Why Generalization in RL is Difficult: Epistemic POMDPs and Implicit Partial Observability

Dibya Ghosh [*1] **Jad Rahme** [*2] **Aviral Kumar** [1] **Amy Zhang** [1 3] **Ryan P. Adams** [2] **Sergey Levine** [1]

Abstract

The full version of this paper is on ArXiv: https://arxiv.org/abs/2107.06277

1. Introduction

Generalization is a central challenge in reinforcement learning (RL), and several works have observed empirically (1; 2; 3; 4) that generalization to new situations poses a significant problem to RL policies learned from a fixed training set of situations. In standard supervised learning, it is known that in the absence of distribution shift and with appropriate inductive biases, optimizing for performance on the training set (i.e., empirical risk minimization) translates into good generalization performance. It is tempting to suppose that the generalization challenges in RL can be solved in the same manner as empirical risk minimization in supervised learning: when provided a training set of contexts, learn the optimal policy within these contexts and then use that policy in new contexts at test-time.

Perhaps surprisingly, we show that such "empirical risk minimization" approaches can be sub-optimal for generalizing to new contexts in RL, even when new contexts are drawn from the same distribution as the training contexts. As an anecdotal example of why this sub-optimality arises, imagine a robotic zookeeper for feeding otters that must be trained on some set of zoos. When placed in a new zoo, the robot must find and enter the otter enclosure, and has two paths to do so: either peek through all the habitat windows looking for otters, which succeeds with 95% probability in all zoos, or to follow an image of a hand-drawn map of the zoo that unambiguously identifies the otter enclosure, which will succeed as long as the agent is able to successfully parse the image. In every training zoo, the otters can be found more reliably using the image of the map, and so an agent trained to seek the optimal policy in the training zoos would learn a classifier to predict the identity of the otter enclosure from the map, and enter the predicted enclosure. This classification strategy is optimal on the training environments because the agent can learn to perfectly classify the training zoo maps, but it is *sub-optimal* for generalization, because the learned classifier will not be able to perfectly classify every new zoo map at test-time.[1] If the learned map classifier succeeds on anything less than 95% of new zoos at test-time, the strategy of peeking through the windows, although sub-optimal in all the training environments, turns out to be a more reliable strategy for finding the otter habitat in a *new* zoo, and results in higher expected returns at test-time.

In the zookeeper example, although the hand-drawn map provides the exact location of the otter enclosure (and so the enclosure's location is technically fully observed), the agent cannot identify the true parameters of the map classifier from the small set of maps seen at training time, and so the location of the otters is implicitly obfuscated from the agent. More generally, we make the observation that, even in fully-observable domains, the agent's epistemic uncertainty renders the environment *implicitly* partially observed at test-time. We formalize this observation, and show that generalizing optimally at test-time corresponds to solving a partially-observed Markov decision process that we call an **epistemic POMDP**, which is induced by the agent's epistemic uncertainty about the test environment.

That uncertainty about MDP parameters can be modelled as a POMDP is well-studied in Bayesian RL when training and testing on a single task in an online setting, primarily in the context of exploration (5; 6; 7; 8). However, as we will discuss, this POMDP interpretation has significant and previously undescribed consequences for the generalization problem in RL, where an agent cannot collect more data online, and must instead learn a policy from a fixed set of training contexts that generalizes to new contexts at test-time. We show that the standard approaches that seek to learn policies under such uncertainty do not appropriately account for the induced partial observability, and can be arbitrarily sub-optimal for test-time generalization in theory and in practice. Maximizing expected return in the epis-

[*]Equal contribution [1]UC Berkeley [2]Princeton [3]Facebook AI Research. Correspondence to: Dibya Ghosh <dibya@berkeley.edu>, Jad Rahme <jrahme@math.princeton.edu>.

RL Theory Workshop at the 38th International Conference on Machine Learning, Copyright 2021 by the author(s).

[1]Much as a supervised learning algorithm cannot attain exactly zero test error, even if it generalizes well.

temic POMDP emerges as a principled approach to learning policies that generalize well, and we propose LEEP, an ensemble-based algorithm derived from the POMDP.

The primary contribution of this paper is to use Bayesian RL techniques to reframe generalization in RL as the problem of solving a partially observed Markov decision process, which we call the *epistemic POMDP*. The epistemic POMDP highlights the additional challenges needed for optimal generalization in RL, as compared to supervised learning. We show both theoretically and empirically the necessity of reasoning about this partial observability in order to maximize test-time performance, and suggest simple methods based on the POMDP formulation for doing so. Empirically, we demonstrate that our algorithm derived from the epistemic POMDP achieves significant gains in performance over current methods on several ProcGen benchmark tasks.

2. Problem Setup

We focus on generalization in contextual MDPs where the agent is only trained on a subsample of contexts, and seeks to generalize well across unseen contexts. A contextual MDP is an MDP $\mathcal{M}$ in which the state can be decomposed as $s_t = (c, s'_t)$, a context vector $c \in \mathcal{C}$ that remains constant throughout an episode, and a sub-state $s' \in \mathcal{S}'$ that may vary: $\mathcal{S} \coloneqq \mathcal{C} \times \mathcal{S}'$. Each context corresponds to a different situation, each with slightly different dynamics and rewards, but some shared structure across which an agent can generalize. During training, the agent is allowed to interact only within a sampled subset of contexts $\mathcal{C}_{\text{train}} \subset \mathcal{C}$. The generalization performance of the agent is measured by the return of the agent's policy in the full contextual MDP $J(\pi)$, corresponding to expected performance when placed in potentially new and unseen contexts. While our experiments will be in contextual MDPs, our theoretical results also apply to other RL generalization settings where the full MDP cannot be inferred unambiguously from the data available during training, for example in offline reinforcement learning (9).

3. Modeling Generalization in RL as an Epistemic POMDP

To understand test-time generalization in RL, we study the problem under a Bayesian perspective on epistemic uncertainty. We show that training on limited training contexts leads to an implicit partial observability at test-time that we describe using a formalism called the epistemic POMDP.

3.1. The Epistemic POMDP

In the Bayesian framework, when learning given a limited amount of evidence $\mathcal{D}$ from an MDP $\mathcal{M}$, we can use a prior distribution $\mathcal{P}(\mathcal{M})$ to construct a posterior belief distribution $\mathcal{P}(\mathcal{M}|\mathcal{D})$ over the identity of the MDP. For learning in a contextual MDP, $\mathcal{D}$ corresponds to the training contexts $\mathcal{C}_{\text{train}}$ that the agent can interact with. Since the agent cannot fully identify the MDP from the evidence, when the agent is evaluated at test-time, it is uncertain as to which MDP from the posterior distribution $\mathcal{P}(\mathcal{M}|\mathcal{D})$ is being acted in. Following a reduction common in Bayesian RL (6; 8), we model this test-time uncertainty using a partially observed MDP $\mathcal{M}^{\text{po}}$ that we will call the **epistemic POMDP**.

The epistemic POMDP is structured as follows: each new episode in the POMDP begins by sampling a single MDP $\mathcal{M} \sim \mathcal{P}(\mathcal{M}|\mathcal{D})$ from the posterior, and then the agent interacts with $\mathcal{M}$ until the episode ends in this MDP. The agent does not observe *which* MDP was sampled, and since the MDP remains fixed for the duration of the episode, this induces implicit partial observability. Effectively, each episode in the epistemic POMDP corresponds to one of the possible situations that the agent believes it might find itself in at test-time.

What makes the epistemic POMDP a useful tool is that the expected return objective in the POMDP corresponds exactly to the expected return of the agent at test-time when the Bayesian prior is accurate (Eq 1). Therefore, the optimal policy in the epistemic POMDP, $\pi^{*\text{po}}$ is the Bayes-optimal policy for maximizing the expected test-time return.

$$J_{\mathcal{M}^{\text{po}}}(\pi) = J(\pi \mid \mathcal{D}) \coloneqq \mathbb{E}_{\mathcal{M}\sim\mathcal{P}(\mathcal{M})}[J_{\mathcal{M}}(\pi) \mid \mathcal{D}]. \quad (1)$$

The epistemic POMDP is based on well-understood concepts in Bayesian reinforcement learning, but we use this construction specifically to understand generalization – a perspective that is distinct from prior work. The equivalence between test-time return and performance in the epistemic POMDP allows us to use the epistemic POMDP as a proxy for understanding how well current RL methods generalize.

