Off-Policy Maximum Entropy RL with Future State and Action Visitation Measures

Adrien Bolland
Montefiore Institute
University of Liège
Liège, Belgium
adrien.bolland@uliege.be

Gaspard Lambrechts
Montefiore Institute
University of Liège
Liège, Belgium
gaspard.lambrechts@uliege.be

Damien Ernst
Montefiore Institute
University of Liège
Liège, Belgium
dernst@uliege.be

Abstract

Maximum entropy reinforcement learning integrates exploration into policy learning by providing additional intrinsic rewards proportional to the entropy of some distribution. In this paper, we propose a novel approach in which the intrinsic reward function is the relative entropy of the discounted distribution of states and actions (or features derived from these states and actions) visited during future time steps. This approach is motivated by two results. First, a policy maximizing the expected discounted sum of intrinsic rewards also maximizes a lower bound on the state-action value function of the decision process. Second, the distribution used in the intrinsic reward definition is the fixed point of a contraction operator. Existing algorithms can therefore be adapted to learn this fixed point off-policy and to compute the intrinsic rewards. We finally introduce an algorithm maximizing our new objective, and we show that resulting policies have good state-action space coverage and achieve high-performance control.

1 Introduction

Many challenging tasks where an agent makes sequential decisions have been solved with reinforcement learning (RL). Examples range from playing games (Mnih et al., 2015; Silver et al., 2017), or controlling robots (Kalashnikov et al., 2018; Haarnoja et al., 2018a), to managing energy systems and markets (Boukas et al., 2021; Aittahar et al., 2024). In practice, many RL algorithms are applied in combination with an exploration strategy to achieve high-performance control. Assuming the agent takes actions in a Markov decision process (MDP), these exploration strategies usually consist of providing intrinsic reward bonuses to the agent for achieving certain behaviors. Typically, the bonus enforces taking actions that reduce the uncertainty about the environment (Pathak et al., 2017; Burda et al., 2018; Zhang et al., 2021b), or actions that enhance the variety of states and actions in trajectories (Bellemare et al., 2016; Lee et al., 2019; Guo et al., 2021; Williams and Peng, 1991; Haarnoja et al., 2019). In many of the latter methods, the intrinsic reward function is the entropy of some distribution over the state-action space. Optimizing jointly the reward function of the MDP and the intrinsic reward function, in order to eventually obtain a high-performing policy, is called Maximum Entropy RL (MaxEntRL) and was shown to be effective in many problems.

The reward of the MDP was already extended with the entropy of the policy in early algorithms (Williams and Peng, 1991) and was only later called MaxEntRL (Ziebart et al., 2008; Toussaint, 2009). This particular reward regularization provides substantial improvements in the robustness of the resulting policy (Ziebart, 2010; Husain et al., 2021; Brekelmans et al., 2022) and provides a learning objective function with good smoothness and concavity properties (Ahmed et al., 2019; Bolland et al., 2023). Several commonly used algorithms can be named, like soft Q-learning (Haarnoja et al., 2017; Schulman et al., 2017a) and soft actor-critic (Haarnoja et al., 2018b, 2019). This MaxEntRL objective

18th European Workshop on Reinforcement Learning (EWRL 2025).

nevertheless only rewards the randomness of actions and neglects the influence of the policy on the visited states, which, in practice, may lead to inefficient exploration.

In order to enhance exploration, Hazan et al. (2019) were the first to propose to intrinsically motivate agents to have a uniform discounted visitation measure over states. Several works have afterward been developed to maximize the entropy of the discounted state visitation measure and the stationary state visitation measure. For discrete state and action spaces, optimal exploration policies, which maximize the entropy of these visitation measures, can be computed to near optimality with off-policy tabular model-based RL algorithms (Hazan et al., 2019; Mutti and Restelli, 2020; Tiapkin et al., 2023). For continuous state and action spaces, alternative methods rely on k nearest neighbors to estimate the density of the visitation measure of states (or features built from the states) and compute the intrinsic rewards, which can afterward be optimized with any RL algorithm (Liu and Abbeel, 2021; Yarats et al., 2021; Seo et al., 2021; Mutti et al., 2021). These methods require sampling new trajectories at each iteration; they are on-policy, and estimating the intrinsic reward function is computationally expensive. Some other methods rely on parametric density estimators to reduce the computational complexity and share information across learning steps (Lee et al., 2019; Guo et al., 2021; Islam et al., 2019; Zhang et al., 2021a). The additional function approximator is typically learned on-policy by maximum likelihood estimation based on batches of truncated trajectories. Alternative methods have adapted this MaxEntRL objective to maximize entropy of states visited in single trajectories (Mutti et al., 2022; Jain et al., 2024). When large and/or continuous state and action spaces are involved, relying on parametric function approximators is likely the best choice. Nevertheless, existing algorithms are on-policy. They require sampling new trajectories from the environment at (nearly) every update of the policy, and cannot be applied using a buffer of arbitrary transitions, in batch-mode RL, or in continuing tasks. Furthermore, learning the discounted visitation measure is more desirable than learning the stationary one, but may be challenging in practice due to the exponentially decreasing influence of the time step at which states are visited (Islam et al., 2019).

The main contribution of this paper is to introduce a MaxEntRL objective relying on a new intrinsic reward function for exploring effectively the state and action spaces, which also alleviates the previous limitations. This intrinsic reward function is the relative entropy of the discounted distribution of states and actions (or features from these states and actions) visited during the next time steps. We prove two results motivating the MaxEntRL objective. First, a policy maximizing the expected discounted sum of intrinsic rewards also maximizes a lower bound on the state-action value function of the decision process. Second, the visitation distribution used in the new intrinsic reward function is the fixed point of a contraction operator. Existing RL algorithms can integrate an additional learning step to approximate this fixed point off-policy, using N-step state-action transitions and bootstrapping the operator. It is then possible to approximate the intrinsic reward function and learn a policy maximizing the extended rewards with the adapted algorithm. We illustrate this methodology on off-policy actor-critic (Degris et al., 2012). The resulting MaxEntRL algorithm is off-policy; it efficiently computes exploration policies with good discounted visitation probability coverage and high-performing control policies.

The visitation measure of future states and actions, which we use to extend the reward function in this article, has a well-established history in the development of RL algorithms. It was popularized by Janner et al. (2020), who learned the distribution of future states as a generalization of the successor features (Barreto et al., 2017). They demonstrated that this distribution allows expressing the state-action value function by separating the influence of the dynamics and the reward function, and that it could be learned off-policy by exploiting its recursive expression. Several algorithms have been proposed to learn this distribution, either by maximum likelihood estimation (Janner et al., 2020), by contrastive learning (Mazoure et al., 2023b), or using diffusion models (Mazoure et al., 2023c). These distributions of future states and actions have found applications in goal-based RL (Eysenbach et al., 2020, 2022), in offline pre-training with expert examples (Mazoure et al., 2023a), in model-based RL (Ma et al., 2023), or in planning (Eysenbach et al., 2023). We are the first to integrate them into the MaxEntRL framework for enhancing exploration through learning.

