Unsupervised Reinforcement Learning in Multiple Environments

Mirco Mutti[1,2,*], **Mattia Mancassola**[1,*], and **Marcello Restelli**[1]

[1] Politecnico di Milano, Milan, Italy
[2] Università di Bologna, Bologna, Italy
mirco.mutti@polimi.it, mattia.mancassola@mail.polimi.it, marcello.restelli@polimi.it

Abstract

Several recent works have been dedicated to unsupervised reinforcement learning in a single environment, in which a policy is first pre-trained with unsupervised interactions, and then fine-tuned towards the optimal policy for several downstream supervised tasks defined over the same environment. Along this line, we address the problem of unsupervised reinforcement learning in a class of multiple environments, in which the policy is pre-trained with interactions from the whole class, and then fine-tuned for several tasks in any environment of the class. Notably, the problem is inherently multi-objective as we can trade off the pre-training objective between environments in many ways. In this work, we foster an exploration strategy that is sensitive to the most adverse cases within the class. Hence, we cast the exploration problem as the maximization of the mean of a critical percentile of the state visitation entropy induced by the exploration strategy over the class of environments. Then, we present a policy gradient algorithm, αMEPOL, to optimize the introduced objective through mediated interactions with the class. Finally, we empirically demonstrate the ability of the algorithm in learning to explore challenging classes of continuous environments and we show that reinforcement learning greatly benefits from the pre-trained exploration strategy w.r.t. learning from scratch.

1 Introduction

The typical Reinforcement Learning (RL, Sutton and Barto 2018) setting involves a learning agent interacting with an environment in order to maximize a reward signal. In principle, the reward signal is a given and perfectly encodes the task. In practice, the reward is usually hand-crafted, and designing it to make the agent learn a desirable behavior is often a huge challenge. This poses a serious roadblock on the way of autonomous learning, as any task requires a costly and specific formulation, while the synergy between solving one RL problem and another is very limited. To address this crucial limitation, several recent works (Mutti, Pratissoli, and Restelli 2021; Liu and Abbeel 2021b,a; Seo et al. 2021; Yarats et al. 2021) have been dedicated to *unsupervised* RL. In this framework, originally envisioned in (Hazan et al. 2019; Mutti and Restelli 2020), the agent first pre-trains its policy by taking a large amount of unsupervised interactions with the environment (*unsupervised pre-training*). Then, the pre-trained policy is transferred to several downstream tasks, each of them defined through a reward function, and the agent has to learn an optimal policy by taking additional supervised interactions with the environment (*supervised fine-tuning*). Whereas most of the existing works in unsupervised RL (Campos et al. (2021) make for a notable exception) converged to a straightforward fine-tuning strategy, in which the pre-trained policy is employed as an exploratory initialization of a standard RL algorithm, there is lesser consensus on which unsupervised objective is best suited for the pre-training phase. Traditional intrinsic motivation bonuses that were originally designed to address exploration in supervised RL (e.g., Pathak et al. 2017; Burda et al. 2019) can be employed in the unsupervised RL setting as well (Laskin et al. 2021). However, these bonuses are designed to vanish over time, which makes it hard to converge to a stable policy during the unsupervised pre-training. The *Maximum State Visitation Entropy* (MSVE, Hazan et al. 2019) objective, which incentives the agent to learn a policy that maximizes the entropy of the induced state visitation, emerged as a powerful alternative in both continuous control and visual domains (Laskin et al. 2021). The intuition underlying the MSVE objective is that a pre-trained exploration strategy should visit with high probability any state where the agent might be rewarded in a subsequent supervised task, so that the fine-tuning to the optimal policy is feasible. Although unsupervised pre-training methods effectively reduce the reliance on a reward function and lead to remarkable fine-tuning performances w.r.t. RL from scratch, all of the previous solutions to unsupervised RL assume the existence of a single environment.

In this work, we aim to push the generality of this framework even further, by addressing the problem of *unsupervised RL in multiple environments*. In this setting, during the pre-training the agent faces a class of reward-free environments that belong to the same domain but differ in their transition dynamics. At each turn of the learning process, the agent is drawn into an environment within the class, where it can interact for a finite number of steps before facing another turn. The ultimate goal of the agent is to pre-train an exploration strategy that helps to solve *any* subsequent fine-tuning task that can be specified over *any* environment of the class.

[*] These authors contributed equally. The appendix of this paper is available at https://arxiv.org/abs/2112.08746. The αMEPOL algorithm is implemented at https://github.com/muttimirco/alphamepol.

Our contribution to the problem of unsupervised RL in multiple environments is three-fold: First, we frame the problem into a tractable *formulation* (Section 4), then, we propose a *methodology* to address it (Section 5), for which we provide a thorough *empirical* evaluation (Section 6). Specifically, we extend the pre-training objective to the multiple-environments setting. Notably, when dealing with multiple environments the pre-training becomes a *multi-objective* problem, as one could establish any combination of preferences over the environments. Previous unsupervised RL methods would blindly optimize the average of the pre-training objective across the class, implicitly establishing a uniform preference. Instead, in this work we consider the mean of a critical percentile of the objective function, i.e., its Conditional Value-at-Risk (CVaR, Rockafellar, Uryasev et al. 2000) at level α, to prioritize the performance in particularly rare or adverse environments. In line with the MSVE literature, we chose the CVaR of the induced state visitation entropy as the pre-training objective, and we propose a policy gradient algorithm (Deisenroth, Neumann, and Peters 2013), *α-sensitive Maximum Entropy POLicy optimization* (αMEPOL), to optimize it via mere interactions with the class of environments. As in recent works (Mutti, Pratissoli, and Restelli 2021; Liu and Abbeel 2021b; Seo et al. 2021), the algorithm employs non-parametric methods to deal with state entropy estimation in continuous and high-dimensional environments. Then, it leverages these estimated values to optimize the CVaR of the entropy by following its policy gradient (Tamar, Glassner, and Mannor 2015). Finally, we provide an extensive experimental analysis of the proposed method in both the unsupervised pre-training over classes of multiple environments, and the supervised fine-tuning over several tasks defined over the class. The exploration policy pre-trained with αMEPOL allows to solve sparse-rewards tasks that are impractical to learn from scratch, while consistently improving the performance of a pre-training that is blind to the unfavorable cases.

2 Related Work

In this section, we revise the works that relates the most with the setting of unsupervised RL in multiple environments. A more comprehensive discussion can be found in Appendix A.

In a previous work, Rajendran et al. (2020) considered a learning process composed of agnostic pre-training (called a *practice*) and supervised fine-tuning (a *match*) in a class of environments. However, in their setting the two phases are alternated, and the supervision signal of the matches allows to learn the reward for the practice through a meta-gradient.