3.2. Optimality in the Epistemic POMDP

We now use the structure of the epistemic POMDP to characterize properties of Bayes-optimal test-time behavior and the sub-optimality of alternative policy learning approaches.

It is well established that optimal POMDP policies are generally memory-based (10), and amongst memoryless policies, the optimal policy may be stochastic (11; 12). Because of the equivalence between the epistemic POMDP and test-time behavior, these maxims are also true for Bayes-optimal behavior for maximizing test-time performance.

Remark 3.1. *The Bayes-optimal policy for maximizing test-time performance is in general non-Markovian. When restricted to Markovian policies, the Bayes-optimal policy is in general stochastic.*

Proofs of all statements in the paper are provided in Appendix B. The fact that acting optimally at test-time can

require adaptivity (or stochasticity for memoryless policies) provides a new perspective on and formal backing to the various empirical studies that have found improved generalization performance using recurrent networks (13; 14) and stochastic regularization penalties (15; 16; 17; 18).

It is also useful to understand when partial observability *does not* play a significant role. When this is true, the POMDP objective can coincide with a surrogate MDP approximation, and Bayes-optimal solutions learned with fully-observed RL algorithms. In the epistemic POMDP,if the optimal behavior in every MDP from the posterior is identical, then acting optimally just involves following this optimal MDP policy, instead of reasoning about the hidden MDP identity. Perhaps surprisingly, being optimal everywhere is also *necessary* for such a strategy to be Bayes-optimal in general. The following proposition states that, even if a policy is optimal in many (but not all) of the possible MDPs from our posterior, this "optimal" policy can generalize poorly at test-time.

Proposition 3.1. *Let $\epsilon > 0$. There exists posterior distributions $\mathcal{P}(\mathcal{M}|\mathcal{D})$ where a deterministic Markov policy π is optimal with probability at least $1-\epsilon$,*

$$P_{\mathcal{M}\sim\mathcal{P}(\mathcal{M}|\mathcal{D})}\left(\pi \in \arg\max_{\pi'} J_{\mathcal{M}}(\pi')\right) \geq 1-\epsilon, \quad (2)$$

but is outperformed by a uniformly random policy in the epistemic POMDP: $J_{\mathcal{M}^{po}}(\pi) < J_{\mathcal{M}^{po}}(\pi_{unif})$.

This proposition indicates the brittleness of learning policies in an MDP model, since the learned policy may perform poorly at test-time, even if it captures behavior on a majority of environments in the posterior (for example, in the RL image classification task). Due to this partial observability, optimal policies for the MDPs in the posterior may be poor guidelines for Bayes-optimal behavior, and in Appendix B.2, we show that the Bayes-optimal policy may take actions that are sub-optimal in *every* environment in the posterior.

As Bayes-optimal memoryless policies are stochastic, one may wonder if simple strategies for inducing stochasticity, such as adding ϵ-greedy noise or entropy regularization, can alleviate sub-optimality. In some cases, this may be true; for certain goal-reaching problems, entropy-regularized RL can be interpreted as optimizing an epistemic POMDP objective for a specific posterior distribution (Appendix B.3) (19). In the more general setting, we show in Appendix B.4 that entropy regularization and other general-purpose techniques can similarly catastrophically fail in epistemic POMDPs.

In general, the structure of the epistemic POMDP indicates that while MDP-based algorithms can serve as a useful starting point for acquiring generalizable skills, maximizing test-time performance may require more nuanced strategies that more carefully reason about this partial observability.

4. Learning Policies in the Epistemic POMDP

We now turn our attention to deriving practical approximate methods for learning policies in the epistemic POMDP.

4.1. A Lower Bound on the POMDP Objective

We begin by developing a lower bound on the return of a policy in the epistemic POMDP, and prove that optimizing this bound recovers the Bayes-optimal policy. We assume access to n candidate MDPs sampled from the posterior distribution: $\{\mathcal{M}_i\}_{i\in[n]} \sim \mathcal{P}(\mathcal{M} \mid \mathcal{D})$. These samples define an empirical approximation of the posterior distribution, and consequently induce an empirical epistemic POMDP $\hat{\mathcal{M}}^{\text{po}}$.

As the empirical epistemic POMDP corresponds to a collection of n MDPs [2], we will decompose the optimization problem to mimic this structure, learning n policies $\pi_1, \cdots, \pi_n$, each one in one of the MDPs from the posterior, and combining these policies together to recover a single policy π. Reducing the policy learning problem into a set of MDP learning problems can allow us to leverage the many advances in deep RL for scalably solving MDPs. The following theorem, which lower-bounds the expected return of a policy π in the empirical epistemic POMDP using the policies $\{\pi_i\}_{i\in[n]}$, provides a natural objective for learning policies in this decoupled manner.

Theorem 4.1. *Let $\pi, \pi_1, \cdots \pi_n$ be $n+1$ memoryless policies, and let $r_{\max} = \max_{i,s,a} |r_{\mathcal{M}_i}(s,a)|$. The expected return of π in the empirical epistemic POMDP $J_{\hat{\mathcal{M}}^{po}}(\pi)$ is bounded below as:*

$$J_{\hat{\mathcal{M}}^{po}}(\pi) \geq \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{\sqrt{2}r_{\max}}{(1-\gamma)^2 n}\sum_{i=1}^{n} \mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi_i(\cdot|s) \;||\; \pi(\cdot|s)\right)}\right]. \quad (3)$$

This theorem indicates that if the policies in the collection $\{\pi_i\}_{i\in[n]}$ all achieve high return in their respective MDPs (first term) and are imitable by a single policy π (second term), then π is guaranteed to achieve high return in the epistemic POMDP. In contrast, if the policies cannot be closely imitated by a single policy, this collection of policies may not be useful for learning in the epistemic POMDP using the lower bound. This could be the case, for example, if each of the policies π_i is trained to maximize return on its MDP $\mathcal{M}_i$ selfishly without any consideration to the other policies or MDPs. To be useful for the lower bound, each policy π_i should balance between maximizing performance on its MDP and minimizing its deviation from the other policies in the set. The following proposition shows that if

[2] Note that when the true environment is a contextual MDP, a sample does not correspond to a single training context within a contextual MDP — it represents an approximation to the *entire* contextual MDP.

the policies are trained jointly to ensure this balance, it not only recovers a good policy, it in fact recovers the optimal policy in the empirical epistemic POMDP.

Proposition 4.1. *Let $f : \{\pi_i\}_{i\in[n]} \mapsto \pi$ be a function that maps n policies to a single policy satisfying $f(\pi, \cdots, \pi) = \pi$ for every policy π, and let α be a hyperparameter satisfying $\alpha \geq \frac{\sqrt{2}r_{max}}{(1-\gamma)^2}$. Then letting $\pi_1^*, \ldots \pi_n^*$ be the optimal solution to the following optimization problem:*

$$\{\pi_i^*\}_{i\in[n]} = \arg\max_{\pi_1,\cdots,\pi_n} \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{\alpha}{n}\sum_{i=1}^{n} \mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi_i(\cdot|s) \;||\; f(\{\pi_i\})(\cdot|s)\right)}\right], \tag{4}$$

the policy $\pi^* := f(\{\pi_i^*\}_{i\in[n]})$ is optimal for the empirical epistemic POMDP.

4.2. A Practical Algorithm: LEEP

We now derive a practical algorithm from Proposition 4.1. To do so, we discuss two problems: how posterior samples $\mathcal{M}_i \sim \mathcal{P}(\mathcal{M}|\mathcal{D})$ can be approximated, and how the function f that combines policies should be chosen.

Approximating the posterior distribution: Although exactly maintaining a posterior distribution over contextual MDPs can be difficult, we can approximate samples from the posterior via a bootstrap sampling technique (20). To sample a candidate MDP $\mathcal{M}_i$, we sample with replacement from the training contexts $\mathcal{C}_{\text{train}}$ to get a new set of contexts $\mathcal{C}^i_{\text{train}}$, and define $\mathcal{M}_i$ to be the empirical MDP on this subset of training contexts. Rolling out trials from the posterior sample $\mathcal{M}_i$ then corresponds to selecting a context at random from $\mathcal{C}^i_{\text{train}}$, and then rolling out that context.

