Meta-Reinforcement Learning Robust to Distributional Shift Via Performing Lifelong In-Context Learning

Tengye Xu [1] Zihao Li [1] Qinyuan Ren [1]

Abstract

A key challenge in Meta-Reinforcement Learning (meta-RL) is the task distribution shift, since the generalization ability of most current meta-RL methods is limited to tasks sampled from the training distribution. In this paper, we propose Posterior Sampling Bayesian Lifelong In-Context Reinforcement Learning (PSBL), which is robust to task distribution shift. PSBL meta-trains a variant of transformer to directly perform amortized inference about the Predictive Posterior Distribution (PPD) of the optimal policy. Once trained, the network can infer the PPD online with frozen parameters. The agent then samples actions from the approximate PPD to perform online exploration, which progressively reduces uncertainty and enhances performance in the interaction with the environment. This property is known as in-context learning. Experimental results demonstrate that PSBL significantly outperforms standard Meta RL methods both in tasks with sparse rewards and dense rewards when the test task distribution is strictly shifted from the training distribution.

1. Introduction

Overfitting and lack of generalization are considered as the two of the main challenges for Deep Reinforcement Learning (RL), which hinder the wide range of application in the Real world (Rajeswaran et al., 2017; Zhang et al., 2018; Cobbe et al., 2019; Fakoor et al., 2019; Song et al., 2019; Mendonca et al., 2020). Recently, a growing number of Meta-Reinforcement Learning (meta-RL) methods have been proposed to tackle these issues by meta-learning policies that quickly adapt to unseen but similar tasks. Given a task distribution, these meta-RL agents are trained to maximize expected returns across any task sampled from the task distribution. However, most meta-RL methods evaluate their performance on tasks drawn from the same distribution, leaving the generalization capability, particularly with frozen parameters, in Out-of-Distribution (OOD) tasks less studied. This aspect is crucial for real-world applications, as obtaining a comprehensive prior of all possible situations an RL agent might encounter is challenging (Beck et al., 2023).

A popular approach in meta-RL is to meta-learn a good initialization of the network across tasks, requiring only a few policy gradient steps for adaptation in test tasks (Finn et al., 2017; Stadie et al., 2018; Zintgraf et al., 2019a). However, when a task distribution shift occurs, these methods struggle to provide a robust initialization for OOD tasks. Meanwhile, the meta-RL algorithm degrades into a conventional RL algorithm, forcing the agent to start almost from scratch and rely on policy gradients to tackle unseen tasks (Beck et al., 2023). This process can necessitate millions of interactions at test time, which is costly in real-world applications.

Another promising meta-RL approach involves meta-training a memory-augmented network to infer the task distribution and a policy network conditioned on this distribution to maximize expected returns (Humplik et al., 2019; Zintgraf et al., 2019b; Liu et al., 2021). Once trained, the memory-augmented network can function as if performing online inference, achieving high data efficiency when adapting new tasks sampled from the training distribution. However, these methods often struggle to generalize to OOD tasks. This is because the policy network, typically a simple feedforward network, lacks the capability to perform approximate inference online, making it difficult to map unseen task distributions to optimal actions.

To tackle these generalization challenges, we propose training a transformer-based network to directly perform amortized inference about the Predictive Posterior Distribution (PPD) of the optimal policy. This can be accomplished within an end-to-end posterior sampling framework in Reinforcement Learning. Once trained, the network can infer the PPD online with frozen parameters. The agent then samples actions from the approximate PPD to perform online exploration, thereby progressively reducing uncertainty and

[1]College of Control Science and Engineering, Zhejiang University, 38 Zheda Road, Hangzhou, Zhejiang, China.. Correspondence to: Qinyuan Ren <renqinyuan@zju.edu.cn>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

improving its performance in the interaction with the environment. This process give rise to an inherent characteristic within the network, namely Lifelong In-context Learning, an extended version of In-context Learning (Von Oswald et al., 2023). Lifelong In-context Learning is defined as the capability of a pretrained network with frozen parameters to continuously improve its policy *in context* over an extended horizon until convergence. Consequently, through performing lifelong in-context learning at meta-test time, the pretrained network can quickly adapt to OOD tasks.

The primary contribution in this work is proposing a meta-RL method called Posterior Sampling Bayesian Lifelong in-context Reinforcement Learning (PSBL). The proposed method is robust to task distribution shift by performing lifelong in-context learning. PSBL is evaluated on the discrete navigation tasks and continuous control tasks, where the test task distribution is strictly different from the training distribution. The experimental results demonstrate that the trained network in PSBL can perform Lifelong in-context Learning across episodes in tasks with sparse rewards and within episodes in tasks with dense rewards. Therefore, through performing Lifelong in-context Learning, PSBL significantly outperforms standard meta RL methods especially in OOD tasks. The open-source implementation can be found at https://github.com/xu-ye/PSBL-MetaRL.

2. Related Work

Meta-Reinforcement Learning Meta-Reinforcement Learning (Meta-RL) primarily focuses on few-shot adaptation to new tasks by leveraging knowledge learned during meta-training Recent approaches mainly include Policy Gradient (PPG) methods, Black-Box methods, and Task Inference methods. PPG methods typically meta-learn a good initialization of the network, requiring only a few policy gradient steps for adaptation to test tasks (Finn et al., 2017; Stadie et al., 2018; Zintgraf et al., 2019a). However, when a task distribution shift occurs, these methods struggle to provide a robust initialization for OOD tasks. Meanwhile, these meta-RL algorithms degrades into a conventional RL algorithm, forcing the agent to start almost from scratch and rely on policy gradients to tackle unseen tasks (Beck et al., 2023). This process can necessitate millions of interactions at test time, which is costly in real-world applications.

In contrast to PPG methods, black-box methods utilize recurrent neural networks to learn history-dependent policies for fast adaptation to new tasks (Duan et al., 2016; Wang et al., 2016; Fakoor et al., 2019). Recently, many black-box methods have shifted from recurrent networks to other memory-augmented networks, such as those incorporating attention mechanisms (Mishra et al., 2017) and transformers (Melo, 2022). If the Supervised Learning (SL) loss, the Posterior Sampling setting and the dynamic window are removed from PSBL, PSBL method reduces to this setting. The main difference is that by minimizing the SL loss and actor loss, the network trained in PSBL can perform approximate Bayesian Inference online. Then our agent can sample from the approximate distribution to do online exploration, progressively improving its performance.

Task Inference methods can be considered as a subset of black box methods (Humplik et al., 2019; Zintgraf et al., 2019b; 2021). Utilizing networks with memory, Task Inference methods meta-train a memory-augmented network to infer the task distribution and a policy network conditioned on the task distribution to maximize the expected returns. The The networks in Task Inference method can be trained with privileged information (Humplik et al., 2019), with contrastive learning loss (Fu et al., 2021), or with self-supervised learning loss (Zhang et al., 2021). Once trained, the network with memory can perform approximate inference about the task distribution online. Similarly, the network trained in PSBL also performs approximate inference online. However, instead of inferring the task distribution, the proposed method PSBL directly infers the distribution of the Bayes-optimal policy, demonstrating superior generalization in OOD tasks.

Memory-based Meta learning Prior studies have demonstrated practically that a parametric memory-based network can be trained to behave as if performing Bayesian inference (Ritchie et al., 2016; Genewein et al., 2023). Recent studies (Ortega et al., 2019; Mikulik et al., 2020) have theoretically proven that Memory-based Meta learning methods can meta-train a fixed-parametric memory-based network to perform amortized Bayesian inference online by minimizing sequential prediction errors over a task distribution. Leveraging this property, we construct an in-context learner capable of performing amortized Bayesian inference by minimizing prediction errors.