The manuscript is organized as follows. In Section 2, the RL problem is reminded, and the MaxEntRL framework is formulated. In Section 3, we introduce and discuss a new MaxEntRL objective. Section 4 details how to learn a model of the conditional state visitation probability that allows estimating this new objective. We finally present experimental results in Section 5 and conclude in Section 6.

2 Background and Preliminaries

2.1 Markov Decision Processes

This paper focuses on problems in which an agent makes sequential decisions in a stochastic environment (Sutton and Barto, 2018). The environment is modeled with an infinite-time Markov decision process (MDP) composed of a state space $\mathcal{S}$, an action space $\mathcal{A}$, an initial state distribution p_0, a transition distribution p, a bounded reward function R, and a discount factor $\gamma \in [0, 1)$. Agents interact in this MDP by providing actions sampled from a policy π. During this interaction, an initial state $s_0 \sim p_0(\cdot)$ is first sampled, then, the agent provides at each time step t an action $a_t \sim \pi(\cdot|s_t)$ leading to a new state $s_{t+1} \sim p(\cdot|s_t, a_t)$. In addition, after each action a_t is executed, a reward $r_t = R(s_t, a_t) \in \mathbb{R}$ is observed. We denote the expected return of the policy π by

$$J(\pi) = \mathop{\mathbb{E}}_{\substack{s_0 \sim p_0(\cdot) \\ a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim p(\cdot|s_t, a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right] . \tag{1}$$

An optimal policy π^* is one with maximum expected return

$$\pi^* \in \arg\max_{\pi} J(\pi) . \tag{2}$$

2.2 Maximum Entropy Reinforcement Learning

In maximum entropy reinforcement learning (MaxEntRL) an optimal policy π^* is approximated by maximizing a surrogate objective function $L(\pi)$, where the reward function from the MDP is extended by an intrinsic reward function. The latter is the (relative) entropy of some particular distribution. A general definition of the MaxEntRL objective function is

$$L(\pi) = \mathop{\mathbb{E}}_{\substack{s_0 \sim p_0(\cdot) \\ a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim p(\cdot|s_t, a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t \left(R(s_t, a_t) + \lambda R^{int}(s_t, a_t) \right) \right] , \tag{3}$$

where this objective depends on the intrinsic reward function R^{int}. We propose a generic formulation that, to the best of our knowledge, encompasses most existing intrinsic rewards from the literature. Given a feature space $\mathcal{Z}$, a conditional feature distribution $q^\pi : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{Z})$, depending on the policy π, and a relative measure $q^* \in \Delta(\mathcal{Z})$, the MaxEntRL intrinsic reward function is

$$R^{int}(s, a) = -KL_z \left[q^\pi(z|s, a) \| q^*(z) \right] = \mathop{\mathbb{E}}_{z \sim q^\pi(\cdot|s,a)} \left[\log q^*(z) - \log q^\pi(z|s, a) \right] . \tag{4}$$

Importantly, the intrinsic reward function is (implicitly) dependent on the policy π through the distribution q^π. We define an optimal exploration policy as a policy that maximizes the expected sum of discounted intrinsic rewards only. Note that a policy maximizing $L(\pi)$ is generally not optimal, due to the potential gap between the optimum of the return $J(\pi)$ and the optimum of the learning objective $L(\pi)$. This subject is inherent to exploration with intrinsic rewards (Bolland et al., 2024).

MaxEntRL algorithms optimize objective functions as defined in equation (3) depending on some intrinsic reward function that can be expressed as in equation (4). The particularity of each algorithm is its estimation of the intrinsic reward and of the stochastic gradient of the learning objective. Often, a pseudo reward $\log q^*(z) - \log q^\pi(z|s, a)$ is computed from a sample $z \sim q^\pi(\cdot|s, a)$ to extend the MDP reward function and used by an existing RL algorithm.

Many of the existing MaxEntRL algorithms optimize an objective that depends on the entropy of the policy for exploring the action space (Haarnoja et al., 2018b; Toussaint, 2009). The feature space is then the actions space $\mathcal{Z} = \mathcal{A}$, and the conditional feature distribution is the policy $q^\pi(z|s, a) = \pi(z|s)$, for all a. Other algorithms optimize objectives enhancing state space exploration (Hazan et al., 2019; Lee et al., 2019; Islam et al., 2019; Guo et al., 2021). The feature space is the state space $\mathcal{Z} = \mathcal{S}$. The conditional feature distribution $q^\pi(z|s, a)$ is either the marginal probability of states in trajectories of T time steps, or the discounted state visitation measure, for all s and a. In the literature, the relative measure $q^*(z)$ is usually a uniform distribution, and the relative entropy is computed as the differential entropy, i.e., by neglecting $\log q^*(z)$ in equation (4). In continuous spaces, the latter is ill-defined and other relative measures may be used.

3 MaxEntRL with Visitation Distributions

3.1 Definition of the MaxEntRL Objective

In the following, we introduce a new MaxEntRL intrinsic reward based on the conditional state-action visitation probability $d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)$ and the conditional state visitation probability $d^{\pi,\gamma}(\bar{s}|s,a)$

$$d^{\pi,\gamma}(\bar{s},\bar{a}|s,a) = (1-\gamma)\pi(\bar{a}|\bar{s}) \sum_{\Delta=1}^{\infty} \gamma^{\Delta-1} p^{\pi}_{\Delta}(\bar{s}|s,a) \tag{5}$$

$$d^{\pi,\gamma}(\bar{s}|s,a) = (1-\gamma) \sum_{\Delta=1}^{\infty} \gamma^{\Delta-1} p^{\pi}_{\Delta}(\bar{s}|s,a)\,, \tag{6}$$

where p^{π}_{Δ} is the transition probability in Δ time steps with the policy π. The distribution from equation (5) can be factorized as a function of the distribution from equation (6) such that $d^{\pi,\gamma}(\bar{s},\bar{a}|s,a) = \pi(\bar{a}|\bar{s})d^{\pi,\gamma}(\bar{s}|s,a)$. The conditional state (respectively, state-action) visitation probability distribution measures the future states (respectively, states and actions) that are visited on expectation over infinite trajectories starting from any state and action. Both distributions generalize the (marginal discounted) state visitation probability measure (Manne, 1960).