Choosing a link function: The link function f in Proposition 4.1 that combines the set of policies together effectively serves as an inductive bias: since policy optimization in practice is approximate, different choices can yield combined policies with different characteristics. Since optimal behavior in the epistemic POMDP must consider all actions, even those that are potentially sub-optimal in all MDPs in the posterior (as discussed in Section 3.2), we use an "optimistic" link function that does not dismiss any action that is considered by at least one of the policies, specifically $f(\{\pi_i\}_{i\in[n]}) = (\max_i \pi_i)(a|s) := \frac{\max \pi_i(a|s)}{\sum_{a'} \max \pi_i(a'|s)}$.

Algorithm: We learn a set of n policies $\{\pi_i\}_{i\in[n]}$, using a policy gradient algorithm to implement the update step. To update the parameters for π_i, we take gradient steps via a surrogate loss constructed via the standard policy gradient, augmented by a disagreement penalty between the policy and the combined policy $f(\{\pi_i\}_{i\in[n]})$ with a penalty

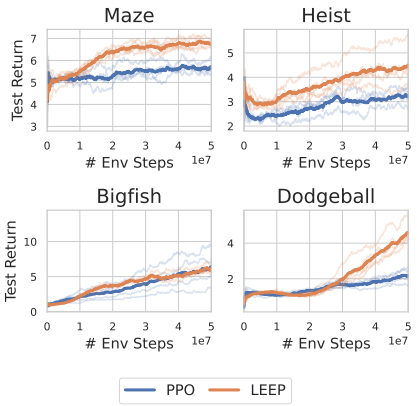

Figure 1. **Procgen.** Test return for LEEP and PPO in four Procgen environments (averaged across 5 random seeds).

parameter $\alpha > 0$, as in Equation 5:

$$\mathcal{L}(\pi_i) = \mathcal{L}^{RL}(\pi_i) + \alpha\mathbb{E}_{\pi_i,\mathcal{M}_i}[D_{KL}(\pi_i(a|s)\| \max_j \pi_j(a|s))]. \tag{5}$$

Combining these elements together leads to our method, LEEP.In summary, LEEP bootstrap samples the training contexts to create overlapping sets of training contexts $\mathcal{C}^1_{\text{train}}, \ldots \mathcal{C}^n_{\text{train}}$. Every iteration, each policy π_i generates rollouts in training contexts chosen uniformly from its corresponding $\mathcal{C}^i_{\text{train}}$, and is then updated according to Equation 5, which both maximizes the expected reward and minimizes the disagreement penalty between each π_i and the combined policy $\pi = \max_j \pi_j$.

4.3. Experimental Results on Procgen

We evaluate LEEP in the Procgen benchmark (21), a challenging suite of tasks testing generalization to unseen contexts. We instantiate our method using an ensemble of $n = 4$ policies, a penalty parameter of $\alpha = 1$, and PPO (22) to train the individual policies (implementation details in Appendix C.1). Here, we display a subset of our results, with the rest in Appendix C.

We evaluate our method on four games in which prior work has found a significant generalization challenge (21; 23; 24): Maze, Heist, BigFish, and Dodgeball. In three of the environments (Maze, Heist, and Dodgeball), our method outperforms PPO significantly, and in all games, the generalization gap between training and test performance is lower for our method. For analysis and ablations, see Appendix C.

5. Discussion

It has often been observed experimentally that generalization in RL poses a significant challenge, but it has so far remained an open question as to whether the RL setting itself presents additional generalization challenges beyond those seen in supervised learning. In this paper, we answer this question in the affirmative, and show that, in contrast

to supervised learning, generalization in RL results in a new type of problem that cannot be solved with standard MDP solution methods, due to partial observability induced by epistemic uncertainty. We call the resulting partially observed setting the epistemic POMDP, where uncertainty about the true underlying MDP results in a challenging partially observed problem. We present a practical approximate method that optimizes a bound for performance in an approximation of the epistemic POMDP, and show empirically that this approach, which we call LEEP, attains significant improvements in generalization over other RL methods that do not properly incorporate the agent's epistemic uncertainty into policy optimization. A limitation of this approach is that it optimizes a crude approximation to the epistemic POMDP with a small number of posterior samples, and may be challenging to scale to better approximations to the true objective. Developing algorithms that better model the epistemic POMDP and optimize policies within is an exciting avenue for future work, and we hope that this direction will lead to further improvements in generalization in RL.

Acknowledgements

This research was supported by an NSF graduate fellowship, the DARPA assured autonomy program, the NSF IIS-2007278 grant, a Princeton SEAS Innovation Grant and compute support from Google and Microsoft. We thank Benjamin Eysenbach, Xinyang Geng, and Justin Fu as well as members of the Princeton Laboratory for Intelligent Probabilistic Systems for helpful discussions and feedback.

References

[1] Jesse Farebrother, Marlos C. Machado, and Michael H. Bowling. Generalization and regularization in dqn. *ArXiv*, abs/1810.00123, 2018.

[2] C. Zhang, Oriol Vinyals, R. Munos, and S. Bengio. A study on overfitting in deep reinforcement learning. *ArXiv*, abs/1804.06893, 2018.

[3] Niels Justesen, R. Torrado, Philip Bontrager, Ahmed Khalifa, J. Togelius, and S. Risi. Illuminating generalization in deep reinforcement learning through procedural level generation. *arXiv: Learning*, 2018.

[4] Xingyou Song, Yiding Jiang, Stephen Tu, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. *ArXiv*, abs/1912.02975, 2020.

[5] R. Dearden, N. Friedman, and Stuart J. Russell. Bayesian q-learning. In *AAAI/IAAI*, 1998.

[6] M. Duff and A. Barto. Optimal learning: computational procedures for bayes-adaptive markov decision processes. 2002.

[7] M. Strens. A bayesian framework for reinforcement learning. In *ICML*, 2000.

[8] M. Ghavamzadeh, Shie Mannor, Joelle Pineau, and Aviv Tamar. Bayesian reinforcement learning: A survey. *Found. Trends Mach. Learn.*, 8:359–483, 2015.

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

[10] George E Monahan. State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms. *Management science*, 28(1):1–16, 1982.

[11] Satinder P. Singh, Tommi S. Jaakkola, and Michael I. Jordan. Learning without state-estimation in partially observable markovian decision processes. In *Proceedings of the Eleventh International Conference on International Conference on Machine Learning*, page 284–292, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[12] Guido Montúfar, K. Zahedi, and N. Ay. Geometry and determinism of optimal stationary control in partially observable markov decision processes. *ArXiv*, abs/1503.07206, 2015.

[13] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[14] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *CoRR*, abs/1710.06537, 2017.

[15] F. Stulp, E. Theodorou, J. Buchli, and S. Schaal. Learning to grasp under uncertainty. *2011 IEEE International Conference on Robotics and Automation*, pages 5703–5708, 2011.

[16] K. Cobbe, Oleg Klimov, Christopher Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *ICML*, 2019.

[17] Maximilian Igl, K. Ciosek, Yingzhen Li, Sebastian Tschiatschek, C. Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. In *NeurIPS*, 2019.

[18] X. Lu, Kimin Lee, P. Abbeel, and Stas Tiomkin. Dynamics generalization via information bottleneck in deep reinforcement learning. *ArXiv*, abs/2008.00614, 2020.

[19] Benjamin Eysenbach and Sergey Levine. If maxent rl is the answer, what is the question? *arXiv preprint arXiv:1910.01913*, 2019.

[20] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[21] K. Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. *ArXiv*, abs/1912.01588, 2020.

[22] John Schulman, F. Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

[23] Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. *ArXiv*, abs/2010.03934, 2020.

[24] Roberta Raileanu, Maxwell Goldstein, Denis Yarats, Ilya Kostrikov, and R. Fergus. Automatic data augmentation for generalization in deep reinforcement learning. *ArXiv*, abs/2006.12862, 2020.

[25] A. Zhang, Yuxin Wu, and Joelle Pineau. Natural environment benchmarks for reinforcement learning. *ArXiv*, abs/1811.06032, 2018.

