MEASURING EMPOWERMENT FROM OFFLINE DATA FOR INTRINSICALLY MOTIVATED EXPLORATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large, unlabeled datasets of prior experience have become increasingly available for agentic tasks in domains such as robotics and coding. A common recipe for these tasks is to train a policy using the prior data, and then further improve it with online rollouts. But while much work has studied how to use both the offline and online data for policy extraction, far less work has been done to understand how the prior data can be used to efficiently *explore* during the online learning phase. Previous work has attempted to formalize this idea by defining the *empowerment*, a measure of how many future states an agent can realize; states with higher empowerment require more exploration as there are more futures to explore. However, existing methods for estimating empowerment require online policy rollouts in the environment. We propose the first method for estimating empowerment from unlabeled prior data, without any environment interaction. By jointly learning an empowerment-maximizing policy together with its discounted state-occupancy measure, our estimator can be trained on unlabeled trajectories generated from any unknown policy. We demonstrate that our empowerment estimator qualitatively gives intuitive empowerment measures and show that the pretrained estimator can be used to improve exploration in the online goal-conditioned reinforcement learning setting.

1 INTRODUCTION

Online exploration is a challenging yet central problem in reinforcement learning, where the agent must traverse the environment to seek out novel states with potentially high reward values. Previous methods tend to place equal weights across all states in the prior, where the agent is optimized to visit all of these states until all of them have been visited. But without any prior, the agent must visit all states, increasing the sample-complexity. This is suboptimal as some states naturally grant the agent more control over what it could do (e.g., picking up a key lets the agent open a door to a new room).

A natural way to characterize the idea above is to consider empowerment (Klyubin et al., 2005), a metric that measures to what degree an agent can influence its future states via its actions. States with high empowerment have many distinct future outcomes that the agent can achieve by taking distinct sequences of actions. Because these high-empowerment states have more future outcomes, it seems reasonable to explore more around these states as an agent has the potential to discover more future outcomes or behaviors. Conversely, in low-empowerment states, there is very little that an agent can do to influence its future outcome, so it should spend less time exploring these regions of the state space as there is less strategic value to be gained.

Previous works have explored using empowerment, or related information-theoretic objectives for online exploration (Sharma et al., 2020a; Kim et al., 2019; Kumar, 2018; Becker-Ehmck et al., 2021; Bharadhwaj et al., 2022), but operate in the purely online setting, where we do not have access to prior data. Luckily, in many practical environments, we have prior data in the form of unlabeled trajectories. Ideally, we would like to use the prior data to inform us how to explore online, but estimating empowerment purely from offline data is difficult: existing empowerment estimators (Mohamed & Jimenez Rezende, 2015; Gregor et al., 2017; Eysenbach et al., 2019; Sharma et al., 2020b;a; Kumar, 2018; Becker-Ehmck et al., 2021; Bharadhwaj et al., 2022; Kim et al., 2019) rely on online samples where a skill-conditioned agent must interact with the environment during the optimization. Furthermore, adapting online methods to the offline setting by simply regularizing the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

policy as is typical in the offline RL literature (Kumar et al., 2020; Fujimoto & Gu, 2021; Kostrikov et al., 2022) does not work (Section 6.2). Existing empowerment estimators look at whether the outcomes achieved by a policy align with its intentions; however, offline, we do not know the policy's intention. **How can we learn empowerment from offline data to improve online exploration?**

In this work, we provide a novel algorithm for learning empowerment purely from *offline data*, without additional environment interaction. Our key contribution is an offline estimator of empowerment: by learning an empowerment-maximizing policy in tandem with its discounted state occupancy, we can extract an estimator of the empowerment from prior data alone. We show that our method produces qualitatively accurate empowerment maps, correctly identifying which states admit the most diverse future outcomes. We further demonstrate that the offline-extracted empowerment estimate can be used to improve exploration in online tasks. Specifically, we find that using the empowerment as an auxiliary loss term improves exploration on downstream, online tasks in OGBench environments (Park et al., 2025). We evaluate our methods in sparse-reward, goal-conditioned environments, where knowing *where* to explore is harder due to the sparsity of the reward, and the agent must prioritize which states to visit. In this setting, our empowerment estimator provides a principled way to characterize the exploration value of a state from offline data, directing the agent toward regions with greater potential for diverse future behaviors.

