Statistical Guarantees for Lifelong Reinforcement Learning using PAC-Bayes Theory

Zhi Zhang[1]   **Chris Chow**[2]   **Yasi Zhang**[1]   **Yanchao Sun**[3]
Haochen Zhang[1]   **Eric Hanchen Jiang**[1]   **Han Liu**[4]   **Furong Huang**[5]
Yuchen Cui[1]   **Oscar Hernan Madrid Padilla**[1]

[1]University of California, Los Angeles   [2]Niantic Labs   [3]Apple Inc.
[4]Northwestern University   [5]University of Maryland, College Park

Abstract

Lifelong reinforcement learning (RL) has been developed as a paradigm for extending single-task RL to more realistic, dynamic settings. In lifelong RL, the "life" of an RL agent is modeled as a stream of tasks drawn from a task distribution. We propose EPIC (Empirical PAC-Bayes that Improves Continuously), a novel algorithm designed for lifelong RL using PAC-Bayes theory. EPIC learns a shared policy distribution, referred to as the *world policy*, which enables rapid adaptation to new tasks while retaining valuable knowledge from previous experiences. Our theoretical analysis establishes a relationship between the algorithm's generalization performance and the number of prior tasks preserved in memory. We also derive the sample complexity of EPIC in terms of RL regret. Extensive experiments on a variety of environments demonstrate that EPIC significantly outperforms existing methods in lifelong RL, offering both theoretical guarantees and practical efficacy through the use of the world policy.

1 INTRODUCTION

Deep reinforcement learning (RL) has excelled in challenging tasks including abstract strategy games (Silver et al., 2017, 2016), visual navigation (Zhu et al., 2017), and control (Mnih et al., 2015; Lillicrap et al., 2015). However, RL is a data intensive learning paradigm, therefore training a policy for every task from scratch is computationally expensive and time-consuming. In reality, many tasks an agent encounters are not entirely

novel but instead belong to a broader task distribution, meaning they share commonalities that can be leveraged. This insight highlights the inefficiency of retraining for every individual task. Lifelong RL, also known as continual RL, emerges as a promising framework where an agent interacts with a sequence of tasks, continuously adapting and improving its policy by leveraging knowledge from past task instances (Khetarpal et al., 2022).

In lifelong RL, an agent's objectives are primarily to achieve **fast adaptation** with limited samples and effective **knowledge retention** (Abel et al., 2024). In other words, lifelong RL agents experience a stability-plasticity dilemma, where the agent must balance retaining useful long-term knowledge with the ability to rapidly adapt to new situations.

Recent methods addressing knowledge retention and transfer in lifelong RL include Q-value function transfer (Lecarpentier et al., 2021), optimizing Q-value function initialization (Abel et al., 2018), decomposing the value function into permanent and transient components (Anand and Precup, 2023), reusing knowledge by sampling from past experiences (Kessler et al., 2023), detecting change points in rewards and environment dynamics (Steinparz et al., 2022), and using a Bayesian approach to learn a common task distribution for better data efficiency and transfer (Fu et al., 2022).

In lifelong RL, domain shifts induce non-stationarity, which occurs not only due to changing transition dynamics and reward functions, but also through variations in available actions or decisions over time (Boutilier et al., 2018; Chandak et al., 2020). Such scenarios are common in real-world applications. For example, in robotics, additional control components are integrated throughout the robot's lifetime, and in medical decision support systems, new treatments and medications must be incorporated.

Furthermore, tasks encountered over an agent's lifetime can be highly diverse, yet certain high-level strategies

that can be shared across tasks. Not only should the world model (Ha and Schmidhuber, 2018; Fu et al., 2022; Anand and Precup, 2023), which captures the general knowledge distribution of tasks, be continuously refined, but it is also crucial for an agent to develop a *world policy*. The key consideration of this *world policy* is to adapt the parameters of the policy so they are captured by a global distribution, which represents the uncertainty over policy parameters. This allows for better generalization and adaptability across tasks.

Motivated by the above need, we raise two questions:

1. Can we extract the common structure present in policies from previously encountered tasks, allowing the agent to quickly learn the policy specific to new task to enable fast adaptation *with* theoretical guarantees?

2. How many samples are required to achieve a given level of performance?

To answer these two questions, we develop a unified framework based on a Bayesian method that learns a rapidly adaptable policy distribution from past tasks, retaining valuable information while remaining capable of quickly adapting to unseen situations.

We also provide a theoretical analysis of the algorithm's generalization performance relative to the number of effective tasks and retained knowledge in the finite Markov Decision Process (MDP) setting, along with its sample complexity to demonstrate efficiency.

When addressing the first question, we must also account for both catastrophic forgetting and generalizability – the aforementioned *stability-plasticity dilemma*. Agents that can quickly solve traditional RL problems risk abruptly losing performance on tasks they have seen before due to their flexibility or plasticity. On the other hand, agents that do not forget any of their past experience may give up a measure of their plasticity. These issues are central in lifelong RL, and can be approached from a Bayesian perspective (Khetarpal et al., 2022). Bayesian methods have been applied to meta learning (Amit and Meir, 2018), lifelong learning for bandits (Flynn et al., 2022), and learning controls for robots in multiple environments (Majumdar et al., 2021), aiming to learn a fast adapted policy. Instead of learning a specific policy, we leverage the PAC-Bayes theory to learn a distribution of policy hypotheses shared across multiple tasks. Further details about PAC-Bayes theory can be found in Section 3.3 and the Related Works section in Appendix §2. When a new task arises, we can initialize a policy hypothesis by sampling from this learned distribution. A well-constructed distribution of hypotheses promotes effective long-term memory, mitigating catastrophic

forgetting. Unlike prior methods, we sample a random policy function according to this distribution. This function sampling approach is seen in modern popular deep learning methods, for example, in-context learning (Garg et al., 2022).

For the second question, an agent has to keep learning as well as forgetting. Too much experienced knowledge kept in memory may decrease the learning efficiency, while too little may be insufficient to learn an effective policy distribution. To address the second question, we derive a relationship between the performance of our algorithm and the number of experienced tasks (denoted as N in a later section) that need to be retained in the agent's memory, based on PAC-Bayes theory. We use the negative expected long term rewards, where the expectation is taken with respect to tasks and policies (also known as the generalization error), as a measure of the algorithm's performance from a statistical perspective.

From our theoretical result, where we provide an expression of this relationship, we discovered a trade-off between this value and the algorithm's performance, which aligns with natural intuition, a double sided effect. In practice, our expression allows us to optimize the performance of our algorithm by optimizing N, although we recommend using hyperparameter tuning. Furthermore, to demonstrate the efficiency of our algorithm, we derive its sample complexity from a RL regret perspective, showing that our algorithm learns an optimal policy as more tasks encountered.

Our Contributions. In this work, we introduce a novel PAC-Bayes framework tailored to lifelong RL, addressing critical challenges including changing decisions, catastrophic forgetting and efficient knowledge retention. Our contributions are summarized as follows:

- We propose EPIC (Empirical PAC-Bayes that Improves Continuously), a lifelong RL algorithm that leverages PAC-Bayes theory to learn a shared policy distribution, referred to as the *world policy*. This world policy enables the agent to quickly adapt to new tasks while retaining useful knowledge from past experiences, providing theoretical guarantees of generalization across tasks.

- We derive a novel PAC-Bayes bound for lifelong RL and provide a theoretical analysis that links long-term rewards to the number of retained past tasks, ensuring a balance between memory usage and performance across diverse tasks. We provide a sample complexity of our approach in terms of RL regret.

- We evaluate EPIC through extensive numerical experiments with common lifelong RL benchmarks, as well as additional environments we created. Our

results show EPIC outperforms prior methods. These results underscore EPIC's effectiveness in lifelong learning scenarios, offering a robust and theoretically grounded solution for continual adaptation in RL.

2 RELATED WORKS

Lifelong Reinforcement Learning: Lifelong learning has been a crucial area of research in machine learning, where the goal is to develop agents that can continuously adapt to new tasks while retaining knowledge from previous experiences. Early foundational works, such as Naik and Mammone (1992) and Thrun and Pratt (1998), explored the basic principles of lifelong learning, setting the stage for more advanced methods. Subsequent research has focused on mitigating catastrophic forgetting and enhancing data efficiency, which are critical challenges in lifelong learning scenarios. Various approaches have been proposed to improve adaptation in lifelong learning. Saxe et al. (2014); Kirkpatrick et al. (2017b); Krähenbühl et al. (2016); Salimans and Kingma (2016) explored strategies for better initialization in deep networks.

Lifelong RL, as an extension of lifelong learning, naturally aligns with the agent-environment interaction framework, making it ideal for continual learning (Khetarpal et al., 2022). Prior works (Lecarpentier et al., 2021; Abel et al., 2018) emphasize value transfer and initialization to boost learning efficiency, while Chandak et al. (2020) tackles the challenge of evolving action sets. Anand and Precup (2023) introduces a dual-component value function approach for balancing long-term stability and short-term adaptability, and Fu et al. (2022) develops a model-based Bayesian framework that enhances both forward and backward transfer by extracting common structures across tasks. Lifelong RL has been further formalized as a framework where agents continuously learn and adapt, moving beyond static solutions (Abel et al., 2024).

Recent baseline algorithms for lifelong RL have made significant advancements. Continual Dreamer (Kessler et al., 2023) employs ensemble networks and is task-agnostic, leveraging a world model that can generate tasks for improving learning efficiency. VBLRL (Fu et al., 2022) is a model-based method that learns a Bayesian posterior distribution shared across tasks to increase sample efficiency in related tasks. LPG-FTW (Mendez et al., 2020) is a policy-gradient-based lifelong method that uses data from previously seen tasks to train policy networks, accelerating the learning of new tasks. EWC (Kirkpatrick et al., 2017a) is a single-model lifelong RL algorithm that avoids forgetting by imposing a quadratic penalty, pulling weights back towards values important for previously learned tasks. T-HiP-

MDP (Killian et al., 2017) is a model-based method that models related tasks using low-dimensional latent embeddings and a Bayesian Neural Network, which captures both shared dynamics across tasks and individual task variations.

Our approach introduces a lifelong RL framework integrating PAC-Bayes theory to learn a policy distribution in non-stationary environments, ensuring effective knowledge retention and adaptability across tasks throughout the agent's lifetime.

PAC-Bayes Theory: PAC-Bayes theory (McAllester, 1999) has been extensively used in supervised and deep learning to study generalization bounds (Langford and Shawe-Taylor, 2002; Seeger, 2002; Germain et al., 2009; Dziugaite et al., 2020; Neyshabur et al., 2018, 2017). In recent years, PAC-Bayes theory has been applied to reinforcement learning (RL) (Schulman et al., 2015; Fard and Pineau, 2010; Fard et al., 2012; Majumdar et al., 2021; Veer and Majumdar, 2020), primarily focused on single-task or offline settings, providing a framework for deriving generalization bounds in dynamic and uncertain environments. Our method uniquely integrates PAC-Bayes theory into lifelong RL, providing a framework for continuous learning and adaptation. Mbacke et al. (2023) is a recent seminal work, both their and our methods make a similar contribution to provide statistical guarantees for particular machine learning methods by using PAC-Bayes theory.

3 PRELIMINARIES

3.1 Reinforcement Learning

In RL, an agent interacts with the environment by taking actions, observing states and receiving rewards. The environment is modeled by a Markov Decision Process (MDP), which is denoted by a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma, \nu \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, T is the transition kernel, R is the reward function, $\gamma \in (0, 1)$ is the discount factor, and ν is the initial state distribution.

A trajectory $\tau \sim \pi$ generated by policy π is a sequence $s_1, a_1, r_1, s_2, a_2, \cdots$, where $s_1 \sim \nu$, $a_t \sim \pi(a|s_t)$, $s_{t+1} \sim T(s|s_t, a_t)$ and $r_t = R(s_t, a_t)$. The goal of an RL agent is to find an optimal policy π^* that maximizes the expected total rewards $J(\pi) = \mathbb{E}_{\tau \sim \pi}[r(\tau)] = \mathbb{E}_{s_1, a_1, \cdots \sim \nu, \pi, T, R}[\sum_{t=1}^{\infty} \gamma^{t-1} r_t]$.

3.2 Lifelong Reinforcement Learning

In lifelong RL, the agent interacts with a (potentially infinite) sequence of tasks, which come from an underlying task distribution (Khetarpal et al., 2022), denoted as $\mathcal{D}_i$, $i = 1, \ldots, \infty$. Suppose that tasks share the same γ, but may have different $\mathcal{S}, \mathcal{A}$, transition probabilities T and rewards R. The learning process is:

1. Initialize a policy π_0;

2. Sample a task (MDP) $\mathcal{M}_i \sim \mathcal{D}_i$;

3. Starting from π_0, learn a policy π_i for task $\mathcal{M}_i$ to maximize rewards.

An effective lifelong RL agent should quickly adapt to new tasks that it encounters throughout its life.

3.3 PAC-Bayes Theory

PAC-Bayes analysis applies to learning algorithms that output a distribution over hypotheses $h \in \mathcal{H}$. This refers to h is sampled independently from a distribution over functions in a function class $\mathcal{H}$. For example, for a linear predictor of d dimension, $h(x) = \langle w, x \rangle$, we let $w \sim \mathcal{N}(0, I_d)$. Generally, such algorithms will be given a prior distribution $\underline{P} \in \mathcal{P}$ at the beginning and learn a posterior distribution $P \in \mathcal{P}$ after observing training data samples $\{z_i\}_{i=1}^N$. We define the expected loss (generalization error) $l_{\mathcal{D}}(P) = \mathbb{E}_{h \sim P}[\mathbb{E}_{z \sim \mathcal{D}}[h(z)]]$, and the empirical loss (training error) $l_{\mathcal{S}}(P) = \mathbb{E}_{h \sim P}\left[\frac{1}{N}\sum_{i=1}^N h(z_i)\right]$, which are under the expectation of hypothesis $h \sim P$.