[26] A. Zhang, Nicolas Ballas, and Joelle Pineau. A dissection of overfitting and generalization in continuous reinforcement learning. *ArXiv*, abs/1806.07937, 2018.

[27] S. Whiteson, B. Tanner, Matthew E. Taylor, and P. Stone. Protecting against evaluation overfitting in empirical reinforcement learning. *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 120–127, 2011.

[28] Zhuang Liu, Xuanlin Li, Bingyi Kang, and Trevor Darrell. Regularization matters in policy optimization – an empirical study on continuous control. *arXiv: Learning*, 2020.

[29] Alex Nichol, Vicki Pfau, Christopher Hesse, Oleg Klimov, and John Schulman. Gotta learn fast: A new benchmark for generalization in rl. *ArXiv*, abs/1804.03720, 2018.

[30] Heinrich Kuttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The nethack learning environment. *ArXiv*, abs/2006.13760, 2020.

[31] Austin Stone, Oscar Ramirez, K. Konolige, and Rico Jonschkowski. The distracting control suite - a challenging benchmark for reinforcement learning from pixels. *ArXiv*, abs/2101.02722, 2021.

[32] Max Jaderberg, V. Mnih, Wojciech Czarnecki, Tom Schaul, Joel Z. Leibo, D. Silver, and K. Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *ArXiv*, abs/1611.05397, 2017.

[33] Adam Stooke, Kimin Lee, P. Abbeel, and M. Laskin. Decoupling representation learning from reinforcement learning. *ArXiv*, abs/2009.08319, 2020.

[34] A. Zhang, Rowan McAllister, R. Calandra, Y. Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *ArXiv*, abs/2006.10742, 2020.

[35] Rishabh Agarwal, Marlos C. Machado, P. S. Castro, and Marc G. Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. *ArXiv*, abs/2101.05265, 2021.

[36] Kimin Lee, Kibok Lee, Jinwoo Shin, and H. Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. In *ICLR*, 2020.

[37] Ilya Kostrikov, Denis Yarats, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *ArXiv*, abs/2004.13649, 2020.

[38] Fereshteh Sadeghi and Sergey Levine. (cad)2rl: Real single-image flight without a single real image. *ArXiv*, abs/1611.04201, 2017.

[39] Joshua Tobin, Rachel Fong, Alex Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017.

[40] A. Rajeswaran, Sarvjeet Ghotra, Sergey Levine, and Balaraman Ravindran. Epopt: Learning robust neural network policies using model ensembles. *ArXiv*, abs/1610.01283, 2017.

[41] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, May 2018.

[42] Katie Kang, Suneel Belkhale, Gregory Kahn, Pieter Abbeel, and Sergey Levine. Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight. In *2019 international conference on robotics and automation (ICRA)*, pages 6008–6014. IEEE, 2019.

[43] Deepak Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In *IJCAI*, 2007.

[44] A. Lazaric and M. Ghavamzadeh. Bayesian multi-task reinforcement learning. In *ICML*, 2010.

[45] Wonseok Jeon, Seokin Seo, and Kee-Eung Kim. A bayesian approach to generative adversarial imitation learning. In *NeurIPS*, 2018.

[46] Luisa M. Zintgraf, K. Shiarlis, Maximilian Igl, Sebastian Schulze, Y. Gal, Katja Hofmann, and S. Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *ArXiv*, abs/1910.08348, 2020.

[47] R. Weber. On the gittins index for multiarmed bandits. *Annals of Applied Probability*, 2:1024–1033, 1992.

[48] P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete bayesian reinforcement learning. *Proceedings of the 23rd international conference on Machine learning*, 2006.

[49] R. Dearden, N. Friedman, and D. Andre. Model based bayesian exploration. In *UAI*, 1999.

[50] Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. In *NIPS*, 2013.

[51] Daniel Russo and Benjamin Van Roy. Learning to optimize via posterior sampling. *Math. Oper. Res.*, 39:1221–1243, 2014.

[52] Stephane Ross, Brahim Chaib-draa, and Joelle Pineau. Bayes-adaptive pomdps. In *NIPS*, pages 1225–1232, 2007.

[53] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *In Proc. 19th International Conference on Machine Learning*. Citeseer, 2002.

A. Related Work

Many empirical studies have demonstrated the tendency of RL algorithms to overfit significantly to their training environments (1; 2; 3; 4), and the more general increased difficulty of learning policies that generalize in RL as compared to seemingly similar supervised learning problems (25; 26; 27; 28). These empirical observations have led to a newfound interest in algorithms for generalization in RL, and the development of benchmark RL environments that focus on generalization to new contexts from a limited set of training contexts sharing a similar structure (state and action spaces) but possibly different dynamics and rewards (29; 16; 30; 21; 31).

Generalization in RL. Approaches for improving generalization in RL have fallen into two main categories: improving the ability of function approximators to generalize better with inductive biases, and incentivizing behaviors that are easier to generalize to unseen contexts. To improve the representations learned in RL, prior work has considered imitating environment dynamics (32; 33), seeking bisimulation relations (34; 35), and more generally, addressing representational challenges in the RL optimization process (17; 23). In image-based domains, inductive biases imposed via neural network design have also been proposed to improve robustness to certain factors of variation in the state (36; 37; 24). The challenges with generalization in RL that we will describe in this paper stem from the deficiencies of MDP objectives, and cannot be fully solved by choice of representations or functional inductive biases. In the latter category, one approach is domain randomization, varying environment parameters such as coefficients of friction or textures, to obtain behaviors that are effective across many candidate parameter settings (38; 39; 40; 41; 42). Domain randomization sits within a class of methods that seek robust policies by injecting noise into the agent-environment loop, whether in the state (15), the action (e.g., via max-entropy RL) (16), or intermediary layers of a neural network policy (e.g., through information bottlenecks) (17; 18). In doing so, these methods effectively introduce partial observability into the problem; while not necessarily equivalent to the partial observability that is induced by the epistemic POMDP, it may indicate why these methods generalize well empirically.

Bayesian RL: Our work recasts generalization in RL within the Bayesian RL framework, the problem of acting optimally under a belief distribution over MDPs (see Ghavamzadeh et al. (8) for a survey). Bayesian uncertainty has been studied in many sub-fields of RL (43; 44; 45; 46), the most prominent being for exploration and learning efficiently in the online RL setting. Bayes-optimal behavior in RL is often reduced to acting optimally in a POMDP, or equivalently, a belief-state MDP (6), of which our epistemic POMDP is a specific instantiation. Learning the Bayes-optimal policy exactly is intractable in all but the simplest problems (47; 48), and many works in Bayesian RL have studied relaxations that remain asymptotically optimal for learning, for example with value of perfect information (5; 49) or Thompson sampling (7; 50; 51). Our main contribution is to revisit these classic ideas in the context of generalization for RL. We find that the POMDP interpretation of Bayesian RL (5; 6; 52) provides new insights on inadequacies of current algorithms used in practice, and explains why generalization in RL can be more challenging than in supervised learning. Being Bayesian in the generalization setting also requires new tools and algorithms beyond those classically studied in Bayesian RL, since test-time generalization is measured using regret over a *single* evaluation episode, instead of throughout an online training process. As a result, algorithms and policies that minimize short-term regret (i.e., are more exploitative) are preferred over traditional algorithms like Thompson sampling that explore thoroughly to ensure asymptotic optimality at the cost of short-term regret.

B. Theoretical Results

B.1. Proposition 5.1

Proposition 3.1. *Let $\epsilon > 0$. There exists posterior distributions $\mathcal{P}(\mathcal{M}|\mathcal{D})$ where a deterministic Markov policy π is optimal with probability at least $1 - \epsilon$,*

$$P_{\mathcal{M}\sim\mathcal{P}(\mathcal{M}|\mathcal{D})}\left(\pi \in \arg\max_{\pi'} J_{\mathcal{M}}(\pi')\right) \geq 1 - \epsilon, \tag{2}$$

but is outperformed by a uniformly random policy in the epistemic POMDP: $J_{\mathcal{M}^{po}}(\pi) < J_{\mathcal{M}^{po}}(\pi_{unif})$.