2 Related Work

Empowerment. Empowerment (Klyubin et al., 2005) measures an agent's control over its future as the mutual information between its action sequence (or skill choice) and the resulting future state. Prior works have estimated empowerment from **online** data by optimizing a variational lower bound (Mohamed & Jimenez Rezende, 2015), and later work estimates the empowerment of an agent's *current* policy from off-policy samples, by maximizing lower bounds on the empowerment (Kumar, 2018; Zhao et al., 2021; Belghazi et al., 2018; Myers et al., 2024; Bharadhwaj et al., 2022; Choi et al., 2021; Zheng et al., 2025). However, these methods estimate the empowerment of the current goal-reaching policy rather than the maximum achievable empowerment, which can be a very weak bound in practice (Poole et al., 2019). Becker-Ehmck et al. (2021) compute empowerment directly, but require finite action spaces and incur cost exponential in the horizon. Schneider et al. (2025) show that empowerment pretraining provides a strong initialization for downstream finetuning, but use a fixed estimator constructed by a human annotator. The closest to our work is EELMA (Song et al., 2026) which estimates the *effective* empowerment (defined in Section 3) using contrastive learning for discrete LLM actions from collected trajectories; we adapt this method to the continuous action space setting and use it as a baseline in Section 6.2. Critically, none of these empowerment estimators work well when adapted to the offline RL setting (Section 6.2). **Our work is the first to estimate the maximal empowerment purely from unlabeled offline data.**

Intrinsic motivation and curiosity-driven exploration. A large body of work in reinforcement learning has studied intrinsic motivation as a mechanism for encouraging exploration in sparse-reward environments. Early approaches include count-based exploration and density modeling methods that assign bonuses to rarely visited states (Tang et al., 2017). More recent methods such as curiosity-driven learning estimate prediction error or learning progress in a forward dynamics model as an intrinsic reward signal (Burda et al., 2019; Kim et al., 2019). Our work builds on these ideas, using empowerment as an exploration bonus that can efficiently leverage prior data.

Exploration from offline data. Offline RL learns policies purely from fixed datasets without further environment interaction. Offline-to-online methods pretrain a policy or value function on the offline data and then fine-tune it with online interaction (Uchendu et al., 2023; Nakamoto et al., 2023). Prior methods have shown that an online RL agent can be trained to explore much more efficiently in the environment by leveraging offline data (Li et al., 2023; Wilcoxson et al., 2025), but do not define a principled way to characterize the "exploration value" of a state. Our work instead learns empowerment offline as a principled exploration objective.

3 PRELIMINARIES

We formulate the environment as a Markov Decision Process (MDP) defined by a state space $\mathcal{S}$, action space $\mathcal{A}$, transition dynamics $P(s'|s,a)$, and a discount factor γ. Let trajectories be denoted by the random variables $s_0, s_1, \ldots, s_T$. We are particularly interested in the distribution of discounted future states, denoted as $s^+ \triangleq s_{t+k}$ where $k \sim \text{Geom}(1-\gamma)$ (Eysenbach et al., 2022). Let $\pi(a \mid s, z) \in \Pi^z$ be a skill-conditioned policy, where $z \in \mathcal{Z} = \{1, \ldots, K\}$ is a latent skill drawn from a fixed, pre-determined set of K skills and represented as a one-hot vector. In line with previous work (Dayan, 1993), we define the discounted state occupancy measures as

$$p^z(s^+ \mid s, a) \triangleq \mathbb{E}_{\substack{s_{t+1}\sim P(\cdot|s_t,a_t),\\ a_t\sim\pi(\cdot|s_t,z)}} \left[\sum_{k=1}^{\infty} (1-\gamma)\,\gamma^{k-1}\, \mathbf{1}\big(s_k = s^+\big) \,\middle|\, s_0 = s, a_0 = a \right], \tag{1}$$

$$p^z(s^+ \mid s) \triangleq \mathbb{E}_{a\sim\pi(\cdot|s,z)} \left[p^z(s^+ \mid s, a) \right]. \tag{2}$$

Learning occupancy measures by temporal differences. Because s^+ is drawn from a geometric mixture over future time steps, the occupancy measures satisfy a Bellman-style recursion (Dayan, 1993):

$$p^z(s^+ \mid s, a) = (1-\gamma)\, P(s^+ \mid s, a) \;+\; \gamma\, \mathbb{E}_{s'\sim P(\cdot|s,a)}\big[p^z(s^+ \mid s')\big], \tag{3}$$

$$p^z(s^+ \mid s) = \mathbb{E}_{a\sim\pi(\cdot|s,z)}\big[p^z(s^+ \mid s, a)\big]. \tag{4}$$

This recursion can be used as a temporal-difference target to learn occupancy measures from off-policy transitions (s, a, s'), without rollouts of π itself. By analogy with value functions, we write $Q^z(s^+ \mid s, a)$ and $V^z(s^+ \mid s)$ for learned models of $p^z(s^+ \mid s, a)$ and $p^z(s^+ \mid s)$, respectively.