The main application of PAC-Bayes analysis in machine learning is to produce high-confidence bounds for the true or generalization error in terms of the training error plus $\mathcal{R}(\mathbb{D}_{KL}(P\|\underline{P}))$, which is a function of the KL divergence between the prior and posterior distributions, as shown below (McAllester, 1999),

$$l_{\mathcal{D}}(P) \leq U(P) \coloneqq l_S(P) + \mathcal{R}(\mathbb{D}_{KL}(P\|\underline{P})), \quad (1)$$

with

$$\begin{aligned} &\mathcal{R}(\mathbb{D}_{KL}(P\|\underline{P})) \\ &\coloneqq \sqrt{\frac{1}{2N}\left[\mathbb{D}_{KL}\left(P\|\underline{P}\right) + \log\left(2N^{1/2}/\delta\right)\right]}, \end{aligned} \quad (2)$$

where $U(P)$ in right-hand side of Equation (1) is called the generalization error bound that depends on P, and minimization of this bound leads to generalization error guarantees.

4 METHODS

We propose a PAC-Bayes lifelong RL algorithm, EPIC (Algorithm 1), to minimize the novel bound in (3). The algorithm utilizes a Bayesian posterior to distill the common policy distribution learned from previous tasks, which is then used to sample the policy and serves as a prior for new tasks. We provide a generalization guarantee for EPIC in Theorem 4.3. Furthermore, we employ the Gaussian family for the posterior and prior in EPICG (Algorithm 2). The sample complexity of EPICG is given in Theorem 4.4.

4.1 PAC-Bayes Framework for Lifelong RL

We learn a general policy distribution P for lifelong RL by leveraging the core concept of the PAC-Bayes Method. We explicitly formulate $U(P)$ for the lifelong RL setting and employ it to propose an algorithm that learn the P by minimizing $U(P)$ to accomplish the lifelong learning objective.

Define $\mathcal{P}$ as the whole policy space for P. Rather than considering a general distribution P for hypotheses where Π can be infinite, we let $\mathcal{P}$ be parameterized by $\theta \in \mathbb{R}^d$ such that $\theta \sim P$. Note θ could be a neural network.

Naturally, the distribution P is the posterior distribution of policy θ in the PAC-Bayes framework. Then let $\underline{P}$ be the prior distribution of the parameter. In the lifelong setting, as the tasks stream in, assume the agent has encountered K tasks so far, then the PAC-Bayes lifelong RL problem is formulated as follows:

$$\begin{aligned} &\min_P U(P) \\ &\coloneqq \frac{1}{K}\sum_{i=1}^K \left\{ \mathbb{E}_{\theta \sim P}[-J_{\mathcal{M}_i}(\pi_\theta)] \right\} + \mathcal{R}(\mathbb{D}_{KL}(P\|\underline{P})), \end{aligned} \quad (3)$$

where $\mathcal{R}(\mathbb{D}_{KL}(P\|\underline{P}))$ is derived later in our theory,

where $J_{\mathcal{M}}(\pi_\theta)$ is the total expected reward of policy π in MDP $\mathcal{M}$, taking the expectation with respect to the posterior distribution P for the parameter θ. The negative sign can be interpreted as the loss on a specific task $\mathcal{M}$.

To be concrete, in the finite MDP setting with length H, for policy π_θ with $\theta \sim P(\theta)$, the total expected reward with task $\mathcal{M}$ is the value function,

$$J_{\mathcal{M}}(\pi_\theta) = \mathbb{E}\left[\sum_{h=1}^{H-1}\gamma^{h-1}r_h | \pi_\theta, s_1, \mathcal{M}\right],$$

from a length of H consecutive sample transitions, $s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_H \sim \pi_\theta \times \mathcal{M}$.

4.2 An Algorithm based on PAC-Bayes Lifelong Framework

We now develop an algorithm to exploit the PAC-Bayes framework to efficiently perform lifelong RL.

Consider a time where we have seen K tasks so far, and denote them $\{\mathcal{M}_i\}_{i=1}^K$. They are drawn from the lifelong task distribution $\{\mathcal{D}_i\}_{i=1}^K$. Each distribution $\mathcal{D}_i$ should possess non-zero support and boundedness both from above and below. Critically, once the agent interacts with a task, revisiting previously encountered MDPs is not guaranteed.

Our objective is to learn a shared lifelong learning model - the distribution of θ using the K tasks the algorithm has encountered so far.

To achieve this, we propose the following lifelong RL learning algorithm based on the learning objective in Equation (3), and provide its theoretical justification. The main idea is to learn a policy distribution P as a policy initializer using the objective in Equation (3), referred to as the *default policy* This approach allows the default policy to capture common knowledge among tasks, addressing the challenge of task divergence.

In the lifelong setting, the agent receives a new task, stores it, learns from it, then forgets. We allow the agent to keep a number of N tasks in memory. We update the default policy every N tasks and estimate the training cost based on the most recent N tasks. At the K-th task, the agent has performed $\lfloor \frac{K}{N} \rfloor$ updates to the default policy so far. At each time step $l = 1, \cdots, \lfloor \frac{K}{N} \rfloor$, the agent has θ_{l-1} as its policy parameters from P_{l-1}. It encounters the ith task $\mathcal{M}_{l,i}$'s MDP, and receives $J_{\mathcal{M}_{l,i}}(\pi_{\theta_{l-1}})$ as the total discounted expected reward. The collects trajectory data of H steps for task $\mathcal{M}_{l,i}$, using $\pi_{\theta_{l-1}}$, resulting in a dataset $\tau_l = (\tau_{l,1}, \ldots, \tau_{l,N})$ with a size of $|\tau_l| = HN$.

The agent uses τ_l to update the default policy P_l by minimizing the generalization error bound in (3), evaluated at the current time's posterior P_{l-1} and prior $\underline{P}_{l-1}$. Before learning starts, the agent initializes a prior policy distribution $\underline{P}_0$ and the same posterior policy distribution P_0 randomly or based on domain knowledge (Lines 2-3 of Algorithm 1). Choosing a good prior policy distribution $\underline{P}_0$ is challenging as it affects the tightness of the bound.

We adopt a Bayesian sequential experiment design (Chaloner and Verdinelli, 1995) and use an evolving prior instead of a fixed one. We gradually move the prior towards the default policy by $\underline{P}_l = (1 - \lambda)P_l + \lambda \times \underline{P}_l$ (Line 11), where $\lambda \in (0, 1)$ controls the moving speed, and λ decays by $\lambda = \lambda \times \alpha$ ($\alpha < 1$) over the tasks. This allows us to find a good prior during learning and leverage it to improve the default policy.

As the agent encounters an increasing number of tasks, each task remains distinct. However, with more exposure to tasks, the agent gradually improves its understanding of the distribution P for π_θ. When a new task emerges, the agent can sample $\theta_l \sim P_{l-1}$, employing θ_l to generate a trajectory for subsequent updates. This allows the agent to learn faster, obtaining higher rewards in a shorter time frame. Next, we derive our main PAC-Bayes theorem for Algorithm 1.

Our learning process involves a loop of times to evolve the policy distribution. So we index the policy dis-

---

Algorithm 1 Empirical PAC-Bayes that Improves Continuously (EPIC)

1: **Input:** Update frequency N; the number of steps allowed in each task H; prior evolving speed λ
2: Initialize prior policy distribution $\underline{P}_0$
3: Initialize default policy distribution $P_0 \leftarrow \underline{P}_0$
4: **for** $i = 1, 2, 3, \cdots, K, \cdots, \infty$ **do**
5:     Receive a new task $\mathcal{M}_i \sim \mathcal{D}_i$ and store it into Memory buffer
6:     **if** $i \bmod N = 0$ **then**
7:         Let $l = i/N$
8:         Sample $\theta_{l-1} \sim P_{l-1}$
9:         Roll out trajectories $\tau_{l,k}$ using $\pi_{\theta_{l-1}}$ and $\{\mathcal{M}_k\}_{k=i-N+1}^{i}$ and store τ_l into Memory.
10:         # Update default policy P_l by using τ_l
11:         $\underline{P}_{l-1} \leftarrow (1 - \lambda)\underline{P}_{l-1} + \lambda P_{l-1}$
12:         $P_l \leftarrow \arg\min_P U(P)$
13:         Decay λ by $\lambda = \lambda \times \alpha$
14:         Empty Memory by clearing dataset τ_l
15:     **end if**
16: **end for**

---

tribution at each time by a subscript. First, denote $\theta := \{\theta_l\}_{l=0}^{\lfloor \frac{K}{N} \rfloor - 1}$ and let $P := P(\{\theta_l\}_{l=0}^{\lfloor \frac{K}{N} \rfloor - 1})$ denote the joint posterior distribution of $\theta_0, \ldots, \theta_{\lfloor \frac{K}{N} \rfloor - 1}$ across all times. And naturally, let $P_l := P(\theta_l | \theta_{l-1})$ be the conditional probability of policy for time l given the policy from time $l - 1$, and specially, let $P_0 := P(\theta_0)$.

Assumption 4.1. (Conditional Independence) Given the previous policy θ_{l-1}, the current policy is conditionally independent of all earlier policies:

$$\theta_l \perp\!\!\!\perp \theta_{l-2}, \ldots, \theta_0 \mid \theta_{l-1}. \tag{4}$$

(Policy Support Bound) Define the smallest nonzero probability across all policies as:

$$s_{\min} = \inf \left\{ \min_{A: P_l(A) > 0} P_l(A) : P_l \in \Pi \right\}. \tag{5}$$

(Radius of Variation) The maximum difference between consecutive policy distributions is bounded by:

$$r = \inf \left\{ c : \sup_{A \in \mathscr{A}} |P_l(A) - P_{l-1}(A)| \leq c, \quad \forall l \right\}, \tag{6}$$

where $\sup_{A \in \mathscr{A}} |P_l(A) - P_{l-1}(A)|$ is the total variation distance of P_l and P_{l-1} such that it measures the worst-case difference over all measurable events.

To simplify the analysis and improve readability, we denote $T = \lfloor \frac{K}{N} \rfloor$ and assume $K \bmod N = 0$ without loss of generality. Based on Assumption 4.1, we arrive at the following relationship:

$$P(\{\theta_l\}_{l=0}^{T-1}) = \quad P_{T-1} \times \cdots \times P_l \times \cdots \times P_0. \quad (7)$$

We also derive a corollary on the decomposition of the training error, which facilitates the subsequent proof. The proof is deferred to Appendix §B.1.

Proposition 4.2 (Decomposition of Training Error)**.** Suppose Assumption 4.1 holds. Then we have:

$$\frac{1}{K}\sum_{i=1}^{K}\mathbb{E}_{\{\theta_l\}_{l=0}^{T-1}\sim P}[-J_{\mathcal{M}_i}(\pi_\theta)]$$
$$= \sum_{l=1}^{T}\sum_{i=1}^{N}\frac{1}{TN}\mathbb{E}_{\theta_{l-1}\sim P_{l-1}}\left[-J_{\mathcal{M}_{l,i}}(\pi_{\theta_{l-1}})\right].$$

Theorem 4.3 (PAC-Bayes Bound for EPIC)**.** Under the settings of Algorithm 1 and Assumption 4.1, further assume i-th finite horizon MDP of task i has a reward that belongs to $[0, 1]$. When running Algorithm 1, we update the default policy distribution P_l for every N-th task by using pairs of $\{(P_l, \underline{P}_l)\}_{l=0}^{T-1}$. Let $\mathscr{T} := \prod_{l=1}^{T}(\theta_{l-1}\times \mathcal{M}_l)$, and let the expected loss over joint policy and joint trajectory space be:

$$\frac{1}{K}\sum_{i=1}^{K}\mathbb{E}_{\{\theta_l\}_{l=0}^{T-1}\sim P}[\mathbb{E}_{\{\tau_l\}_{l=1}^{T}\sim \mathscr{T}}[-J_{\mathcal{M}_i}(\pi_\theta)]].$$

and let the training error be:

$$\frac{1}{K}\sum_{i=1}^{K}\mathbb{E}_{\{\theta_l\}_{l=0}^{T-1}\sim P}[-J_{\mathcal{M}_i}(\pi_\theta)].$$

Then with probability at least $1 - 2\exp(-K^\gamma)$, for any $0 < \gamma < 1$, we have

$$\text{expected loss} \leq \text{training error} + \mathscr{R}(\mathbb{D}_{KL}(P\|\underline{P})),$$

with

$$\mathscr{R}(\mathbb{D}_{KL}(P\|\underline{P}))$$
$$:= \frac{2N^{1/2}H\frac{\lambda r}{1-\alpha}\sqrt{\frac{1-\alpha^{2(K/N-1)}}{s_{\min}(1-\alpha^2)}}}{K^{1/2}} + \frac{2N^{1/2}H}{K^{(1-\gamma)/2}}.$$
$$(8)$$

The proof is deferred to Appendix §B.

Remarks. (1) The tightness of the bound depends on the number of lifelong tasks encountered so far K, the number of tasks memorized N, the trajectory length H, and the KL divergence between P and $\underline{P}$. By letting $\gamma = 1/4$, the difference of training and generalization error is in the order $\mathcal{O}(K^{-3/8})$. (2) The N appears on the right hand side can be understood as the memory size kept in the agent before it refreshes. The larger N will reduce K/N thus could potentially decrease the first term by making $1 - \alpha^{2K/N-2}$ smaller. However, it also increase the value $N^{1/2}$. According

to the experimental results with different seeds, we observe the $N = 25$ performs well and is robust, we recommend as an initial value.

Practically, a strategy to adaptively adjust N by minimizing the $U(P)$ using a neural network can work.

Overall, this theory enables our learner to optimize the right-hand side of Equation (8) and learn the lifelong policy distribution with a guaranteed minimal true cost.

There are several unsolved questions before we propose a practical lifelong RL algorithm. Theorem 4.3 holds for policy distribution P and prior distribution $\underline{P}$ parameterized by θ. However, in practice, determining suitable distributions for $\underline{P}$ and P becomes a crucial challenge. Additionally, computing the posterior distributions $\{P_l\}_{l=0}^{T-1}$ is non-trivial. Moreover, we need to identify the appropriate optimization method to learn parameters for P. To address these questions, the next two sections will provide solutions and propose a practical lifelong RL algorithm based on the proposed Algorithm 1.

4.3 Posterior Distribution and Prior Distribution

In Equation (8), P_l represents the posterior distribution of θ_l. To optimize the posterior P_l, we need to choose appropriate hyperparameters for its distribution. For instance, in a Gaussian distribution, we optimize its mean μ and variance σ^2.