Proof. Consider two deterministic MDPs, $\mathcal{M}_A$, and $\mathcal{M}_B$ that both have two states and two actions: "stay" and "switch". In both MDPs, the reward for the "stay" action is always zero. In $\mathcal{M}_A$ the reward for "switch" is always 1, while in $\mathcal{M}_B$ the reward for "switch" is $-c$ for $c > 0$. The probability of being in $\mathcal{M}_B$ is ϵ while the probability of being in $\mathcal{M}_A$ is $1 - \epsilon$. Clearly, the policy "always switch" is optimal in $\mathcal{M}_A$ and so is ϵ-optimal under the distribution on MDPs. The expected

discounted reward of the "always switch" policy is:

$$J(\pi_{\text{always switch}}) = (1-\epsilon)\frac{1}{1-\gamma} - \epsilon\frac{c}{1-\gamma} = \frac{1}{1-\gamma}(1-(c+1)\epsilon)\,. \tag{6}$$

On the other hand, we can consider a policy which selects actions uniformly at random. In this case, the expected cumulative reward is

$$J(\pi_{\text{random}}) = (1-\epsilon)\frac{1}{2}\frac{1}{1-\gamma} - \epsilon\frac{c}{2}\frac{1}{1-\gamma} = \frac{1}{2}\frac{1}{1-\gamma}(1-(c+1)\epsilon) = \frac{1}{2}J(\pi_{\text{always switch}})\,. \tag{7}$$

Thus for any ϵ we can find a $c > \frac{1}{\epsilon} - 1$ such that both policies have negative expected rewards and we prefer the random policy for being half as negative. □

B.2. Bayes Optimal policy might take suboptimal actions everywhere

Proposition B.1. *There exist posterior distributions $\mathcal{P}(\mathcal{M}|\mathcal{D})$ where the support of the Bayes-optimal memoryless policy $\pi^{*po}(a|s)$ is disjoint with that of the optimal policies in each MDP in the posterior. Formally, writing* $\text{supp}(\pi(a|s)) = \{a \in \mathcal{A} : \pi(a|s) > 0\}$*, then $\forall\mathcal{M}$ with $\mathcal{P}(\mathcal{M}|\mathcal{D}) > 0$ and $\forall s$:*

$$\text{supp}(\pi^{*po}(a|s)) \cap \text{supp}(\pi^*_{\mathcal{M}}(a|s)) = \emptyset$$

Proof. The proof is a simple modification of the construction in Proposition 5.1. Consider two deterministic MDPs, $\mathcal{M}_A$, and $\mathcal{M}_B$ with equal support under the posterior, where both have two states and three actions: "stay", "switch 1", and "switch 2". In both MDPs, the reward for the "stay" action is always zero. In $\mathcal{M}_A$ the reward for "switch" is always 1, while in $\mathcal{M}_B$ the reward for "switch" is -2. The reward structure for "switch 2" is flipped: in $\mathcal{M}_A$, the reward for "switch 2" is -2, and in $\mathcal{M}_B$, the reward is 1. Then, the policy "always switch" is optimal in $\mathcal{M}_A$, and the policy "always switch 2" is optimal in $\mathcal{M}_B$. However, any memoryless policy that takes either of these actions receives negative reward in the epistemic POMDP, and is dominated by the Bayes-optimal memoryless policy "always stay", which achieves 0 reward. □

B.3. MaxEnt RL is optimal for a choice of Prior

We describe a special case of the construction of Eysenbach and Levine (19), which shows that maximum-entropy RL in a bandit problem recovers the Bayes-optimal POMDP policy in an epistemic POMDP similar to that described in the RL image classification task.

Consider the family of MDPs $\{\mathcal{M}_k\}_{k\in[n]}$ each with one state and n actions, where taking action k in MDP $\mathcal{M}_k$ yields zero reward and the episode ends, and taking any other action yields reward -1 and the episode continues. Effectively, $\mathcal{M}_k$ corresponds to a first-exit problem with "goal action" k. Note that this MDP structure is exactly what we have for the RL image classification task for a single image. Also consider the surrogate bandit MDP $\hat{\mathcal{M}}$, also with one state and n actions, but in which taking action k yields reward r_k with immediate episode termination. The following proposition shows that running max-ent RL in $\hat{\mathcal{M}}$ recovers the optimal memoryless policy in a particular epistemic POMDP supported on $\{\mathcal{M}_k\}_{k\in[n]}$.

Proposition B.2. *Let $\pi^* = \arg\max_{\pi\in\Pi} J_{\hat{\mathcal{M}}}(\pi) + \mathcal{H}(\pi)$ be the max-ent solution in the surrogate bandit MDP $\hat{\mathcal{M}}$. Define the distribution $\mathcal{P}(\mathcal{M}|\mathcal{D})$ on $\{\mathcal{M}_k\}_{k\in[n]}$ as $\mathcal{P}(\mathcal{M}_k|\mathcal{D}) = \frac{\exp(2r_k)}{\sum_j \exp(2r_j)}$. Then, π is the optimal memoryless policy in the epistemic POMDP $\mathcal{M}^{po}$ defined by $\mathcal{P}(\mathcal{M}|\mathcal{D})$.*

Proof. See Eysenbach and Levine (19, Lemma 4.1). The optimal policy π^* is given by $\pi^*(a=k) = \frac{\exp(r_k)}{\sum_j \exp(r_j)}$. We know from Appendix **??** that this policy is optimal for epistemic POMDP $\mathcal{M}^{\text{po}}$ when $\gamma = 1$. □

If allowing for time-varying reward functions, this construction can be extended beyond seeking to epistemic POMDPs beyond bandits, and towards a more general MDP setting, where the agent seeks to reach a specific goal state, but the identity of the goal state hidden from the agent (19, Lemma 4.2).

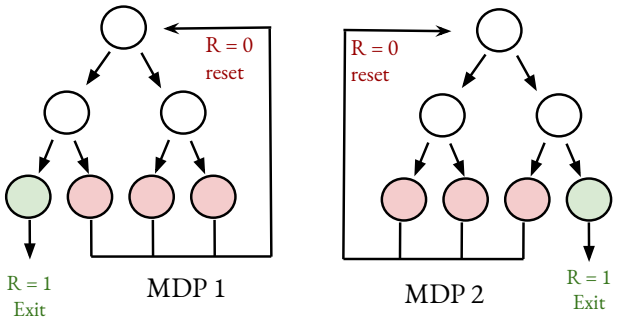

Figure 2. Visual description of Binary Tree MDPs described in proof of Proposition B.3 with depth $n = 3$.

B.4. Failure of MaxEnt RL and Uncertainty-Agnostic Regularizations

Here, we formalize the remark made in the main text that while the Bayes-optimal memoryless policy is stochastic, methods that promote stochasticity in an uncertainty-agnostic manner can fail catastrophically. We begin by explaining the significance of this result: it is well-known that stochastic policies can be arbitrarily sub-optimal in a single MDP, and can be outperformed by deterministic policies. The result we describe is more subtle than this: there are epistemic POMDPs where any attempt at being stochastic in an uncertainty-agnostic manner is sub-optimal, and *also* any attempt at acting completely deterministically is also sub-optimal. Rather, the characteristic of Bayes-optimal behavior is to be stochastic in *some* states (where it has high uncertainty), and not stochastic in others, and a useful stochastic regularization method must modulate the level of stochasticity to calibrate with regions where it has high epistemic uncertainty.