In-Context Learning In-context Learning is an approach where pre-trained models improve their policies by utilizing the context provided during inference, without the need for gradient updates. An increasing number of works focus on studying this phenomenon and try to provide theoretical explanations (Akyürek et al., 2022; Von Oswald et al., 2023). In-context Learning has initially emerged from large language models and now is being extended to decision-making (Laskin et al., 2022) and other domains. Recently, Grigsby (2023) has proposed AMAGO, an in-context learning method designed to address meta-RL tasks and long-horizon RL tasks. However, the experimental results in Section 5 demonstrate that PSBL outperforms AMAGO both in asymptotic performance and training data efficiency.

Posterior Sampling in Reinforcement Learning Posterior Sampling in Reinforcement Learning refers to a set of methods (Osband et al., 2013; Rakelly et al., 2019) where the agent samples a MDP from its posterior distribution and taking optimal actions based on the sampled MDP. As the agent observes new data, this distribution of MDPs is updated, allowing it to make decisions that balance exploration and exploitation based on the uncertainty in its knowledge. These methods (Osband et al., 2013; Rakelly et al., 2019) have demonstrated high data efficiency with performance comparable to state-of-the-art RL methods. However, because exploration is guided by a single sampled MDP rather than the full distribution, these traditional Posterior Sampling methods can lead to suboptimal exploration (Beck et al., 2023). Therefore, instead of sampling an MDP from its distribution, PSBL directly sample a near-optimal action from the posterior distribution of the optimal policy.

3. BACKGROUND

A standard Markov decision process can be defined by a tuple $M = (\mathcal{O}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \gamma, H)$. $\mathcal{O}$ is a set of observations, $\mathcal{A}$ stands for a set of actions. $\mathcal{R}$ represents a reward function and $\mathcal{T}$ is the transition function. γ is the discount factor and H represents the horizon. The objective of the agent is to maximize the discounted accumulated return defined by $G = \sum \lambda^t R(s_t, a_t)$. In this work, a meta-reinforcement learning setting is considered for training a lifelong in-context learner that is robust to distributional shift.

3.1. Training Setup

In the standard meta-reinforcement learning setting, there is a distribution $P(M)$ over MDPs representing various tasks that differ in rewards and transition functions but share a similar structure. The training horizon extends to H^+, encompassing N episodes $H^+ = NH$. At meta-training time, batches of MDPs are repeatedly sampled from the distribution $M \sim P(M)$, and the agent is trained to maximize the accumulated returns across these tasks. At meta-test time, the meta-RL agent is evaluated on its ability to adapt quickly to new tasks sampled from the same distribution $P(M)$ used during training.

The standard meta-reinforcement learning setting is modified in this work to promote training a lifelong in-context learner that is robust to distributional shift. The evaluation horizon at meta-test time is extended from H^+ to L , a significantly longer period than the training horizon. This extended horizon allows the agent to continuously improve its performance through interaction with the environment until convergence. Moreover, the tasks at meta-test time are set to be drawn from a distribution strictly different from the training distribution, enabling the evaluation of robustness to distributional shifts.

To train a lifelong in-context learner that is robust to distributional shift, we propose training a transformer-based network to directly perform amortized Bayesian inference online within an Posterior Sampling framework for Meta-Reinforcement Learning.

3.2. Posterior Sampling for Meta-Reinforcement Learning

In the formulation of Posterior Sampling for Meta-Reinforcement Learning, it is assumed that each MDP M is attached with an optimal policy. Thus, we can define the world model of each task by a transition model $P(o_{t+1}|\theta, \tau_t)$, a reward model $P(r_t|\theta, \tau_t)$ and an optimal policy $P(a_{t+1}|\theta, \tau_t)$, where the θ is the parameters of the world model and $\tau_t = \{o_0, a_0, r_0, o_1, a_1, r_1, \dots, o_t, a_t\}$ is the interaction sequences from time step 0 to time step t. The distribution over interaction sequences under the optimal policy of MDP M is defined as follows:

$$P(\tau_T|\theta) = \prod_t^T P(r_{t-1}|\theta, \tau_{t-1}) P(o_t|\theta, \tau_{t-1}) P(a_t|\theta, \tau_{t-1}). \tag{1}$$

When the MDP is unknown, the distribution over interaction sequences can inferred as follows,

$$P(\tau_T) = \int P(\tau_T|\theta) P(\theta). \tag{2}$$

Posterior Sampling can be characterized as directly sampling predicted optimal action, observation and reward from the Posterior Predictive Distribution of these variables in the world model,

$$a_{t+1} \sim P(a_{t+1}|\tau_t) = \int P(a_{t+1}|\theta, \tau_t) P(\theta|\tau_t), \tag{3}$$

$$o_{t+1} \sim P(o_{t+1}|\tau_t) = \int P(o_{t+1}|\theta, \tau_t) P(\theta|\tau_t), \tag{4}$$

$$r_t \sim P(r_t|\tau_t) = \int P(r_t|\theta, \tau_t) P(\theta|\tau_t). \tag{5}$$

The objective of Posterior Sampling is to minimize the cross entropy between the PPD and its approximation,

$$\mathcal{J}(q_\delta) = \mathbb{E}_{\tau_t \sim P(\tau_t)} \left[\mathrm{H}\left(P(\cdot \mid \tau_t), q_\phi(\cdot \mid \tau_t) \right) \right] \tag{6}$$

where q is the memory-augmented network to be trained in Posterior Sampling with parameters ϕ. By minimizing the prediction error, the network can directly perform amortized inference about the PPD of the optimal policy, as theoretically proven in (Ortega et al., 2019; Mikulik et al., 2020). Furthermore, the PPD of the optimal policy captures both the predictive optimal actions and their associated uncertainties. By repeatedly sampling actions from the approximate PPD, the agent can efficiently explore online and adapt to new tasks with high data efficiency.

Posterior Sampling is powerful for training a network to perform amortized inference of the PPD about the optimal policy. However, applying the Posterior Sampling framework in meta-reinforcement learning, particularly for training a lifelong in-context learner, can be nearly intractable. The main challenges are as follows:

- The true optimal policy for each MDP is unknown,
- Computation of the PPD is often intractable, and
- Even with an accurate PPD, Posterior Sampling methods can only train agent in a fixed horizon whereas lifelong in-context learning requires an almost infinite evaluation horizon.

4. Posterior Sampling Bayesian Lifelong In-context Reinforcement Learning

In this section, a new meta-reinforcement learning method called Posterior Sampling Bayesian Lifelong In-context Reinforcement Learning (PSBL) is proposed to address the aforementioned challenges. This section starts by explaining how to approximate the PPD using a memory-based network. This section then details a method to obtain a Bayes-optimal policy for supervisory signals and presents an approach to enable lifelong in-context learning over an extended evaluation horizon. Finally, this section describes an specially designed Transformer-based network and integrates all these components to summarize the new approach PSBL.