Definition 3.1. Let us consider the feature space $\mathcal{Z}$ and a feature distribution $h : \mathcal{S}\times\mathcal{A} \to \Delta(\mathcal{Z})$. The intrinsic reward is defined by equation (4), for any relative measure q^*, with conditional distribution

$$q^{\pi}(z|s,a) = \int h(z|\bar{s},\bar{a}) d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)\, d\bar{s}\, d\bar{a}\,. \tag{7}$$

Optimal exploration policies are here intrinsically motivated to take actions so that the discounted visitation measure of future features is distributed according to q^* in each state and for each action. It allows to select features that must be visited during trajectories according to prior knowledge about the problem if any. Alternatively, it allows to only explore lower dimensional feature spaces, or to explore sufficient statistics from the state-action pairs.

The MaxEntRL objective from Definition 3.1 can be optimized by any existing RL algorithm that is adapted to compute for each state s and action a the additional (pseudo) reward

$$R^{int}(s,a) = \log q^*(z) - \log q^{\pi}(z|s,a)\,, \tag{8}$$

where $z \sim q^{\pi}(\cdot|s,a)$. This reward is a single-sample Monte-Carlo estimate of equation (4), unbiased for fixed q^{π}. This computation requires sampling features z from the conditional distribution q^{π} and estimating the probability of these samples $q^{\pi}(z|s,a)$. It can be achieved by solving the integral equation (7), e.g., numerically by sampling states $\bar{s} \sim d^{\pi,\gamma}(\cdot|s,a)$, actions $\bar{a} \sim \pi(\cdot|\bar{s})$, and finally features $z \sim h(\cdot|\bar{s},\bar{a})$. This particular sampling procedure requires access to the unknown conditional state visitation probability. Section 4 provides a method for learning that distribution off-policy.

3.2 Relationship with Alternative MaxEntRL Objectives

Let us relate MaxEntRL with the new intrinsic reward function to the maximization of a lower bound on the state-action value function of the MDP (computed without intrinsic rewards). We rely on Theorem 3.2, shown in Appendix A, close to the results from Kakade and Langford (2002).

Theorem 3.2. *Let the reward function $R(s,a)$ be non-negative, let π be a policy with state-action value function $Q^{\pi}(s,a)$, and let the visitation measures be non-zero over their support, then,*

$$Q^{\pi}(s,a) \geq Q^{\pi^*}(s,a)\, \exp\left(- \left\| \log \frac{d^{\pi,\gamma}(\cdot,\cdot|s,a)}{d^{\pi^*,\gamma}(\cdot,\cdot|s,a)} \right\|_{\infty} \right)\,, \tag{9}$$

where $\|f\|_{\infty} = \sup_x |f(x)|$ is the L_{∞}-norm of f.

Let us again consider that the feature distribution h is the identity map, so that $z = (\bar{s},\bar{a})$, and apply the triangle inequality on equation (9). For any policy π, we get the bound

$$Q^{\pi}(s,a) \geq Q^{\pi^*}(s,a)\, \exp\left(- \left\| \log \frac{d^{\pi,\gamma}(\cdot,\cdot|s,a)}{q^*(\cdot,\cdot)} \right\|_{\infty} \right) \exp\left(- \left\| \log \frac{d^{\pi^*,\gamma}(\cdot,\cdot|s,a)}{q^*(\cdot,\cdot)} \right\|_{\infty} \right)\,. \tag{10}$$

The bound on the state-action value function of any policy π in equation (10) is an exponentially decreasing function of the two error terms $\| \log d^{\pi,\gamma}(\cdot,\cdot|s,a) - \log q^*(\cdot,\cdot)||_\infty$ and $\| \log d^{\pi^*,\gamma}(\cdot,\cdot|s,a) - \log q^*(\cdot,\cdot)||_\infty$. The first can be minimized as a function of π while the second is independent of the policy, and can thus not be reduced. Let us assume that an optimal exploration policy has zero expected discounted sum of intrinsic rewards, and that the target measure and the visitation measures are smooth. Then, an optimal exploration policy maximizes the bound in equation (10). Optimizing the MaxEntRL objective we introduce can be seen as a practical algorithm to compute a policy that maximizes the lower bound equation (10). The quality of the resulting policy then only depends on the choice of the distribution q^*.

4 Off-policy Learning of Conditional Visitation Models

4.1 Fixed-Point Properties of Conditional Visitation

As explained in Section 3.1, the MaxEntRL intrinsic reward function in Definition 3.1 can be computed from samples of the conditional distribution $q^\pi(z|s,a)$, which in turn can be computed based on samples of the conditional state visitation distribution $d^{\pi,\gamma}(\bar{s}|s,a)$. In this section, we establish useful properties of this visitation distribution.

Let us first recall that the conditional state visitation distribution accepts a recursive definition (Janner et al., 2020) that is a trivial fixed point of the operator $\mathcal{T}^\pi$ from Definition 4.1.

Definition 4.1. The operator $\mathcal{T}^\pi$ is defined over the space of conditional state distribution as

$$\mathcal{T}^\pi q(\bar{s}|s,a) = (1-\gamma)p(\bar{s}|s,a) + \gamma \mathop{\mathbb{E}}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(\cdot|s')}} \left[q(\bar{s}|s',a')\right] . \tag{11}$$

In Theorem 4.2, we establish that the operator $\mathcal{T}^\pi$ is a contraction mapping, which furthermore implies the uniqueness of its fixed point. Assuming the result of the operator could be computed (or estimated), the fixed point could also be computed by successive application of this operator. It would allow computing the conditional state visitation distribution and the intrinsic reward function.

Theorem 4.2. *The operator $\mathcal{T}^\pi$ is γ-contractive in $\bar{L}_n$-norm, where $\bar{L}_n(f)^n = \sup_y \int |f(x|y)|^n \, dx$.*

The theorem is shown in Appendix A.

4.2 TD Learning of Conditional Visitation Models

In practice, computing the result of the operator $\mathcal{T}^\pi$ (and $(\mathcal{T}^\pi)^N$ after N applications) may be intractable when large or continuous state-action spaces are at hand. It furthermore requires having a model of the MDP. A common alternative approach is to rely on a function approximator d_ψ to approximate the fixed point. Theorem 4.2 suggests optimizing the parameters of this model d_ψ to minimize the residual of the operator, measured with the $\bar{L}_n$-norm for which the operator is γ-contractive, similarly to TD-learning methods (Sutton and Barto, 2018). Nevertheless, measuring the residual with the $\bar{L}_n$-norm requires estimating the MDP transition function (Janner et al., 2020), and can therefore not be trivially minimized by stochastic gradient descent using sampled transitions. We therefore propose to solve as surrogate a minimum cross-entropy problem, in which stochastic gradient descent can be applied afterward. For any policy π, the distribution is approximated with a function approximator d_ψ with parameter ψ optimized to solve

$$\arg\min_\psi \mathop{\mathbb{E}}_{\substack{s,a\sim g(\cdot,\cdot)\\ \bar{s}\sim(\mathcal{T}^\pi)^N d_\psi(\cdot|s,a)}} \left[-\log d_\psi(\bar{s}|s,a)\right] , \tag{12}$$

where g is an arbitrary distribution over the state and action spaces, and where N is any positive integer. This optimization problem is related to minimizing the KL-divergence instead of an $\bar{L}_n$-norm (Bishop and Nasrabadi, 2006).