Parisi et al. (2021) addresses the unsupervised RL in multiple environments concurrently to our work. Whereas their setting is akin to ours, they come up with an essentially orthogonal solution. Especially, they consider a pre-training objective inspired by count-based methods (Bellemare et al. 2016) in place of our entropy objective. Whereas they design a specific bonus for the multiple-environments setting, they essentially establish a uniform preference over the class instead of prioritizing the worst-case environment as we do.

Finally, our framework resembles the *meta-RL* setting (Finn, Abbeel, and Levine 2017), in which we would call *meta-training* the unsupervised pre-training, and *meta-testing* the supervised fine-tuning. However, none of the existing works combine unsupervised meta-training (Gupta et al. 2018a) with a multiple-environments setting.

3 Preliminaries

A vector $\boldsymbol{v}$ is denoted in bold, and v_i stands for its i-th entry.

Probability and Percentiles Let X be a random variable distributed according to a cumulative density function (cdf) $F_X(x) = Pr(X \leq x)$. We denote with $\mathbb{E}[X]$, $\mathbb{V}\mathrm{ar}[X]$ the expected value and the variance of X respectively. Let $\alpha \in (0, 1)$ be a confidence level, we call the α-percentile (shortened to $\alpha\%$) of the variable X its Value-at-Risk (VaR), which is defined as

$$\mathrm{VaR}_\alpha(X) = \inf \big\{ x \mid F_X(x) \geq \alpha \big\}.$$

Analogously, we call the mean of this same α-percentile the Conditional Value-at-Risk (CVaR) of X,

$$\mathrm{CVaR}_\alpha(X) = \mathbb{E}\big[X \mid X \leq \mathrm{VaR}_\alpha(X)\big].$$

Markov Decision Processes A Controlled Markov Process (CMP) is a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, P, D)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, the transition model $P(s'|a, s)$ denotes the conditional probability of reaching state s' when selecting action a in state s, and D is the initial state distribution. The behavior of an agent is described by a policy $\pi(a|s)$, which defines the probability of taking action a in s. Let Π be the set of all the policies. Executing a policy π in a CMP over T steps generates a trajectory $\tau = (s_{0,\tau}, a_{0,\tau}, \ldots, a_{T-2,\tau}, s_{T-1,\tau})$ such that $p_{\pi,\mathcal{M}}(\tau) = D(s_{0,\tau}) \prod_{t=0}^{T-1} \pi(a_{t,\tau}|s_{t,\tau}) P(s_{t+1,\tau}|s_{t,\tau}, a_{t,\tau})$ denotes its probability. We denote the state-visitation frequencies induced by τ with $d_\tau(s) = \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{1}(s_{t,\tau} = s)$, and we call $d_\pi^{\mathcal{M}} = \mathbb{E}_{\tau \sim p_{\pi,\mathcal{M}}}[d_\tau]$ the marginal state distribution. We define the differential entropy (Shannon 1948) of d_τ as $H(d_\tau) = -\int_{\mathcal{S}} d_\tau(s) \log d_\tau(s) \, \mathrm{d}s$. For simplicity, we will write $H(d_\tau)$ as a random variable $H_\tau \sim \delta(h - H(d_\tau)) p_{\pi,\mathcal{M}}(\tau)$, where $\delta(h)$ is a Dirac delta.

By coupling a CMP $\mathcal{M}$ with a reward function R we obtain a Markov Decision Process (MDP, Puterman 2014) $\mathcal{M}^R := \mathcal{M} \cup R$. Let $R(s, a)$ be the expected immediate reward when taking $a \in \mathcal{A}$ in $s \in \mathcal{S}$ and let $R(\tau) = \sum_{t=0}^{T-1} R(s_{t,\tau})$, the *performance* of a policy π over the MDP $\mathcal{M}^R$ is defined as

$$\mathcal{J}_{\mathcal{M}^R}(\pi) = \mathop{\mathbb{E}}_{\tau \sim p_{\pi,\mathcal{M}}} \big[R(\tau)\big]. \tag{1}$$

The goal of reinforcement learning (Sutton and Barto 2018) is to find an optimal policy $\pi^*_{\mathcal{J}} \in \arg\max \mathcal{J}_{\mathcal{M}^R}(\pi)$ through sampled interactions with an unknown MDP $\mathcal{M}^R$.

4 Unsupervised RL in Multiple Environments

Let $\boldsymbol{\mathcal{M}} = \{\mathcal{M}_1, \ldots, \mathcal{M}_I\}$ be a class of unknown CMPs, in which every element $\mathcal{M}_i = (\mathcal{S}, \mathcal{A}, P_i, D)$ has a specific transition model P_i, while $\mathcal{S}, \mathcal{A}, D$ are homogeneous across the class. At each turn, the agent is able to interact with a single environment $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. The selection of the environment to interact with is mediated by a distribution $p_{\boldsymbol{\mathcal{M}}}$ over $\boldsymbol{\mathcal{M}}$,

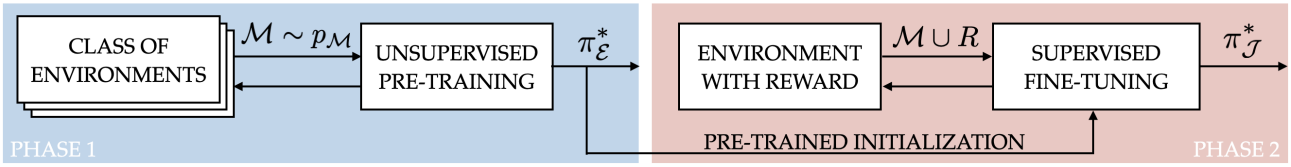


Figure 1: On the left, we highlight the unsupervised pre-training, in which the agent iteratively interacts with a CMP $\mathcal{M} \in \boldsymbol{\mathcal{M}}$ drawn from $p_{\mathcal{M}}$. The pre-trained policy $\pi^*_{\mathcal{E}}$ conveys the initialization to the subsequent supervised fine-tuning (on the right), which outputs a reward maximizing policy $\pi^*_{\mathcal{J}}$ for an MDP $\mathcal{M} \cup R$ that pairs $\mathcal{M} \in \boldsymbol{\mathcal{M}}$ with an arbitrary reward function R.