4.1. End to End PPD Approximation With LILTrans

Computing the Predictive Posterior Distribution in closed form is almost intractable for most cases, since the true model of the current MDP is unknown and marginalising over MDPs is computationally infeasible. Consequently, the PPD can only be approximated for most cases. Classic Posterior Sampling methods, such as PSRL (Osband et al., 2013), PSDRL (Sasso et al., 2023), PEARL (Rakelly et al., 2019), typically approximate the posterior distribution of MDPs $P(\theta|\tau_t)$ first, then sample one MDP θ_1 from the posterior distribution,

$$\theta_1 \sim P(\theta \mid \tau_t) = \frac{P(\theta) P(\tau_t \mid \theta)}{\int P(\theta') P(\tau_t \mid \theta') d\theta'}. \tag{7}$$

Based on the sampled MDP, these methods compute and follow its optimal policy in the following episode,

$$a_{t+1} \sim P(a_{t+1} \mid \theta_1, \tau_t). \tag{8}$$

As shown in (7)(8), the policy used for exploration heavily relies on the sampled MDP rather than the full distribution of MDPs. This process can result in sub-optimal exploration. A more desirable approach is to directly sample action from the approximated PPD of the optimal policy rather than first sampling a MDP from the approximate PPD of MDPs and then sampling actions from a policy based on the sampled MDP. The desirable approach is as follows,

$$a_{t+1} \sim P(a_{t+1}|\tau_t) = \int P(a_{t+1}|\theta, \tau_t) P(\theta|\tau_t). \tag{9}$$

Considering a prior work (Muller et al., 2021) has proved that Transformers can perform Bayesian Inference, a variant of the transformer network, named Lifelong In-context Learning Transformer (LILTrans), is used in this work to directly approximate the PPD of the optimal policy. The LILTrans, parameterized by q_ϕ, functions as follows,

$$r_t, o_{t+1}, a_{t+1} = q_\phi(\tau_t). \tag{10}$$

To train the network in terms of cross entropy, we proposed a Supervised Learning (SL) loss for the predicted rewards and observations, whose true values are accessible in the next time step. The SL loss is defined as follows,

$$\ell_s = \mathbb{E}_{\tau_t \sim P(\tau_t)}[-\log q_\phi(o_{t+1}, r_t|\tau_t)], \tag{11}$$

Assuming an optimal policy exists for each MDP, the optimal policy loss is defined as follows,

$$\ell_a = \mathbb{E}_{\tau_t}\left[D_{KL}\left(q_\phi(a_{t+1} \mid \tau_t) \middle\| P(a_{t+1} \mid \tau_t)\right)\right]. \tag{12}$$

It is proved in Appendix A that minimizing the SL loss ℓ_s and the optimal policy loss ℓ_a can lead to an approximation of the PPD in terms of cross-entropy.

4.1.

4.2. Meta Learning an Bayes-Optimal Policy

The optimal policy loss (18) requires expert-provided optimal actions as supervisory signals. However, expert policies are not available for most cases except for toy tasks. Thus, a meta reinforcement learning method is used in this work to obtain a Bayes-Optimal Policy.

The Bay-Optimal policy can be characterized as maximizing the expected accumulated returns. The training objective is as follows,

$$\mathcal{J}^+(\pi) = \mathbb{E}_{\tau,\pi}\left[\sum_{t=0}^{H^+} \gamma^t R(\tau_t, o_{t+1}, a_{t+1})\right]. \tag{13}$$

Since the optimal policy in unknown, during training we only have access to interaction sequences generated by a current policy π. Therefore, the distribution over interaction sequences under the current policy is as follows,

Algorithm 1 PSBL Meta-Training

```
Require: Batch of training MDPs sampled from $P(M)$, dynamic window capacity $b$
Initialize replay buffer $\mathcal{D}$
while not done do
  for each sampled MDP $M$ do
    Initialize interaction sequences $\tau$
    for $k$ = 1, ..., $N$ do
      for $l$ = 1, ..., $H$ do
        At time step $t = kH + l$
        Sample $a_{t+1} \sim q_\phi(a_{t+1} \mid \tau)$
        Gather data $o_{t+1}, r_t$ from the MDP
        Update $\tau$ with $\{r_t, t+1, d_{t+1}, o_{t+1}, a_{t+1},\}$
      end for
      Update the replay buffer $\mathcal{D}$ with $\tau$
      if $k > b$ then
        Discard the earliest $k - b$ episode from $\tau$
      end if
    end for
  end for
  for step in training steps do
    for batches of $\tau$ sampled from $\mathcal{D}$ do
      Compute the PPD from $q_\phi(o, r \mid \tau)$
      $\mathcal{L}_s \leftarrow \mathcal{L}_s + \ell_s(q_\phi(o, r \mid \tau), \tau)$
      Sample $a \sim q_\phi(a \mid \tau)$
      $\mathcal{L}_c \leftarrow \mathcal{L}_c + \ell_c(a, \tau)$
      $\mathcal{L}_a \leftarrow \mathcal{L}_a + \ell_a(a, \tau)$
    end for
    $\phi \leftarrow \phi - \alpha \bigtriangledown_\phi (\lambda_s \mathcal{L}_s + \lambda_a \mathcal{L}_a + \lambda_c \mathcal{L}_c)$
  end for
end while
```

$$P^\pi(\tau_t) = \int_\theta P(\theta) \prod_t^T P(r_{t-1} \mid \theta, \tau_{t-1}) P(o_t \mid \theta, \tau_{t-1}) \pi(a_t \mid \tau_{t-1}). \tag{14}$$

Furthermore, we define a state-action function $Q^\pi(a_{t+1}, \tau_t)$ for the policy π as the expected accumulated discounted rewards given a past interaction sequence. The state-action function is defined as

$$Q^\pi(a_{t+1}, \tau_t) = \mathbb{E}^\pi \Big[\sum_{k=t}^{H} \lambda^{(k-t)} R(\tau_k, a_{k+1}, o_{k+1}) \Big| \tau_t, a_{t+1} \Big], \tag{15}$$

where $\mathbb{E}^\pi := \mathbb{E}_{\tau_t \sim P^\pi(\tau_t)}$ denotes an expectation w.r.t. the distribution $P^\pi(\tau_t)$. Thus, an optimal policy is one that maximizes this function at any time step for any MDP.

The state-action function can be approximated by a Q network parameterized by ϕ. We define an instantaneous loss

Algorithm 2 PSBL Meta-Testing

```
Require: test MDPs $M$ sampled from $P(M)$ or out of the $P(M)$, network $q_\phi$, evaluation horizon $L$
Initialize interaction sequences $\tau$
for $k$ = 1, ..., $L$ do
  for $l$ = 1, ..., $H$ do
    At time step $t = kH + l$
    Sample $a_{t+1} \sim q_\phi(a_{t+1} \mid \tau)$
    Roll out action $a_{t+1}$ and collect data $o_{t+1}, r_t$
    Update $\tau$ with $\{r_t, t+1, d_{t+1}, o_{t+1}, a_{t+1},\}$
  end for
  if $k > b$ then
    Discard the earliest $k - b$ episode from $\tau$
  end if
end for
```

function ℓ_c based on the TD error as follows,

$$\ell_c = \mathbb{E}^\pi \Big[Q^\pi(a_{t+1}, \tau_t) - (r_{t+1} + \lambda \overline{Q}^\pi(a_{t+2}, \tau_{t+1})) \Big]^2, \tag{16}$$

where $a_{t+2} \sim \pi(a_{t+2} \mid \tau_{t+1})$ samples from the current policy π, $\overline{Q}$ is the target Q network.

To regress the policy network to the true optimal policy, we use simulated annealing (Ortega et al., 2019). It can gradually converge to the optimal policy. The Bayes-optimal policy can be computed as

$$P(a_{t+1} \mid \tau_t) \approx \pi(a_{t+1} \mid \tau_t) = \frac{\exp\left(\frac{1}{\alpha} Q(a_{t+1}, \tau_t)\right)}{\mathcal{Z}(\tau_t)}. \tag{17}$$

where $\mathcal{Z}(\tau_t) = \sum_{a'_{t+1}} \exp\left(\frac{1}{\alpha} Q(a'_{t+1} \mid \tau_t)\right)$.

According to (11) (17), we can get the final optimal policy loss ℓ_a as follows,

$$\ell_a = \mathbb{E}_{\tau_t} \left[D_{KL} \left(q_\phi(a_{t+1} \mid \tau_t) \middle\| \frac{\exp\left(\frac{1}{\alpha} Q(a_{t+1}, \tau_t)\right)}{\mathcal{Z}(\tau_t)} \right) \right]. \tag{18}$$

Therefore, the overall training objective is to maximize,

$$\mathcal{L} = \lambda_s \ell_s + \lambda_a \ell_a + \lambda_c \ell_c, \tag{19}$$

where the λ_s, λ_a, λ_c are the weighted coefficients of SL loss ℓ_s, optimal policy loss ℓ_a, critic loss ℓ_c, individually.