Empowerment (Klyubin et al., 2005; Mohamed & Jimenez Rezende, 2015) quantifies the maximal influence an agent can exert over its future outcomes from a given state, typically measured as the channel capacity between a control or intention and a future outcome. Prior work has offered many variations of the specific form. We use the definition from Gregor et al. (2017), which defines the empowerment in state s as the maximum possible mutual information between a uniformly-distributed random latent skill $z \in \mathcal{Z} = \{1, \ldots, K\}$ and the resulting future state s^+, conditioned on the current state:

$$\mathcal{E}(s) \triangleq \max_{\pi\in\Pi^z} I^\pi(s^+; z \mid s_t = s). \tag{5}$$

We call this the *potential* or *maximal empowerment*, to distinguish it from the *effective empowerment* defined next. Intuitively, empowerment should be high when a policy can realize a diverse set of outcomes where each outcome corresponds to a distinct skill. For example, an agent stuck in the dead end of a maze has very low empowerment, as its actions have almost no outcome.

We take Π^z to be the class of Gaussian policies with a learned mean $\mu_\theta(s, z)$ and a fixed standard deviation σ_a, clipped to the action bounds: $a = \text{clip}(\mu_\theta(s, z) + \eta,\, -1,\, 1)$ with $\eta \sim \mathcal{N}(0, \sigma_a^2 I)$. For a deterministic policy in a deterministic environment, each occupancy measure $p^z(\cdot \mid s)$ is a set of point masses on the states of a single trajectory, so it has no density over the continuous state space, and the mutual information theoretically saturates: $I^\pi(s^+; z \mid s) = \log K$ whenever no two skills' trajectories share a state, no matter how close together they are. With fixed action noise, each skill's occupancy measure is spread over a neighborhood of its trajectory instead, so the skills' occupancy measures overlap and the mutual information reflects how far apart the skills' futures are.

While potential empowerment represents the theoretical maximum control, evaluating it directly would require searching over all policies. Instead, we can define the *effective empowerment* as a lower bound induced by a specific, known policy π:

$$\tilde{\mathcal{E}}(s; \pi) \triangleq I^\pi(s^+; z \mid s_t = s) \;=\; \frac{1}{K} \sum_{z\in\mathcal{Z}} \mathbb{E}_{s^+\sim p^z(\cdot|s)} \left[\log \frac{p^z(s^+ \mid s)}{\frac{1}{K}\sum_{z'\in\mathcal{Z}} p^{z'}(s^+ \mid s)} \right], \tag{6}$$

where the equality uses that z is uniform; $\tilde{\mathcal{E}}(s; \pi) \le \mathcal{E}(s)$, with equality for the maximizing π.

Our method iteratively discovers a policy to maximize this effective empowerment, thereby approximating the potential empowerment $\mathcal{E}(s)$, using samples (s, a, s') from an offline dataset $\mathcal{D}$.

Problem formulation. We assume access only to a static dataset $\mathcal{D}$ of transitions (s, a, s') collected by an unknown behavior policy π_β, with no further environment interaction. Our goal is to use $\mathcal{D}$ to obtain an estimator $\hat{\mathcal{E}}(s)$ of the potential empowerment $\mathcal{E}(s)$ (Eq. 5) at any state s.

4 AN ALGORITHM FOR OFFLINE EMPOWERMENT

In this section, we present our algorithm for estimating empowerment from offline data. The key insight is that the effective empowerment of a policy depends on that policy only through its discounted state occupancy measures (Eq. 6), which can be learned off-policy using temporal differences (Eqs. 3–4). Our method jointly learns **occupancy models** $Q_\theta^z(s^+ \mid s, a)$ and $V_\theta^z(s^+ \mid s)$ for a skill-conditioned policy by TD on offline transitions (Section 4.1), trains that **policy** $\pi(a \mid s, z)$ to maximize the effective empowerment computed from these models so that it approaches the potential empowerment $\mathcal{E}(s)$ (Section 4.2), and reads off an **empowerment estimator** $\hat{\mathcal{E}}(s)$ from these models (Section 4.3). Throughout training, we sample skills $z \sim p(z)$ uniformly from $\mathcal{Z}$.