Let us make explicit how samples from the distribution $(\mathcal{T}^\pi)^N d_\psi(\bar{s}|s,a)$ can be generated from the MDP. By definition of the operator $\mathcal{T}^\pi$, the distribution $(\mathcal{T}^\pi)^N d_\psi(\bar{s}|s,a)$ is the mixture

$$(\mathcal{T}^\pi)^N d_\psi(\bar{s}|s,a) = \left(\sum_{\Delta=1}^{N}(1-\gamma)\gamma^{\Delta-1}p_\Delta^\pi(\bar{s}|s,a)\right) + \gamma^N \mathop{\mathbb{E}}_{\substack{s'\sim p_N^\pi(\cdot|s,a)\\ a'\sim\pi(\cdot|s')}}\left[d_\psi(\bar{s}|s',a')\right] \tag{13}$$

$$= \sum_{\Delta=1}^{\infty}\mathcal{G}_{1-\gamma}(\Delta)b_{\psi,\pi}^\beta(\bar{s}|s,a,\Delta)\big|_{\beta=\pi}\,, \tag{14}$$

where $\mathcal{G}_{1-\gamma}(\Delta)$ is the probability of Δ from a geometric distribution of parameter $1-\gamma$, and

$$b_{\psi,\pi}^\beta(\bar{s}|s,a,\Delta) = \begin{cases} p_\Delta^\beta(\bar{s}|s,a) & \Delta \leq N \\ \mathop{\mathbb{E}}_{\substack{s'\sim p_N^\beta(\cdot|s,a)\\ a'\sim\pi(\cdot|s')}}\left[d_\psi(\bar{s}|s',a')\right] & \Delta > N \end{cases}\,. \tag{15}$$

Sampling from $(\mathcal{T}^\pi)^N d_\psi(\bar{s}|s,a)$ consists in sampling from the mixture. First, Δ is drawn from a geometric distribution of parameter $1-\gamma$. Second, a state is sampled as $\bar{s}\sim p_\Delta^\pi(\cdot|s,a)$ if $\Delta\leq N$ or as $\bar{s}\sim d_\psi(\cdot|s',a')$ otherwise; where $s'\sim p_N^\pi(\cdot|s,a)$ and $a'\sim\pi(\cdot|s')$.

Let us reformulate the problem equation (12) to highlight the previous sampling procedure, and such that it can be estimated from transitions sampled from an arbitrary policy β in the MDP. To that end, we apply importance weighting and get the equivalent optimization problem

$$\arg\min_\psi \mathop{\mathbb{E}}_{\substack{s,a\sim g(\cdot,\cdot)\\ \Delta\sim\mathcal{G}_{1-\gamma}(\cdot)\\ \bar{s}\sim b_{\psi,\pi}^\beta(\cdot|s,a,\Delta)}}\left[-\frac{b_{\psi,\pi}^\pi(\bar{s}|s,a,\Delta)}{b_{\psi,\pi}^\beta(\bar{s}|s,a,\Delta)}\log d_\psi(\bar{s}|s,a)\right]\,. \tag{16}$$

In the particular cases where $\beta=\pi$ or where $N=1$, the importance weight simplifies to one, otherwise it can be simplified to a (finite) product of ratios of policies.

Learning d_ψ from samples can be achieved by solving problem equation (16) as an intermediate step to any RL algorithm. First, the objective function is estimated as described using transitions stored in a batch or generated with a behavior policy β. The sample $\bar{s}=s_{t+\Delta}$ is available in the batch or replay buffer if $\Delta\leq N$, or $\bar{s}\sim d_\psi(\cdot|s_{t+N},a'_{t+N})$ is bootstrapped otherwise; where $a'_{t+N}\sim\pi(\cdot|s_{t+N})$ and $\Delta\sim\mathcal{G}_{1-\gamma}(\cdot)$. Second, this estimate is differentiated, and the parameter ψ is updated by gradient descent steps. In practice, the gradients generated by differentiating this loss function are biased. The influence of the parameter ψ on the probability of the sample z is neglected when bootstrapping, i.e., the partial derivative of $(\mathcal{T}^\pi)^N d_\psi(\bar{s}|s_t,a_t)$ with respect to ψ is neglected, and a target network is used. This is analogous to SARSA and TD-learning strategies (Sutton and Barto, 2018). Furthermore, we suggest neglecting the importance weights, which introduces a dependency of the distribution d_ψ on the policy β. Finally, the model d_ψ is used to compute the intrinsic rewards and update the policy.

5 Experiments

5.1 Experimental Setting

Illustrative experiments are performed on adapted environments from the Minigrid suite (Chevalier-Boisvert et al., 2023). In the latter, an agent must travel across a grid containing walls and passages in order to reach a goal. The size of the grid and the number of passages and walls depend on the environment. The state space is composed of the agent's orientation, its position on the grid, as well as the positions of the passages in the walls and their orientations. In some environments, the goal to be reached is randomly generated and is also part of the state. The agent can take four different actions: turn left, turn right, move forward, or stand still. The need for exploration comes from the sparsity of the reward function, which is zero everywhere and equals one in the state to be reached.

As explained, the model d_ψ is learned during an intermediate step added to an arbitrary RL algorithm that evaluates and optimizes the intrinsic rewards with the MDP rewards. Experiments were performed using off-policy actor-critic (Degris et al., 2012), i.e., an approximate policy iteration algorithm,

adapted as advocated. This new algorithm is detailed in Appendix B and is called off-policy actor-critic with conditional visitation measures (OPAC+CV) in the remainder of the paper. For the Minigrid environments, the features $z \in \mathcal{Z}$ are the pairs of horizontal and vertical positions of the agent in the environment, the function h is a deterministic mapping that computes these positions based on the state-action pairs, and the relative measure q^* is uniform. The pseudo-code is provided in Appendix B, and the implementation choices and hyperparameters are detailed in Appendix C.

This new MaxEntRL algorithm is compared to two alternative algorithms. The first concurrent method is soft actor-critic (SAC) (Haarnoja et al., 2018b). It is a commonly-used MaxEntRL algorithm where the feature space is the action space $\mathcal{Z} = \mathcal{A}$, the conditional distribution is the policy $q^\pi(z|s,a) = \pi(z|s)$ for all a, and the relative measure q^* is uniform. To the best of our knowledge, the MaxEntRL objective optimized in soft actor-critic is also the only alternative where policies can be computed off-policy when the state and action space is large or continuous.