4.3. Meta-train an Lifelong In-context Learner

With the overall training objective (19), PSBL meta-trains the LILTrans network to perform in-context learning within a fixed horizon H^+ when adapting to an unseen task. However, if the test task distribution differs significantly from the training distribution, the fixed horizon H^+ may be insufficient for the agent to continuously improve its performance until convergence. Therefore, a dynamic window and time

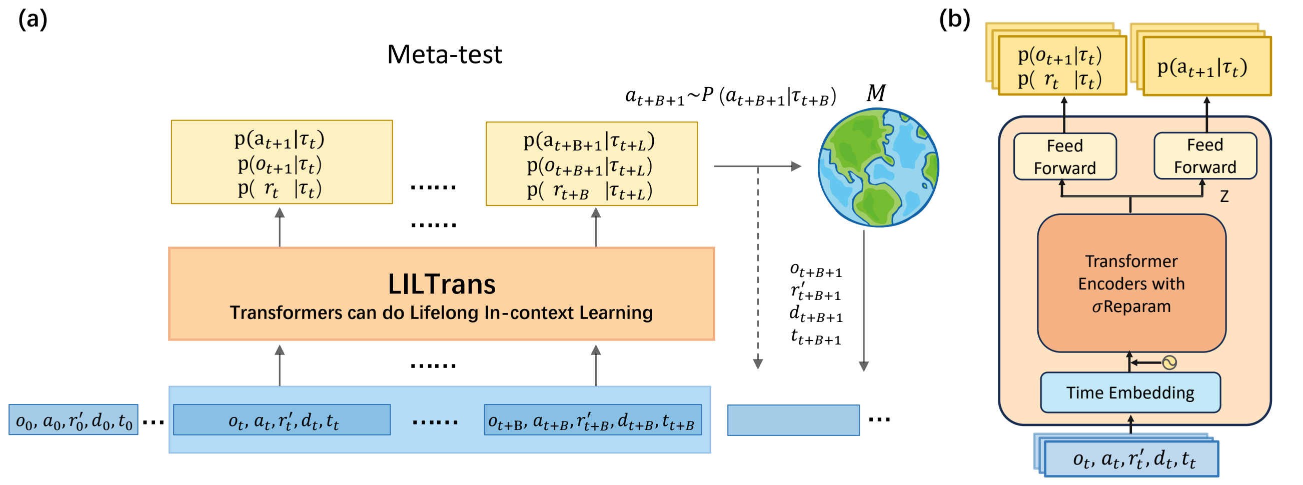
(a)
Meta-test
p(a_{t+1}|τ_t) p(o_{t+1}|τ_t) p(r_t|τ_t)
p(a_{t+B+1}|τ_{t+L}) p(o_{t+B+1}|τ_{t+L}) p(r_{t+B}|τ_{t+L})
a_{t+B+1}~P(a_{t+B+1}|τ_{t+B})
M
LILTrans
Transformers can do Lifelong In-context Learning
o_{t+B+1} r'_{t+B+1} d_{t+B+1} t_{t+B+1}
o_0, a_0, r'_0, d_0, t_0
o_t, a_t, r'_t, d_t, t_t
o_{t+B}, a_{t+B}, r'_{t+B}, d_{t+B}, t_{t+B}
(b)
p(o_{t+1}|τ_t) p(r_t|τ_t)
p(a_{t+1}|τ_t)
Feed Forward
Feed Forward
Z
Transformer Encoders with σReparam
Time Embedding
o_t, a_t, r'_t, d_t, t_t

Figure 1. **(a) Meta-Testing Procedure**: The LILTrans uses the interaction sequences τ to infer the posterior distribution of the Bayes-optimal policy, reward and next observation. Then the action is directly sampled from the approximate PPD of the Bayes-optimal policy. After executing the action, the data, including the next observation and reward, is collected to form the next interaction sequences τ. The dynamic window rolls forward, prompting LILTrans to carry out the next round of amortized Bayesian inference. This process allows the agent to progressively reduce uncertainty about the optimal action and improve its performance. In essence, LILTrans is performing lifelong in-context learning during the meta-test phase. **(b) LILTrans Network Architecture**: Once trained, LILTrans is robust to distributional shift by performing lifelong in-context learning.

embedding are specially designed in this work to enable an almost infinite evaluation horizon.

The Dynamic Window includes only the most recent b episodes, allowing the transformer-based network to process interaction sequences within this window. Thus, a transformer-based network trained on a fixed horizon can be applied to lifelong interactions by continuously rolling the dynamic window forward.

However, a dynamic window alone is insufficient for training a lifelong in-context learner. When the MDP is unknown, the agent's behavior should vary across different episodes. For instance, the agent might focus on exploration during the first b episodes and shift to exploitation in subsequent episodes. To distinguish these interaction sequences, we add time embedding to the interaction sequences τ. The updated interaction sequence is defined as follows:

$$\tau_{t_0:t_1} = \{\hat{t}_0, d_{t_0}, o_{t_0}, a_{t_0}, r_{t_0}, \dots, \hat{t}_1, d_{t_1}, o_{t_1}, a_{t_1}\} \quad (20)$$

where $\hat{t}_0 = \frac{t_0}{H^+}$ represents the relative time step, and d_t denotes whether the episode is reset at time step.

At meta-training time, the dynamic window rolls forward $N - b$ times, enabling the agent to learn to balance exploration and exploitation across different episodes. Consequently, the training objective for the Bayes-optimal policy is redefined as follows:

$$\mathcal{J}^+(\pi) = \mathbb{E}_{\tau,\pi}\left[\sum_{k=0}^{N}\sum_{l=0}^{H}\gamma^t R\left(\tau_{t_0:t}, o_{t+1}, a_{t+1}\right)\right], \quad (21)$$

where k is the episode index and l indicates the time index within an episode. $t = kH + l$ denotes the current time step, $t_0 = max[(k - b + 1)H, 0]$ represents the start time step of the dynamic window.

Therefore, we summarize our training procedure in Algorithm 1 and describe Meta-testing procedure in Figure 2 and Algorithm 2.

4.4. Adapting Transformers for Lifelong In-context Learning

In this work, a variant of transformer network is specifically proposed for tackling the Lifelong In-Context Learning problem, called LILTrans. Essentially, LILTrans functions as a state machine, making it well-suited for online learning.

The LILTrans, visualized in Figure 2a, is based on Transformer encoders with a time Embedding network and two feedforward networks. To prevent performance collapse during training, we adopt several existing techniques to modify the original structure from (Vaswani et al., 2017). Details can be found in Appendix B.1.