Proposition B.3. *Let $\alpha > 0, c > 0$. There exist posterior distributions $\mathcal{P}(\mathcal{M}|\mathcal{D})$, where the Bayes-optimal memoryless policy π^{*po} is stochastic. However, every memoryless policy π_s that is "everywhere-stochastic", in that $\forall s \in \mathcal{S}$: $\mathcal{H}(\pi_s(a|s)) > \alpha$, can have performance arbitrarily close to the uniformly random policy:*

$$\frac{J(\pi_s) - J(\pi_{unif})}{J(\pi^{*po}) - J(\pi_{unif})} < c$$

Proof. Consider two binary tree MDP with n levels, $\mathcal{M}_1$ and $\mathcal{M}_2$. A binary tree MDP, visualized in Figure 2, has n levels, where level k has 2^k states. On any level $k < n$, the agent can take a "left" action or a "right" action, which transitions to the corresponding state in the next level. On the final level, if the state corresponds to the terminal state (in green), then the agent receives a reward of 1, and the episode exits, and otherwise a reward of 0, and the agent returns to the top of the binary tree. The two binary tree MDPs $\mathcal{M}_1$ and $\mathcal{M}_2$ are identical except for the final terminal state: in $\mathcal{M}_1$, the terminal state is the left-most state in the final level, and in $\mathcal{M}_2$, the terminal state is the right-most state. Reaching the goal in $\mathcal{M}_1$ corresponds to taking the "left" action repeatedly, and reaching the goal in $\mathcal{M}_2$ corresponds to taking the "right" action repeatedly. We consider the posterior distribution that places equal mass on $\mathcal{M}_1$ and $\mathcal{M}_2$, $\mathcal{P}(\mathcal{M}_1|\mathcal{D}) = \mathcal{P}(\mathcal{M}_2|\mathcal{D}) = \frac{1}{2}$. A policy that reaches the correct terminal state with probability p (otherwise reset) will visit the initial state a Geom(p) number of times, and writing $\overline{\gamma} := \gamma^n$, will achieve return $\frac{\overline{\gamma}p}{1-\overline{\gamma}+p\overline{\gamma}} = \frac{1}{1+\frac{1}{p}\frac{1-\overline{\gamma}}{\overline{\gamma}}}$.

Uniform policy: A uniform policy randomly chooses between "left" and "right" at all states, and will reach all states in the final level equally often, so the probability it reaches the correct goal state is $\frac{1}{2^n}$. Therefore, the expected return is $J(\pi_{\text{unif}}) = \frac{1}{1+2^n\frac{1-\overline{\gamma}}{\overline{\gamma}}}$.

Bayes-optimal memoryless policy: The Bayes-optimal memoryless policy $\pi^{*\text{po}}$ chooses randomly between "left" and "right" at the top level; on every subsequent level, if the agent is in the left half of the tree, the agent deterministically picks "left" and on the right half of the tree, the agent deterministically picks "right". Effectively, this policy either visits the left-most state or the right-most state in the final level. The Bayes-optimal memoryless policy returns to the top of the tree a Geom$(p = \frac{1}{2})$ number of times, and the expected return is given by $J(\pi^{*\text{po}}) = \frac{1}{1+2\frac{1-\overline{\gamma}}{\overline{\gamma}}}$.

Everywhere-stochastic policy: Unlike the Bayes-optimal policy, which is deterministic in all levels underneath the first, an everywhere-stochastic policy will sometimes take random actions at these lower levels, and therefore can reach states at the final level that are neither the left-most or right-most states (and therefore always bad). We note that if $\mathcal{H}(\pi(a|s)) > \alpha$,

then there is some $\beta > 0$ such that $\max_a \pi(a|s) < 1 - \beta$. For an α-everywhere stochastic policy, the probability of taking at least one incorrect action increases as the depth of the binary tree grows, getting to the correct goal at most probability $\frac{1}{2}(1-\beta)^{n-1}$. The maximal expected return is therefore $J(\pi_s) \leq \frac{1}{1+2(\frac{1}{1-\beta})^{n-1}\frac{1-\overline{\gamma}}{\overline{\gamma}}}$

$$J(\pi^{*\text{po}}) = \frac{1}{1+2\frac{1-\overline{\gamma}}{\overline{\gamma}}} \qquad J(\pi_s) = \frac{1}{1+2(\frac{1}{1-\beta})^{n-1}\frac{1-\overline{\gamma}}{\overline{\gamma}}} \qquad J(\pi_{\text{unif}}) = \frac{1}{1+2^n\frac{1-\overline{\gamma}}{\overline{\gamma}}}$$

As $n \to \infty$, $J(\pi^{*\text{po}})$, $J(\pi_s)$ and $J(\pi_{\text{unif}})$ will converge to zero. Using asymptotic analysis we can determine their speed of convergence and find that:

$$J(\pi^{*\text{po}}) \sim \frac{\overline{\gamma}}{2} \qquad J(\pi_s) \sim \frac{\overline{\gamma}}{2(\frac{1}{1-\beta})^{n-1}} \qquad J(\pi_{\text{unif}}) \sim \frac{\overline{\gamma}}{2^n}$$

Using these asymptotics, we find that:

$$\frac{J(\pi_s) - J(\pi_{\text{unif}})}{J(\pi^{*\text{po}}) - J(\pi_{\text{unif}})} \sim \frac{1}{(\frac{1}{1-\beta})^{n-1}} = (1-\beta)^{n-1}$$

Which shows that this ratio can be made arbitrarily small as we increase n. □

An aside: deterministic policies While our proposition only discusses the failure mode of stochastic policies, *all* deterministic memoryless policies in this environment also fail. A deterministic policy π_d in this environment continually loops through one path in the binary tree repeatedly, and therefore will only ever reach one goal state; the best deterministic policy then either constantly takes the "left" action (which is optimal for $\mathcal{M}_1$), or constantly takes the "right" action (which is optimal for $\mathcal{M}_2$). Any other deterministic policy reaches a final state that is neither the left-most nor the right-most state, and will always get 0 reward. The expected return of the optimal deterministic policy is $J(\pi_d) = \frac{\overline{\gamma}}{2}$, receiving $\overline{\gamma}$ reward in one of the MDPs, and 0 reward in the other. When the discount factor γ is close to 1, the maximal expected return of a deterministic policy is approximately $\frac{1}{2}$, while the expected return of the Bayes-optimal policy is approximately 1, indicating a sub-optimality gap.

□

B.5. Proof of Theorem 6.1

Theorem 4.1. *Let $\pi, \pi_1, \cdots \pi_n$ be $n+1$ memoryless policies, and let $r_{\max} = \max_{i,s,a} |r_{\mathcal{M}_i}(s,a)|$. The expected return of π in the empirical epistemic POMDP $J_{\hat{\mathcal{M}}^{po}}(\pi)$ is bounded below as:*

$$\begin{aligned} J_{\hat{\mathcal{M}}^{po}}(\pi) &\geq \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) \\ &- \frac{\sqrt{2}r_{\max}}{(1-\gamma)^2 n}\sum_{i=1}^{n} \mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi_i(\cdot|s) \;||\; \pi(\cdot|s)\right)}\right]. \end{aligned} \tag{3}$$

Proof. Before we begin, we recall some basic tools from analysis of MDPs. For a memoryless policy π, the state-action value function $Q^\pi(s,a)$ is given by $Q^\pi(s,a) = \mathbb{E}_\pi[\sum_{t\geq 0}\gamma^t r(s_t,a_t)|s_0 = s, a_0 = a]$. The advantage function $A^\pi(s,a)$ is defined as $A^\pi(s,a) = Q^\pi(s,a) - \mathbb{E}_{a\sim\pi(\cdot|s)}[Q^\pi(s,a)]$. The performance difference lemma (53) relates the expected return of two policies π and π' in an MDP $\mathcal{M}$ via their advantage functions as

$$J_{\mathcal{M}}(\pi') = J_{\mathcal{M}}(\pi) + \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi'}_{\mathcal{M}}}[\mathbb{E}_{a\sim\pi'}[A^\pi_{\mathcal{M}}(s,a)]]. \tag{8}$$

We now begin the derivation of our lower bound:

$$\begin{aligned} J_{\hat{\mathcal{M}}^{\text{po}}} &= \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi) \\ &= \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) + \frac{1}{n}\sum_{i=1}^{n}\left[J_{\mathcal{M}_i}(\pi) - J_{\mathcal{M}_i}(\pi_i)\right] \\ &= \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{1}{n(1-\gamma)}\sum_{i=1}^{n}\mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\mathbb{E}_{a\sim\pi_i}\left[A^{\pi}_{\mathcal{M}_i}(s,a)\right]\right] \\ &= \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{1}{n(1-\gamma)}\sum_{i=1}^{n}\mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\mathbb{E}_{a\sim\pi_i}\left[A^{\pi}_{\mathcal{M}_i}(s,a)\right] - \mathbb{E}_{a\sim\pi}\left[A^{\pi}_{\mathcal{M}_i}(s,a)\right]\right] \end{aligned} \tag{9}$$