The second concurrent method intrinsically motivates agents to have uniform (marginal) discounted visitation measures as originally proposed by Hazan et al. (2019) and discussed in Section 1. To that end, we adapt the algorithm from Zhang et al. (2021a). First, to improve sample efficiency, policies are optimized using off-policy actor-critic (Degris et al., 2012) instead of PPO (Schulman et al., 2017b). Second, we use a categorical distribution rather than a variational auto-encoder to approximate the visitation measure, which is made possible as the state-action space is discrete. It allows optimizing the approximator without relying on the evidence lower bound. We refer to that adapted algorithm as off-policy actor-critic with marginal visitation measures (OPAC+MV). Here, the feature space $\mathcal{Z}$ is the same as in OPAC+CV, the conditional distribution $q^\pi(z|s,a)$ is the discounted visitation measure of features for each state s and action a, and the relative measure q^* is uniform. In practice, the feature probability and intrinsic reward are computed as for OPAC+CV; more details are available in Appendix B. Even if off-policy actor-critic is off-policy, learning the model of the visitation measure requires online buffer updates. The final algorithm is therefore on-policy.

5.2 Exploring Sparse-Reward Environments

The feature space coverage of optimal exploration policies computed with OPAC+CV, OPAC+MV, and SAC is first compared. In Figure 1, the evolution of the entropy of the discounted visitation measure of features is shown as a function of the number of algorithm iterations, when only the intrinsic rewards are considered. For each environment, the entropy increases rapidly with the OPAC+CV and OPAC+MV algorithms, and a high-entropy policy results from the optimization. In most environments, OPAC+MV achieves the highest entropy, followed closely by OPAC+CV, while

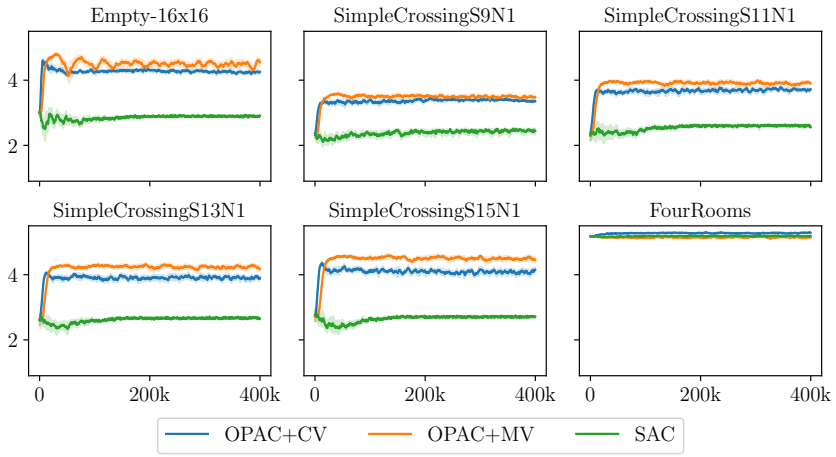

Figure 1: Evolution of the entropy of the discounted visitation probability measure of the position of the agent on the grid when computing exploration policies (i.e., when neglecting the rewards of the MDP). The entropy is computed empirically with Monte Carlo simulations. For each iteration, the interquartile mean over 15 runs is reported, along with its 95% confidence interval.

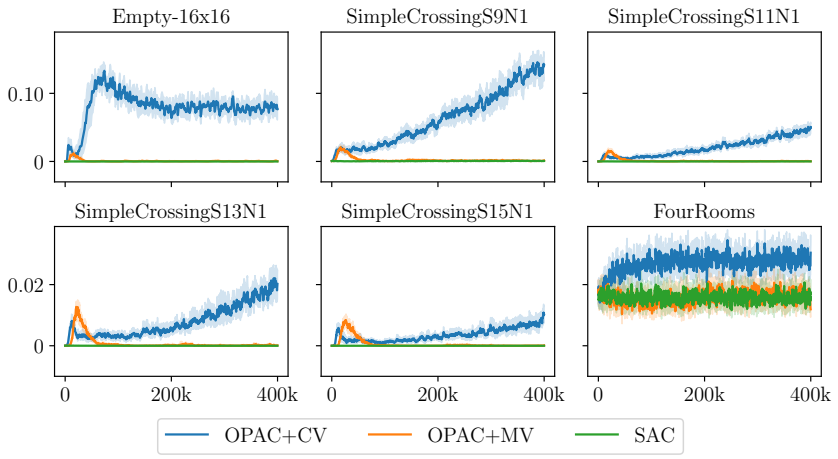

Figure 2: Expected return during the (exploration) policy optimization with OPAC+CV and OPAC+MV. The expectation is computed empirically with Monte Carlo simulations. For each iteration, the interquartile mean over 15 runs is reported, along with its 95% confidence interval.

SAC performs poorly. It is worth noting that OPAC+CV performs competitively with concurrent method despite optimizing a different objective than the reported discounted visitation measure.

In Figure 2, the evolution of the expected returns of the policies is reported during learning. As can be seen, optimizing the exploration objective presented in Section 3.1 with OPAC+CV provides optimal exploration policies with significantly higher expected return compared to OPAC+MV and SAC. Importantly, comparing Figure 1 and Figure 2, one can see that policies with small differences in the entropy of the discounted visitation measure may achieve very different expected returns.

In the literature, feature exploration is usually used to compute optimal exploration policies as an initialization when extrinsic rewards are not available. Our method is an off-policy alternative yielding policies with good feature space coverage and larger expected return.

5.3 Controlling Sparse-Reward Environments

The objective of MaxEntRL is to provide intrinsic motivations to explore in order to compute a high-performance policy. In Figure 3, the expected return of OPAC+CV is compared to that of SAC and OPAC+MV. As can be seen, our method always performs at least as well as SAC. In the `SimpleCrossing`-environments, the two methods perform equivalently for the first one, OPAC+CV performs similarly to the lucky realizations of SAC for the second one, and only OPAC+CV computes (with high probability) policies with non-zero return for the last two. These environments are open grids of different sizes where the agent shall cross a wall through a small passage to reach the target. The larger the environment, the lower the probability of reaching the goal with a uniform policy, and the worse the performance of SAC. The same can be observed in the `Empty-16x16`-environment. On the contrary, both MaxEntRL methods perform equivalently in the `FourRooms`-environment, where complex exploration is apparently not necessary to solve the problem. Finally, our method slightly outperforms OPAC+MV in all environments, except in `SimpleCrossingS15N1` where the concurrent method performs best. Two factors may influence the performance. First, the intrinsic reward functions have different scales, and the weight λ is constant. Second, the expected returns of optimal exploration policies are different; see Figure 2. Probably the most important is that both methods allow computing policies with non-zero rewards. With an appropriate scheduling on λ, both methods could eventually compute high-performing policies.