Let $\tau_l \sim \theta_{l-1}\times \mathcal{M}_1\cdots\times\cdots\times\mathcal{M}_N$ be the data induced from previous θ_{l-1}, and define the likelihood function $p(g(\tau_l)|\theta_{l-1})$. Suppose the prior distribution is a probability density function $\underline{p}(\theta_{l-1}; q)$, parametrized by q, such as a Gaussian prior $\underline{P}_l = \mathcal{N}(\underline{\mu}_l, \underline{\sigma}_l)$, where $q := (\underline{\mu}_l, \underline{\sigma}_l)$.

Based on Bayes' Rule, the posterior distribution is uniquely given by $p(\theta_{l-1}|g(\tau_l); q) = \frac{p(g(\tau_l)|\theta_{l-1})\underline{p}(\theta_{l-1};q)}{c(q)}$, where $c(q)$ is the normalization constant depends on q. We can optimize the hyperparameter q using the following equation:

$$\min_q U_l(P; q) := \sum_{i=1}^{N}\mathbb{E}_{\theta_{l-1}\sim p(\theta_{l-1}|g(\tau_l);q)}\left[-J_{\mathcal{M}_{l,i}}(\pi_{\theta_{l-1}})\right]$$
$$+ \mathscr{R}(\mathbb{D}_{KL}(P\|\underline{P};q)),$$
$$(9)$$

where $\mathscr{R}(\mathbb{D}_{KL}(P\|\underline{P};q))$ is defined in (8). In Equation (9), the posterior distribution is unknown, and obtaining an explicit expression requires knowing the data likelihood. In the RL regime, data samples consist of states, actions, and value functions, and one approach is to use the exponential of the negative squared temporal difference (TD) error as an unnor-

malized likelihood, as suggested in Dann et al. (2021), which is left for future research.

In PAC-Bayes, the prior and posterior distributions can belong to different families. However, it is often practical to consider them belonging to a common distribution family, as it simplifies the computation of KL divergence. Hence, we assume the default and prior policy distributions for θ to be d-dimensional Gaussians with unknown parameters $\theta \sim \mathcal{N}(\mu, \sigma^2)$. These parameters are updated by minimizing the upper bound.

Based on Equation (9), we solve the following problem where $\phi(\theta; \mu, \sigma)$ is the Multivariate Gaussian PDF:

$$\min_{\mu, \sigma} U(P; \mu, \sigma) := \sum_{i=1}^{N} \int -J_{\mathcal{M}_{l,i}}(\pi_{\theta_{l-1}}) \phi(\theta; \mu, \sigma) d\theta$$
$$+ \mathcal{R}\left(\mathbb{D}_{KL}\left(\mathcal{N}(\mu, \sigma^2) \| \mathcal{N}(\underline{\mu}; \underline{\sigma}^2)\right)\right). \quad (10)$$

Evaluating the integral in Equation (10) analytically is intractable in practice. Therefore, we resort to Monte Carlo Methods, where we sample $\theta_{l-1,j\, j \in [M]}$ to approximate the gradient descent updates by:

$$\nabla_{\mu, \sigma} \hat{U}(P, \{\mathcal{M}_i\}_{i \in [N]}, \{\theta_{l-1,j}\}_{j \in [M]}; \mu, \sigma)$$
$$:= \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{N} -\nabla_{\mu, \sigma} \big\{ J_{\mathcal{M}_{l,i}}(\pi_{\theta_{l,j}}) \quad (11)$$
$$+ \mathcal{R}\left(\mathbb{D}_{KL}\left(\mathcal{N}(\mu, \sigma^2) \| \mathcal{N}(\underline{\mu}; \underline{\sigma}^2)\right)\right) \big\},$$

where $\theta_{l-1,j}$ is a sample drawn from P_{l-1} to perform gradient descent during optimization in each iteration.

Moreover, to ensure the parameters μ and σ can be updated, we use indirect sampling by first sampling a multivariate standard normal distribution ϵ_j. The randomness of the parameter θ_j is then defined as:

$$\theta_j = \mu + \sigma \odot \epsilon_j, \quad \epsilon_j \sim \mathcal{N}(0, I_d). \quad (12)$$

According to Equation (12), the parameter θ_j is multivariate normal distributed with $\theta_j \sim \mathcal{N}(\mu, \sigma^2)$.

4.4 A Practical EPIC Algorithm

We propose a practical EPIC algorithm, called EPICG, as presented in Algorithm 2. In this algorithm, a policy is defined as a Gibbs distribution in a linear combination of features: $\pi_\theta(s, a) = \frac{\exp(\theta^\top \psi_{s,a})}{\sum_b \exp(\theta^\top \psi_{s,b})}$. Here, θ can be replaced by a neural network.

For the parameterization of θ using a neural network, we provide the details in Appendix §D.1. In each iteration, the agent samples a set of policies $\theta_{j\, j \in [M]} \sim P$ from the "posterior" policy distribution for every N tasks (Lines 8-9). It then rolls out a set of trajectories

---

Algorithm 2 Empirical PAC-Bayes that Improves Continuously Under Gaussian Prior) (EPICG)

Input: policy dimension d; learning rate β; update frequency N; failure probability δ; the number of steps allowed in each task H; prior evolving speed λ

2: Initialize prior policy mean and derivation $\underline{\mu}_0, \underline{\sigma}_0 \in \mathbb{R}^d$
 Initialize default policy mean and derivation $\mu_0 \leftarrow \underline{\mu}_0, \sigma_0 \leftarrow \underline{\sigma}_0$

4: **for** $i = 1, 2, 3, \cdots, K, \cdots \infty$ **do**
 Receive a new task $\mathcal{M}_i \sim \mathcal{D}_i$ and store it in memory.

6: **if** $i \bmod N = 0$ **then**
 Let $l = i/N$

8:   Sample $\{\theta_{l-1,j}\}_{j \in [M]} \sim \mathcal{N}(\mu_{l-1}, \sigma_{l-1}^2)$ by sample $\epsilon_j \sim \mathcal{N}(0, I_d)$
 Set initial policy, i.e., initialize parameters for neural network $\theta_{l-1,j} \leftarrow \{\mu_{l-1} + \epsilon_j \odot \sigma_{l-1}\}$

10:   Roll out trajectories $\tau_{k,j}$ using $\{\pi_{\theta_{l-1,j}}\}_{j \in [M]}$ and $\{\mathcal{M}_k\}_{k=i-N+1}^{i}$ and store into $\tau_{l,j}$
 \# Update default and Prior parameters by using $\tau_{l,j}$

12:   $\mu_l \leftarrow \mu_{l-1} - \beta \nabla_\mu \hat{U}(P, \{\mathcal{M}_k\}, \{\theta_{l-1,j}\}_{j \in [M]}; \mu, \sigma)$
 $\sigma_l \leftarrow \sigma_{l-1} - \beta \nabla_\sigma \hat{U}(P, \{\mathcal{M}_k\}, \{\theta_{l-1,j}\}_{j \in [M]}; \mu, \sigma)$

14:   $\underline{\mu}_l \leftarrow (1-\lambda)\underline{\mu}_l + \lambda \mu_l; \underline{\sigma} \leftarrow (1-\lambda)\underline{\sigma}_l + \lambda \sigma_l$
 Empty Memory by clearing dataset τ_l

16: **end if**
 end for

$\tau = \tau_{i \times j \in [N] \times [M]}$ for each task and estimates the cost (Lines 10-11). More specifically, an action a is sampled as $a \sim \pi_{\theta_{l,j}}(s, a)$, and a state s is sampled using a transition kernel determined by task $\mathcal{M}_i$.

The gradient is taken with respect to the objective function $\hat{U}$ (Equation (10)) which with respect to the cost function and with respect to the KL divergence function expressed in Equation (8).

EPICG uses gradient descent in the space of P to find the policy that minimizes the expected loss, i.e., $P^* \in \arg\inf_{P \in \Pi} \frac{1}{K} \sum_{i=1}^{K} \mathbb{E}_{\{\theta_l\}_{l=0}^{T-1} \sim P}[\mathbb{E}_{\{\tau_l\}_{l=1}^{T} \sim \mathscr{T}}[-J_{\mathcal{M}_i}(\pi_\theta)]]$, where we assume the P^* exists.

We denote the optimal expected return as $J^* := \frac{1}{K} \sum_{i=1}^{K} \mathbb{E}_{\{\theta_l\}_{l=0}^{T-1} \sim P^*}[\mathbb{E}_{\{\tau_l\}_{l=1}^{T} \sim \mathscr{T}}[J_{\mathcal{M}_i}(\pi_\theta)]]$. We denote the return as $\tilde{J} := \frac{1}{K} \sum_{i=1}^{K} \mathbb{E}_{\{\theta_l\}_{l=0}^{T-1} \sim P}[J_{\mathcal{M}_i}(\pi_\theta)]$., which is also equal to the negative training error. We provide the sample complexity of EPICG.

Theorem 4.4 (Sample Complexity)**.** Consider the setting of Theorem 4.3. Given a small $\epsilon > 0$, if the

(a) HalfC.-gravity **(b)** HalfC.-bodyp. **(c)** Hopper-gravity **(d)** Hop.-bodyp. **(e)** Walker-gravity **(f)** Walker-bodyp.
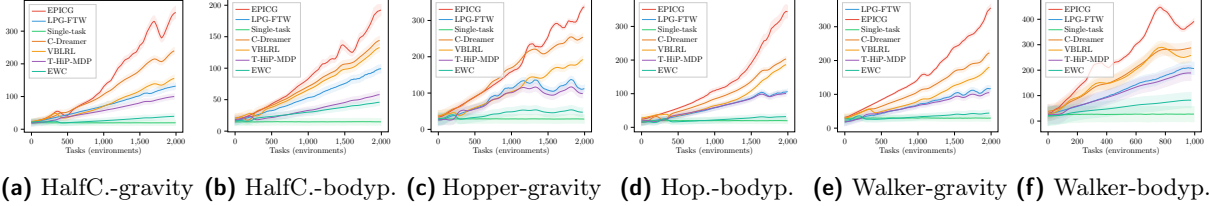
Figure 1: Comparison between EPICG and baselines on lifelong RL benchmarks. X-axis: tasks, Y-axis: reward. CartPole-Goal with $x_{goal} \sim \mathcal{N}(0, 0.1)$ and $x_{goal} \sim \mathcal{N}(0, 0.5)$, LunarLander, CartPole-Mass with $\mu_c = 0.5$ and $\mu_c = 1.0$, and Swimmer.

number of tasks K satisfies

$$K = \max \left(\frac{16NH^2\lambda^2 r^2}{s_{\min}(1-\alpha)^3(1+\alpha)\epsilon^2}, \left(\frac{16NH^2}{\epsilon^2} \right)^{\frac{1}{1-\gamma}} \right)$$
$$+ \widetilde{\mathcal{O}}(N\epsilon^{-4}),$$

then with high probability,

$$J^* - \tilde{J} \leq \mathcal{O}(\epsilon),$$

where $\widetilde{\mathcal{O}}(\cdot)$ suppresses logarithmic dependence.

Proof. The central part of the proof is Theorem 4.3 (detailed proof in Appendix §B.2), and the remaining parts are provided in Appendix §B.5. □

Algorithm 2 learns a general policy distribution. However, if we are interested in a policy for any specific task, we can sample a policy from P and use an appropriate single-task learning method to fine-tune the policy. Hence, every task gets a "customized" policy. We also provide an Algorithm 3 to reflect this in Section 5.3.

5 EXPERIMENTS

5.1 Experimental Setup

We experiment with common tasks in lifelong-RL benchmarks used in prior works (Mendez et al., 2020; Fu et al., 2022), including HalfCheetah-gravity, HalfCheetah-bodyparts, Hopper-gravity, Hopper-bodyparts, Walker-gravity, Walker-bodyparts. To increase the diversity of lifelong environments, we also create several more lifelong environments, Cartpole-GMM, LunarLander-Uniform, Ant-Direction-Uniform, Ant-Forward-Backward-Bernoulli, Swimmer-Uniform, Humanoid-Direction-Uniform. Details about the above environments can be found in Appendix §E.2 and in Table 2. In each lifelong environment, the agents are tested across 2,000 or 1,000 tasks. Each environment has a distinct maximum H. As the sequence of 2,000 or 1,000 tasks unfolds, we update the default policy every N tasks. The effectiveness of our approach is

assessed by how how fast it learns to maximize return as new tasks emerge.

5.2 Effective Lifelong Learning

Figure 1 evaluates EPICG[1] across several control tasks, all of which are lifelong-RL benchmarks used in prior works (Mendez et al., 2020; Fu et al., 2022). This reveals EPICG's noticeable advantage in most scenarios. EPICG consistently outperforms others. We compared EPICG against: 1. Continual Dreamer (Kessler et al., 2023), state-of-the-art lifelong RL method, 2. VBLRL (Fu et al., 2022), Model based Bayesian lifelong RL method; 3. LPG-FTW (Mendez et al., 2020), a lifelong RL method which assumes a factored representations of the policy parameter space; 4. EWC (Kirkpatrick et al., 2017a), which is a single-model lifelong RL algorithm that achieves comparable performance with LPG-FTW as shown in the latter paper; 5. T-HiP-MDP (Killian et al., 2017), which is a model-based lifelong RL baseline; and 6. Single-Task RL, which let the agent learn the task policy from scratch for every new task and does not use the world policy to help learning. Further details on each baseline can be found in Appendix §2.

5.3 Further Improvement

EPICG effectively learns a shared distribution P of policy parameters for different tasks. Upon receiving new tasks, we learn the policy distribution by using the sampled policy parameter θ. At this point, this approach has already shown effectiveness in our lifelong learning setting. Additionally, we can further improve this θ by optimizing it using data from the new task, customizing it for that particular task. Below we introduce Algorithm 3 (EPICG-SAC), which integrates the EPICG framework with the single task algorithm Soft Actor Critic.

We then compare EPICG-SAC and EPICG for different environments. Figure 2 shows EPICG-SAC achieves faster learning than EPICG.

---

[1]Our method is publicly available at https://zzh237.github.io/EPIC/.