4.1 LEARNING THE OCCUPANCY MEASURE

To evaluate effective empowerment offline (Section 3), we must estimate the distribution of future states without environment interactions. We achieve this by learning function approximators Q_θ and V_θ that model the discounted future state occupancy measures so that $Q_\theta^z(s^+|s,a) \approx p^z(s^+|s,a)$ and $V_\theta^z(s^+|s) \approx p^z(s^+|s)$. Each training example consists of a transition $(s, a, s') \sim \mathcal{D}$ together with a random state s^+ drawn uniformly from all states in $\mathcal{D}$, independently of the transition. Because the empowerment-maximizing policy differs from the behavior policy, these sampled futures are not technically samples from $p^z(\cdot \mid s, a)$, but the recursive backup allows us to still learn these quantities.

Bellman Backups. We train the occupancy networks by using the recursion of Eq. (3)–4 as a TD target, with the next state s' sampled from the dataset and the expectation over actions in Eq. (4) taken under the current policy $\pi(\cdot \mid s, z)$. A caveat is that the first term of Eq. (3), $(1-\gamma)P(s^+ \mid s, a)$, is almost never observed, as a random state s^+ drawn from the dataset almost surely differs from the next state s'. We resolve this by adding a second loss where $s^+ = s'$ instead of the randomly sampled offline data state, so that its target includes the $(1-\gamma)$ term. Equivalently, we sum two regression losses with equal weight,

$$\mathcal{L}_Q^{\text{boot}} \;=\; \mathbb{E}_{(s,a,s',s^+)}\Big[\, \ell\big(\, Q_\theta^z(s^+ \mid s,a),\;\; \gamma\, V_{\bar{\theta}}^z(s^+ \mid s')\,\big)\Big], \tag{7}$$

$$\mathcal{L}_Q^{\text{self}} \;=\; \mathbb{E}_{(s,a,s',s^+)}\Big[\, \ell\big(\, Q_\theta^z(s' \mid s,a),\;\; (1-\gamma) + \gamma\, V_{\bar{\theta}}^z(s' \mid s')\,\big)\Big], \tag{8}$$

with total objective $\mathcal{L}_Q = \mathcal{L}_Q^{\text{boot}} + \mathcal{L}_Q^{\text{self}}$, where ℓ is a clipped linear exponential regression loss used in prior works for stability (Parsian & Kirmani, 2002) (Section 4.4). The V-loss is

$$\mathcal{L}_V \;=\; \mathbb{E}_{(s,a,s',s^+)}\Big[\, \ell\big(\, V_\theta^z(s^+ \mid s),\;\; \mathbb{E}_{a' \sim \pi(\cdot|s,z)}\big[Q_{\bar{\theta}}^z(s^+ \mid s,a')\big]\,\big)\Big] \tag{9}$$

Note that $Q_{\bar{\theta}}, V_{\bar{\theta}}$ denote target networks for Q and V. We update the networks using Polyak averaging (Lillicrap et al., 2016) with an update rate of $\tau = 0.005$.

Latent Space Parameterization. Recall that computing empowerment (Eq. (6)) requires samples from $p^z(s^+ \mid s)$. We now describe a parameterization of V_θ that makes such sampling tractable.

To tractably model these high-dimensional distributions, we project states and actions into a shared representation space of dimension d. We define non-linear encoders $\phi_Q(s, a, z)$, $\phi_V(s, z)$, and $\psi(s^+)$, and parameterize the log-occupancies as scaled negative Euclidean distances:

$$\log Q^z(s^+|s,a) = -\tfrac{1}{d}\|\phi_Q(s,a,z) - \psi(s^+)\|^2, \quad \log V^z(s^+|s) = -\tfrac{1}{d}\|\phi_V(s,z) - \psi(s^+)\|^2 \tag{10}$$

This formulation allows us to sample directly from the discounted state-occupancy measure as an isotropic Gaussian in the latent space $p(\psi \mid \phi) = \mathcal{N}(\phi, \frac{d}{2}I)$, which is needed to compute the empowerment. Note that while we cannot sample s^+, we can still sample $\psi(s^+)$ which is sufficient for computing $V^z(s^+ \mid s)$ as we can plug the sampled latent into Eq. (10).

It is possible to use separate networks for both ϕ_Q, ϕ_V, or alternatively, only learn $Q^z(s^+ \mid s, a)$ and simply use $V^z(s^+ \mid s) = \mathbb{E}_{a \sim \pi(\cdot|s,z)}\big[Q^z(s^+ \mid s, a)\big]$. We find that using separate networks for Q and V leads to more stable training and qualitatively better estimates of the empowerment.