5.4 Discussion of Experiments

Several phenomena influence the learning of the visitation model. First, when γ is close to one, the learning becomes unstable in practice. We hypothesize this results from the increased importance of future states. Increasing parameter N helps mitigate the issue as there is less bootstrapping, reducing

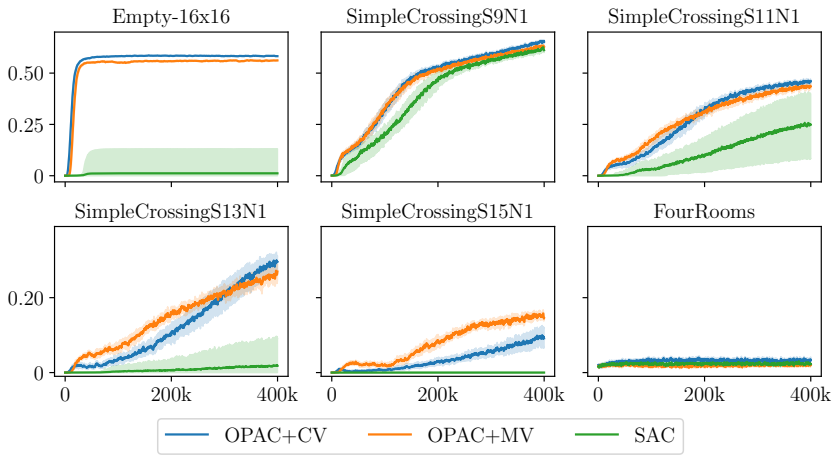

Figure 3: Expected return during the policy optimization with OPAC+CV, OPAC+MV, and SAC. The expectation is computed empirically with Monte Carlo simulations. For each iteration, the interquartile mean over 15 runs is reported, along with its 95% confidence interval.

the risk of learning a biased target. Second, we neglect the importance weights in practice to reduce variance, which makes d_ψ partially dependent on the behavior policy β. In our online setting, a relatively small buffer is refreshed sufficiently often to mitigate this dependence. Bootstrapping still propagates long-term effects of the policy, which also allows batch-mode RL with a biased model.

In the experiments, OPAC+CV reached strong marginal coverage quickly. In general, our exploration objective is not expected to improve the marginal visitation coverage compared to its direct maximization. In practice, the off-policiness and the stability noted above can still make OPAC+CV competitive. We also observed that maximizing our objective leads to higher conditional visitation entropy, given the initial state, meaning a wider set of features is explored within each independent trajectory. This could explain the higher returns observed in Figure 2.

Finally, we relied on off-policy actor-critic for concreteness, yet the MaxEntRL objective is agnostic to the control backbone, and similar results should hold with other RL methods. Our method offers a practical alternative to directly maximizing marginal visitation, without focusing on the potential theoretical advantages of different exploration objectives.

6 Conclusion

In this paper, we presented a new MaxEntRL objective providing intrinsic reward bonuses proportional to the entropy of the distribution of features built from the states and actions visited by the agent in future time steps. The reward bonus can be estimated efficiently by sampling from the conditional distribution of states visited, which we proved to be the fixed point of a contraction mapping. It can therefore be learned for any policy relying on batches of arbitrary transitions. We proposed an end-to-end off-policy algorithm maximizing our objective that allows exploring effectively the state and action spaces. The algorithm is benchmarked on several control problems. The method we developed is easy to implement and can be integrated into already existing RL algorithms.

In this paper, experiments were limited to relatively small-scale environments. Future work should focus on benchmarking the method in more challenging environments, including environments with larger or continuous state-action spaces. For the continuous case, this will require adapting the density estimator and the algorithm accordingly. Furthermore, in this paper, the feature space to explore is fixed a priori, but could be learned. A potential avenue is to explore reward-predictive feature spaces. Finally, the distribution that is learned for exploration purposes can be used to generate new samples to enhance sample efficiency when learning the critic. The integration of this approach into the MaxEntRL framework is left for future work.

References

Ahmed, Z., Le Roux, N., Norouzi, M., and Schuurmans, D. (2019). Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*, pages 151–160. PMLR.

Aittahar, S., Bolland, A., Derval, G., and Ernst, D. (2024). Optimal control of renewable energy communities subject to network peak fees with model predictive control and reinforcement learning algorithms. *arXiv preprint arXiv:2401.16321*.

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., van Hasselt, H. P., and Silver, D. (2017). Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. *Advances in Neural Information Processing Systems*, 29.

Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

Bolland, A., Lambrechts, G., and Ernst, D. (2024). Behind the myth of exploration in policy gradients. *arXiv preprint arXiv:2402.00162*.

Bolland, A., Louppe, G., and Ernst, D. (2023). Policy gradient algorithms implicitly optimize by continuation. *Transactions on Machine Learning Research*.

Boukas, I., Ernst, D., Théate, T., Bolland, A., Huynen, A., Buchwald, M., Wynants, C., and Cornélusse, B. (2021). A deep reinforcement learning framework for continuous intraday market bidding. *Machine Learning*, 110:2335–2387.

Brekelmans, R., Genewein, T., Grau-Moya, J., Delétang, G., Kunesch, M., Legg, S., and Ortega, P. (2022). Your policy regularizer is secretly an adversary. *arXiv preprint arXiv:2203.12592*.

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., and Efros, A. A. (2018). Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*.

Chevalier-Boisvert, M., Dai, B., Towers, M., de Lazcano, R., Willems, L., Lahlou, S., Pal, S., Castro, P. S., and Terry, J. (2023). Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, abs/2306.13831.

Degris, T., White, M., and Sutton, R. S. (2012). Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*.

Eysenbach, B., Myers, V., Levine, S., and Salakhutdinov, R. (2023). Contrastive representations make planning easy. In *NeurIPS 2023 Workshop on Generalization in Planning*.

Eysenbach, B., Salakhutdinov, R., and Levine, S. (2020). C-learning: Learning to achieve goals via recursive classification. *arXiv preprint arXiv:2011.08909*.

Eysenbach, B., Zhang, T., Levine, S., and Salakhutdinov, R. R. (2022). Contrastive learning as goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*, 35:35603–35620.

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*.

Haarnoja, T., Pong, V., Zhou, A., Dalal, M., Abbeel, P., and Levine, S. (2018a). Composable deep reinforcement learning for robotic manipulation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 6244–6251. IEEE.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*, pages 1352–1361. PMLR.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018b). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. (2019). Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*.

Hazan, E., Kakade, S., Singh, K., and Van Soest, A. (2019). Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pages 2681–2691. PMLR.

Huang, S., Dossa, R. F. J., Ye, C., Braga, J., Chakraborty, D., Mehta, K., and Araújo, J. G. (2022). Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18.

Husain, H., Ciosek, K., and Tomioka, R. (2021). Regularized policies are reward robust. In *International Conference on Artificial Intelligence and Statistics*, pages 64–72. PMLR.