outside the control of the agent. The aim of the agent is to pre-train an exploration strategy that is general across all the MDPs $\mathcal{M}^R$ one can build upon $\boldsymbol{\mathcal{M}}$. In a single-environment setting, this problem has been assimilated to learning a policy that maximizes the entropy of the induced state visitation frequencies (Hazan et al. 2019; Mutti and Restelli 2020). One can straightforwardly extend the objective to multiple environments by considering the expectation over the class of CMPs, $\mathcal{E}_{\boldsymbol{\mathcal{M}}}(\pi) = \mathbb{E}_{\substack{\mathcal{M}\sim p_{\boldsymbol{\mathcal{M}}} \\ \tau\sim p_{\pi,\mathcal{M}}}}\big[H_\tau\big]$, where the usual entropy objective over the single environment $\mathcal{M}_i$ can be easily recovered by setting $p_{\mathcal{M}_i} = 1$. However, this objective function does not account for the tail behavior of H_τ, i.e., for the performance in environments of $\boldsymbol{\mathcal{M}}$ that are rare or particularly unfavorable. This is decidedly undesirable as the agent may be tasked with an MDP built upon one of these adverse environments in the subsequent supervised fine-tuning, where even an optimal strategy w.r.t. $\mathcal{E}_{\boldsymbol{\mathcal{M}}}(\pi)$ may fail to provide sufficient exploration. To overcome this limitation, we look for a more nuanced exploration objective that balances the expected performance with the sensitivity to the tail behavior. By taking inspiration from the risk-averse optimization literature (Rockafellar, Uryasev et al. 2000), we consider the CVaR of the state visitation entropy induced by π over $\boldsymbol{\mathcal{M}}$,

$$\begin{aligned}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi) &= \mathrm{CVaR}_\alpha(H_\tau) \\ &= \mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim p_{\boldsymbol{\mathcal{M}}} \\ \tau\sim p_{\pi,\mathcal{M}}}}\big[H_\tau \mid H_\tau \leq \mathrm{VaR}_\alpha(H_\tau)\big], \end{aligned} \tag{2}$$

where α is a confidence level and $\mathcal{E}^1_{\boldsymbol{\mathcal{M}}}(\pi) := \mathcal{E}_{\boldsymbol{\mathcal{M}}}(\pi)$. The lower we set the value of α, the more we hedge against the possibility of a bad exploration outcome in some $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. In the following sections, we propose a method to effectively learn a policy $\pi^*_{\mathcal{E}} \in \arg\max \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi)$ through mere interactions with $\boldsymbol{\mathcal{M}}$, and we show how this serves as a pre-training for RL (the full process is depicted in Figure 1). A preliminary theoretical characterization of the problem of optimizing $\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi)$ is provided in Appendix B.

5 A Policy Gradient Approach

In this section, we present an algorithm, called α-*sensitive Maximum Entropy POLicy optimization* (αMEPOL), to optimize the exploration objective in (2) through mediated interactions with a class of continuous environments.

αMEPOL operates as a typical policy gradient approach (Deisenroth, Neumann, and Peters 2013). It directly searches for an optimal policy by navigating a set of parametric differentiable policies $\Pi_\Theta := \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^n\}$. It does so by repeatedly updating the parameters $\boldsymbol{\theta}$ in the gradient direction, until a stationary point is reached. This update has the form

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \beta \nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}),$$

where β is a learning rate, and $\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ is the gradient of (2) w.r.t. $\boldsymbol{\theta}$. The following proposition provides the formula of $\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$. The derivation follows closely the one in (Tamar, Glassner, and Mannor 2015, Proposition 1), which we have adapted to our objective function of interest (2).

Proposition 5.1. *The policy gradient of the exploration objective $\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ w.r.t. $\boldsymbol{\theta}$ is given by*

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim p_{\boldsymbol{\mathcal{M}}} \\ \tau\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}}\Bigg[\Bigg(\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_{t,\tau}|s_{t,\tau})\Bigg) \\ \times \Big(H_\tau - \mathrm{VaR}_\alpha(H_\tau)\Big)\Big| H_\tau \leq \mathrm{VaR}_\alpha(H_\tau)\Bigg].\end{aligned}$$

However, in this work we do not assume full knowledge of the class of CMPs $\boldsymbol{\mathcal{M}}$, and the expected value in Proposition 5.1 cannot be computed without having access to $p_{\boldsymbol{\mathcal{M}}}$ and $p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}$. Instead, αMEPOL computes the policy update via a Monte Carlo estimation of $\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$ from the sampled interactions $\{(\mathcal{M}_i, \tau_i)\}_{i=1}^N$ with the class of environments $\boldsymbol{\mathcal{M}}$. The policy gradient estimate itself relies on a Monte Carlo estimate of each entropy value H_{τ_i} from τ_i, and a Monte Carlo estimate of $\mathrm{VaR}_\alpha(H_\tau)$ given the estimated $\{H_{\tau_i}\}_{i=1}^N$. The following paragraphs describe how these estimates are carried out, while Algorithm 1 provides the pseudocode of αMEPOL. Additional details and implementation choices can be found in Appendix D.

Entropy Estimation We would like to compute the entropy H_{τ_i} of the state visitation frequencies d_{τ_i} from a single realization $\{s_{t,\tau_i}\}_{t=0}^{T-1} \subset \tau_i$. This estimation is notoriously challenging when the state space is continuous and high-dimensional $\mathcal{S} \subseteq \mathbb{R}^p$. Taking inspiration from recent works pursuing the MSVE objective (Mutti, Pratissoli, and Restelli 2021; Liu and Abbeel 2021b; Seo et al. 2021), we employ a principled k-Nearest Neighbors (k-NN) entropy estimator (Singh et al. 2003) of the form

$$\widehat{H}_{\tau_i} \propto -\frac{1}{T}\sum_{t=0}^{T-1} \log \frac{k\,\Gamma(\frac{p}{2}+1)}{T\left\|s_{t,\tau_i} - s^{k\text{-NN}}_{t,\tau_i}\right\|^p \pi^{\frac{p}{2}}}, \tag{3}$$

where Γ is the Gamma function, $\|\cdot\|$ is the Euclidean distance, and $s^{k\text{-NN}}_{t,\tau_i} \in \tau_i$ is the k-nearest neighbor of s_{t,τ_i}. The intuition behind the estimator in (3) is simple: We can suppose

Algorithm 1: αMEPOL

Input: percentile α, learning rate β
Output: policy $\pi_{\boldsymbol{\theta}}$

initialize $\boldsymbol{\theta}$
for epoch $= 0, 1, \ldots,$ until convergence **do**
  for $i = 1, 2, \ldots, N$ **do**
    sample an environment $\mathcal{M}_i \sim p_{\boldsymbol{\mathcal{M}}}$
    sample a trajectory $\tau_i \sim p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}_i}$
    estimate H_{τ_i} with (3)
  end for
  estimate $\mathrm{VaR}_\alpha(H_\tau)$ with (4)
  estimate $\nabla_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ with (5)
  update parameters $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \beta \widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$
end for

the state visitation frequencies d_{τ_i} to have a high entropy as long as the average distance between any encountered state and its k-NN is large. Despite its simplicity, a Euclidean metric suffices to get reliable entropy estimates in continuous control domains (Mutti, Pratissoli, and Restelli 2021).