At time step t, the time Embedding network f_t first embeds the interaction sequences τ. The transformer Encoders then process the embedding and output two latent variables z_1, z_2,

$$z_1, z_2 = Encoders(f_t(\tau)). \quad (22)$$

The latent variables $Z = [z_1, z_2]$ provide a compact repre-

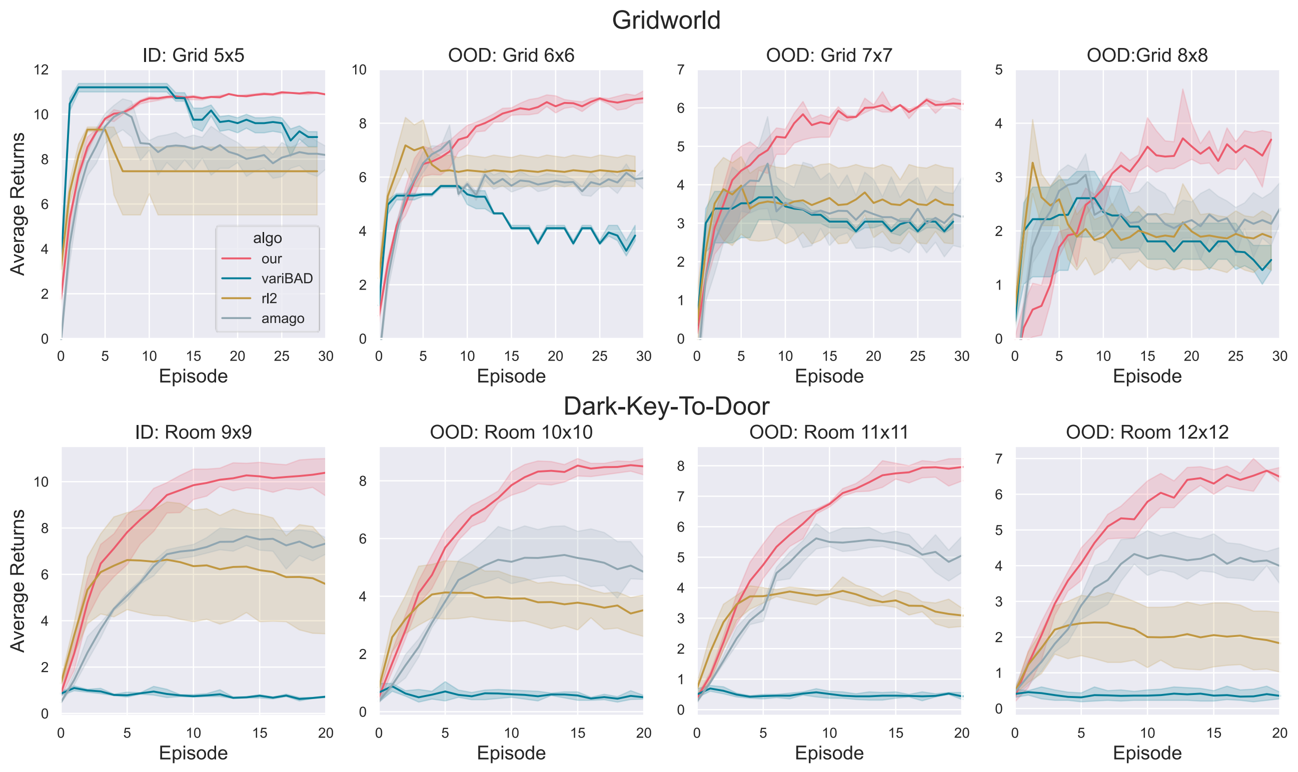

Figure 2. Average test performance in both ID and strictly OOD tasks in 64 parallel environments (using 3 seeds). PSBL enables the trained network to perform lifelong in-context learning across episodes with sparse rewards. Furthermore, PSBL significantly outperforms other methods in OOD tasks.

sentation of the interaction sequences τ, which are utilized in the state-action network $Q^{\pi}(a_{t+1}, Z)$.

Finally, the two latent variables are individually fed into two feedforward networks, f_1 and f_2. These networks return the approximate PPD of the optimal policy, rewards, and next observations. Therefore, the proposed LILTrans network can compute two functions g_ϕ and h_ϕ with the paramaters ϕ as follows,

$$\begin{aligned} \widetilde{a}_{t+1} &= h_\phi(a_t, (o_t, r_{t-1}), \tau_{t-1}) \\ (\widetilde{o}_{t+1}, \widetilde{r}_t) &= g_\phi(a_t, (o_t, r_{t-1}), \tau_{t-1}) \end{aligned} \tag{23}$$

that map the input current action, previous state-reward pair and previous interaction sequence $(a_t, (o_t, r_{t-1}), \tau_{t-1})$ to the next action $\widetilde{a}_{t+1}$ and the next state-reward pair $(\widetilde{o}_{t+1}, \widetilde{r}_t)$, respectively. Here, h_ϕ is the output function and g_ϕ is the state transition function. Therefore, equation (23) defines a state machine that is well-suited for online learning or in-context learning.

5. Experiments

In this section, various experiments, including discrete navigation tasks with sparse rewards and continuous control tasks with dense rewards, are conducted to evaluated the performance of PSBL. The experimental results demonstrate that the trained network in PSBL can perform Lifelong in-context Learning across episodes in tasks with sparse rewards and within episodes in tasks with dense rewards. Furthermore, PSBL significantly outperforms standard meta RL methods especially in OOD tasks. Finally, several ablation studies are conducted to to better understand the proposed method PSBL.

5.1. Discrete Navigation Meta-RL Tasks

We consider environments that requires task inference and online exploration. The first is Gridworld, a discrete navigation environment where the agent needs to infer the optimal route to an unknown goal position. The agent receives a reward of $+1$ upon reaching the goal and -1 otherwise. The second is Dark-Key-to-Door, a more complex navigation environment where the agent needs to find a key first and then use the key to open the door. The start, key and door position are unknown to the agent. The agent receives a reward of $+1$ both in finding the key and opening the door.

To evaluate the performance of PSBL in environments with

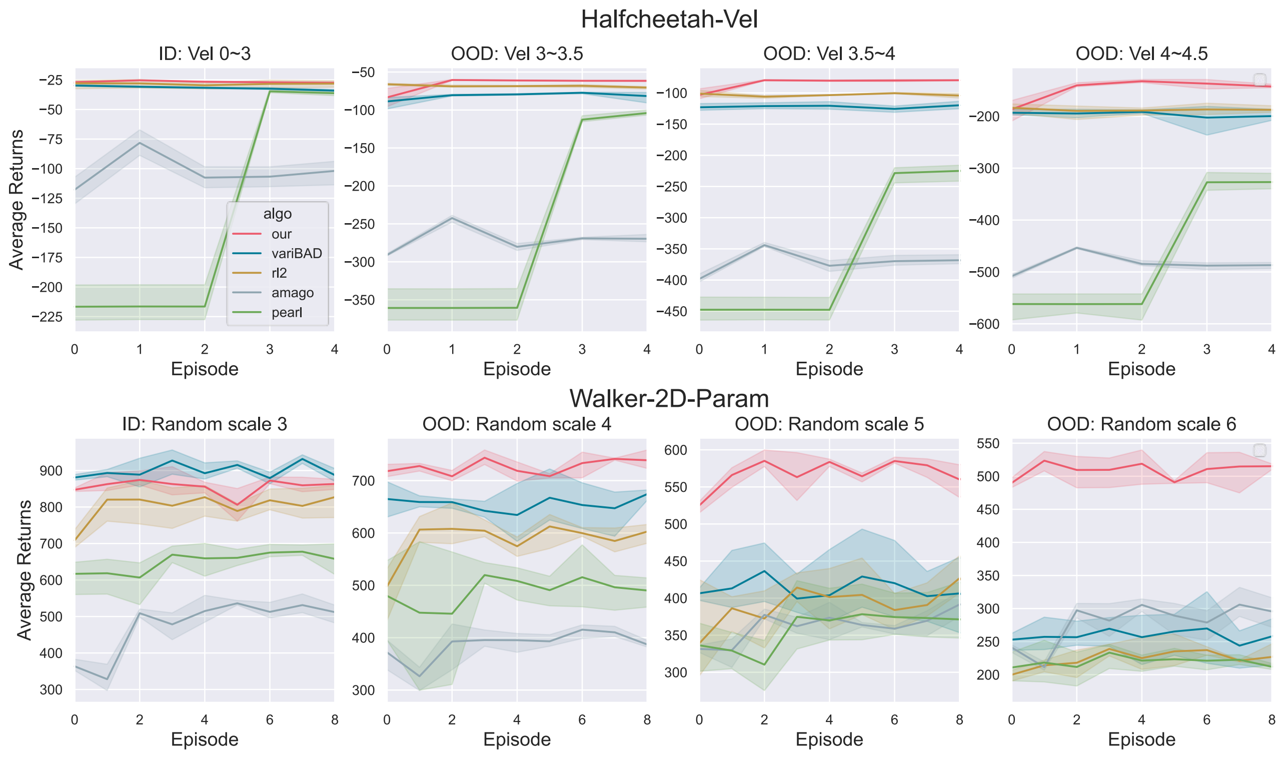

Figure 3. Average test performance in both ID and strictly OOD tasks in 64 parallel environments (using 3 seeds). PSBL enables the trained network to perform lifelong in-context learning within one or two episodes with dense rewards. Furthermore, PSBL significantly outperforms other methods in OOD tasks.

sparse rewards, we compare PSBL with state-of-the-art in-context RL methods and context-based Meta RL methods, including AMAGO (Grigsby et al., 2023), VariBAD (Zintgraf et al., 2019b), RL^2 (Duan et al., 2016). Detailed introductions of these methods are provided in Appendix B.2.