Islam, R., Ahmed, Z., and Precup, D. (2019). Marginalized state distribution entropy regularization in policy optimization. *arXiv preprint arXiv:1912.05128*.

Jain, A. K., Lehnert, L., Rish, I., and Berseth, G. (2024). Maximum state entropy exploration using predecessor and successor representations. *Advances in Neural Information Processing Systems*, 36.

Janner, M., Mordatch, I., and Levine, S. (2020). Generative temporal difference learning for infinite-horizon prediction. *arXiv preprint arXiv:2010.14496*.

Kakade, S. and Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, volume 19. Citeseer.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. (2018). Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*.

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. (2021). Behavior from the void: Unsupervised active pre-training. *Advances in Neural Information Processing Systems*, 34:18459–18473.

Ma, Y. J., Sivakumar, K., Yan, J., Bastani, O., and Jayaraman, D. (2023). Learning policy-aware models for model-based reinforcement learning via transition occupancy matching. In *Learning for Dynamics and Control Conference*, pages 259–271. PMLR.

Manne, A. S. (1960). Linear programming and sequential decisions. *Management Science*, 6(3):259–267.

Mazoure, B., Bruce, J., Precup, D., Fergus, R., and Anand, A. (2023a). Accelerating exploration and representation learning with offline pre-training. *arXiv preprint arXiv:2304.00046*.

Mazoure, B., Eysenbach, B., Nachum, O., and Tompson, J. (2023b). Contrastive value learning: Implicit models for simple offline rl. In *Conference on Robot Learning*, pages 1257–1267. PMLR.

Mazoure, B., Talbott, W., Bautista, M. A., Hjelm, D., Toshev, A., and Susskind, J. (2023c). Value function estimation using conditional diffusion models for control. *arXiv preprint arXiv:2306.07290*.

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

Mutti, M., De Santi, R., and Restelli, M. (2022). The importance of non-markovianity in maximum state entropy exploration. *arXiv preprint arXiv:2202.03060*.

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*, volume 35, pages 9028–9036.

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*, volume 34, pages 5232–5239.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR.

Schulman, J., Chen, X., and Abbeel, P. (2017a). Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017b). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Seo, Y., Chen, L., Shin, J., Lee, H., Abbeel, P., and Lee, K. (2021). State entropy maximization with random encoders for efficient exploration. In *International Conference on Machine Learning*, pages 9443–9454. PMLR.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359.

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

Tiapkin, D., Belomestny, D., Calandriello, D., Moulines, E., Munos, R., Naumov, A., Perrault, P., Tang, Y., Valko, M., and Menard, P. (2023). Fast rates for maximum entropy exploration. In *International Conference on Machine Learning*, pages 34161–34221. PMLR.

Toussaint, M. (2009). Robot trajectory optimization using approximate inference. In *International Conference on Machine Learning*, volume 26, pages 1049–1056.

Williams, R. J. and Peng, J. (1991). Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. (2021). Reinforcement learning with prototypical representations. In *International Conference on Machine Learning*, pages 11920–11931. PMLR.

Zhang, C., Cai, Y., Huang, L., and Li, J. (2021a). Exploration by maximizing rényi entropy for reward-free rl framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10859–10867.

Zhang, T., Xu, H., Wang, X., Wu, Y., Keutzer, K., Gonzalez, J. E., and Tian, Y. (2021b). Noveld: A simple yet effective exploration criterion. *Advances in Neural Information Processing Systems*, 34:25217–25230.

Ziebart, B. D. (2010). *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. PhD thesis, Carnegie Mellon University.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *AAAI*, volume 8, pages 1433–1438. Chicago, IL, USA.

A Proofs of Theorems

Proof Theorem 3.2. Let us express the state-action value function as a function of the conditional state-action visitation distribution (Eysenbach et al., 2020; Janner et al., 2020)

$$
\begin{aligned}
Q^\pi(s,a) &= \frac{1}{1-\gamma}\int d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)R(\bar{s},\bar{a})\,d\bar{s}\,d\bar{a} \\
&= \frac{1}{1-\gamma}\int \frac{d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)}{d^{\pi^*,\gamma}(\bar{s},\bar{a}|s,a)} d^{\pi^*,\gamma}(\bar{s},\bar{a}|s,a)R(\bar{s},\bar{a})\,d\bar{s}\,d\bar{a} \\
&\geq Q^{\pi*}(s,a)\inf_{\bar{s},\bar{a}} \frac{d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)}{d^{\pi^*,\gamma}(\bar{s},\bar{a}|s,a)} \qquad (17) \\
&= Q^{\pi*}(s,a)\exp\inf_{\bar{s},\bar{a}}\left(\log\frac{d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)}{d^{\pi^*,\gamma}(\bar{s},\bar{a}|s,a)}\right) \\
&= Q^{\pi*}(s,a)\exp\left(\inf_{\bar{s},\bar{a}}\left(\log d^{\pi,\gamma}(\bar{s},\bar{a}|s,a) - \log d^{\pi^*,\gamma}(\bar{s},\bar{a}|s,a)\right)\right) \\
&= Q^{\pi*}(s,a)\exp\left(-\sup_{\bar{s},\bar{a}}\left(\log d^{\pi^*,\gamma}(\bar{s},\bar{a}|s,a) - \log d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)\right)\right) \\
&\geq Q^{\pi*}(s,a)\exp\left(-\sup_{\bar{s},\bar{a}}\left|\log d^{\pi^*,\gamma}(\bar{s},\bar{a}|s,a) - \log d^{\pi,\gamma}(\bar{s},\bar{a}|s,a)\right|\right) \qquad (18) \\
&= Q^{\pi*}(s,a)\exp\left(-\|\log d^{\pi^*,\gamma}(\cdot,\cdot|s,a) - \log d^{\pi,\gamma}(\cdot,\cdot|s,a)\|_\infty\right)\,.
\end{aligned}
$$

Inequation (17) holds by the monotonicity of the (Lebesgue) integral, and inequation (18) holds as $\sup_x f(x) \leq \sup_x |f(x)|$ for any function f.