VaR Estimation The last missing piece to get a Monte Carlo estimate of the policy gradient $\nabla_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}$ is the value of $\mathrm{VaR}_\alpha(H_\tau)$. Being $H_{[1]}, \ldots, H_{[N]}$ the order statistics out of the estimated values $\{\widehat{H}_{\tau_i}\}_{i=1}^N$, we can naïvely estimate the VaR as

$$\widehat{\mathrm{VaR}}_\alpha(H_\tau) = H_{[\lceil \alpha N \rceil]}. \tag{4}$$

Albeit asymptotically unbiased, the VaR estimator in (4) is known to suffer from a large variance in finite sample regimes (Kolla et al. 2019), which is aggravated by the error in the upstream entropy estimates, which provide the order statistics. This variance is mostly harmless when we use the estimate to filter out entropy values beyond the $\alpha\%$, i.e., the condition $H_\tau \leq \mathrm{VaR}_\alpha(H_\tau)$ in Proposition 5.1. Instead, its impact is significant when we subtract it from the values within the $\alpha\%$, i.e., the term $H_\tau - \mathrm{VaR}_\alpha(H_\tau)$ in Proposition 5.1. To mitigate this issue, we consider a convenient baseline $b = -\mathrm{VaR}_\alpha(H_\tau)$ to be subtracted from the latter, which gives the Monte Carlo policy gradient estimator

$$\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \sum_{i=1}^N f_{\tau_i} \, \widehat{H}_{\tau_i} \, \mathbb{1}(\widehat{H}_{\tau_i} \leq \widehat{\mathrm{VaR}}_\alpha(H_\tau)), \tag{5}$$

where $f_{\tau_i} = \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_{t,\tau_i} | s_{t,\tau_i})$. Notably, the baseline b trades off a lower estimation error for a slight additional bias in the estimation (5). We found that this baseline leads to empirically good results and we provide some theoretical corroboration over its benefits in Appendix D.1.

6 Empirical Evaluation

We provide an extensive empirical evaluation of the proposed methodology over the two-phase learning process described in Figure 1, which is organized as follows:

6.1 We show the ability of our method in pre-training an exploration policy in a class of continuous gridworlds, emphasizing the importance of the percentile sensitivity;

6.2 We discuss how the choice of the percentile of interest affects the exploration strategy;

6.3 We highlight the benefit that the pre-trained strategy provides to the supervised fine-tuning on the same class;

6.4 We verify the scalability of our method with the size of the class, by considering a class of 10 continuous gridworlds;

6.5 We verify the scalability of our method with the dimensionality of the environments, by considering a class of 29D continuous control Ant domains;

6.6 We verify the scalability of our method with visual inputs, by considering a class of 147D MiniGrid domains;

6.7 We show that the pre-trained strategy outperforms a policy meta-trained with MAML (Finn, Abbeel, and Levine 2017; Gupta et al. 2018a) on the same class.

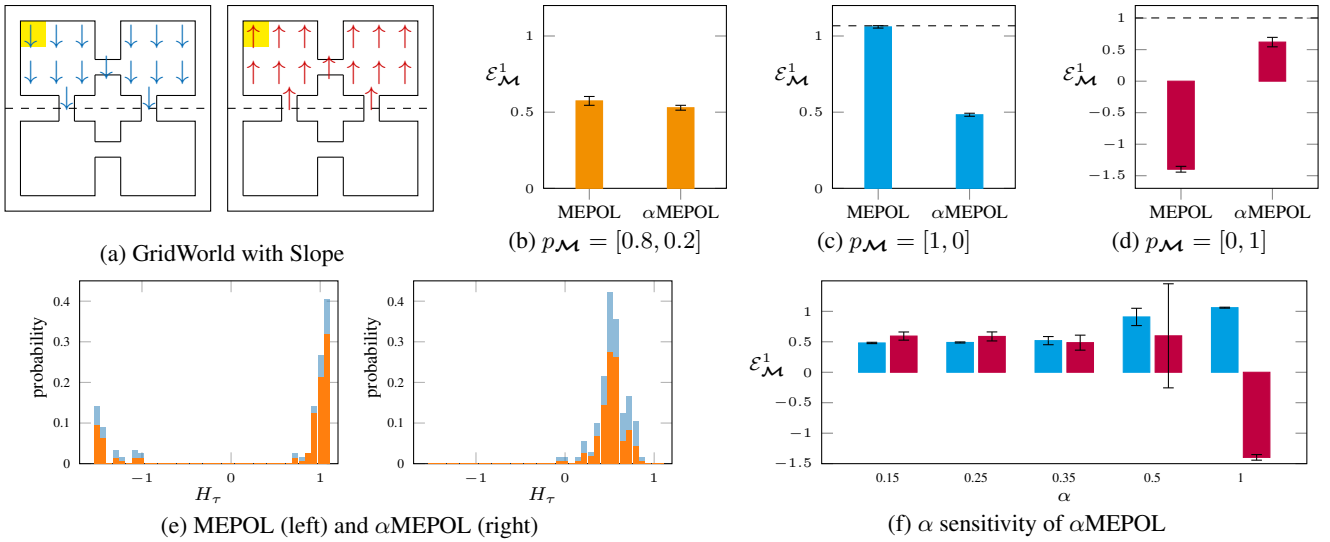
A thorough description of the experimental setting is provided in Appendix E.

6.1 Unsupervised Pre-Training with Percentile Sensitivity

We consider a class $\boldsymbol{\mathcal{M}}$ composed of two different configurations of a continuous gridworld domain with 2D states and 2D actions, which we call the *GridWorld with Slope*. In each configuration, the agent navigates through four rooms connected by narrow hallways, by choosing a (bounded) increment along the coordinate directions. A visual representation of the setting can be found in Figure 2a, where the shaded areas denote the initial state distribution and the arrows render a slope that favors or contrasts the agent's movement. The configuration on the left has a south-facing slope, and thus it is called GridWorld with South slope (GWS). Instead, the one on the right is called GridWorld with North slope (GWN) as it has a north-facing slope. This class of environments is unbalanced (and thus interesting to our purpose) for two reasons: First, the GWN configuration is more challenging from a pure exploration standpoint, since the slope prevents the agent from easily reaching the two bottom rooms; secondly, the distribution over the class is also unbalanced, as it is $p_{\boldsymbol{\mathcal{M}}} = [Pr(\text{GWS}), Pr(\text{GWN})] = [0.8, 0.2]$. In this setting, we compare αMEPOL against MEPOL (Mutti, Pratissoli, and Restelli 2021), which is akin to αMEPOL with $\alpha = 1$,[1] to highlight the importance of percentile sensitivity w.r.t. a naïve approach to the multiple-environments scenario. The methods are evaluated in terms of the state visitation entropy $\mathcal{E}^1_{\boldsymbol{\mathcal{M}}}$ induced by the exploration strategies they learn.