(a) CartP.-Unif. **(b)** LunarL.-GMM **(c)** AntD.-Unif. **(d)** AntF.B.-Bern. **(e)** Swi.-Unif. **(f)** Hum.D.-Unif.
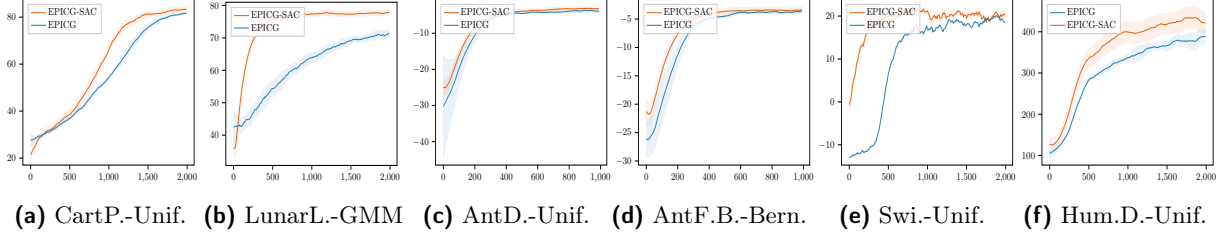
Figure 2: Average reward obtained by EPICG-SAC and EPICG in different environments with different lifelong learning settings. X-axis: tasks, Y-axis: reward.

Algorithm 3 EPICG-SAC

1: **Input:** Same setting as Algorithm 2
2: **for** $i = 1, 2, 3, \cdots, K, \cdots \infty$ **do**
3: Receive a new task $\mathcal{M}_i$.
4: **if** $i \bmod N = 0$ **then**
5:    Do EPICG Policy Distribution Learning.
6: **end if**
7: **SAC single-task train-eval loop.**
8: **end for**

5.4 Ablation on KL divergence regularization

We first verify the empirical performance when we add the regularizer in (11) compared to having no regularizer by a comparison study. The results are shown in Figure 3, where we observe that adding the regularizer facilitates fast adaptation, leads to learning a higher reward, and also reduces the variance, which leads to a more stable learning compared to having no regularize.
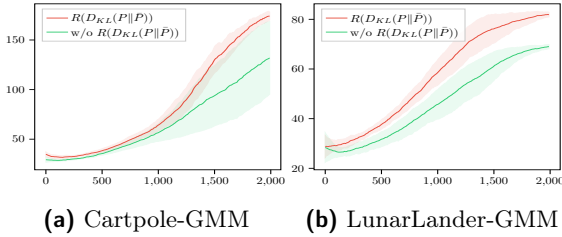

(a) Cartpole-GMM   **(b)** LunarLander-GMM

Figure 3: Comparison of adding $\mathscr{R}(\mathbb{D}_{KL}(P\|\bar{P}))$ vs. not. X-axis: tasks, Y-axis: reward.

5.5 Experiments on Memory Size N
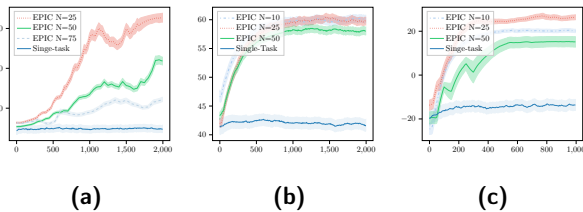

(a)   **(b)**   **(c)**

Figure 4: Comparison of different update frequency N on (a) CartPole-Uniform; (b) LunarLander-Uniform; (c) Swimmer-Uniform. X-axis: tasks, Y-axis: reward

As we discussed in Theorem 4.3, there is a performance trade-off on the number of tasks N retained in memory. Experimental results have verified this theoretical finding. We can see that in Figure 4 the practical effect of N on the performance of learning is double-sided.

6 CONCLUSION AND FUTURE WORKS

In this work, we address the challenging problem of lifelong RL and propose a novel algorithm, EPIC(G), for distribution learning and policy sampling. Our approach leverages the concept of a *world policy*, a shared policy distribution across tasks. This world policy is updated continuously, enabling our algorithm to handle both non-stationarity and catastrophic forgetting, achieving best-in-class performance across a suite of complex lifelong RL benchmarks.

Future directions include exploring more accurate ways to obtain the posterior distribution of the policy parameters, as discussed in Section 4.3. Additionally, since the optimization objective in Equation (11) is nonconvex, better performance guarantees could be achieved by investigating multiple optimizations across tasks.

References

Abel, D., Barreto, A., Van Roy, B., Precup, D., van Hasselt, H. P., and Singh, S. (2024). A definition of continual reinforcement learning. Advances in Neural Information Processing Systems, 36.

Abel, D., Jinnai, Y., Guo, S. Y., Konidaris, G., and Littman, M. (2018). Policy and value transfer in lifelong reinforcement learning. In International Conference on Machine Learning, pages 20–29. PMLR.

Amit, R. and Meir, R. (2018). Meta-learning by adjusting priors based on extended pac-bayes theory. In International Conference on Machine Learning, pages 205–214. PMLR.

Anand, N. and Precup, D. (2023). Prediction and control in continual reinforcement learning. In

Advances in Neural Information Processing Systems (NeurIPS).

Boutilier, C., Cohen, A., Daniely, A., Hassidim, A., Mansour, Y., Meshi, O., Mladenov, M., and Schuurmans, D. (2018). Planning and learning with stochastic action sets. arXiv preprint arXiv:1805.02363.

Chaloner, K. and Verdinelli, I. (1995). Bayesian experimental design: A review. Statistical science, pages 273–304.

Chandak, Y., Theocharous, G., Nota, C., and Thomas, P. (2020). Lifelong learning with a changing action set. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 3373–3380.

Dann, C., Mohri, M., Zhang, T., and Zimmert, J. (2021). A provably efficient model-free posterior sampling method for episodic reinforcement learning. Advances in Neural Information Processing Systems, 34:12040–12051.

Ding, Q., Kang, Y., Liu, Y.-W., Lee, T. C. M., Hsieh, C.-J., and Sharpnack, J. (2022). Syndicated bandits: A framework for auto tuning hyper-parameters in contextual bandit algorithms. Advances in Neural Information Processing Systems, 35:1170–1181.

Donsker, M. D. and Varadhan, S. S. (1983). Asymptotic evaluation of certain markov process expectations for large time. iv. Communications on pure and applied mathematics, 36(2):183–212.

Dziugaite, G. K., Hsu, K., Gharbieh, W., Arpino, G., and Roy, D. M. (2020). On the role of data in PAC-Bayes bounds. arXiv:2006.10929 [cs, stat].

Fard, M. and Pineau, J. (2010). PAC-Bayesian Model Selection for Reinforcement Learning. Advances in Neural Information Processing Systems, 23:1624–1632.

Fard, M. M., Pineau, J., and Szepesvari, C. (2012). PAC-Bayesian Policy Evaluation for Reinforcement Learning. arXiv:1202.3717 [cs, stat].

Flynn, H., Reeb, D., Kandemir, M., and Peters, J. (2022). Pac-bayesian lifelong learning for multi-armed bandits. Data Mining and Knowledge Discovery, 36(2):841–876.

Freedman, D. A. (1975). On tail probabilities for martingales. the Annals of Probability, pages 100–118.

Fu, H., Yu, S., Littman, M., and Konidaris, G. (2022). Model-based lifelong reinforcement learning with bayesian exploration. Advances in Neural Information Processing Systems, 35:32369–32382.

Garg, S., Tsipras, D., Liang, P. S., and Valiant, G. (2022). What can transformers learn in-context? a case study of simple function classes. Advances in Neural Information Processing Systems, 35:30583–30598.

Germain, P., Lacasse, A., Laviolette, F., and Marchand, M. (2009). PAC-Bayesian learning of linear classifiers. In Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, pages 353–360, New York, NY, USA. Association for Computing Machinery.

Ha, D. and Schmidhuber, J. (2018). World models. arXiv preprint arXiv:1803.10122.

Kang, Y., Hsieh, C.-J., and Lee, T. (2024). Online continuous hyperparameter optimization for generalized linear contextual bandits. Transactions on Machine Learning Research.

Kang, Y., Hsieh, C.-J., and Lee, T. C. M. (2022). Efficient frameworks for generalized low-rank matrix bandit problems. Advances in Neural Information Processing Systems, 35:19971–19983.

Kang, Y., Hsieh, C.-J., and Lee, T. C. M. (2023). Robust lipschitz bandits to adversarial corruptions. Advances in Neural Information Processing Systems, 36:10897–10908.

Kessler, S., Ostaszewski, M., Bortkiewicz, M., Żarski, M., Wolczyk, M., Parker-Holder, J., Roberts, S. J., Mi, P., et al. (2023). The effectiveness of world models for continual reinforcement learning. In Conference on Lifelong Learning Agents, pages 184–204. PMLR.

Khetarpal, K., Riemer, M., Rish, I., and Precup, D. (2022). Towards continual reinforcement learning: A review and perspectives. Journal of Artificial Intelligence Research, 75:1401–1476.

Killian, T. W., Daulton, S., Konidaris, G., and Doshi-Velez, F. (2017). Robust and efficient transfer learning with hidden parameter markov decision processes. Advances in neural information processing systems, 30.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017a). Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences, 114(13):3521–3526.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. (2017b). Overcoming catastrophic forgetting in neural net-

works. Proceedings of the National Academy of Sciences, 114(13):3521–3526.

Krähenbühl, P., Doersch, C., Donahue, J., and Darrell, T. (2016). Data-dependent Initializations of Convolutional Neural Networks. arXiv:1511.06856 [cs].

Langford, J. and Shawe-Taylor, J. (2002). PAC-Bayes & margins. In Proceedings of the 15th International Conference on Neural Information Processing Systems, NIPS'02, pages 439–446, Cambridge, MA, USA. MIT Press.

Lecarpentier, E., Abel, D., Asadi, K., Jinnai, Y., Rachelson, E., and Littman, M. L. (2021). Lipschitz lifelong reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 8270–8278.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Liu, X.-H., Xue, Z., Pang, J., Jiang, S., Xu, F., and Yu, Y. (2021). Regret minimization experience replay in off-policy reinforcement learning. Advances in neural information processing systems, 34:17604–17615.

Majumdar, A., Farid, A., and Sonar, A. (2021). Pac-bayes control: learning policies that provably generalize to novel environments. The International Journal of Robotics Research, 40(2-3):574–593.

Mbacke, S. D., Clerc, F., and Germain, P. (2023). Statistical guarantees for variational autoencoders using pac-bayesian theory. Advances in Neural Information Processing Systems, 36:56903–56915.

McAllester, D. A. (1999). Some PAC-Bayesian Theorems. Machine Learning, 37(3):355–363.

Mendez, J., Wang, B., and Eaton, E. (2020). Lifelong policy gradient learning of factored policies for faster training without forgetting. Advances in Neural Information Processing Systems, 33:14398–14409.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540):529–533.

Naik, D. K. and Mammone, R. J. (1992). Meta-neural networks that learn by learning. In [Proceedings 1992] IJCNN International Joint Conference on Neural Networks, volume 1, pages 437–442 vol.1.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and

Srebro, N. (2017). Exploring Generalization in Deep Learning. arXiv:1706.08947 [cs].

Neyshabur, B., Bhojanapalli, S., and Srebro, N. (2018). A PAC-Bayesian Approach to Spectrally-Normalized Margin Bounds for Neural Networks. arXiv:1707.09564 [cs].

Pang, J.-C., Xu, T., Jiang, S., Liu, Y.-R., and Yu, Y. (2021). Reinforcement learning with sparse-executing actions via sparsity regularization. arXiv preprint arXiv:2105.08666.

Salimans, T. and Kingma, D. P. (2016). Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. arXiv:1602.07868 [cs].

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. arXiv:1312.6120 [cond-mat, q-bio, stat].

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In International conference on machine learning, pages 1889–1897. PMLR.

Seeger, M. (2002). PAC-Bayesian Generalisation Error Bounds for Gaussian Process Classification. Journal of Machine Learning Research, 3(Oct):233–269.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. Nature, 529(7587):484–489.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv preprint arXiv:1712.01815.

Steinparz, C. A., Schmied, T., Paischer, F., Dinu, M.-C., Patil, V. P., Bitto-Nemling, A., Eghbal-zadeh, H., and Hochreiter, S. (2022). Reactive exploration to cope with non-stationarity in lifelong reinforcement learning. In Conference on Lifelong Learning Agents, pages 441–469. PMLR.

Thrun, S. and Pratt, L. (1998). Learning to Learn: Introduction and Overview. In Thrun, S. and Pratt, L., editors, Learning to Learn, pages 3–17. Springer US, Boston, MA.

Veer, S. and Majumdar, A. (2020). Probably Approximately Correct Vision-Based Planning using Motion Primitives. arXiv:2002.12852 [cs, eess, math].

Wainwright, M. J. (2019). High-dimensional statistics: A non-asymptotic viewpoint, volume 48. Cambridge university press.

Yuan, R., Gower, R. M., and Lazaric, A. (2022). A general sample complexity analysis of vanilla policy gradient. In International Conference on Artificial Intelligence and Statistics, pages 3332–3380. PMLR.

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., and Farhadi, A. (2017). Target-driven visual navigation in indoor scenes using deep reinforcement learning. In 2017 IEEE international conference on robotics and automation (ICRA), pages 3357–3364. IEEE.

Checklist

1. For all models and algorithms presented, check if you include:

   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]

   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]

   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes/No/Not Applicable]

2. For any theoretical claim, check if you include:

   (a) Statements of the full set of assumptions of all theoretical results. [Yes]

   (b) Complete proofs of all theoretical results. [Yes]

   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:

   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]

   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]

   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes]

   (b) The license information of the assets, if applicable. [Yes]

   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]

   (d) Information about consent from data providers/curators. [Not Applicable]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [Not Applicable]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Supplementary Material

A Broader Impact and Ethical Statement

This paper contributes to the ongoing development of Machine Learning, adhering to standard ethical guidelines in research. Our work, aligning with common advancements in the field, does not present any unique ethical dilemmas or societal consequences that require special emphasis. We recognize the importance of responsible use and development of Machine Learning technologies and their potential impact on society. However, our research does not delve into areas that might raise significant ethical or societal concerns. We commit to ethical research practices and consider the broader implications of our work to be aligned with the typical advancements in Machine Learning.