□

Proof Theorem 4.2. For all conditional distributions q and q'

$$
\begin{aligned}
\bar{L}_n(\mathcal{T}^\pi q(\cdot|s,a), \mathcal{T}^\pi q'(\cdot|s,a))^n &= \sup_{s,a}\int |\mathcal{T}^\pi q(\bar{s}|s,a) - \mathcal{T}^\pi q'(\bar{s}|s,a)|^n\,d\bar{s} \\
&= \gamma\sup_{s,a}\int \left|\mathop{\mathbb{E}}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(\cdot|s')}}[q(\bar{s}|s',a') - q'(\bar{s}|s',a')]\right|^n\,d\bar{s} \\
&\leq \gamma\sup_{s,a}\int \mathop{\mathbb{E}}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(\cdot|s')}}[|q(\bar{s}|s',a') - q'(\bar{s}|s',a')|^n]\,d\bar{s} \\
&= \gamma\sup_{s,a}\mathop{\mathbb{E}}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(\cdot|s')}}\left[\int |q(\bar{s}|s',a') - q'(\bar{s}|s',a')|^n\,d\bar{s}\right] \\
&\leq \gamma\sup_{s,a}\sup_{s',a'}\left(\int |q(\bar{s}|s',a') - q'(\bar{s}|s',a')|^n\,d\bar{s}\right) \\
&= \gamma\sup_{s',a'}\int |q(\bar{s}|s',a') - q'(\bar{s}|s',a')|^n\,d\bar{s} \\
&= \gamma\bar{L}_n(q(\cdot|s,a), q'(\cdot|s,a))^n\,.
\end{aligned}
$$

□

B Off-Policy RL with Conditional Visitation Measure

In the following, we adapt soft actor-critic (Haarnoja et al., 2018b), itself an adaptation of off-policy actor-critic (Degris et al., 2012), according to the procedure from Section 4. In essence, soft actor-critic estimates the state-action value function with a parameterized critic Q_ϕ, which is learned using expected SARSA (sometimes called generalized SARSA), and updates the parameterized policy π_θ with approximate policy iteration (i.e., off-policy policy gradient), all based on one-step transitions stored in a replay buffer $\mathcal{D}$. The actor and critic loss functions are furthermore extended with the log-likelihood of actions weighted by the parameter λ_{SAC}, therefore called soft and considered a MaxEntRL algorithm using the entropy of policies as intrinsic reward. In the particular case where λ equals zero, the algorithm boils down to a slightly revisited implementation of off-policy actor-critic.

Soft actor-critic is adapted to MaxEntRL with the intrinsic reward function defined in Section 3.1, as follows. First, N-step transitions are stored in the buffer $\mathcal{D}$ instead of one-step transitions. Second, the conditional state visitation distribution is estimated with a function approximator d_ψ and learned with stochastic gradient descent. Third, at each iteration of the critic updates, the reward provided by the MDP is extended with the intrinsic reward.

Formally, the parameterized critic Q_ϕ is iteratively updated by performing stochastic gradient descent steps on the loss function

$$\mathcal{L}(\phi) = \mathop{\mathbb{E}}_{s_t, a_t \sim \mathcal{D}} \left[\left(Q_\phi(s_t, a_t) - y \right)^2 \right] \tag{19}$$

$$y = R(s_t, a_t) + \lambda R^{int}(s_t, a_t) + \gamma \left(Q_{\phi'}(s_{t+1}, a_{t+1'}) - \lambda_{SAC} \log \pi_\theta(a_{t+1'} | s_{t+1}) \right) , \tag{20}$$

where $a_{t+1'} \sim \pi_\theta(\cdot | s_{t+1})$, and where ϕ' is the target network parameter.

Furthermore, the policy π_θ is updated performing gradient descent steps on the loss function

$$\mathcal{L}(\theta) = - \mathop{\mathbb{E}}_{s_t, a_t \sim \mathcal{D}} \left[\log \pi_\theta(a_{t'} | s_t) A(s_t, a_{t'}) \right] \tag{21}$$

$$A(s_t, a_{t'}) = Q_\phi(s_t, a_{t'}) - \lambda_{SAC} \log \pi_\theta(a_{t'} | s_t) , \tag{22}$$

where $a_{t'} \sim \pi_\theta(\cdot | s_t)$.

Algorithm 1 summarizes the learning steps during each iteration.[1] It differs slightly from the original soft actor-critic (Haarnoja et al., 2018b). The loss equation (21) is based on the log-trick instead of the reparametrization trick, the expected SARSA update in equation (19) is approximated by sampling, and a single value function is learned, as implemented in CleanRL (Huang et al., 2022). These changes are of minor importance in our experiments.

C Hyperparameters Experiments

In practice, the agent observes the concatenation of the one-hot-encoding of the components of the state space and takes actions in one-hot-encoding format too. The policy π_θ is a neural network that outputs a categorical distribution over the action representation. The critic Q_ϕ is a neural network that takes as input the concatenation of the state and action representations and outputs a scalar. In OPAC+CV, the visitation distribution model d_ψ is also a neural network that takes the same input as the critic Q_ϕ and outputs, for each component of the state space, a categorical distribution over its one-hot-encoding representation. In OPAC+MV, the visitation distribution model d_ψ is a marginal distribution over the same one-hot-encoding representation. In both algorithms, this amounts to assuming the conditional independence of the future state components given the state and action taken as input. This implementation choice mitigates the curse of dimensionality. In addition, it allows computing the probability of a feature in closed form. The probability equals the product of the probability of the vertical position and the probability of the horizontal position provided in one-hot-encoding by the model d_ψ. Table 1 summarizes the hyperparameters used in the experiments. In practice, the parameter λ_{SAC} is constant for SAC, OPAC+CV, and OPAC+MV simulations.

[1] Implementation: `https://github.com/adrienBolland/future-visitation-exploration`

Algorithm 1 SAC with conditional visitation measure for exploration

Initialize the policy π_θ, the soft critic Q_ϕ, and the visitation model d_ψ
Initialize the critic target $Q_{\phi'}$ and visitation target $d_{\psi'}$
Initialize the replay buffer with random N-step transitions
while Learning **do**
  Sample transitions from the policy π_θ and add them to the buffer
  while Update the visitation model **do**
    Sample a batch of N-step transitions from the buffer
    Perform a stochastic gradient descent step on $\mathcal{L}(\psi)$
  end while
  while Update the critic **do**
    Sample a batch of N-step transitions from the buffer (use only the 1-step transitions)
    For each element of the batch sample $z_t \sim q^\pi(\cdot|s_t, a_t)$
    Estimate the intrinsic reward $R^{int}(s_t, a_t) = \log q^*(z_t) - \log q^\pi(z_t|s_t, a_t)$
    Perform a stochastic gradient descent step on $\mathcal{L}(\phi)$
  end while
  Sample a batch of N-step transitions from the buffer (use only the 1-step transitions)
  Perform a stochastic gradient descent step on $\mathcal{L}(\theta)$
  Update the target parameters with Polyak averaging
end while

Table 1: Hyperparameters

Parameter	Value
Neurons for each network layers	256
Layers policy	2
Layers critic	2
Learning rate policy	10^{-5}
Learning rate critic	10^{-4}
Maximum trajectory length	200
Buffer size	1000
Batch size	32
Critic target update weight τ	0.1
Discount factor γ	0.98
SAC λ_{SAC}	0.002
Layers visitation model OPAC+CV	2
Learning rate visitation model	10^{-5}
MaxEntRL λ	0.01
Density model target update weight τ	1
Bootstrap horizon N	10