In Figure 2, we compare the performance of the optimal exploration strategy obtained by running αMEPOL ($\alpha = 0.2$) and MEPOL for 150 epochs on the GridWorld with Slope class ($p_{\boldsymbol{\mathcal{M}}} = [0.8, 0.2]$). We show that the two methods achieve a very similar expected performance over the class (Figure 2b). However, this expected performance is the result of a (weighted) average of very different contributions. As anticipated, MEPOL has a strong performance in GWS ($p_{\boldsymbol{\mathcal{M}}} = [1, 0]$, Figure 2c), which is close to the configuration-specific optimum (dashed line), but it displays a bad showing in the adverse GWN ($p_{\boldsymbol{\mathcal{M}}} = [0, 1]$, Figure 2d). Conversely,

[1] The pseudocode is identical to Algorithm 1 except that all trajectories affect the gradient estimate in (5).



(a) GridWorld with Slope (b) $p_{\mathcal{M}} = [0.8, 0.2]$ (c) $p_{\mathcal{M}} = [1, 0]$ (d) $p_{\mathcal{M}} = [0, 1]$

(e) MEPOL (left) and αMEPOL (right) (f) α sensitivity of αMEPOL

Figure 2: Pre-training performance $\mathcal{E}^1_{\mathcal{M}}$ obtained by αMEPOL ($\alpha = 0.2$) and MEPOL in the *GridWorld with Slope* domain **(a)**. The polices are trained on **(b)** and tested on **(b, c, d)**. The dashed lines in **(c, d)** represent the optimal performance. The empirical distribution having mean in **(b)** is reported in **(e)**. The behaviour of αMEPOL with different α is reported in **(f)**. For every plot, we provide 95% c.i. over 10 runs.

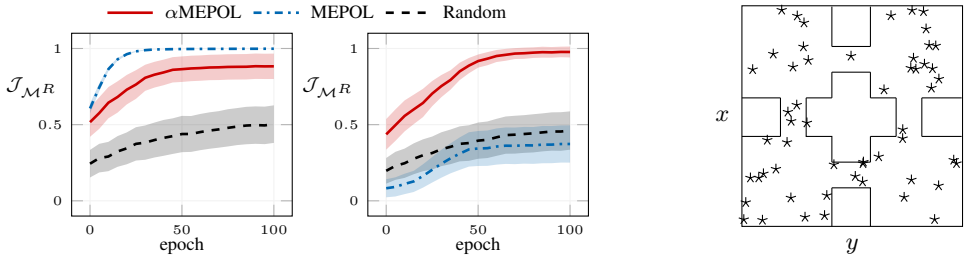
αMEPOL learns a strategy that is much more robust to the configuration, showing a similar performance in GWS and GWN, as the percentile sensitivity prioritizes the worst case during training. To confirm this conclusion, we look at the actual distribution that is generating the expected performance in Figure 2b. In Figure 2e, we provide the empirical distribution of the trajectory-wise performance (H_τ), considering a batch of 200 trajectories with $p_{\mathcal{M}} = [0.8, 0.2]$. It clearly shows that MEPOL is heavy-tailed towards lower outcomes, whereas αMEPOL concentrates around the mean. *This suggests that with a conservative choice of α we can induce a good exploration outcome for every trajectory (and any configuration), while without percentile sensitivity we cannot hedge against the risk of particularly bad outcomes.* However, let us point out that not all classes of environments would expose such an issue for a naïve, risk-neutral approach (see Appendix E.4 for a counterexample), but it is fair to assume that this would arguably generalize to any setting where there is an imbalance (either in the hardness of the configurations, or in their sampling probability) in the class. These are the settings we care about, as they require nuanced solutions (e.g., αMEPOL) for scenarios with multiple environments.

6.2 On the Value of the Percentile

In this section, we consider repeatedly training αMEPOL with different values of α in the GridWorld with Slope domain, and we compare the resulting exploration performance $\mathcal{E}^1_{\mathcal{M}}$ as before. In Figure 2f, we can see that the lower α we choose, the more we prioritize GWN (right bar for every α) at the expense of GWS (left bar). Note that this trend carries on with increasing α, ending in the values of Figures 2c, 2d. The reason for this behavior is quite straightforward, the smaller is α, the larger is the share of trajectories from the adverse configuration (GWN) ending up in the percentile at first, and thus the more GWN affects the policy update (see the gradient in (5)). *Note that the value of the percentile α should not be intended as a hyper-parameter to tune via trial and error, but rather as a parameter to select the desired risk profile of the algorithm.* Indeed, there is not a way to say which of the outcomes in Figure 2f is preferable, as they are all reasonable trade-offs between the average and worst-case performance, which might be suited for specific applications. For the sake of consistency, in every experiment of our analysis we report results with a value of α that matches the sampling probability of the worst-case configuration, but similar arguments could be made for different choices of α.

6.3 Supervised Fine-Tuning

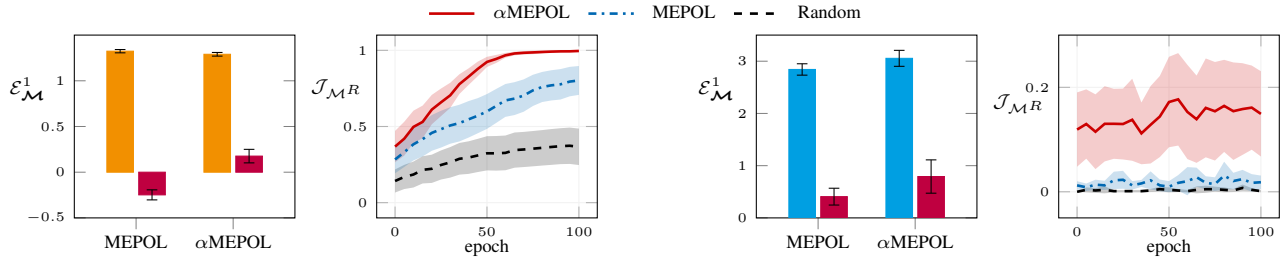
To assess the benefit of the pre-trained strategy, we design a family of MDPs $\mathcal{M}^R$, where $\mathcal{M} \in \{\text{GWS}, \text{GWN}\}$, and R is any sparse reward function that gives 1 when the agent reaches the area nearby a random goal location and 0 otherwise. On this family, we compare the performance achieved by TRPO (Schulman et al. 2015) with different initializations: The exploration strategies learned (as in Section 6.1) by αMEPOL ($\alpha = 0.2$) and MEPOL, or a randomly initialized policy (Random). These three variations are evaluated in terms of their average return $\mathcal{J}_{\mathcal{M}^R}$, which is defined in (1), over 50 randomly generated goal locations (Figure 3b). As expected, the performance of TRPO with MEPOL is competitive in the GWS configuration (Figure 3), but it falls sharply in the GWN configuration, where it is not significantly better than TRPO with Random. Instead, the performance of TRPO with αMEPOL is strong on both GWS and GWN. Despite the simplicity of the domain, solving an RL problem in GWN with an adverse goal location is far-fetched for both a random initialization and a naïve solution to the problem of unsupervised RL in multiple environments.