B Detailed Proofs

B.1 Proof of Proposition 4.2

Proposition (Decomposition of Training Error). Assume Assumption 4.1 holds, then it holds true that

$$\frac{1}{K}\sum_{i=1}^{K}\mathbb{E}_{\{\theta_l\}_{l=0}^{T-1}\sim P}[-J_{\mathcal{M}_i}(\pi_\theta)] = \frac{1}{T}\sum_{l=1}^{T}\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\theta_{l-1}\sim P_{l-1}}\left[-J_{\mathcal{M}_{l,i}}(\pi_{\theta_{l-1}})\right].$$

Proof. By Assumption 4.1,

$$
\begin{aligned}
&P(\theta_{T-1},\ldots,\theta_0)\\
=&P(\theta_{T-1}|\theta_{T-2},\ldots,\theta_0)\times\cdots\times P(\theta_1|\theta_0)P(\theta_0)\\
=&P(\theta_{T-1}|\theta_{T-2})\times\cdots\times P(\theta_1|\theta_0)P(\theta_0)\\
:=&P_{T-1}\times\cdots\times P_l\times\cdots\times P_0
\end{aligned}
\tag{13}
$$

By law of total expectation, we have

$$
\begin{aligned}
&\mathbb{E}_{\{\theta_l\}_{l=0}^{T-1}\sim P}[-J_{\mathcal{M}_i}(\pi_\theta)] = \mathbb{E}_{\{\theta_l\sim P_l\}_{l=0}^{T-1}}[-J_{\mathcal{M}_i}(\pi_\theta)]\\
&=\frac{1}{N}\sum_{i=1}^{N}E_{P_{T-2},\ldots,P_0}E_{\theta_{T-1}\sim P_{T-1}|P_{T-2},\ldots,P_0}[-J_{\mathcal{M}_{T,i}}(\pi_{\theta_{T-1}})]+\\
&\cdots+\frac{1}{N}\sum_{i=1}^{N}E_{\theta_0\sim P_0}[-J_{\mathcal{M}_{T,i}}(\pi_{\theta_0})]\\
&=\frac{1}{N}\sum_{i=1}^{N}E_{P_{T-2}}E_{\theta_{T-1}\sim P_{T-1}|P_{T-2}}[-J_{\mathcal{M}_{T,i}}(\pi_{\theta_{T-1}})]+\\
&\cdots+\frac{1}{N}\sum_{i=1}^{N}E_{\theta_0\sim P_0}[-J_{\mathcal{M}_{T,i}}(\pi_{\theta_0})]\\
&=\frac{1}{N}\sum_{i=1}^{N}E_{\theta_{T-1}\sim P_{T-1}}[-J_{\mathcal{M}_{T,i}}(\pi_{\theta_{T-1}})]+\\
&\cdots+\frac{1}{N}\sum_{i=1}^{N}E_{\theta_0\sim P_0}[-J_{\mathcal{M}_{T,i}}(\pi_{\theta_0})]\\
&=\frac{1}{T}\sum_{l=1}^{T}\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\theta_{l-1}\sim P_{l-1}}\left[-J_{\mathcal{M}_{l,i}}(\pi_{\theta_{l-1}})\right].
\end{aligned}
$$

$\square$

B.2 Azuma-Hoeffiding or Freedmans inequality for martingale difference sequences for RL

We let the Algorithm (1) experience K tasks, for every N task Algorithm (1) performs lifelong learning, i.e., learns the posterior distribution hyperparameters for policy.

Remember, in Algorithm ([1](#)), where the the lifelong setting happens with K tasks streaming in, we update the default policy every N tasks and estimate the training cost based on the most recent N tasks. The entire learning process consists of a total of T episodes of updates.

Let the distribution of policy has a parameter mean μ and variance-covariance σ^2 for illustration purposes. For each $l \in [T]$ episode, Algorithm ([1](#)) proceeds as:

1. Sample $\theta_{l-1} \sim P_{l-1}(\theta; \mu_{l-1}, \sigma_{l-1}^2)$ learnt from the previous episode.
2. Using θ_{l-1} and N tasks $\{\mathcal{M}_{i,l}\}_{i \in [N]}$ from current episode to collect data τ_l;
3. Using an optimization algorithm and data from Step 1 to learn the posterior distribution $P_l(\theta; \mu_l, \sigma_l^2)$'s hyper parameters;

For l's episode, for policy π_{θ_l}, which is learned using data roll-out by $\theta_{l-1} \sim P_{l-1}(\theta; \mu_{l-1}, \sigma_{l-1}^2)$, its value function with task $\mathcal{M}_{l,i}$ is defined as the total discounted expected rewards,

$$V_{\mathcal{M}_{l,i}}^{\pi_{\theta_l}}(s_{l,1}) := J_{\mathcal{M}_{l,i}}(\pi_{\theta_l}) = \mathbb{E}\left[\sum_{h=1}^{H-1} \gamma^{h-1} r_{l,h} | \pi_{\theta_l}, s_{l,1}, \mathcal{M}_{l,i}\right], \tag{14}$$

from a length of H consecutive sample transitions, $s_{l,1}, a_{l,1}, r_{l,1}, s_{l,2}, a_{l,2}, r_{l,2}, \ldots, s_{l,H} \sim \pi_{\theta_l} \times \mathcal{M}_{l,i}$. Note, θ_l is a function of $\tau_l, \mu_{l-1}, \sigma_{l-1}^2$, which is random, and the randomness is dependent on $\tau_l, \mu_{l-1}, \sigma_{l-1}^2$.

The posterior distribution $P(\theta; \mu, \sigma^2)$ defines a randomized θ. Algorithm ([1](#)) draws a θ according to $\theta \sim P(\theta; \mu, \sigma^2)$ at each round of the whole process and applies it to learn the hyperparameter of $P(\theta; \mu, \sigma^2)$ on the next round. For notation-wise, if we do not use subscript l, it means the statement holding for general.

For any θ, let S_T be the difference between the expected and empirical objective value function after the T-th round,

$$S_T := \sum_{l=1}^{T} D_l, \quad l \in [T], \tag{15}$$

where $D_l := \sum_{i=1}^{N} \mathbb{E}_{\tau_l \sim \theta_{l-1} \times \mathcal{M}_l}\left[V_{\mathcal{M}_{l,i}}^{\pi_{\theta_l}}(s_{l,1}) | \mathcal{F}_{l-1}\right] - \sum_{i=1}^{N} V_{\mathcal{M}_{l,i}}^{\pi_{\theta_l}}(s_{l,1})$. And the filtration $\mathcal{F}_{l-1} = \sigma(\{\theta_k\}_{k \leq l-2}, \{\mathcal{M}_{i,k-1}\}_{i \in [N], k \leq l-1})$ is the σ-algebra generated by the random variables $\{\theta_k\}_{k \leq l-2}$ and $\{\mathcal{M}_{i,k-1}\}_{i \in [N], k \leq l-1}$.

We first show that using Algorithm ([1](#)), after T-th lifelong learning updates, with probability at least $1 - \delta$, for a small $\delta \in (0,1)$, $S_T = \mathcal{O}(\sqrt{T})$.

Theorem B.1. Let $\{D_l\}_{l \leq T}$, and S_T be defined in Equation ([15](#)). For fixed N and H, then with probability at least $1 - \delta$,

$$|S_T| \lesssim \sqrt{\frac{1}{2}\left(\ln \frac{2}{\delta}\right) T N^2 H^2}. \tag{16}$$

Furthermore, if $|D_l| \leq b$ for all $l \leq T$, and let

$$\tilde{S}_T = \sum_{l=1}^{T} \mathbb{E}\left[D_l^2 | \mathcal{F}_{l-1}\right], \tag{17}$$

then with probability at least $1 - \delta$, for $\lambda \in [0, \frac{1}{b}]$,

$$|S_T| \lesssim \frac{1}{\lambda} \ln \frac{2}{\delta} + \lambda \tilde{S}_T \leq \frac{1}{\lambda} \ln \frac{2}{\delta} + \lambda T N^2 H^2. \tag{18}$$

Proof. Firstly, the θ_l comes from the posterior distribution $P(\theta; \mu, \sigma^2)$, which depends on $\{\theta_k\}_{k \leq l-1}$ and $\{\mathcal{M}_{i,l}\}_{i \in [N]}$. Furthermore, for a fixed θ_l, we see that D_l is $\mathcal{F}_{l-1}$ measurable. So given that $D_l = \sum_{i=1}^{N} \mathbb{E}\left[V_{\mathcal{M}_{i,l}}^{\pi_{\theta_l}}(s_{l,1}) | \mathcal{F}_{l-1}\right] - \sum_{i=1}^{N} V_{\mathcal{M}_{i,l}}^{\pi_{\theta_l}}(s_{l,1})$, we have $\mathbb{E}[D_l | \mathcal{F}_{l-1}] = 0$. And $\{D_l\}_{l \in [T]}$ is a martingale difference sequence of functions of θ. Furthermore, $\sum_{i=1}^{N} V_{\mathcal{M}_{i,l}}^{\pi_{\theta_l}}(s_{T,1}) \leq NH$ because of Equation ([14](#)) and the reward $r_{l,h}$ is

in $[0, 1]$ in Theorem 4.3. And $\{D_l\}_{l \in [T]}$ is a bounded martingale difference sequence. In other words, $D_l \in [a_l, b_l]$, with $a_l = -NH$, $b_l = NH$, and $b = NH$, this is true because in Equation (14), the reward $r_{l,h}$ is in $[0, 1]$ in Theorem 4.3. Then based on the conclusion of Theorem C.3, the Azuma-Hoeffding Inequality for bounded martingale difference sequence, we get the result in Equation (16).

For Equation (18), we use Theorem C.4, the Freedmans Inequality for martingale difference sequence. For any $\lambda \in [0, \frac{1}{b}]$, where $|D_l| \leq b$, we have

$$\mathbb{P}\{S_T \geq t | \mathcal{F}_{T-1}\} = \mathbb{P}\{e^{\lambda S_T} \geq e^{\lambda t} | \mathcal{F}_{T-1}\} \leq e^{-\lambda t} \mathbb{E}\left[e^{\lambda S_T} | \mathcal{F}_{T-1}\right]$$
$$\leq e^{-\lambda t} e^{\lambda^2 \tilde{S}_T} = e^{-\lambda t + \lambda^2 \tilde{S}_T}. \tag{19}$$

Thus,

$$\mathbb{P}\{S_T \geq t\} \leq \mathbb{E}_{\mathcal{F}_{T-1}}\left[e^{-\lambda t + \lambda^2 \tilde{S}_T}\right]. \tag{20}$$

Repeating this argument for $-S_T$, we get the same bound, so overall,

$$\mathbb{P}\{|S_T| \geq t\} \leq 2\mathbb{E}_{\mathcal{F}_{T-1}}\left[e^{-\lambda t + \lambda^2 \tilde{S}_T}\right]. \tag{21}$$

Let $\mathbb{E}_{\mathcal{F}_{T-1}}\left[2e^{-\lambda t + \lambda^2 \tilde{S}_T}\right] = \delta$, we get $t = \frac{1}{\lambda} \ln \frac{2}{\delta} + \lambda \tilde{S}_T$. Therefore, with probability at least $1 - \delta$, for $\lambda \in [0, \frac{1}{b}]$,

$$|S_T| \lesssim \frac{1}{\lambda} \ln \frac{2}{\delta} + \lambda \tilde{S}_T \overset{(a)}{\leq} \frac{1}{\lambda} \ln \frac{2}{\delta} + \lambda T N^2 H^2, \tag{22}$$

where (a) holds since $D_l \in [a_l, b_l]$ almost surely, the conditioned variable $D_l | \mathcal{F}_{l-1}$ also belongs to this interval almost surely, then we have $\tilde{S}_T \leq \sum_{l=1}^{T} \frac{(a_l - b_l)^2}{4} \leq T N^2 H^2$ by Lemma C.1. $\qquad \square$

Remark of Theorem B.1. Note, if we minimize the right hand side of Equation (18) with respect to λ, we get the optimal $\lambda = \sqrt{\frac{\ln \frac{2}{\delta}}{T N^2 H^2}}$, and $|S_T| \lesssim 2\sqrt{\ln \frac{2}{\delta} T N^2 H^2}$. Hence, Equation (18) (derived from Freedmans's Inequality) matches Equation (16) (derived from Azuma-Hoeffding Inequality) up to minor constants and logarithmic factors in the general case, and can be much tighter when the variance $\tilde{S}_T = \sum_{l=1}^{T} \mathbb{E}\left[D_l^2 | \mathcal{F}_{l-1}\right]$ is small.

B.3 PAC-Bayes Bound

The quantity S_T is of our interest as it is the difference between the expected and empirical objective value after the T-th round. In the previous Theorem B.1, we show that S_T is bounded for the sampled θ. Our Algorithm 1 keep updating P and sampling θ, rendering $\{P_l\}$ and $\{\theta_l\}$, $l = 0, \ldots, T-1$. To abuse the notation, without further reminder, in the following proof, we use $P(\{\theta_l\})$ and $\underline{P}(\{\theta_l\})$ to denote the joint posterior and prior, and use θ to denote the set for all θ_l, similarly for the hyperparameter in the distribution. However, since Algorithm (1) draws θ from posterior distribution $P(\theta; \mu, \sigma^2)$, we are interested in the expected value of S_T, which is $\mathbb{E}_P[S_T]$. At the same time, we will relate all possible $P(\theta; \mu, \sigma^2)$ to its corresponding "reference" distribution $\underline{P}$, the prior distribution of θ, which is selected before we do step 3. Let $q = (\mu, \sigma)$. In the next Lemma, we will control $\mathbb{E}_P[S_T]$ for any q.