All of these methods are trained from scratch in the default environment setting. The detailed environmental setting are in Appendix B.3. Hyperparameters of PSBL during training are provided in Appendix B.4. The learning curves of all these algorithms, shown in the Appendix B.5, show that PSBL is much sample efficient than baselines. Once trained, we evaluate the trained networks both in in-distribution (ID) and out-of-distribution (OOD) tasks. Specifically speaking, in gridworld environments, we evaluate the networks trained on 5x5 grid worlds by navigating 5x5, 6x6, 7x7 and 8x8 grid worlds with randomly sampled goal positions. In Dark-Key-to-Door environments, we evaluate networks trained on 9x9 rooms worlds by navigating 9x9, 10x10, 11x11 and 12x12 rooms with randomly sampled start, key and door positions. For fair comparison, each method is evaluated in 64 parallel randomly sampled environments using 3 seeds, and the averaged returns are used to compare their test performance.

Figure 2 shows the evaluation results of PSBL and the baseline methods. It can be seen that at meta-test time, the trained network in PSBL progressively improve its performance until convergence both in ID and OOD test tasks through interactions with the environments. Furthermore, when the evaluation time step exceeds the learning horizon, PSBL continues to demonstrate improvement until convergence, whereas the performance of other baseline methods declines. These phenomenon indicates the emergence of Lifelong In-context Learning across episodes within the trained network in PSBL.

Through performing lifelong in-context learning, PSBL significantly outperform other methods in OOD tasks. Although the VariBAD method adapts more quickly to ID tasks than PSBL, its test performance declines dramatically when the task distribution shifts. This is because VariBAD's base policy network, being a simple feedforward network without online inference capability, struggles to map the unseen task distribution to a near-optimal policy.

Overall, the experimental results confirm that the PSBL-trained network can perform lifelong in-context learning across episodes in tasks with sparse rewards, significantly outperforming other methods in OOD tasks.

5.2. Mujoco Continuous Control Meta-RL Tasks

We demonstrate that PSBL can scale to continuous control tasks with dense rewards by evaluating it in Mujoco environments, which are commonly used as benchmarks (Zintgraf et al., 2019b; Rakelly et al., 2019). These locomotion tasks require online inference and fast adaptation across the reward functions (randomly generated target velocity for HalfCheetah-Vel) and the transition functions (randomly sampled environment properties for Walker-2D-Param).

To evaluate the performance of PSBL in continuous tasks with dense rewards, we compare the PSBL to four methods, including AMAGO, VariBAD, RL^2, PEARL (Rakelly et al., 2019). Detailed introductions of these methods are provided in Appendix B.2. All methods are trained from scratch in the default environment settings, detailed in Appendix B.3. Learning curves for all algorithms, shown in Appendix B.5, indicate that PSBL achieves comparable data efficiency to PEARL and is more sample-efficient than the other baselines.

Once trained, we evaluate the trained networks both in ID tasks and OOD tasks. Specifically speaking, in the benchmark of Half-Cheetah-Vel, we evaluate the networks trained at velocity of $0 \sim 3m/s$ by following the new target velocities sampled from $0 \sim 3, 3 \sim 3.5, 3.5 \sim 4$ and $4 \sim 4.5$ m/s . In the benchmark of Walker-2D-Param, we evaluate the networks trained in environments with parameters random scale at 3 by controlling the walker to walk forward with parameters random at 3, 4, 5 and 6. For fair comparison, these methods are evaluated in 64 parallel randomly sampled environments using 3 seeds, and the averaged returns are used to compare their test performance.

Figure 4 presents the experimental results in continuous control tasks. We can observe that PSBL adapts to OOD tasks almost within the first or second episodes, achieving significantly higher average returns than other methods especially in OOD tasks. That is because PSBL quickly reduces the uncertainty of near-optimal policy in tasks with dense rewards, allowing the uncertainty to converge to a rough point mass within one or two episodes. Consequently, PSBL enables lifelong in-context learning within episodes for these tasks. Moreover, PSBL significantly outperforms other methods in OOD tasks, and its superiority becomes more pronounced as the distance between test and training task distributions increases.

5.3. Ablations

In this section, we ablate our approach to better understand its key features. Several ablation experiments are conducted to evaluate the importance of the dynamic window and its hyperparameters. In the grid-world tasks, the dynamic window in PSBL is set to contain 10 episodes (Memory Length) while the training horizon contains 20 episodes (Horizon Length). We consider two other configurations for the dynamic window:

- PSBL method with a horizon length of 20 and a memory length of 20.
- PSBL method with a horizon length of 12 and a memory length of 6.

The test performance of these variants is evaluated in the OOD task Grid-World 6x6. The results, shown in Figure 4, demonstrate that PSBL with a horizon length of 20 and a memory length of 10 outperforms PSBL with a horizon length of 20 and a memory length of 20, as the dynamic window in the latter does not progress forward. Furthermore, the results show that PSBL with a horizon length of 20 and a memory length of 10 outperforms PSBL with a horizon length of 12 and a memory length of 6. Hence, a longer memory length and horizon length prove beneficial for retaining the past interaction sequence and improving policy in the context

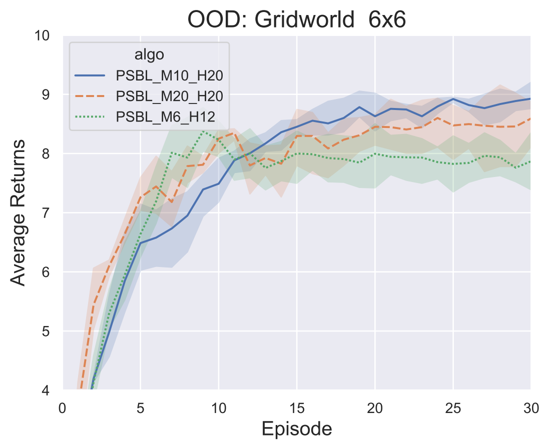

Figure 4. Ablation Experiment Results: We compare PSBL to two variants with different horizon and memory lengths.

6. Conclusion

In this work, we propose PSBL, a new Meta-RL method robust to distribution shifts through lifelong in-context learning. PSBL is evaluated on discrete navigation tasks and continuous control tasks, where test task distributions differ significantly from training distributions. Experimental results demonstrate that PSBL enables lifelong in-context learning across episodes for tasks with sparse rewards and within episodes for tasks with dense rewards. Consequently, through performing Lifelong in-context Learning, PSBL significantly outperforms standard Meta-RL methods, especially in OOD tasks.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgements

This work was supported by the National Natural Science Foundation, China (No.62173300)

References

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Beck, J., Vuorio, R., Liu, E. Z., Xiong, Z., Zintgraf, L., Finn, C., and Whiteson, S. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 1282–1289. PMLR, 2019.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. Rl ˆ2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Fakoor, R., Chaudhari, P., Soatto, S., and Smola, A. J. Meta-q-learning. *arXiv preprint arXiv:1910.00125*, 2019.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Fu, H., Tang, H., Hao, J., Chen, C., Feng, X., Li, D., and Liu, W. Towards effective context for meta-reinforcement learning: an approach based on contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7457–7465, 2021.