4.2 POLICY OPTIMIZATION VIA THE EMPOWERMENT GRADIENT

Now that we can learn and sample from the state occupancies, we optimize a policy $\pi(a \mid s, z)$ to maximize its effective empowerment. Because empowerment depends only on the state, we need a

way to describe how a skill's action distribution influences it. We can define $\mathcal{M}^{z'}(s^+ \mid s, \pi, z)$ as the marginal modified discounted state occupancy measure, representing the distribution over s^+ if skill z draws its action at s from $\pi(\cdot \mid s, z)$ and all skills follow the current policy thereafter:

$$\mathcal{M}^{z'}(s^+ \mid s, \pi, z) = \mathbf{1}\{z' = z\}\, \mathbb{E}_{a\sim\pi(\cdot|s,z)}\big[Q^{z'}(s^+ \mid s, a)\big] + \mathbf{1}\{z' \neq z\} V^{z'}(s^+ \mid s). \tag{11}$$

We can then define the (π, z)-modified empowerment,

$$\mathcal{E}_\Delta(s; \pi, z) \;=\; \frac{1}{K} \sum_{z' \in \mathcal{Z}} \mathbb{E}_{s^+\sim\mathcal{M}^{z'}(\cdot|s,\pi,z)} \left[\log \left(\frac{\mathcal{M}^{z'}(s^+ \mid s, \pi, z)}{\frac{1}{K}\sum_{z'' \in \mathcal{Z}} \mathcal{M}^{z''}(s^+ \mid s, \pi, z)} \right) \right], \tag{12}$$

which quantifies the local change in empowerment caused by changing the action distribution of skill z at s. Increasing this encourages the skill-conditioned policy to take actions that make its future state occupancies under different skills distinguishable. Since we can easily sample from the marginal occupancy measure $V^{z'}$ via our Gaussian parameterization from above, we rewrite this as an importance-weighted expectation and estimate it by Monte Carlo sampling:

$$\mathcal{E}_\Delta(s; \pi, z) = \frac{1}{K} \sum_{z' \in \mathcal{Z}} \mathbb{E}_{s^+\sim V^{z'}(\cdot|s)} \left[\frac{\mathcal{M}^{z'}(s^+ \mid s, \pi, z)}{V^{z'}(s^+ \mid s)} \log \frac{\mathcal{M}^{z'}(s^+ \mid s, \pi, z)}{\frac{1}{K}\sum_{z'' \in \mathcal{Z}} \mathcal{M}^{z''}(s^+ \mid s, \pi, z)} \right]. \tag{13}$$

The complete policy objective is:

$$\pi \leftarrow \arg\max_{\pi}\; \mathbb{E}_{(s,a)\sim\mathcal{D},\, z\sim p(z)} \Big[\mathcal{E}_\Delta\big(s; \pi, z\big) + \alpha_{\text{bc}} \log \pi(a \mid s, z) \Big] \tag{14}$$

where α_{bc} is a hyperparameter that enforces a behavioral constraint. We train the Q, V networks and the policy jointly, taking a gradient step on their combined losses at every iteration (Algorithm 1).

4.3 Estimating the Empowerment

Once training has converged, the learned policy π approximately maximizes the effective empowerment, so its effective empowerment $\tilde{\mathcal{E}}(s; \pi)$ serves as an estimate of the potential empowerment $\mathcal{E}(s)$. We compute it by substituting the learned occupancy model V_θ^z into Eq. (6) and approximating the expectation by sampling latents from the Gaussian parameterization of V_θ^z in Eq. (10):

$$\hat{\mathcal{E}}(s) \;=\; \frac{1}{K} \sum_{z \in \mathcal{Z}} \frac{1}{M} \sum_{m=1}^{M} \left[\log V_\theta^z(\psi_{z,m} \mid s) - \log \frac{1}{K} \sum_{z' \in \mathcal{Z}} V_\theta^{z'}(\psi_{z,m} \mid s) \right], \tag{15}$$

where $\psi_{z,m} \sim \mathcal{N}\big(\phi_V(s, z), \frac{d}{2} I\big)$ are M latent samples per skill and $V_\theta^z(\psi \mid s)$ denotes Eq. (10) evaluated at latent ψ. This is the estimator we evaluate in Section 6.

4.4 Additional Details

Action noise. Every expectation over $a \sim \pi(\cdot \mid s, z)$, in $\mathcal{L}_V$ and in the policy objective (Section 4.2), is estimated with reparameterized samples $a = \text{clip}(\mu_\theta(s, z) + \eta,\, -1,\, 1)$. Since the policy's actions are small perturbations of its mean, we also perturb the dataset actions with the same noise when training Q_θ, $\tilde{a} = \text{clip}(a + \eta,\, -1,\, 1)$, so that Q_θ is trained on the same kind of slightly noisy actions at which it is later evaluated. We use $\sigma_a = 0.01$ in all experiments. A deterministic policy ($\sigma_a = 0$) also works in practice, but a small amount of noise yields more consistent empowerment estimates across training runs by keeping the skills better separated; Section A.2 ablates this choice.