Corollary B.2 (Uniform control of all distributions). Let $\theta \sim P(\theta; q)$ come from a parametrized distribution with the same family as $\underline{P}$, where $\underline{P}$ is a given prior distribution. Let $\{D_l\}_{l \leq T}$ and S_T follow the same definition in Theorem B.1. For any $\lambda > 0$, let $g(\theta) := \lambda S_T(\theta) - \lambda^2 \tilde{S}_T$, where $\tilde{S}_T$ is defined either

$$\tilde{S}_T = \sum_{l=1}^{T} \frac{(a_l - b_l)^2}{8}, \quad \text{(Align with Equation (16))} \tag{23}$$

or

$$\tilde{S}_T = \sum_{l=1}^{T} \mathbb{E}\left[D_l^2 | \mathcal{F}_{l-1}\right], \quad \text{(Align with Equation (18))} \tag{24}$$

then with probability at least $1 - \delta$, and for all $P(\theta; q)$,

$$|\mathbb{E}_P\left[S_T(\theta)\right]| \leq \frac{\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}}{\lambda} + \lambda \mathbb{E}_P\left[\tilde{S}_T\right], \tag{25}$$

with $\lambda > 0$ for (23), and $\lambda \in [0, \frac{1}{b}]$ required for (24), where $|D_l| \leq b$ and $|a_l| \leq b, |b_l| \leq b$.

Proof. Based on Theorem C.5, the Donsker−Varadhans Representation formula, we have

$$
\begin{aligned}
\mathbb{E}_P\left[|\lambda S_T| - \lambda^2 \tilde{S}_T\right] &\leq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right]\right) \\
&\overset{(a)}{\lesssim} \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\frac{1}{\delta}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right]\right]\right) \quad \text{(with probability at least } 1 - \delta) \\
&\leq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\frac{1}{\delta}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{\lambda \max\{S_T, -S_T\} - \lambda^2 \tilde{S}_T}\right]\right]\right) \\
&\leq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\frac{1}{\delta}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{\lambda S_T - \lambda^2 \tilde{S}_T} + e^{\lambda(-S_T) - \lambda^2 \tilde{S}_T}\right]\right]\right) \\
&\leq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\frac{1}{\delta}\left(\mathbb{E}_{\underline{P}}\left[\mathbb{E}_{\{D_l\}_{l\leq T}}\left[e^{\lambda S_T - \lambda^2 \tilde{S}_T}\right] + \mathbb{E}_{\{D_l\}_{l\leq T}}\left[e^{\lambda(-S_T) - \lambda^2 \tilde{S}_T}\right]\right]\right)\right) \\
&\overset{(b)}{\leq} \mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}.
\end{aligned}
\tag{26}
$$

Where (a) is due to the Lemma C.2, the Markov's Inequality. For the $\tilde{S}_k$ defined in Equation (23), (b) holds because of Equation (45) in Theorem C.3. For the $\tilde{S}_k$ defined in Equation (24), it is based on Theorem C.4.

Moving $\lambda \mathbb{E}_P\left[\tilde{S}_T\right]$ to the other side of Equation (26), and dividing by λ from both sides,

$$|\mathbb{E}_P\left[S_T(\theta)\right]| \leq \frac{\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}}{\lambda} + \lambda \mathbb{E}_P\left[\tilde{S}_T\right]. \tag{27}$$

$\square$

Corollary B.3 (Proof of Theorem 4.3)**.** If we let $\tilde{S}_T$ follow the definition in Equation (23), for any $\lambda > 0$,

$$|\mathbb{E}_P\left[S_T(\theta)\right]| \leq \sqrt{2}NH\sqrt{T\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta} + \frac{\ln 2}{2\ln c}\left(\frac{\mathbb{D}_{KL}(P\|\underline{P})}{\ln(\frac{2}{\delta})} + 1\right)\right)}, \tag{28}$$

and if we let $\tilde{S}_T$ follow the definition in Equation (24), for any $\lambda \in [0, \frac{1}{b}]$, where $|D_l| \leq b$,

$$|\mathbb{E}_P\left[S_T(\theta)\right]| \leq \min\left\{2NH\sqrt{T\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{\mathcal{O}(\ln T)}{\delta}\right)}, 2NH\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{\mathcal{O}(\ln T)}{\delta}\right)\right\}. \tag{29}$$

Proof. The next step would be to optimize the λ in Equation (25), to get the tightest upper bound. However, the value of λ that minimizes Equation (25) depends on P, whereas we would like to have a result that holds for all possible distributions simultaneously, which is not possible. So we do a discretization of λ. We make a grid of λ's value in a form of a geometric sequence and for each value of $\mathbb{D}_{KL}(P\|\underline{P})$, we pick a value of λ from the grid, which is the closest to the one that minimizes the right-hand side of Equation (25) upon to some minor errors.

First, in Equation (25), we get

$$\lambda^* = \arg\min_\lambda \frac{\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}}{\lambda} + \lambda\mathbb{E}_P\left[\tilde{S}_T\right] = \sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}}{\mathbb{E}_P\left[\tilde{S}_T\right]}}. \tag{30}$$

Then putting λ^* back to Equation (25), we get

$$|\mathbb{E}_P\left[S_T(\theta)\right]| \leq 2\sqrt{\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}\right)\mathbb{E}_P\left[\tilde{S}_T\right]}. \tag{31}$$

Moreover, $\mathbb{D}_{KL}(P\|\underline{P}) \geq 0$; note that, if the $\mathbb{D}_{KL}(P\|\underline{P}) = 0$, we get

$$\lambda^{**} = \arg\min_\lambda \frac{\ln\frac{2}{\delta}}{\lambda} + \lambda\mathbb{E}_P\left[\tilde{S}_T\right] = \sqrt{\frac{\ln\frac{2}{\delta}}{\mathbb{E}_P\left[\tilde{S}_T\right]}}, \tag{32}$$

putting λ^{**} back to Equation (25) with $\mathbb{D}_{KL}(P\|\underline{P}) = 0$, we get

$$|\mathbb{E}_P\left[S_T(\theta)\right]| \leq 2\sqrt{\ln\frac{2}{\delta}\mathbb{E}_P\left[\tilde{S}_T\right]}. \tag{33}$$

Here λ^{**} is also a lower bound for the λ.

Now if we let $\tilde{S}_T$ follow the definition in (24), we have $\mathbb{E}_P\left[\tilde{S}_T\right] \leq Tb^2$ by the definition of $\tilde{S}_T$ in Equation (23) and (24), and $|D_l| \leq b$. Thus, we have $\lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \min\left\{\sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P})+\ln\frac{2}{\delta}}{\mathbb{E}_P[\tilde{S}_T]}}, \frac{1}{b}\right\}\right]$, note for such $\tilde{S}_T$, we required $\lambda \leq \frac{1}{b}$.

However, in the setting $\mathbb{D}_{KL}(P\|\underline{P}) > 0$, the value of λ that minimizing right hand side of Equation (25) is given by Equation (30), which depends on P, as early mentioned, thus we use the geometric sequence $\{\lambda_j\}_{j=0}^{J-1}$ over the range $\left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \frac{1}{b}\right]$, for $\lambda_j = c^j\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}$, for some $c > 1$ and $j = 0, \ldots, J-1$. Now we have the geometric series satisfy $c^{J-1}\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}} \leq \frac{1}{b}$ so as long as $J - 1 = \left\lceil\frac{1}{\ln c}\ln\sqrt{\frac{T}{\ln\frac{2}{\delta}}}\right\rceil$.

If $\min\left\{\sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P})+\ln\frac{2}{\delta}}{\mathbb{E}_P[\tilde{S}_T]}}, \frac{1}{b}\right\} = \sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P})+\ln\frac{2}{\delta}}{\mathbb{E}_P[\tilde{S}_T]}}$, so there are at most total $J = \left\lceil\frac{1}{\ln c}\ln\sqrt{\frac{T}{\ln\frac{2}{\delta}}}\right\rceil + 1$ λ's.

We go back to the proof of Corollary B.2, let $\delta = \sum_{j=0}^J \delta_j$, with $\delta_j = \frac{1}{J}\delta$, $j \in \mathbb{N}_{\geq 0}$. Then for any $\delta_j = \frac{1}{J}\delta$, we have

$$\begin{aligned}
&\mathbb{P}\left(\mathbb{E}_P\left[|\lambda S_T| - \lambda^2\tilde{S}_T\right] \geq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta_j}\middle|\lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \frac{1}{b}\right]\right) \\
&\stackrel{(a)}{\leq} \mathbb{P}\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T|-\lambda^2\tilde{S}_T}\right]\right)\right. \\
&\left.\geq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T|-\lambda^2\tilde{S}_T}\right]\right]\right)\middle|\lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \frac{1}{b}\right]\right),
\end{aligned} \tag{34}$$

where (a) holds because $\mathbb{E}_P\left[|\lambda S_T| - \lambda^2\tilde{S}_T\right] \leq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T|-\lambda^2\tilde{S}_T}\right]\right)$ almost surely, and $\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta_j} \geq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T|-\lambda^2\tilde{S}_T}\right]\right]\right)$ almost surely. Here $\middle|$ indicates given not conditional on. Following this, we have

$$\mathbb{P}\left(\mathbb{E}_P\left[|\lambda S_T| - \lambda^2 \tilde{S}_T\right] \geq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta_j} \,\middle|\, \lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \frac{1}{b}\right]\right)$$

$$\leq \mathbb{P}\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right]\right)\right.$$

$$\left. \geq \mathbb{D}_{KL}(P\|\underline{P}) + \ln\left(\frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right]\right]\right) \,\middle|\, \lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \frac{1}{b}\right]\right)$$

$$= \mathbb{P}\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right] \geq \frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right]\right] \,\middle|\, \lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \frac{1}{b}\right]\right) \tag{35}$$

$$\leq \max_{\lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \frac{1}{b}\right]} \mathbb{P}\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right] \geq \frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right]\right]\right)$$

$$\stackrel{(a)}{=} \max_{\lambda \in \{\lambda_1,\ldots,\lambda_J\}} \mathbb{P}\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right] \geq \frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2 \tilde{S}_T}\right]\right]\right)$$

$$\leq \sum_{j=1}^{J} \mathbb{P}\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda_j S_T| - \lambda_j^2 \tilde{S}_T}\right] \geq \frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda_j S_T| - \lambda_j^2 \tilde{S}_T}\right]\right]\right)$$

$$\stackrel{(b)}{\leq} J \times \delta_j = \delta,$$

where (a) we use the discretization of λ, where (b) we apply Lemma C.2.

Thus, we get for all $\lambda \in \left[\frac{1}{b}\sqrt{\frac{\ln\frac{2}{\delta}}{T}}, \min\left\{\sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P})+\ln\frac{2}{\delta}}{\mathbb{E}_P[\tilde{S}_T]}}, \frac{1}{b}\right\}\right]$, we can obtain the inequality as in Equation (31)

$$|\mathbb{E}_P[S_T(\theta)]| \leq 2\sqrt{\mathbb{E}_P\left[\tilde{S}_T\right]\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta_j}\right)}$$

$$\leq 2\sqrt{\mathbb{E}_P\left[\tilde{S}_T\right]\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2J}{\delta}\right)} \tag{36}$$

$$= 2NH\sqrt{T\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2J}{\delta}\right)} = 2NH\sqrt{T\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{\mathcal{O}(\ln T)}{\delta}\right)},$$

with probability at least $1-\delta$.

If $\min\left\{\sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P})+\ln\frac{2}{\delta}}{\mathbb{E}_P[\tilde{S}_T]}}, \frac{1}{b}\right\} = \frac{1}{b}$, which implies

$$\mathbb{E}_P\left[\tilde{S}_T\right] \leq b^2(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}) \tag{37}$$

, and for this value of $\lambda = \frac{1}{b}$, we put Equation (37) back to Equation (30), then we get,

$$|\mathbb{E}_P[S_T(\theta)]| \leq b\mathbb{D}_{KL}(P\|\underline{P}) + b\ln\frac{2}{\delta} + \frac{1}{b}\mathbb{E}_P\left[\tilde{S}_T\right]$$

$$\leq 2b(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}) \leq 2NH(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}). \tag{38}$$

Then under the same argument we did previously for Equation (36), we have

$$\begin{aligned}
|\mathbb{E}_P\left[S_T(\theta)\right]| &\leq 2b(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta_j}) \\
&\leq 2NH(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2J}{\delta}) = 2NH\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{\mathcal{O}(\ln T)}{\delta}\right).
\end{aligned} \tag{39}$$

Note, the range of λ depends on the $\tilde{S}_T$, which is sample dependent, thus we have the bound also depends on the sample.

Now if we let $\tilde{S}_T$ follow the definition in (23), now $\mathbb{E}_P\left[\tilde{S}_T\right] = \tilde{S}_T$ since $\tilde{S}_T$ is not random. Now the λ does not have an upper bound. We use the geometric sequence over the range of $\left[\sqrt{\frac{\ln\frac{2}{\delta}}{\tilde{S}_T}}, \infty\right]$, where $\sqrt{\frac{\ln\frac{2}{\delta}}{\tilde{S}_T}}$ is given when $\mathbb{D}_{KL}(P\|\underline{P}) = 0$. We use the same argument, let $\lambda_j = c^j\sqrt{\frac{\ln\frac{2}{\delta}}{\tilde{S}_T}}$ for some $c > 1$ and $j \geq 0$. For given value of $\mathbb{D}_{KL}(P\|\underline{P})$, the optimal λ_j in (25) equals to $\sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}}{\tilde{S}_T}}$, which requires j is the solution of $c^j\sqrt{\frac{\ln\frac{2}{\delta}}{\tilde{S}_T}} = \sqrt{\frac{\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta}}{\tilde{S}_T}}$, and we floor the value of j to the nearest integer, which is $\left\lfloor \ln\left(\frac{\mathbb{D}_{KL}(P\|\underline{P})}{\ln(\frac{2}{\delta})} + 1\right)/(2\ln c)\right\rfloor \leq \left(\ln\left(\frac{\mathbb{D}_{KL}(P\|\underline{P})}{\ln(\frac{2}{\delta})}\right) + 1\right)/(2\ln c)$.

As the same procedures in Equation (35) we used for deriving Equation (36), we go back to the proof of Corollary B.2, we let $\delta = \sum_{j=0}^{\infty}\delta_j = \sum_{j=0}^{\infty}2^{-(j+1)}\delta$, with $\delta_j = 2^{-(j+1)}\delta$, $j \in \mathbb{N}_{\geq 0}$.