Genewein, T., Delétang, G., Ruoss, A., Wenliang, L. K., Catt, E., Dutordoir, V., Grau-Moya, J., Orseau, L., Hutter, M., and Veness, J. Memory-based meta-learning on non-stationary distributions. *arXiv preprint arXiv:2302.03067*, 2023.

Grigsby, J., Fan, L., and Zhu, Y. Amago: Scalable in-context reinforcement learning for adaptive agents. *arXiv preprint arXiv:2310.09971*, 2023.

Humplik, J., Galashov, A., Hasenclever, L., Ortega, P. A., Teh, Y. W., and Heess, N. Meta reinforcement learning as task inference. *arXiv preprint arXiv:1905.06424*, 2019.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Laskin, M., Wang, L., Oh, J., Parisotto, E., Spencer, S., Steigerwald, R., Strouse, D., Hansen, S., Filos, A., Brooks, E., et al. In-context reinforcement learning with algorithm distillation. *arXiv preprint arXiv:2210.14215*, 2022.

Liu, E. Z., Raghunathan, A., Liang, P., and Finn, C. Decoupling exploration and exploitation for meta-reinforcement learning without sacrifices. In *International conference on machine learning*, pp. 6925–6935. PMLR, 2021.

Melo, L. C. Transformers are meta-reinforcement learners. In *international conference on machine learning*, pp. 15340–15359. PMLR, 2022.

Mendonca, R., Geng, X., Finn, C., and Levine, S. Meta-reinforcement learning robust to distributional shift via model identification and experience relabeling. *arXiv preprint arXiv:2006.07178*, 2020.

Mikulik, V., Delétang, G., McGrath, T., Genewein, T., Martic, M., Legg, S., and Ortega, P. Meta-trained agents implement bayes-optimal agents. *Advances in neural information processing systems*, 33:18691–18703, 2020.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017.

Muller, S., Hollmann, N., Arango, S. P., Grabocka, J., and Hutter, F. Transformers can do bayesian inference. *ArXiv*, abs/2112.10510, 2021. URL `https://api.semanticscholar.org/CorpusID:245334722`.

Ortega, P. A., Wang, J. X., Rowland, M., Genewein, T., Kurth-Nelson, Z., Pascanu, R., Heess, N., Veness, J., Pritzel, A., Sprechmann, P., et al. Meta-learning of sequential strategies. *arXiv preprint arXiv:1905.03030*, 2019.

Osband, I., Russo, D., and Van Roy, B. (more) efficient reinforcement learning via posterior sampling. *Advances in Neural Information Processing Systems*, 26, 2013.

Rajeswaran, A., Lowrey, K., Todorov, E. V., and Kakade, S. M. Towards generalization and simplicity in continuous control. *Advances in Neural Information Processing Systems*, 30, 2017.

Rakelly, K., Zhou, A., Finn, C., Levine, S., and Quillen, D. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pp. 5331–5340. PMLR, 2019.

Ritchie, D., Horsfall, P., and Goodman, N. D. Deep amortized inference for probabilistic programs. *arXiv preprint arXiv:1610.05735*, 2016.

Sasso, R., Conserva, M., and Rauber, P. Posterior sampling for deep reinforcement learning. *arXiv preprint arXiv:2305.00477*, 2023.

Shleifer, S., Weston, J., and Ott, M. Normformer: Improved transformer pretraining with extra normalization. *arXiv preprint arXiv:2110.09456*, 2021.

Song, X., Jiang, Y., Tu, S., Du, Y., and Neyshabur, B. Observational overfitting in reinforcement learning. *arXiv preprint arXiv:1912.02975*, 2019.

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.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023.

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.

Zhai, S., Likhomanenko, T., Littwin, E., Busbridge, D., Ramapuram, J., Zhang, Y., Gu, J., and Susskind, J. M. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, pp. 40770–40803. PMLR, 2023.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018.

Zhang, J., Wang, J., Hu, H., Chen, T., Chen, Y., Fan, C., and Zhang, C. Metacure: Meta reinforcement learning with empowerment-driven exploration. In *International Conference on Machine Learning*, pp. 12600–12610. PMLR, 2021.

Zintgraf, L., Shiarli, K., Kurin, V., Hofmann, K., and Whiteson, S. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pp. 7693–7702. PMLR, 2019a.

Zintgraf, L., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., and Whiteson, S. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019b.

Zintgraf, L. M., Feng, L., Lu, C., Igl, M., Hartikainen, K., Hofmann, K., and Whiteson, S. Exploration in approximate hyper-state space for meta reinforcement learning. In *International Conference on Machine Learning*, pp. 12991–13001. PMLR, 2021.

A. End to End PPD Approximation Guarantee

A.1. Supervised Learning Loss

Insight 1. The proposed SL loss ℓ_s is equal to the expectation of the cross-entropy between the PPD and its approximation q_ϕ: $\ell_s = \mathbb{E} * \tau_t \left[H\left(P(\cdot \mid \tau_t), q * \delta(\cdot \mid \tau_t)\right)\right]$

Proof.

$$\begin{aligned}\ell_s &= -\int_{o_{t+1}, r_t, \tau_t} p(o_{t+1}, r_t, \tau_t) \log q_\phi(o_{t+1}, r_t \mid \tau_t) = -\int_{\tau_t} p(\tau_t) \int_{o_{t+1}, r_t} p(o_{t+1}, r_t \mid \tau_t) \log q_\phi(o_{t+1}, r_t \mid \tau_t) \\ &= \int_{\tau_t} p(\tau_t) \mathrm{H}\left(p(\cdot \mid \tau_t), q_\phi(\cdot \mid \tau_t)\right) = \mathbb{E} * \tau_t \sim p(\tau_t) \left[\mathrm{H}\left(p(\cdot \mid \tau_t), q * \phi(\cdot \mid \tau_t)\right)\right]\end{aligned} \tag{24}$$

□

A.2. Actor Loss

Insight 2. The proposed actor loss ℓ_a is equal to the expectation of the cross-entropy between the PPD and its approximation, up to an additive constant: q_ϕ: $\ell_a = \mathbb{E} * \tau_t \left[H\left(P(\cdot \mid \tau_t), q * \delta(\cdot \mid \tau_t)\right)\right] + C$

Proof.

$$\begin{aligned}&\mathbb{E} * \tau_t \left[D * KL\left(q_\phi(a_{t+1} \mid \tau_t) \middle\| P(a_{t+1} \mid \tau_t)\right)\right] \\ &= -\mathbb{E} * \tau_t \left[\int * a_{t+1} P(a_{t+1} \mid \tau_t) \log \frac{q_\delta(a_{t+1} \mid \tau_t)}{P(a_{t+1} \mid \tau_t)}\right] \\ &= -\mathbb{E} * \tau_t \left[\int * a_{t+1} P(a_{t+1} \mid \tau_t) \log q_\delta(a_{t+1} \mid \tau_t)\right] + \mathbb{E} * \tau_t \left[\int * a_{t+1} P(a_{t+1} \mid \tau_t) \log P(a_{t+1} \mid \tau_t)\right] \\ &= \mathbb{E} * \tau_t \left[H\left(P(\cdot \mid \tau_t), q * \delta(\cdot \mid \tau_t)\right)\right] - \mathbb{E} * \tau_t \left[H\left(P(\cdot \mid \tau_t)\right)\right] \\ &= \mathbb{E} * \tau_t \left[H\left(P(\cdot \mid \tau_t), q_\delta(\cdot \mid \tau_t)\right)\right] + C\end{aligned} \tag{25}$$

□

B. Implementation Details

B.1. LILTrans Network Architecture

To prevent performance collapse during training, particularly during long-horizon meta training, we incorporate three existing methods: extra LayerNorms from Normformer (Shleifer et al., 2021), $\alpha-$ Reparam from (Zhai et al., 2023), and the use of Leaky ReLU from AMAGO (Grigsby et al., 2023).