(a) Fine-tuning on GWS (left) and GWN (right)

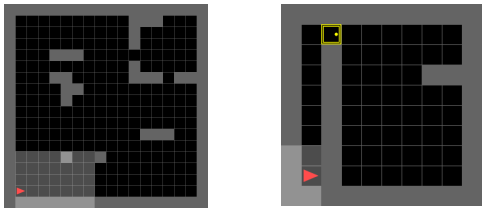
(b) Sampled goals

Figure 3: Fine-tuning performance $\mathcal{J}_{\mathcal{M}^R}$ as a function of learning epochs achieved by TRPO initialized with αMEPOL ($\alpha = 0.2$), MEPOL, and random exploration strategies, when dealing with a set of RL tasks specified on the *GridWorld with Slope* domain **(a)**. We provide 95% c.i. over 50 randomly sampled goal locations **(b)**.

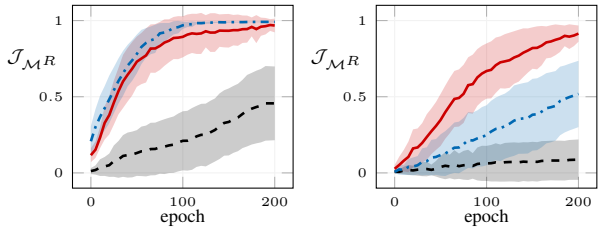


(a) MultiGrid: Pre-training (left) and fine-tuning (right)

(b) Ant: Pre-training (left) and fine-tuning (right)



(c) MiniGrid: EasyG (left) and AdvG (right)



(d) MiniGrid: fine-tuning on EasyG (left) and AdvG (right)

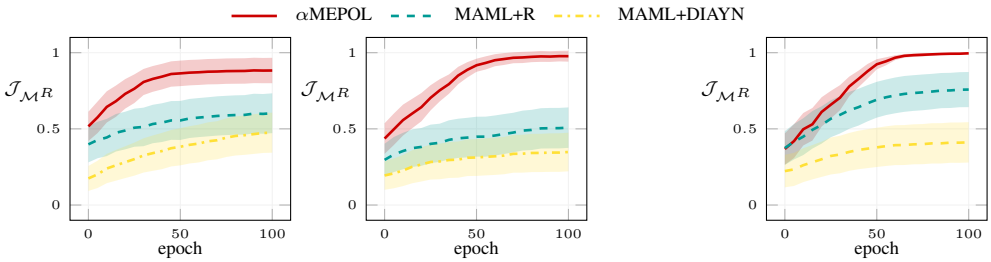
Figure 4: Pre-training performance $\mathcal{E}^1_{\mathcal{M}}$ (95% c.i. over 10 runs) achieved by αMEPOL ($\alpha = 0.1$ **(a)**, $\alpha = 0.2$ **(b)**) and MEPOL in the in the *MultiGrid* **(a)** and *Ant* **(b)** domains. Fine-tuning performance $\mathcal{J}_{\mathcal{M}^R}$ (95% c.i. over 50 tasks **(a)**, 8 tasks **(b)**, 13 tasks **(d)**) obtained by TRPO with corresponding initialization (αMEPOL, MEPOL, Random), in the *MultiGrid* **(a)**, *Ant* **(b)**, and *MiniGrid* **(d)** domains. *MiniGrid* domains are illustrated in **(c)**.

6.4 Scaling to Larger Classes of Environments

In this section, we consider a class $\mathcal{M}$ composed of 10 different configurations of the continuous gridworlds presented in Section 6.1 (including the GWN as the worst-case configuration) which we call the *MultiGrid* domain. As before, we compare αMEPOL ($\alpha = 0.1$) and MEPOL on the exploration performance $\mathcal{E}^1_{\mathcal{M}}$ achieved by the optimal strategy, in this case considering a uniformly distributed $p_{\mathcal{M}}$. While the average performance of MEPOL is slightly higher across the class (Figure 4a left, left bar), αMEPOL still has a decisive advantage in the worst-case configuration (Figure 4a left, right bar). Just as in Section 6.3, this advantage transfer to the fine-tuning, where we compare the average return $\mathcal{J}_{\mathcal{M}^R}$ achieved by TRPO with αMEPOL, MEPOL, and Random initializations over 50 random goal locations in the GWN configuration (Figure 4a right). *Whereas in the following sections we will only consider classes of two environments, this experiment shows that the arguments made for small classes of environments can easily generalize to larger classes.*

6.5 Scaling to Increasing Dimensions

In this section, we consider a class $\mathcal{M}$ consisting of two Ant environments, with 29D states and 8D actions. In the first, sampled with probability $p_{\mathcal{M}_1} = 0.8$, the Ant faces a wide descending staircase (*Ant Stairs Down*). In the second, the Ant faces a narrow ascending staircase (*Ant Stairs Up*, sampled with probability $p_{\mathcal{M}_2} = 0.2$), which is significantly harder to explore than the former. In the mold of the gridworlds in Section 6.1, these two configurations are specifically designed to create an imbalance in the class. As in Section 6.1, we compare αMEPOL ($\alpha = 0.2$) against MEPOL on the exploration performance $\mathcal{E}^1_{\mathcal{M}}$ achieved after 500 epochs. αMEPOL fares slightly better than MEPOL both in the worst-case configuration (Figure 4b left, right bar) and,



(a) GridWorld with Slope: GWS (left) and GWN (right) (b) MultiGrid

Figure 5: Fine-tuning performance $\mathcal{J}_{\mathcal{M}^R}$ achieved by TRPO initialized with αMEPOL ($\alpha = 0.2$ **(a)**, $\alpha = 0.1$ **(b)**), a MAML+R meta-policy, and a MAML+DIAYN meta-policy, when dealing with a set of RL tasks in the *GridWorld with Slope* **(a)** and the *MultiGrid* **(b)** domains. We provide 95% c.i. over 50 tasks.

surprisingly, in the easier one (Figure 4b left, left bar).[2] Then, we design a set of incrementally challenging fine-tuning tasks in the *Ant Stairs Up*, which give reward 1 upon reaching a certain step of the staircase. Also in this setting, TRPO with αMEPOL initialization outperforms TRPO with MEPOL and Random in terms of the average return $\mathcal{J}_{\mathcal{M}^R}$ (Figure 4b right). Note that these sparse-reward continuous control tasks are particularly arduous: TRPO with MEPOL and Random barely learns anything, while even TRPO with αMEPOL does not handily reach the optimal average return (1).