Then with a similar argument in Equation (35), for any $\delta_j = 2^{-(j+1)}\delta$, we have we have

$$\begin{aligned}
&\mathbb{P}\left(\mathbb{E}_P\left[|\lambda S_T| - \lambda^2\tilde{S}_T\right] >= \mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta_j}\middle|\lambda \in \left[\sqrt{\frac{\ln\frac{2}{\delta}}{\tilde{S}_T}}, \infty\right]\right) \\
&\stackrel{\sim}{=} \max_{\lambda \in \{\lambda_1, \ldots\}}\mathbb{P}\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2\tilde{S}_T}\right] \geq \frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda S_T| - \lambda^2\tilde{S}_T}\right]\right]\right) \\
&\leq \sum_{j=1}^{\infty}\mathbb{P}\left(\mathbb{E}_{\underline{P}}\left[e^{|\lambda_j S_T| - \lambda_j^2\tilde{S}_T}\right] \geq \frac{1}{\delta_j}\mathbb{E}_{\{D_l\}_{l\leq T}}\left[\mathbb{E}_{\underline{P}}\left[e^{|\lambda_j S_T| - \lambda_j^2\tilde{S}_T}\right]\right]\right) \\
&\leq \sum_{j=1}^{\infty}\times\delta_j = \sum_{j=1}^{\infty}\delta 2^{-(j+1)} = \delta.
\end{aligned} \tag{40}$$

In the end, we get for all $\lambda \in \left[\sqrt{\frac{\ln\frac{2}{\delta}}{\tilde{S}_T}}, \infty\right]$, we can obtain the inequality as in Equation (31)

$$\begin{aligned}
|\mathbb{E}_P\left[S_T(\theta)\right]| &\leq 2\sqrt{\tilde{S}_T\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta_j}\right)} \\
&\leq 2\sqrt{\tilde{S}_T\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2 * 2^j}{\delta}\right)} \\
&\leq 2\sqrt{\sum_{l=1}^{T}\frac{(a_l - b_l)^2}{8}\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta} + \frac{\ln 2}{2\ln c}\left(\ln\left(\frac{\mathbb{D}_{KL}(P\|\underline{P})}{\ln(\frac{2}{\delta})}\right) + 1\right)\right)} \\
&\leq \sqrt{2}NH\sqrt{T\left(\mathbb{D}_{KL}(P\|\underline{P}) + \ln\frac{2}{\delta} + \frac{\ln 2}{2\ln c}\left(\ln\left(\frac{\mathbb{D}_{KL}(P\|\underline{P})}{\ln(\frac{2}{\delta})}\right) + 1\right)\right)}.
\end{aligned} \tag{41}$$

By utilizing Equation (13), the joint distribution can be decomposed into products of independent distributions that are solely dependent on the preceding episode, which can be successively absorbed into the filtration we

defined. By $P = \prod_{l=0}^{T-1} P_l$ and $\underline{P} = \prod_{l=0}^{T-1} \underline{P}_l$, we have

$$\mathbb{D}_{KL}(P\|\underline{P}) = \sum_{l=1}^{T} \mathbb{D}_{KL}(P_{l-1}\|\underline{P}_{l-1}), \tag{42}$$

putting Equation (42) back into (41), we have

$$|\mathbb{E}_P[S_T(\theta)]| \leq \sqrt{2}NH\sqrt{T\left(\sum_{l=1}^{T}\mathbb{D}_{KL}(P_{l-1}\|\underline{P}_{l-1}) + \ln\frac{2}{\delta} + \frac{\ln 2}{2\ln c}\left(\ln\left(\frac{\sum_{l=1}^{T}\mathbb{D}_{KL}(P_{l-1}\|\underline{P}_{l-1})}{\ln(\frac{2}{\delta})}\right) + 1\right)\right)}. \tag{43}$$

Next, by Lemma B.4, we have

$$\sum_{l=1}^{T-1} \mathbb{D}_{KL}(P_l\|\underline{P}_l) \leq \frac{2\lambda^2 r^2}{s_{\min}(1-\alpha)^2} \frac{1-\alpha^{2(T-1)}}{1-\alpha^2}.$$

Return to $T := K/N$, then take $\delta = 2\exp(-K)$, choose c such that $\frac{\ln 2}{2\ln c} = 1$, and by the inequality $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ and $\ln(K) + K \leq \sqrt{2}K$ for any $K > 0$, some basic algebra, we get the final bound in Theorem 4.3 Equation (8). □

Lemma B.4. Suppose Assumption 4.1 holds. Then, for any $l \in \{1, \ldots, T-1\}$, the following bound on the KL divergence holds:

$$\sum_{l=1}^{T-1} \mathbb{D}_{KL}(P_l\|\underline{P}_l) \leq \frac{2\lambda^2 r^2}{s_{\min}(1-\alpha)^2} \cdot \frac{1-\alpha^{2(T-1)}}{1-\alpha^2}.$$

Proof. For any l, we denote lth λ as λ_l, so $\lambda_1 = \lambda$, $\lambda_l = \alpha^{l-1}\lambda$, where λ and α are introduced in Algorithm 1. First by the property of KL divergence, we have

$$\mathbb{D}_{KL}(P_l\|\underline{P}_l) \leq \frac{2\|P_l - \underline{P}_l\|_\infty^2}{s_{min}}.$$

Further, given the updating rule, $\underline{P}_l = \lambda_l \underline{P}_{l-1} + (1-\lambda_l)P_l$, we have

$$\begin{aligned}
\|P_l - \underline{P}_l\|_\infty &= \|P_l - \lambda_l\underline{P}_{l-1} - (1-\lambda_l)P_l\|_\infty = \|\lambda_l P_l - \lambda_l\underline{P}_{l-1}\|_\infty \\
&= \|\lambda_l P_l - \lambda_l(\lambda_{l-1}\underline{P}_{l-2} + (1-\lambda_{l-1})P_{l-1})\|_\infty = \|\lambda_l(P_l - P_{l-1}) + \lambda_l\lambda_{l-1}(P_{l-1} - \underline{P}_{l-2})\|_\infty \\
&= \|\lambda_l(P_l - P_{l-1}) + \lambda_l\lambda_{l-1}(P_{l-1} - (\lambda_{l-2}\underline{P}_{l-3} + (1-\lambda_{l-2})P_{l-2}))\|_\infty \\
&= \|\lambda_l(P_l - P_{l-1}) + \lambda_l\lambda_{l-1}(P_{l-1} - P_{l-2}) + \lambda_l\lambda_{l-1}\lambda_{l-2}(P_{l-2} - \underline{P}_{l-3})\|_\infty \\
&= \cdots \\
&= \|\lambda_l(P_l - P_{l-1}) + \lambda_l\lambda_{l-1}(P_{l-1} - P_{l-2}) + \cdots + \lambda_l\lambda_{l-1}\cdots\lambda_2(P_2 - P_1) + \lambda_l\lambda_{l-1}\lambda_{l-2}\cdots\lambda_1(P_1 - \underline{P}_0)\|_\infty \\
&= \|\alpha^{l-1}\lambda_1(P_l - P_{l-1}) + \alpha^{l-1+l-2}\lambda_1(P_{l-1} - P_{l-2}) + \cdots + \alpha^{l-1+l-2+\cdots+1}\lambda_1(P_1 - \underline{P}_0)\|_\infty \\
&\leq \lambda_1 r\frac{\alpha^{l-1}}{1-\alpha},
\end{aligned}$$

where we use, $\underline{P}_0 = P_0$, $\lambda_l = \alpha\lambda_{l-1}$, and $\|P_l - P_{l-1}\|_\infty \leq r$, we have $\mathbb{D}_{KL}(P_l\|\underline{P}_l) \leq \frac{2\lambda^2 r^2 \alpha^{2l-2}}{s_{\min}(1-\alpha)^2}$, thus we have

$\sum_{l=1}^{T-1} \mathbb{D}_{KL}(P_l\|\underline{P}_l) \leq \sum_{l=1}^{T-1} \frac{2\lambda^2 r^2 \alpha^{2l-2}}{s_{\min}(1-\alpha)^2} \leq \frac{2\lambda^2 r^2}{s_{\min}(1-\alpha)^2} \frac{1-\alpha^{2(T-1)}}{1-\alpha^2}$.

□

B.4 Proof of Theorem 4.4

Lemma B.5 (Sample Complexity For Policy Gradient). Consider the setting of Theorem 4.3. Given a small $\epsilon > 0$, with proper choice of learning rate β, If the number of iterations T satisfies $T = \widetilde{\mathcal{O}}(\epsilon^{-4})$. Denote $\bar{J} = \frac{1}{K} \sum_{i=1}^{K} \mathbb{E}_{\{\theta_l\}_{l=0}^{T-1} \sim P} [\mathbb{E}_{\{\tau_l\}_{l=1}^{T} \sim \mathscr{T}} [J_{\mathcal{M}_i}(\pi_\theta)]]$.

Then $J^* - \bar{J} \leq \mathcal{O}(\epsilon)$.

Proof. Refer the Corollary C.1 in Yuan et al. (2022) for details. $\qquad\square$

We then begin to prove Theorem 4.4.

Proof. By the proof of Theorem 4.3 , we have $|\bar{J} - \tilde{J}| \leq \mathscr{R}(\mathbb{D}_{KL}(P\|\underset{\sim}{P}))$, then the following conditions holds

1. let $\mathscr{R}(\mathbb{D}_{KL}(P\|\underset{\sim}{P})) \leq \frac{\epsilon}{2}$,

2. and let $J^* - \bar{J} \leq \mathcal{O}(\frac{\epsilon}{2})$,

where condition 1 holds by Theorem 4.3, and condition 2 holds by Lemma B.5. By satisfying both conditions 1 and 2, we obtain, $J^* - \tilde{J} \leq \mathcal{O}(\epsilon)$. The value of K can then be determined to satisfy these conditions. $\qquad\square$

C   Auxiliary Theorems and Lemmas

Lemma C.1 (Popoviciu's inequality on variances). For bounded random variable $x \in [a, b]$, then $\text{Var}[x] \leq \frac{(b-a)^2}{4}$

Lemma C.2 (Markov's Inequality, Equation 2.1 in Wainwright (2019)). For any non-negative random variable x, it holds that $\mathbb{P}(x \geq t) \leq \frac{\mathbb{E}[x]}{t}$. Taking $t = \frac{\mathbb{E}[x]}{\delta}$, where $\delta \in (0, 1)$, it results in with probability at least $1 - \delta$, $0 \leq x \leq \frac{\mathbb{E}[x]}{\delta}$.

Theorem C.3 (Azuma-Hoeffding Inequality, Corollary 2.20 in Wainwright (2019)). For a sequence of Martingale Difference Sequence random variable $\{D_l\}_{l=1}^{T}$, if we have $D_l \in [a_l, b_l]$ almost sure for some constant $[a_l, b_l]$ and $l = 1, 2, \ldots, T$, the summation $S_T := \sum_{l=1}^{T} D_l$, and let $\tilde{S}_T = \frac{\sum_{l=1}^{T}(b_l - a_l)^2}{8}$ then:

$$\mathbb{P}(|S_T| \geq t) \leq 2e^{\frac{-t^2}{\tilde{S}_T}} \tag{44}$$

Equivalently, the moment-generating function satisfies

$$\mathbb{E}\left[e^{\lambda S_T}\right] \leq e^{\lambda^2 \tilde{S}_T}. \tag{45}$$

Furthermore, if we choose $t = \sqrt{\frac{1}{2} \ln \frac{2}{\delta} \sum_{l=1}^{T}(b_l - a_l)^2}$, we get $\mathbb{P}\left(|S_T| > \sqrt{\frac{1}{2} \ln \frac{2}{\delta} \sum_{l=1}^{T}(b_l - a_l)^2}\right) \leq \delta$.

Theorem C.4 (Freedman's inequality, Theorem 1.6 in Freedman (1975)). Let $\mathcal{F}_T$ and $\{D_l\}_{l \leq T}$ follow the definition in Equation (15), and let $|D_l| \leq b$ with probability at least 1 and $\mathbb{E}[D_l|\mathcal{F}_{l-1}] = 0$. Let $S_T := \sum_{l=1}^{T} D_l$ and $\tilde{S}_T := \sum_{l=1}^{T} \mathbb{E}[D_l^2|\mathcal{F}_{l-1}]$. Then for any $\lambda \in [0, \frac{1}{b}]$

$$\mathbb{E}_{\{D_l\}_{l \leq T}}\left[e^{\lambda S_T - \lambda^2 \tilde{S}_T}\right] \leq 1 \tag{46}$$

Proof.

$$\mathbb{E}_{D_T}\left[e^{\lambda D_T}|\mathcal{F}_{T-1}\right] \overset{(a)}{\leq} \mathbb{E}_{D_T}\left[1 + \lambda D_T + \lambda^2 D_T^2|\mathcal{F}_{T-1}\right]$$
$$= 1 + \lambda^2 \mathbb{E}_{D_T}\left[D_T^2|\mathcal{F}_{T-1}\right] \tag{47}$$
$$\overset{(b)}{\leq} e^{\lambda^2 \mathbb{E}_{D_T}[D_T^2|\mathcal{F}_{T-1}]}$$

Where (a) holds since $e^x \leq 1 + x + x^2$ for $0 < x \leq 1$, thus, we require $\lambda D_T \leq 1$, so $\lambda \leq \frac{1}{b}$, and (b) holds by $1 + x \leq e^x$. Now we have

$$
\begin{aligned}
\mathbb{E}_{\{D_l\}_{l\leq T}}\left[e^{\lambda S_T - \lambda^2 \tilde{S}_T}\right] &\overset{(a)}{=} \mathbb{E}_{\{D_l\}_{l\leq T}}\left[e^{\lambda S_{T-1} - \lambda^2 \tilde{S}_{T-1} + \lambda D_T - \lambda^2 \mathbb{E}\left[(D_T)^2 | \mathcal{F}_{T-1}\right]}\right] \\
&\overset{(b)}{=} \mathbb{E}_{\{D_l\}_{l\leq T-1}}\left[e^{\lambda S_{T-1} - \lambda^2 \tilde{S}_{T-1}} \times \mathbb{E}_{D_T}\left[e^{\lambda D_T} | \mathcal{F}_{T-1}\right] \times e^{-\lambda^2 \mathbb{E}\left[(D_T)^2 | \mathcal{F}_{T-1}\right]}\right] \\
&\overset{(c)}{\leq} \mathbb{E}_{\{D_l\}_{l\leq T-1}}\left[e^{\lambda S_{T-1} - \lambda^2 \tilde{S}_{T-1}}\right] \\
&\leq \dots \\
&\leq 1.
\end{aligned}
\tag{48}
$$

where (a) holds by the definition of $\tilde{S}_T$, and (b) holds by the definition of the definition $D_T | \mathcal{F}_{T-1}$, where (c) holds by (47), and in the last step above we have recursively applied the above argument. $\qquad\square$

Theorem C.5 (Donsker−Varadhan's Representation formula, Donsker and Varadhan (1983)). Given a probability space $(\mathcal{X}, \mathcal{B})$ and a bounded real-valued function f, where $f(x)$ is a measurable function $f : \mathcal{X} \to \mathbb{R}$, x is a random variable, and any two probability distributions P_0 and P over $\mathcal{X}$ (or, if $\mathcal{X}$ is uncountably infinite, two probability density functions),

$$
\mathbb{D}_{KL}\left(P \| \underline{P}\right) \geq \mathbb{E}_P\left[f(x)\right] - \ln \mathbb{E}_{\underline{P}}\left[e^{f(x)}\right].
\tag{49}
$$

The $\ln \mathbb{E}_{\underline{P}}\left[e^{f(x)}\right]$ on the right-hand side is the cumulant generating function. These lemmas have been commonly used in the theory of online learning (Pang et al., 2021; Kang et al., 2022; Yuan et al., 2022; Liu et al., 2021; Kang et al., 2023).