B.2. Baselines

- **AMAGO** (Grigsby et al., 2023). A Transformers-based In-context Reinforcement Learning method which specializes in meta-learning, generalization and long-term memory. We use the open-source reference implementation of AMAGO at https://github.com/ut-austin-rpl/amago. We keep all hyperparameters of AMAGO as the Default.
- **VariBAD** (Zintgraf et al., 2019b). A variational Bayes-Adaptive Deep RL which can online infer the task distribution and incorporate task uncertainty during action selection. We use the open-source reference implementation of variBAD at https://github.com/lmzintgraf/varibad to report the results of variBAD method. We keep all hyperparameters of VariBAD as the Default.
- **PEARL** (Rakelly et al., 2019). A Probabilistic Context Variables based Meta RL method, which infer latent task variables online and Posterior samples from the inferred distribution. We use the open-source reference implementation of PEARL at https://github.com/ut-austin-rpl/amago. We keep all hyperparameters of PEARL as the Default.

- RL^2. (Duan et al., 2016) A classic context-based RL which utilize a recurrent neural network to quickly adapt to the new task.We use the open-source reference implementation of RL^2 at https://github.com/katerakelly/oysterto report the results of RL^2 method. We keep all hyperparameters of RL^2 as the Default.

B.3. Environments Setting

- **Gridworld**. The horizon is set to the default value of 15, as described in VarBAD (Zintgraf et al., 2019b). The gridworld is configured as a 5x5 grid. During training, the goal position is randomly sampled from any of the 25 grids except (0,0), (0,1), (1,0), and (1,1). During meta-testing, the goal position is randomly selected from the 5x5 grid in ID tasks and from 6x6, 7x7, and 8x8 grids in OOD tasks.
- **Dark-Key-To-Door** Consistent with AMAGO (Grigsby et al., 2023), the horizon is set to the default value of 50. The training environment is a 9x9 room, with the start, key, and door positions randomly sampled from all possible locations within the room. During meta-testing, these positions are randomly sampled from the 9x9 room in ID tasks, and from 10x10, 11x11, and 12x12 rooms in OOD tasks.
- **HalfCheetah-Vel** Similar to VarBAD (Zintgraf et al., 2019b), the horizon is set to the default value of 200. The target speed during training is randomly sampled from the range . During meta-testing, the target is totally randomly sampled from the $0 \sim 3$ in ID tasks and from $3 \sim 3.5$, $3.5 \sim 4$ and $4 \sim 4.5$ in OOD tasks .
- **Walker-2D-Param** As in VarBAD (Zintgraf et al., 2019b), the horizon is set to the default value of 200. During training, the environment parameters, including body mass, degree of freedom (dof) damping, body inertia, and geom friction, are randomly altered by a factor of 3 at the beginning of each episode. During evaluation, the random scaling factor for these parameters is set to 3 for ID tasks, and to 4, 5, and 6 for OOD tasks.

B.4. Hyperparameters

The network architecture details of PSBL for all test environments are listed in Table 1. The training hyperparameters for PSBL across all test environments are provided in Table 2.

Table 1. PSBL Network Architecture Details For All Test Environments.

TRANSFORMER ENCODERS	
MODEL DIM	256
FF DIM	1024
HEADS	8
LAYERS	3
OTHER NETWORKS	
TIME EMBEDDING	(128,128,64)
CRITIC	(256,256)
FEEDFORWARD NETWORK	(128,128)

B.5. Learning Curves in Experiments

Figure 5 shows the learning curves for all environments presented in Section 5 for all methods. Since the open-source reference implementation of PEARL can only address continuous tasks, the PEARL method is evaluated exclusively on continuous tasks, including HalfCheetah-Vel and Walker-2D-Param.

It is evident that our method, PSBL, demonstrates higher sample efficiency than other methods in discrete tasks. However, in continuous tasks, PEARL is slightly more sample-efficient in training than PSBL. This difference is attributed to the memory-augmented network in PSBL, which is based on transformers and naturally requires more training data than recurrent networks. In the Dark-Key-To-Door tasks, PSBL achieves significantly higher average returns than AMAGO, VariBAD, and RL^2. Despite extensive hyperparameter tuning and more than 20 trials, the VariBAD method only converges to suboptimal performance with average returns near 1 in the Dark-Key-To-Door tasks

Table 2. PSBL Trainning Hyperparameters

	GRIDWORLD	DARK-KEY-DOOR	HALFCHEEH-VEL	WALKER-2D-PARAM
LEARNING RATE	5E-5	5E-5	3E-4	5E-4
L2 PENALTY	1E-4	1E-4	1E-4	1E-4
BATCH SIZE	24	32	24	24
MAX BUFFER SIZE	20000	20000	20000	20000
GRADIENT UPDATES PER EPOCH	2000	2000	2000	2000
TARGET UPDATE τ	0.03	0.03	0.03	0.03
GRADIENT CLIP (NORM)	1	1	1	1
ACTOR LOSS WEIGHT	1	1	1	1
CRITIC LOSS WEIGHT	10	10	10	10
SL LOSS WEIGHT	0.5	0.8	0.5	0.5
PARALLEL ACTORS	24	24	24	24
MEMORY LENGTH(EPISODES)	10	8	2	3
HORIZON LENGTH (EPISODES)	20	10	4	5
EXPLORATION MAX ϵAT EP. START	0.8	0.8	0.8	0.8
EXPLORATION MAX ϵ AT EP. END	0.05	0.05	0.05	0.05

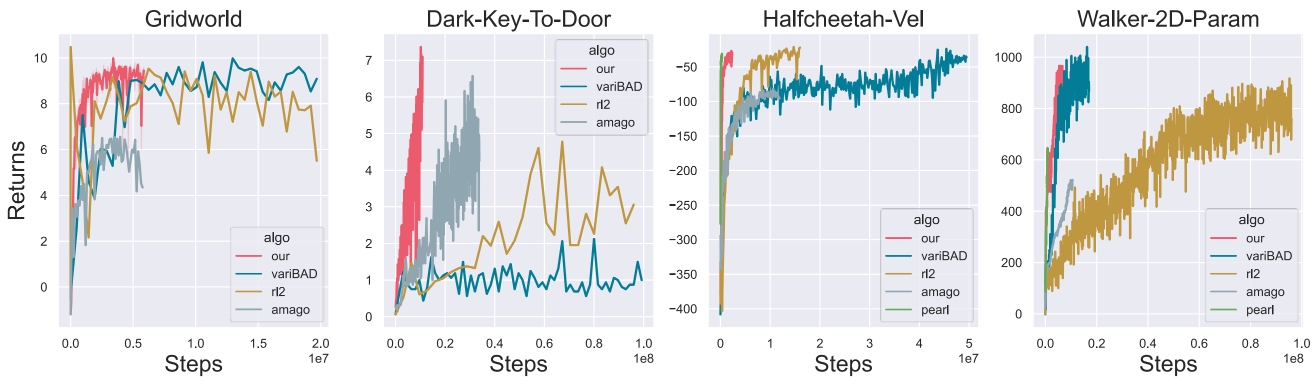

Figure 5. The learning curves for results presented in Section 5. Each figure shows the average returns per episodes for these methods.