In the last equality we used the fact that $\mathbb{E}_{a\sim\pi}\left[A^{\pi}(s,a)\right] = 0$. From there we proceed to derive a lower bound:

$$\begin{aligned} \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi) &= \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{1}{n(1-\gamma)}\sum_{i=1}^{n}\mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\mathbb{E}_{a\sim\pi_i}\left[A^{\pi}_{\mathcal{M}_i}(s,a)\right] - \mathbb{E}_{a\sim\pi}\left[A^{\pi}_{\mathcal{M}_i}(s,a)\right]\right] \\ &\geq \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{2r_{max}}{n(1-\gamma)^2}\sum_{i=1}^{n}\mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[D_{TV}\left(\pi_i(\cdot\mid s);\pi(\cdot\mid s)\right)\right] \\ &\geq \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{\sqrt{2}r_{max}}{(1-\gamma)^2 n}\sum_{i=1}^{n}\mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi_i(\cdot\mid s)\;||\;\pi(\cdot\mid s)\right)}\right] \end{aligned} \tag{10}$$

where the first inequality is since $|A^{\pi}_{\mathcal{M}_i}(s,a)| \leq \frac{r_{\max}}{1-\gamma}$ and the second from Pinsker's inequality. Our intention in this derivation is not to obtain the tighest lower bound possible, but rather to illustrate how bounding the advantage can lead to a simple lower bound on the expected return in the POMDP. The inequality can be made tighter using other bounds on $|A^{\pi}_{\mathcal{M}_i}(s,a)|$, for example using $A_{\max} = \max_{i,s,a}|A^{\pi}_{\mathcal{M}_i}(s,a)|$, or potentially a bound on the advantage that varies across state. □

B.6. Proof of Proposition 6.1

Proposition 4.1. *Let $f : \{\pi_i\}_{i\in[n]} \mapsto \pi$ be a function that maps n policies to a single policy satisfying $f(\pi,\cdots,\pi) = \pi$ for every policy π, and let α be a hyperparameter satisfying $\alpha \geq \frac{\sqrt{2}r_{max}}{(1-\gamma)^2}$. Then letting $\pi^*_1,\ldots\pi^*_n$ be the optimal solution to the following optimization problem:*

$$\begin{aligned} \{\pi^*_i\}_{i\in[n]} = \arg\max_{\pi_1,\cdots,\pi_n} &\frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) \\ &- \frac{\alpha}{n}\sum_{i=1}^{n}\mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi_i(\cdot|s)\;||\;f(\{\pi_i\})(\cdot|s)\right)}\right], \end{aligned} \tag{4}$$

the policy $\pi^* \coloneqq f(\{\pi^*_i\}_{i\in[n]})$ is optimal for the empirical epistemic POMDP.

Proof. By Theorem 4.1 we have that $\forall\alpha \geq \frac{\sqrt{2}r_{\max}}{(1-\gamma)^2 n}$:

$$J_{\hat{\mathcal{M}}^{\text{po}}}(f(\{\pi^*_i\})) \geq \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi^*_i) - \alpha\sum_{i=1}^{n}\mathbb{E}_{s\sim d^{\pi^*_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi^*_i(\cdot|s)\;||\;f(\{\pi^*_i\})(\cdot|s)\right)}\right]. \tag{11}$$

Now, write $\pi'^* \in \arg\max_\pi J_{\hat{\mathcal{M}}^{\text{po}}}(\pi)$ to be an optimal policy in the empirical epistemic POMDP, and consider the collection

of policies $\{\pi'^*, \pi'^*, \ldots, \pi'^*\}$. Since $\{\pi_i^*\}$ is the optimal solution to Equation 4, we have

$$\begin{aligned} J_{\hat{\mathcal{M}}^{\text{po}}}(f(\{\pi_i^*\})) &\geq \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi'^*) - \alpha \sum_{i=1}^{n} \mathbb{E}_{s\sim d^{\pi'^*}_{\mathcal{M}_i}} \left[\sqrt{D_{KL}\left(\pi'^*(\cdot|s) \;||\; f(\{\pi'^*\})(\cdot|s)\right)}\right] \\ &= \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi'^*) \\ &= J_{\hat{\mathcal{M}}^{\text{po}}}(\pi'^*), \end{aligned} \tag{12}$$

where the second line here uses the fact that $f(\pi'^*, \ldots, \pi'^*) = \pi'^*$. Therefore $\pi^* := f(\{\pi_i^*\})$ is optimal for the empirical epistemic POMDP. □

C. Experiments

We evaluate LEEP in the Procgen benchmark (21), a challenging suite of tasks testing generalization to unseen contexts. Our experiments seek to answer the following questions:

1. Does LEEP (derived from the epistemic POMDP) lead to improved test-time performance over standard RL methods?
2. Can LEEP prevent overfitting when provided a limited number of training contexts?
3. How do different algorithmic components of LEEP affect test-time performance ?

The Procgen benchmark is a set of procedurally generated games, each with different generalization challenges. In each game, during training, the algorithm can interact with 200 training levels, before it is asked to generalize to the full distribution of levels. The agent receives a $64 \times 64 \times 3$ image observation, and must output one of 15 possible actions. We instantiate our method using an ensemble of $n = 4$ policies, a penalty parameter of $\alpha = 1$, and PPO (22) to train the individual policies (implementation details in Appendix C.1).

We evaluate our method on four games in which prior work has found a large gap between training and test performance, and which we therefore conclude pose a significant generalization challenge (21; 23; 24): Maze, Heist, BigFish, and Dodgeball. In Figure 3 (left), we compare the test-time performance of the policies learned using our method to those learned by a PPO agent with entropy regularization. In three of these environments (Maze, Heist, and Dodgeball), our method outperforms PPO by a significant margin, and in all cases, we find that the generalization gap between training and test performance is lower for our method than PPO (Appendix **??**). To understand how LEEP behaves with fewer training contexts, we ran on the Maze task with only 50 levels (Figure 3 (center)); the test return of the PPO policy decreases through training, leading to final performance worse than the starting random policy, but our method avoids this degradation.

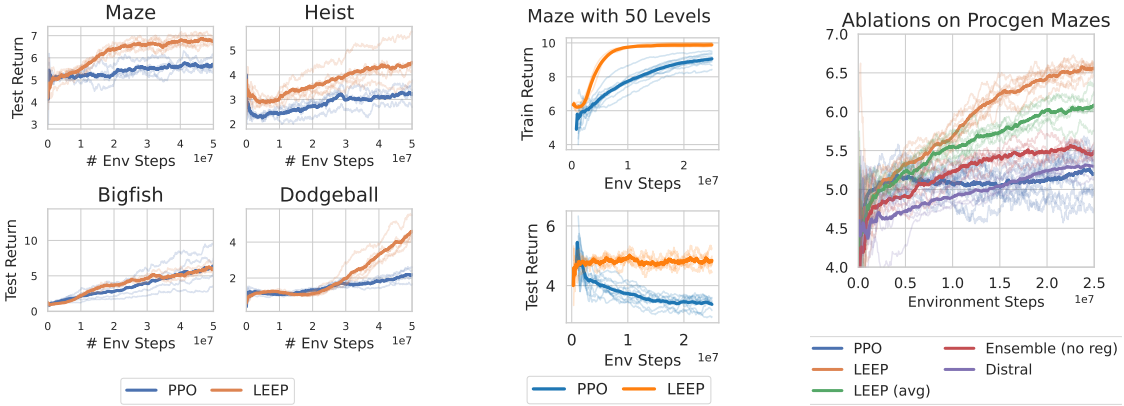

Figure 3. **Procgen results.** *(left)* Test set return for LEEP and PPO throughout training in four Procgen environments (averaged across 5 random seeds). *(center)* Performance on Maze with 50 training levels. *(right)* Ablations of LEEP and comparisons in Maze.

We perform an ablation study on the Maze and Heist environments (Maze in Figure 3 (right), Heist in Appendix **??**) to rule out potential confounding causes for the improved generalization that our method displays on the Procgen benchmark tasks.

