{B}_N)$ or simply $\mathcal{M}^K$ for brevity. An illustration of our synthetic

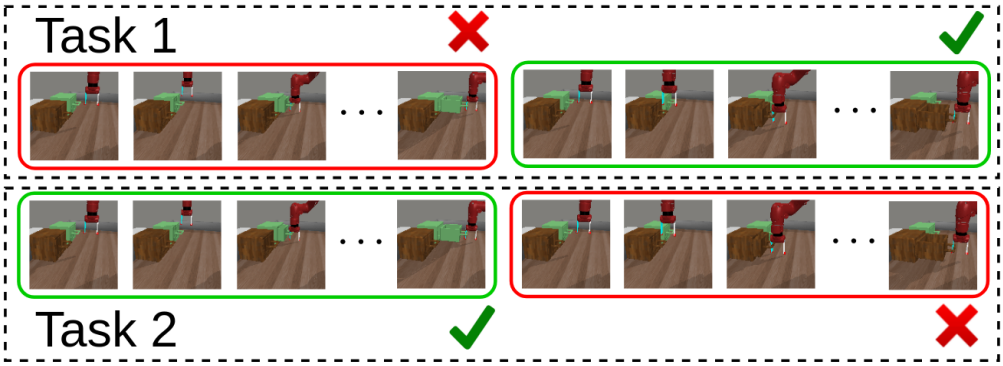

Figure 1: Illustration of our synthetic exploration construction. The setup consists of two drawers, the goal of Task 1 is to open the left drawer, while the goal of Task 2 is to open the right drawer. The agent receives reward of 1 for successfully completing a task and 0 otherwise. During deployment the agent does not have a direct indication of which is the target drawer to open and needs to carry out exploration in the environment. We randomly concatenate rollouts from different tasks together to create extended exploration runs. Furthermore, for each meta-trajectory we choose a particular task and zero out rewards for all trajectories that do not carry out that task. We then train a policy using offline reinforcement learning on the constructed meta-trajectories with the augmented sequence of rewards.

exploration approach is shown in Figure 1. Since we consider realistic settings with sparse rewards, this process does not fundamentally alter the reward distribution encountered by the agent during exploration at test time. Furthermore, since we sample the "ground-truth" target task randomly after constructing the extended rollouts, this approach fundamentally removes spurious correlations between the state-action distribution and the task distribution. In this case the inference model is forced to integrate not only the environment interactions, but the rewards as well, which matches the inference problem at test time.

4.2 Variational Model Training

We follow [28] in training a high-dimensional variational model in order to build the belief distribution for the associated meta-learning POMDP. The procedure closely follows the methods outlined in Section 3.1 , however we operate on batches of meta-exploration trajectories, constructed using the method from Section 4.1. In addition, since task identification depends on the obtained rewards, in order to build a belief representation over tasks, we also include the sequence of rewards as both an input to the inference model, as well as a reconstruction term. Our final optimization objective, is an extended version of the Evidence Lower Bound:

$$\max_{\theta} \widehat{\mathbb{E}}_{\mathbf{s}_{1:TK} \sim q_\theta} \Bigg[\sum_{t=1}^{TK} \log \widehat{O}_\theta(\mathbf{o}_t|\mathbf{s}_t) + \beta \log \widehat{R}_\theta(\hat{\mathbf{r}}_t|\mathbf{s}_t) - \sum_{t=0}^{TK-1} \mathbb{D}_{KL}[q_\theta(\mathbf{s}_{t+1}|\mathbf{o}_{t+1}, \hat{\mathbf{r}}_{t+1}, \mathbf{s}_t, \mathbf{a}_t) || \widehat{P}_\theta(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)] \Bigg]$$

where the empirical expectation is computed using batches from $\mathcal{M}^K$ and samples from the belief distribution q_θ. Here β is a parameter we use to weigh reward prediction (indicative of task prediction) over image-reconstruction, which is task-agnostic. Since the model is optimized using supervised training to predict rewards, the inference network needs to aggregate long-term evidence based on environment interactions and rewards in order to identify the corresponding ground-truth task. Note that we do not reset the model states between episodes in the meta-exploration rollout, since it's crucial to maintain information across the rollout in order to identify the task.