D   Policy Function Parameter θ With A Gaussian Prior

D.1   Neural Network Parametrization

In Equation (10), the parameter θ can represent the weights of a neural network. Here, we provide details on how we set up the parameter updates for the neural network weights. Let $\theta = (w_r, b_r)$ denote the random weights and biases of the r-th ($r \in \mathbb{N}_{\geq 1}$) network layer. Additionally, let ϵ_r and ϵ_{b_r} be multivariate standard normal distributed random variables. The random weights w_r and biases b_r are defined as follows:

$$
w_r = \mu_r \odot (1 + \gamma_r \epsilon_r), \gamma_r = \ln(1 + \exp(\delta_r)),
\tag{50}
$$

$$
b_r = \mu_{b_r} \odot (1 + \gamma_{b_r} \epsilon_{b_r}), \gamma_{b_r} = \ln(1 + \exp(\delta_{b_r})).
\tag{51}
$$

This implies that w_r and b_r are multivariate normal distributed according to:

$$
w_r \sim \mathcal{N}(\mu_r, \gamma_r^2 \mathrm{diag}\left(\mu_r^2\right)), \quad b_r \sim \mathcal{N}(\mu_{b_r}, \gamma_{b_r}^2 \mathrm{diag}\left(\mu_{b_r}^2\right)).
\tag{52}
$$

During optimization in each iteration, a sample of w_r and b_r is drawn from the random network parameters to perform gradient descent.

The indirect sampling according to Equations (50) and (51) ensures that the parameters $\mu_r, \gamma_r, \mu_{b_r}, \gamma_{b_r}$ can be updated. The normal prior $\underline{p}(\theta)$ is defined as:

$$
w_{\underline{r}} \sim \mathcal{N}(\underline{\mu}_r, \underline{\gamma}_r^2), \quad b_{\underline{r}} \sim \mathcal{N}(\underline{\mu}_{b_r}, \underline{\gamma}_{b_r}^2).
\tag{53}
$$

Thus, the posterior distribution for the neural network is given as $p(\theta|D) = p(\theta)p(D|\theta)/\int p(\theta)p(D|\theta)d\theta$, where $p(D|\theta) := p(g(\tau)|\theta)$ is the data likelihood. The exact likelihood function $p(D|\theta)$ and posterior policy $p(\theta|D)$ are left as future research, as mentioned in Section 4.3 "Posterior Distribution and Prior Distribution".

Instead of analytically deriving $p(\theta|D)$, we assume it belongs to a common distribution family of the prior, but with unknown parameters, which are updated by minimizing the upper bound. Therefore, we approximate the posterior $p(\theta|D)$ by a proposed distribution $q(\theta)$.

Following this approach, we can approximate the posterior $p(\theta|D)$ by updating the parameters of $q(\theta)$ using the indirect sampling chain rule. We first sample a $\theta \sim (\mathcal{N}_{w_r}, \mathcal{N}_{b_r})$, then evaluate the right-hand side in Equation (11).

We can calculate the derivatives of $\hat{U}(P, \{\mathcal{M}_i\}_{i\in[N]}, \{\theta_{l-1,j}\}_{j\in[M]}; \mu, \sigma, \underline{\mu}, \underline{\sigma})$ with respect to $\mu_r, \mu_{b_r}, \delta_r, \delta_{b_r}$ and $\underline{\mu}_r, \underline{\mu}_{b_r}, \underline{\delta}_r, \underline{\delta}_{b_r}$ in Equations (50) and (51), which we used in our implementation.

E   Environment and Experiment

E.1   Supplementary Experiment

We conducted additional experiments to examine the influence of different λ values on the final rewards for selective environments as shown in Table 1.

Table 1: Final Rewards for Different λ Values

Experiments	$\lambda = 0.84$	$\lambda = 0.86$	$\lambda = 0.88$	$\lambda = 0.90$	$\lambda = 0.92$	$\lambda = 0.94$
HalfCheetah (bodyparts)	162 (7.9)	176 (6.6)	181 (8.1)	**192 (10.2)**	184 (9.8)	174 (9.5)
Hopper (bodyparts)	302 (18.1)	318 (14.2)	321 (20.9)	**345 (14.5)**	328 (18.1)	312 (15.9)
Walker (gravity)	305 (18.8)	321 (23.6)	**357 (21.7)**	341 (24.4)	329 (26.3)	314 (21.5)
Ant (Forward-Backward)	-4.48 (0.4)	-4.17 (0.4)	-4.10 (0.3)	**-3.59 (0.3)**	-3.97 (0.4)	-4.28 (0.3)
Swimmer (Uniform)	14 (1.9)	16 (1.7)	**18 (1.1)**	17 (1.3)	16 (1.8)	15 (1.6)
Humanoid (Direction)	350 (16.8)	373 (12.4)	**388 (17.1)**	386 (11.7)	377 (16.8)	362 (15.4)

The results show that λ controls the trade-off between exploration and exploitation. Larger λ emphasizes exploration by increasing the KL penalty, while smaller λ prioritizes exploitation but risks overfitting. Note that $\lambda = 0.9$ consistently produces good result, so it can be set as the default and fine-tuned around this value if needed.

E.2   OpenAI and MAMuJoCo Environment

(a) CartPole   **(b)** LunarLander   **(c)** Swimmer   **(d)** Ant

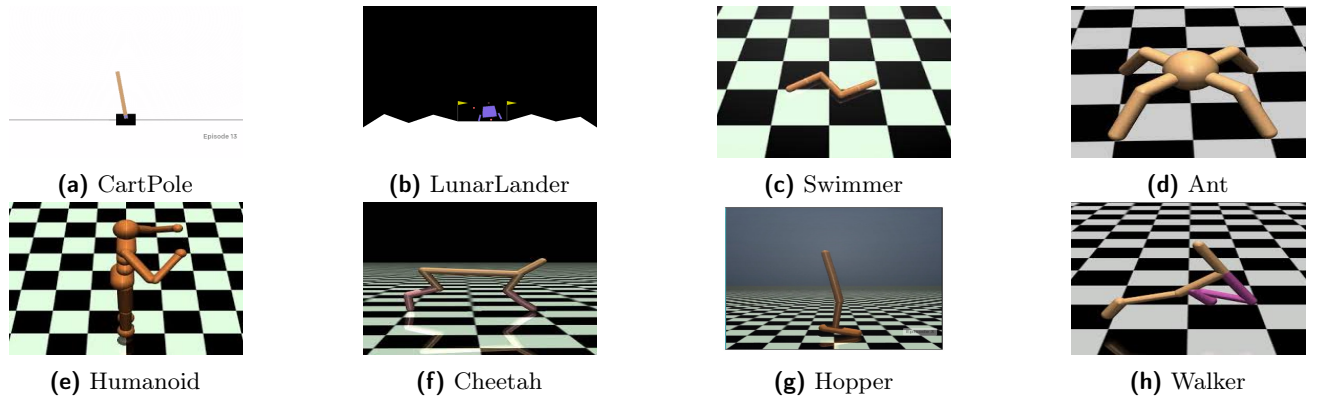**(e)** Humanoid   **(f)** Cheetah   **(g)** Hopper   **(h)** Walker

Figure 5: The illustration of environments. (a) CartPole, (b) LunarLander, (c) Swimmer, (d) Ant, (e) Humanoid, (f) Cheetah, (g) Hopper, (h) Walker

We evaluate our method on various OpenAI Gym and MuJoCo-based lifelong RL environments (see Figure 5), introducing structured variations in key dynamics to encourage continual adaptation. Table 2 summarizes these modifications.

CartPole and **LunarLander** modify mass, engine power, and length using Gaussian mixtures and uniform distributions. **Swimmer** and **Ant** introduce random movement directions, requiring adaptive control for diverse locomotion tasks. **HalfCheetah, Hopper, and Walker** adjust gravity and body morphology, simulating real-world physical variations that impact stability and efficiency.

Table 2: Different Lifelong Environments

CartPole-GMM	cart mass	$0.15[\mathcal{N}(1, 0.1^2) + 0.15\mathcal{N}(5, 0.1^2)] + 0.18[\mathcal{N}(2, 0.1^2) + 0.18\mathcal{N}(4, 0.1^2)] + 0.34\mathcal{N}(3, 0.1^2)$
	pole mass	$0.15[\mathcal{N}(0.4, 0.01^2) + \mathcal{N}(0.5, 0.01^2)] + 0.18[\mathcal{N}(0.2, 0.01^2) + \mathcal{N}(0.3, 0.01^2)] + 0.34\mathcal{N}(0.1, 0.01^2)$
	pole length	$0.15[\mathcal{N}(0.3, 0.01^2) + \mathcal{N}(0.7, 0.01^2)] + 0.18[\mathcal{N}(0.4, 0.01^2) + \mathcal{N}(0.6, 0.01^2)] + 0.34\mathcal{N}(0.5, 0.01^2)$
CartPole-Uniform	cart mass	$\mathcal{U}(1, 5)$
	pole mass	$\mathcal{U}(0.1, 0.5)$
	pole length	$\mathcal{U}(0.3, 0.7)$
LunarLander-GMM	main engine power	$0.15[\mathcal{N}(11, 0.1^2) + 0.18\mathcal{N}(12, 0.1^2)] + 0.34[\mathcal{N}(13, 0.1^2) + 0.18\mathcal{N}(14, 0.1^2)] + 0.15\mathcal{N}(15, 0.1^2)$
	side engine power	$0.15[\mathcal{N}(0.45, 0.01^2) + 0.18\mathcal{N}(0.55, 0.01^2)] + 0.34[\mathcal{N}(0.65, 0.01^2) + 0.18\mathcal{N}(0.75, 0.01^2)] + 0.15\mathcal{N}(0.85, 0.01^2)$
LunarLander-Uniform	main engine power	$\mathcal{U}(3, 20)$
	side engine power	$\mathcal{U}(0.15, 0.95)$
Swimmer-Uniform	movement direction	$\theta \sim \mathcal{U}(0, \pi)$
Humanoid-Direction-Uniform	movement direction	$\theta \sim \mathcal{U}(0, 2\pi)$
Ant-Direction-Uniform	goal direction	$\theta \sim \mathcal{U}(0, 2\pi)$
Ant-Forward-Backward-Bernoulli	movement direction	$\theta \sim \text{Categorical}(0, \pi; 0.5)$
Physics-based Variations See Mendez et al. (2020); Fu et al. (2022)	HalfCheetah-gravity	Gravity sampled from $\mathcal{U}(0.5g, 1.5g)$
	HalfCheetah-bodyparts	Mass and size scaling of torso, thigh, leg $\sim \mathcal{U}(0.5, 1.5)$
	Hopper-gravity	Gravity sampled from $\mathcal{U}(0.5g, 1.5g)$
	Hopper-bodyparts	Mass and size scaling of body parts $\sim \mathcal{U}(0.5, 1.5)$
	Walker-gravity	Gravity sampled from $\mathcal{U}(0.5g, 1.5g)$
	Walker-bodyparts	Mass and size scaling of body parts $\sim \mathcal{U}(0.5, 1.5)$

Note: g represents the standard gravity acceleration (9.81 m/s^2).

E.3 Hyper-Parameters

Table 3 list hyperparameters used in EPICG. Among these hyperparameters, the frequency of lifelong update, i.e., N is very important and closely related to the performance of both algorithm. Therefore, N is choosen carefully for each environment, whose values are shown in Table 4. For hyperparameters of other methods, we use the original source code with parameters and model architectures suggested in the original paper. We believe hyperparameters can further be tuned online and it will be our future work (Ding et al., 2022; Kang et al., 2024). The experiments were done in the GeForce RTX 2080 Ti GPU with 10 GB Memories.

Hyperparameters	Values
taks (K)	2000 or 1000
learning rate	10^{-4}
β	10^{-4}
N	chosen the best from $\{5, 10, 25, 50\}$
initial value of λ	0.9
decay factor of λ	0.95

Table 3: Hyparameters of EPICG

Environments	EPICG
Cartpole-GMM	25
Cartpole-Uniform	25
LunarLander-GMM	25
LunarLander-Uniform	25
Ant-Direction-Uniform	25
Ant-Forward-Backward-Bernoulli	10
Swimmer-Uniform	25
Humanoid-Direction-Uniform	10
HalfCheetah-gravity	10
HalfCheetah-bodyparts	10
Hopper-gravity	25
Hopper-bodyparts	25
Walker-gravity	25
Walker-bodyparts	25

Table 4: Lifelong update frequency of EPICG