First, to see if the performance benefit derives solely from the use of ensembles, we compare LEEP to a Bayesian model averaging strategy that trains an ensemble of policies without regularization ("Ensemble (no reg)"), and uses a mixture of these policies. This strategy does improve performance over the PPO policy, but does not match LEEP, indicating the usefulness of the regularization. Second, we compared to a version of LEEP that combines the ensemble policies together using the average $\frac{1}{n}\sum_{i=1}^{n}\pi_i(a|s)$ ("LEEP (avg)") achieves worse test-time performance than the optimistic version, which indicates that the inductive bias conferred by the $\max_i \pi_i$ link function is a useful component of the algorithm. Finally, we compare to Distral, a multi-task learning method with different motivations but similar structure to LEEP: this method helps accelerate learning on the provided training contexts (see figures on next page), but does not improve generalization performance as LEEP does.

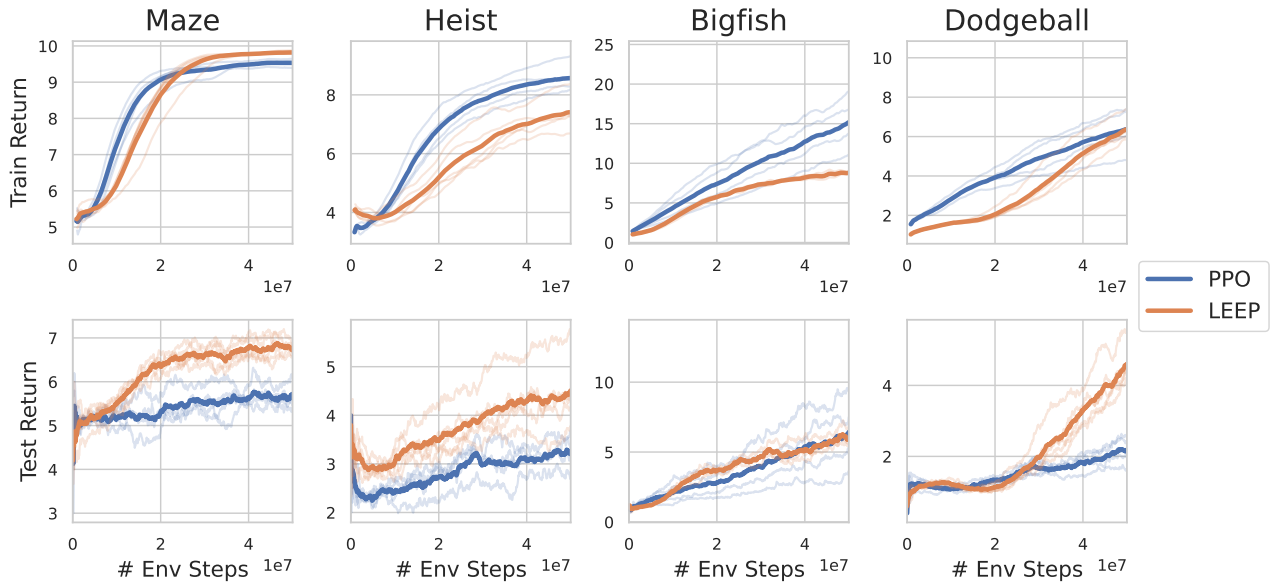

Figure 4. Training (top) and test (bottom) returns for LEEP and PPO on four Procgen environments. Results averaged across 5 random seeds. LEEP achieves equal or higher training return compared to PPO, while having a lower generalization gap between test and training returns.

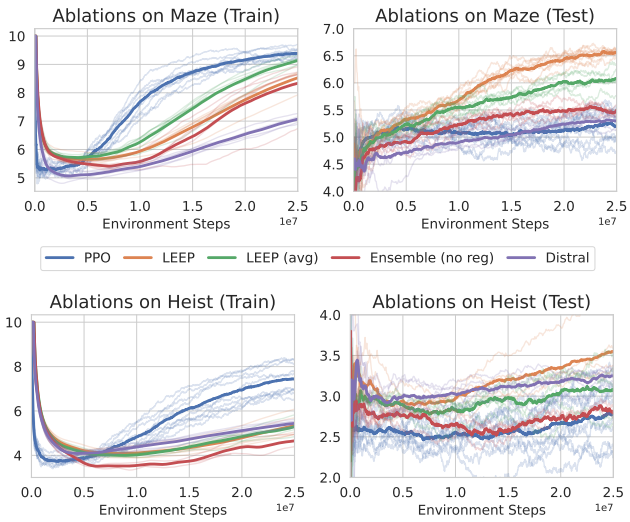

Figure 5. Training and test returns for various ablations and comparisons of LEEP.

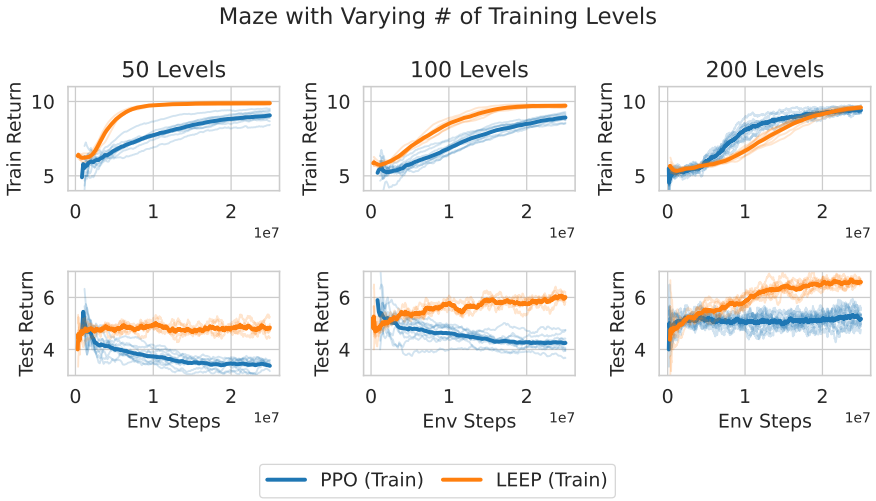

Figure 6. Performance of LEEP and PPO as the number of training levels provided varies. While the learned performance of the PPO policy is worse than a *random policy* with less training levels, LEEP avoids this overfitting and in general, demonstrates a smaller train-test performance gap than PPO.

C.1. Procgen Implementation and Experimental Setup

We follow the training and testing scheme defined by Cobbe et al. (21) for the Procgen benchmarks: the agent trains on a fixed set of levels, and is tested on the full distribution of levels. Due to our limited computational budget, we train on the so-called "easy" difficulty mode using the recommended 200 training levels. Nonetheless, many prior work has found a significant generalization gap between test and train performance even in this easy setting, indicating it a useful benchmark for generalization (21; 24; 23). We implemented LEEP on top of an existing open-source codebase released by Jiang et al. (23). Full code is provided in the supplementary for reference.

LEEP maintains $n = 4$ policies $\{\pi_i\}_{i\in[n]}$, each parameterized by the ResNet architecture prescribed by (21). In LEEP, each policy is optimized to maximize the entropy-regularized PPO surrogate objective alongside a one-step KL divergence penalty between itself and the linked policy $\max_i \pi_i$; gradients are not taken through the linked policy.

$$\mathbb{E}_{\pi_i}[\min(r_t(\pi)A^\pi(s,a), \text{clip}(r_t(\pi), 1-\epsilon, 1+\epsilon)A^\pi(s,a) + \beta\mathcal{H}(\pi_i(a|s)) - \alpha D_{KL}(\pi_i(a|s)\|\max_j \pi_j(a|s))]$$

The penalty hyperparameter α was obtained by performing a hyperparameter search on the Maze task for all the comparison methods (including LEEP) amongst $\alpha \in [0.01, 0.1, 1.0, 10.0]$. Since LEEP trains 4 policies using the same environment budget as a single PPO policy, we change the number of environment steps per PPO iteration from 16384 to 4096, so that the PPO baseline and each policy in our method takes the same number of PPO updates. All other PPO hyperparameters are taken directly from (23).

In our implementation, we parallelize training of the policies across GPUs, using one GPU for each policy. We found it infeasible to run more ensemble members due to GPU memory constraints without significant slowdown in wall-clock time. Running LEEP on one Procgen environment for 50 million steps requires approximately 5 hrs in our setup on a machine with four Tesla T4 GPUs.