Clipped Linear Exponential Loss. Because Q and V operate in log-space, we regress in the exponentiated domain using a clipped linear-exponential loss for numerical stability, with the clipping threshold set to $t = 5$ to avoid exploding gradients; its exact form is given in Section A.5.

5 Experimental Setup

Our experiments have two stages. We first train our empowerment estimator on offline OGBench datasets and compare it against offline adaptations of existing empowerment estimators (Sections 6.1

Algorithm 1: Offline Empowerment Estimation

Require: Offline dataset $\mathcal{D}$ of transitions (s, a, s'); number of skills K; discount γ; behavioral constraint coefficient α_{bc}; Polyak rate τ

Initialize occupancy networks Q_θ, V_θ, target networks $Q_{\bar{\theta}} \leftarrow Q_\theta$, $V_{\bar{\theta}} \leftarrow V_\theta$, and policy $\pi(a \mid s, z)$
for each training step **do**
  Sample a batch of transitions $(s, a, s') \sim \mathcal{D}$
  Sample a random state $s^+ \sim \mathcal{D}$ and a skill $z \sim \mathrm{Unif}\{1, \dots, K\}$ for each transition
  Compute the occupancy losses $\mathcal{L}_Q = \mathcal{L}_Q^{\text{boot}} + \mathcal{L}_Q^{\text{self}}$ and $\mathcal{L}_V$ ▷ *Sec. 4.1*
  Sample latent futures $\psi_{z',m} \sim \mathcal{N}\big(\phi_V(s, z'), \frac{d}{2} I\big)$ for each $z' \in \mathcal{Z}$, $m = 1, \dots, M$
  Sample actions $a_m \sim \pi(\cdot \mid s, z)$ with the reparameterization trick
  Estimate $\mathcal{E}_\Delta(s; \pi, z)$ with Eq. 13, averaging over the samples ▷ *Sec. 4.2*
  Compute the policy loss $\mathcal{L}_\pi$ from Eq. 14
  Take one gradient step on $\mathcal{L}_Q + \mathcal{L}_V + \mathcal{L}_\pi$
  $\bar{\theta} \leftarrow \tau\theta + (1 - \tau)\bar{\theta}$ ▷ *Polyak update of $Q_{\bar{\theta}}, V_{\bar{\theta}}$*
return $\hat{\mathcal{E}}(s)$ computed from V_θ with Eq. 15 ▷ *Sec. 4.3*

and 6.2). We then freeze the estimator and use it to guide online exploration on sparse-reward, goal-reaching tasks (Section 6.3). Since computing the true empowerment over all policies is intractable in continuous environments, we evaluate the estimator qualitatively, in environments whose structure dictates where the empowerment should be high (e.g., toward the center of a maze rather than in its dead ends). Hyperparameters are given in Table 1.

5.1 Baselines

Our method is the first to learn empowerment in continuous environments from offline data, so there are no direct baselines to compare to; instead, we adapt four existing online empowerment estimators to the offline setting. We give the core idea behind each baseline below; full derivations, architectures, and hyperparameters are given in Section A.6.

Empowerment baselines. The baselines differ from each other, and from our method, along two orthogonal axes: *which* mutual information they measure and *how* they estimate it. Three measure the *action empowerment* $I(s^+; a \mid s)$, the mutual information between the action taken at s in the dataset and the future state s^+ (Section A.6), and one measures the skill empowerment $I(s^+; z \mid s)$, as our method does; each estimates it with a variational bound, whereas our method computes it directly from learned occupancy measures. **CF-Empowerment** (contrastive features) estimates $I(s^+; a \mid s)$ with a binary NCE contrastive critic (Eysenbach et al., 2022) using in-batch negatives. **CF-FlowBC-Empowerment** replaces those in-batch negatives with actions from a learned flow-matching behavior-cloned policy (Lipman et al., 2023). **DV-Empowerment** is architecturally identical but optimizes the Donsker–Varadhan bound (Belghazi et al., 2018) instead of InfoNCE. **DADS-Empowerment** estimates the skill empowerment $I(s^+; z \mid s)$, matching our formalism, by jointly learning a discrete skill labeling and transition channel that maximize the Barber–Agakov bound (Sharma et al., 2020b; Barber & Agakov, 2003).