6.6 Scaling to Visual Inputs

In this section, we consider a class $\boldsymbol{\mathcal{M}}$ of two partially-observable MiniGrid (Chevalier-Boisvert, Willems, and Pal 2018) environments, in which the observation is a 147D image of the agent's field of view. In Figure 4c, we provide a visualization of the domain: The easier configuration (EasyG, left) is sampled with probability $p_{\mathcal{M}_1} = 0.8$, the adverse configuration (AdvG, right) is sampled with probability $p_{\mathcal{M}_2} = 0.2$. Two factors make the AdvG more challenging to explore, which are the presence of a door at the top-left of the grid, and reversing the effect of agent's movements (e.g., the agent goes backward when it tries to go forward). Whereas in all the previous experiments we estimated the entropy on the raw input features, visual inputs require a wiser choice of a metric. As proposed in (Seo et al. 2021), we process the observations through a random encoder before computing the entropy estimate in (3), while keeping everything else as in Algorithm 1. We run this slightly modified version of αMEPOL ($\alpha = 0.2$) and MEPOL for 300 epochs. Then, we compare TRPO with the learned initializations (as well as Random) on sparse-reward fine-tuning tasks defined upon the class. As in previous settings, TRPO with αMEPOL results slightly worse than TRPO with MEPOL in the easier configuration (Figure 4d, left), but significantly better in the worst-case (Figure 4d, right). Notably, TRPO from scratch struggles to learn the tasks, especially in the AdvG (Figure 4d, right). *Although the MiniGrid domain is extremely simple from a vision standpoint, we note that the same architecture can be employed in more challenging scenarios (Seo et al. 2021), while the focus of this experiment is the combination between visual inputs and multiple environments.*

[2]Note that this would not happen in general, as we expect αMEPOL to be better in the worst-case but worse on average. In this setting, the percentile sensitivity positively biases the average performance due to the peculiar structure of the environments.

6.7 Comparison with Meta-RL

In this section, we compare our approach against meta-training a policy with MAML (Finn, Abbeel, and Levine 2017) on the same *GridWorld with Slope* ($p_{\boldsymbol{\mathcal{M}}} = [0.8, 0.2]$) and *MultiGrid* (uniformly distributed $p_{\boldsymbol{\mathcal{M}}}$) domains that we have previously presented. Especially, we consider two relevant baselines. The first is MAML+R, to which we provide full access to the tasks (i.e., rewards) during meta-training. Note that this gives MAML+R an edge over αMEPOL, which operates reward-free training. The second is MAML+DIAYN (Gupta et al. 2018a), which operates unsupervised meta-training through an intrinsic reward function learned with DIAYN (Eysenbach et al. 2018). As in previous sections, we consider the average return $\mathcal{J}_{\mathcal{M}^R}$ achieved by TRPO initialized with the exploration strategy learned by αMEPOL or the meta-policy learned by MAML+R and MAML+DIAYN. TRPO with αMEPOL fares clearly better than TRPO with the meta-policies in all the configurations (Figures 5a, 5b). Even if it works fine in fast adaptation (see Appendix E.5), *MAML struggles to encode the diversity of task distribution into a single meta-policy and to deal with the most adverse tasks in the long run.* Moreover, DIAYN does not specifically handle multiple environments, and it fails to cope with the larger *MultiGrid* class.

7 Conclusions

In this paper, we addressed the problem of unsupervised RL in a class of multiple environments. First, we formulated the problem within a tractable objective, which is inspired by MSVE but includes an additional percentile sensitivity. Then, we presented a policy gradient algorithm, αMEPOL, to optimize this objective. Finally, we provided an extensive experimental analysis to show its ability in the unsupervised pre-training and the benefits it brings to the subsequent supervised fine-tuning. We believe that this paper motivates the importance of designing specific solutions to the relevant problem of unsupervised RL in multiple environments.

References

Achiam, J.; Edwards, H.; Amodei, D.; and Abbeel, P. 2018. Variational option discovery algorithms. *arXiv preprint arXiv:1807.10299*.

Ajgl, J.; and Šimandl, M. 2011. Differential entropy estimation by particles. *IFAC Proceedings Volumes*.

Bellemare, M.; Srinivasan, S.; Ostrovski, G.; Schaul, T.; Saxton, D.; and Munos, R. 2016. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*.

Burda, Y.; Edwards, H.; Storkey, A.; and Klimov, O. 2019. Exploration by random network distillation. In *International Conference on Learning Representations*.

Campos, V.; Sprechmann, P.; Hansen, S.; Barreto, A.; Kapturowski, S.; Vitvitskyi, A.; Badia, A. P.; and Blundell, C. 2021. Beyond fine-tuning: Transferring behavior in reinforcement learning. *arXiv preprint arXiv:2102.13515*.

Chevalier-Boisvert, M.; Willems, L.; and Pal, S. 2018. Minimalistic gridworld environment for openai gym. *GitHub repository*.

Chow, Y.; and Ghavamzadeh, M. 2014. Algorithms for CVaR optimization in MDPs. In *Advances in Neural Information Processing Systems*.

Csiszár, I.; and Talata, Z. 2006. Context tree estimation for not necessarily finite memory processes, via BIC and MDL. *IEEE Transactions on Information Theory*.

Deisenroth, M.; Neumann, G.; and Peters, J. 2013. A survey on policy search for robotics. *Foundations and Trends in Robotics*.

Duan, Y.; Chen, X.; Houthooft, R.; Schulman, J.; and Abbeel, P. 2016. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the International Conference on Machine Learning*.

Eysenbach, B.; Gupta, A.; Ibarz, J.; and Levine, S. 2018. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*.

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

Gregor, K.; Rezende, D. J.; and Wierstra, D. 2016. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*.