4.3 Policy Training

We choose to use CQL [16] as our model-free optimization method, as it achieves good results on offline RL benchmarks [6]. In addition our synthetic exploration approach creates multi-modal data

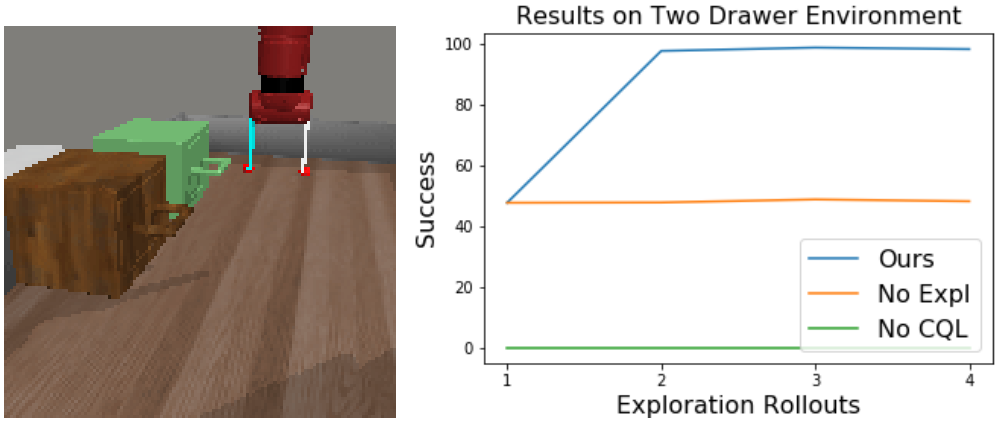

Figure 2: **Left**: This environment consist of a Sawyer robot arm and two drawers. The agent receives a reward of 1 for successfully opening the correct drawer, however it does not not which one that is apriori and needs to carry out extended exploration to verify the particular task. **Right**: Results on the two-drawer domain The $x-$ axis shows number of exploration rollouts, while the $y-$ axis shows success rate of opening the correct drawer. We see that after a single exploration rollout our method (labeled "Ours") successfully identifies the correct drawer to open, while a naive combination of offline training and meta-learning methods (labeled "No Expl" as it does not use our synthetic exploration procedure) fails to reliably succeed in the task. Naive application of online learning methods ("No CQL") completely fails to learn meaningful behaviours.

distributions, which policy constraint-based methods [14, 26] tend to struggle with. We optimize our policy using Eq. 3 on top of the learned belief representation. In particular, we compute *belief states* $\mathbf{b}_{1:TK} = q_\theta(\mathcal{M})$ over batches of trajectories from the meta-exploration distribution $\mathcal{M}^K$ and using the belief network q_θ. In practice q_θ is usually parameterized as a diagonal Gaussian distribution and we represent the belief state as a concatenation of the mean and variance vectors of the distribution. Finally we use the following TD learning objective:

$$Q^{j+1} \leftarrow \arg\min_Q \beta(\mathbb{E}_{\mathbf{b}\sim q_\theta(\mathcal{M}^K),\mathbf{a}\sim\mu(\cdot|\mathbf{b})}\left[Q(\mathbf{b},\mathbf{a})\right] - \mathbb{E}_{\mathbf{b},\mathbf{a}\sim q_\theta(\mathcal{M}^K)}\left[Q(\mathbf{b},\mathbf{a})\right])$$
$$+\frac{1}{2}\mathbb{E}_{\mathbf{b},\mathbf{a},\mathbf{b}'\sim q_\theta(\mathcal{M}^K)}\left[\left(Q(\mathbf{b},\mathbf{a}) - \widehat{\mathcal{B}}^\pi Q^j(\mathbf{b},\mathbf{a}))\right)^2\right].$$

The policy is then optimized in the standard way

$$\max_\psi \mathbb{E}_{\mathbf{b}\sim q_\theta(\mathcal{M}^K)}\mathbb{E}_{\mathbf{a}\sim\pi_\psi(\mathbf{a}|\mathbf{b})}[Q_\theta(\mathbf{b},\mathbf{a})]$$

In [29] the authors make connections between this approach and BAMDPs and show significant improvements in exploration efficiency over posterior sampling methods used in previous works [22].