Online, errors in their learned models (e.g., dynamics or skill labelings) are corrected by fresh data; offline, distribution shift and approximation errors bias the estimate (Section 6.2).

5.2 Online Exploration with Offline Empowerment

We also use the empowerment estimator $\hat{\mathcal{E}}(s)$ (Eq. (15)), trained on $\mathcal{D}$ and then frozen, to guide exploration when an agent later solves online, sparse-reward, goal-reaching tasks in the same environment.

Base agent. All online agents are built on contrastive RL (CRL) (Eysenbach et al., 2022), a goal-conditioned actor-critic. The critic $f(s, a, g) = \phi(s, a)^\top \psi(g)$ is trained with a contrastive loss to tell whether g is a future state of (s, a) on the same trajectory, and the policy $\pi(a \mid s, g)$ maximizes f along with a SAC-style auto-tuned entropy term (Haarnoja et al., 2018; 2019). Every online agent is

trained with Reinforcement Learning with Prior Data (RLPD) (Ball et al., 2023): each training batch is drawn half from the online replay buffer and half from $\mathcal{D}$.

Empowerment bonus. A naive way to use empowerment is to add it as an additional reward; however, this can have the undesired consequence of keeping the policy from leaving high-empowerment states. Instead, we would like the policy to visit high-empowerment states more often so it can explore around them, but not be penalized for leaving those high-empowerment states and continuing its trajectory into other parts of the state space. Consequently, we train a network $B_\omega(s, a)$ to predict the *maximum* empowerment reached anywhere along the trajectory containing the transition (s, a). For every complete trajectory $\tau = (s_0, a_0, \ldots, s_T)$ in either the online or offline buffer, each of its transitions is regressed onto the highest empowerment reached along τ:

$$\min_{\omega} \; \mathbb{E}_{\tau,\, t}\Big[\big(B_\omega(s_t, a_t) - \max_{0 \le k \le T} \hat{\mathcal{E}}(s_k)\big)^2\Big]. \tag{16}$$

Since the max is taken over both the future *and* the history, leaving a high-empowerment state is not penalized, while actions that lead to one are rewarded. The policy then maximizes

$$\mathbb{E}_{(s,g),\; a\sim\pi(\cdot|s,g)}\big[f(s, a, g) + \beta\, B_\omega(s, a) - \alpha_{\text{ent}} \log \pi(a \mid s, g)\big], \tag{17}$$

where β is the bonus weight, and α_{ent} is the automatically tuned entropy temperature.

Exploration baselines. We compare against standard CRL and two exploration methods. **Random Network Distillation (RND)** (Burda et al., 2019) rewards novel states, measured by how poorly a trained predictor matches a fixed, randomly initialized network; this reward is learned into a separate critic $Q_{\text{RND}}(s, a)$, and the CRL policy maximizes $f(s, a, g) + \beta_{\text{RND}}\, Q_{\text{RND}}(s, a)$ along with the entropy term. **SUPE** (Wilcoxson et al., 2025) pretrains a latent skill model on $\mathcal{D}$, then trains a high-level SAC policy online that reselects a skill every H steps; offline data is relabeled with a skill and an optimistic reward (a learned model plus an RND bonus) is mixed into training with RLPD.

Combining empowerment with other exploration methods. Because our bonus only adds a term to the policy objective, it can be combined with other exploration methods. We test the effect of combining our empowerment bonus with RND and also adding it to the SUPE high-level policy loss.

Evaluation. We evaluate on sparse-reward, goal-conditioned variants of OGBench environments in which the agent starts from a fixed state and must reach goals spread across the environment, as exploration is especially critical in these environments due to the sparsity of the reward signal: *antmaze-medium* (an ant starts in a corner of the medium maze and must reach other cells of the maze), *pointmaze-teleport* (a point must navigate a maze with random teleporters), *antsoccer-arena* (the ant must push a ball to different goals), and *cube-single* (a robot arm must move a cube to a different position on the ground). Section A.3.1 describes each task in detail. Each task is paired with one or more OGBench datasets for $\mathcal{D}$, which differ in data quality: *navigate* and *play* contain long, goal-directed trajectories, *stitch* contains only short trajectory segments that must be stitched together, *noisy* adds heavy action noise, and *explore* contains highly suboptimal, random-walk-like trajectories. The empowerment estimator for each task is trained on the same dataset $\mathcal{D}$ used by the online agent for RLPD. Implementation details and hyperparameters are given in Section A.3.