Guo, Z. D.; Azar, M. G.; Saade, A.; Thakoor, S.; Piot, B.; Pires, B. A.; Valko, M.; Mesnard, T.; Lattimore, T.; and Munos, R. 2021. Geometric entropic exploration. *arXiv preprint arXiv:2101.02055*.

Gupta, A.; Eysenbach, B.; Finn, C.; and Levine, S. 2018a. Unsupervised meta-learning for reinforcement learning. *arXiv preprint arXiv:1806.04640*.

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

Hallak, A.; Di Castro, D.; and Mannor, S. 2015. Contextual Markov decision processes. *arXiv preprint arXiv:1502.02259*.

Hazan, E.; Kakade, S.; Singh, K.; and Van Soest, A. 2019. Provably efficient maximum entropy exploration. In *Proceedings of the International Conference on Machine Learning*.

Jin, C.; Krishnamurthy, A.; Simchowitz, M.; and Yu, T. 2020. Reward-free exploration for reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.

Kolla, R. K.; Prashanth, L.; Bhat, S. P.; and Jagannathan, K. 2019. Concentration bounds for empirical conditional value-at-risk: The unbounded case. *Operations Research Letters*.

Kwon, J.; Efroni, Y.; Caramanis, C.; and Mannor, S. 2021. RL for latent MDPs: Regret guarantees and a lower bound. In *Advances in Neural Information Processing Systems*.

L.A., P.; Jagannathan, K.; and Kolla, R. 2020. Concentration bounds for CVaR estimation: The cases of light-tailed and heavy-tailed distributions. In *Proceedings of the International Conference on Machine Learning*.

Laskin, M.; Yarats, D.; Liu, H.; Lee, K.; Zhan, A.; Lu, K.; Cang, C.; Pinto, L.; and Abbeel, P. 2021. URLB: Unsupervised reinforcement learning benchmark. In *Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Lee, L.; Eysenbach, B.; Parisotto, E.; Xing, E.; Levine, S.; and Salakhutdinov, R. 2019. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*.

Liu, H.; and Abbeel, P. 2021a. Aps: Active pretraining with successor features. In *Proceedings of the International Conference on Machine Learning*.

Liu, H.; and Abbeel, P. 2021b. Behavior from the void: Unsupervised active pre-training. In *Advances in Neural Information Porcessing Systems*.

Metelli, A. M.; Mutti, M.; and Restelli, M. 2018. Configurable Markov decision processes. In *Proceedings of the International Conference on Machine Learning*.

Metelli, A. M.; Papini, M.; Faccio, F.; and Restelli, M. 2018. Policy optimization via importance sampling. In *Advances in Neural Information Processing Systems*.

Mutti, M.; Pratissoli, L.; and Restelli, M. 2021. Task-agnostic exploration via policy gradient of a non-parametric state entropy estimate. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Mutti, M.; and Restelli, M. 2020. An intrinsically-motivated approach for learning highly exploring and fast mixing policies. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Ostrovski, G.; Bellemare, M. G.; Oord, A.; and Munos, R. 2017. Count-based exploration with neural density models. In *International Conference on Machine Learning*.

Parisi, S.; Dean, V.; Pathak, D.; and Gupta, A. 2021. Interesting object, curious agent: Learning task-agnostic exploration. In *Advances in Neural Information Processing Systems*.

Parisi, S.; Pirotta, M.; and Restelli, M. 2016. Multi-objective reinforcement learning through continuous pareto manifold approximation. *Journal of Artificial Intelligence Research*.

Pathak, D.; Agrawal, P.; Efros, A. A.; and Darrell, T. 2017. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the International Conference on Machine Learning*.

Pathak, D.; Gandhi, D.; and Gupta, A. 2019. Self-supervised exploration via disagreement. In *Proceedings of the International Conference on Machine Learning*.

Pirotta, M.; Restelli, M.; and Bascetta, L. 2015. Policy gradient in Lipschitz Markov decision processes. *Machine Learning*.

Puterman, M. L. 2014. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons.

Rajendran, J.; Lewis, R.; Veeriah, V.; Lee, H.; and Singh, S. 2020. How should an agent practice? In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Rajeswaran, A.; Ghotra, S.; Ravindran, B.; and Levine, S. 2016. EPOpt: Learning robust neural network policies using model ensembles. In *Proceedings of the International Conference on Learning Representations*.

Rockafellar, R. T.; Uryasev, S.; et al. 2000. Optimization of conditional value-at-risk. *Journal of Risk*.

Satia, J. K.; and Lave Jr, R. E. 1973. Markovian decision processes with uncertain transition probabilities. *Operations Research*.

Schmidhuber, J. 1991. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proceedings of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning*.

Seo, Y.; Chen, L.; Shin, J.; Lee, H.; Abbeel, P.; and Lee, K. 2021. State entropy maximization with random encoders for efficient exploration. In *Proceedings of the International Conference on Machine Learning*.

Shannon, C. E. 1948. A mathematical theory of communication. *The Bell System Technical Journal*.

Singh, H.; Misra, N.; Hnizdo, V.; Fedorowicz, A.; and Demchuk, E. 2003. Nearest neighbor estimates of entropy. *American Journal of Mathematical and Management Sciences*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Tamar, A.; Glassner, Y.; and Mannor, S. 2015. Optimizing the CVaR via sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Tang, H.; Houthooft, R.; Foote, D.; Stooke, A.; Chen, X.; Duan, Y.; Schulman, J.; De Turck, F.; and Abbeel, P. 2017. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in Neural Information Processing Systems*.

Tarbouriech, J.; and Lazaric, A. 2019. Active exploration in Markov decision processes. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*.

Villani, C. 2008. *Optimal transport: old and new*. Springer Science & Business Media.

Xu, T.; Liu, Q.; Zhao, L.; and Peng, J. 2018. Learning to explore via meta-policy gradient. In *Proceedings of the International Conference on Machine Learning*.

Yarats, D.; Fergus, R.; Lazaric, A.; and Pinto, L. 2021. Reinforcement learning with prototypical representations. In *Proceedings of the International Conference on Machine Learning*.

Zhang, C.; Cai, Y.; and Li, L. H. J. 2021. Exploration by maximizing Rényi entropy for reward-free RL framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Zhang, X.; Ma, Y.; and Singla, A. 2020. Task-agnostic exploration in reinforcement learning. In *Advances in Neural Information Processing Systems*.

Zintgraf, L.; Shiarlis, K.; Igl, M.; Schulze, S.; Gal, Y.; Hofmann, K.; and Whiteson, S. 2019. VariBAD: A very good method for bayes-adaptive deep RL via meta-learning. In *Proceedings of the International Conference on Learning Representations*.