5 Experiments

Environment: Our evaluation environment is shown in Fig. 2 on the left. The environment consist of a Sawyer robot arm and two drawers. The agent receives a reward of 1 for successfully opening the correct drawer, however it does not not which one that is apriori and needs to carry out extended exploration to verify the particular task. Following [21], the agent has access to 1000 training rollouts for each individual task. This environment very clearly demonstrates the issues with naively using offline methods to train meta-RL agents. In particular it shows two main issues:

1. The task being performed in a rollout from the dataset can be correctly inferred from the first few frames, simply by the direction of the robot arm. At test time this yields a very poor exploration performance as obviously the task is independent of the robot motion.

2. The dataset contains reltively narrow data that does not explore other tasks. In particular at test time, the robot can still end up opening the wrong drawer and receive a reward of 0. This is an out-of-distribution event, since drawer opening always results in positive reward in the offline dataset, but the agent fails to include rewards in the task-based inference process.

Results: The above problems are clearly visible in the results shown in Fig. 2 on the right. We observe that after a single exploration rollout our method (labeled "Ours") successfully identifies the correct drawer to open, while a naive combination of offline training and meta-learning methods (labeled "No Expl" as it does not use our synthetic exploration procedure) fails to reliably succeed in the task. In essence this approach learns to always open the left drawer, as during training any drawer opening results in the same reward, and the left one is closer to the initial position of the arm. However, since at test time we randomly sample between the two tasks, this method ends up achieving only 50% success rate. Finally, naive application of online learning methods ("No CQL") completely fail to learn meaningful behaviours, due to test-time distribution shift, which the policy is not equipped to handle.

6 Conclusion

In this work we presented a method for offline meta-learning from images in tasks with sparse rewards, which significantly outperforms naive combinations of meta-learning and offline-RL methods on a realistic meta-learning task involving robot manipulation. Our frameworks allows us to add additional data and new tasks and seamless expand the capabilities of meta-RL agents without the need for additional environmental interactions. Currently we're working towards deploying this algorithm on more complex robotic manipulation domains from [23] that involve both qualitative and visual generalization to new objects and tasks. We also believe that our approach has significant potential for real-world applications and are currently evaluating it on a real-world multi-task desk setup.

References

[1] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

[2] J. Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. *ArXiv*, abs/1905.00360, 2019.

[3] Ron Dorfman and Aviv Tamar. Offline meta reinforcement learning. *arXiv preprint arXiv:2008.02598*, 2020.

[4] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

[6] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

[7] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018.

[8] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *International Conference on Learning Representations*, 2020.

[9] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *International Conference on Machine Learning*, 2019.

[10] Jan Humplik, Alexandre Galashov, Leonard Hasenclever, Pedro A Ortega, Yee Whye Teh, and Nicolas Heess. Meta reinforcement learning as task inference. *arXiv preprint arXiv:1905.06424*, 2019.

[11] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[13] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019.

[14] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *NeurIPS*, 2019.

[15] Aviral Kumar, Anikait Singh, Stephen Tian, Chelsea Finn, and Sergey Levine. A workflow for offline model-free robotic reinforcement learning, 2021.

[16] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Conference on Neural Information Processing Systems*, 2020.

[17] Alex X. Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*, 2019.

[18] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[19] Eric Mitchell, Rafael Rafailov, Xue Bin Peng, Sergey Levine, and Chelsea Finn. Offline meta-reinforcement learning with advantage weighting. *International Conference on Machine Learning*, 2021.

[20] Rémi Munos and Csaba Szepesvari. Finite-time bounds for fitted value iteration. *J. Mach. Learn. Res.*, 9:815–857, 2008.

[21] Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. *Learning for Dynamics and Control*, 2020.

[22] Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International Conference on Machine Learning*, 2019.

[23] Avi Singh, Huihan Liu, Gaoyue Zhou, Albert Yu, Nicholas Rhinehart, and Sergey Levine. Parrot: Data-driven behavioral priors for reinforcement learning. *International Conference on Learning Representations*, 2021.

[24] Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[25] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[26] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[27] Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization, 2021.

[28] Tony Z. Zhao, Anusha Nagabandi, Kate Rakelly, Chelsea Finn, and Sergey Levine. Meld: Meta-reinforcement learning from images via latent state models. *Conference on Robot Learning*, 2021.

[29] Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. 2020.

A Appendix

Optionally include extra information (complete proofs, additional experiments and plots) in the appendix. This section will often be part of the supplemental material.