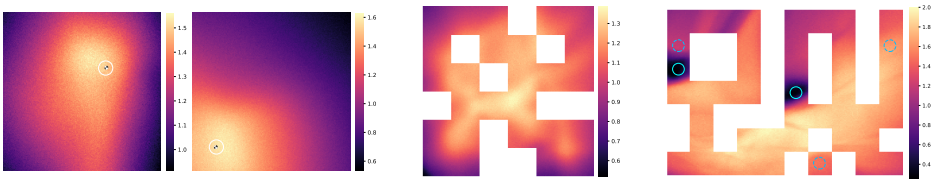
6 Experiments

We answer two questions: (1) Does our method recover meaningful empowerment estimates from offline data? (2) Can we use the offline empowerment estimator to improve online exploration?

6.1 Offline Empowerment Plots

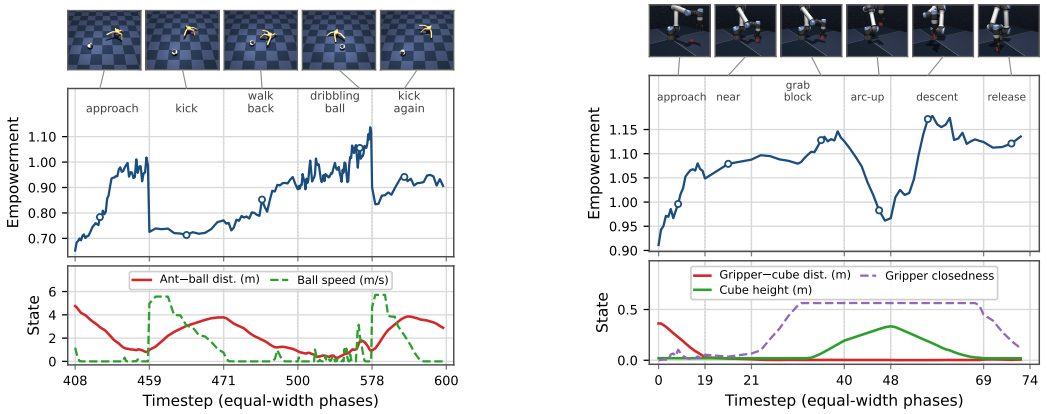
Fig. 1 shows the empowerment recovered by our offline estimator on three OGBench environments; in each case it matches what we intuitively expect. On *antsoccer-arena-navigate-v0* it peaks immediately around the ball in both plots, where the ant can freely choose whether to approach, dribble, or move away from it. On *antmaze-medium-navigate-v0* it peaks at the maze's two corridor junctions and falls off in dead ends and corners. On *pointmaze-teleport-navigate-v0* it collapses at each one-way teleporter entrance, where the agent has no control over its random next state.

Fig. 2 shows the empowerment estimate along real dataset trajectories. In *antsoccer* (Fig. 2a), empowerment rises as the ant approaches the ball, drops sharply the moment it kicks it, and stays low



(a) Ant-Soccer Arena. (b) AntMaze. (c) PointMaze-Teleport.

Figure 1: Empowerment recovered by our offline estimator; axes are the agent's (x, y) position. (a) *antsoccer-arena-navigate-v0* with the ball (circled in white) fixed off-center (left) and cornered (right); each point's color is the empowerment with the ant there. (b) *antmaze-medium-navigate-v0*. (c) *pointmaze-teleport-navigate-v0*, with teleporter entrances (solid circles) and exits (dashed).



(a) Ant-Soccer. (b) Cube.

Figure 2: Empowerment (middle panel) recovered by our offline estimator along real dataset trajectories, alongside the underlying state (bottom). (a) *antsoccer*: the ant walks up to the ball, kicks it away, walks back, dribbles it, and kicks it again. (b) *cube-single*: one cycle in which the arm grabs the block, picks it up, and places it down in a new spot. Rendered frames (top) show the scene at the timestep marked by each hollow circle. The x-axis has timesteps at the phase boundaries as tick labels.

while the ball rolls out of reach, then climbs again as the ant walks back and dribbles, peaking just before the second kick with the ball at its feet. In *cube* (Fig. 2b), empowerment rises as the gripper approaches and grasps the block. Once the arm carries the block up into the air, the empowerment falls: this is because the gripper must stay closed to retain control of the block, giving the agent fewer degrees of freedom to control its future state. Empowerment rises as the arm descends and releases the block; here empowerment is high because, as in *antsoccer*, the arm can choose to back off from the cube, or grab it and move it.

6.2 Qualitative Comparison to Baselines

Fig. 3 shows $\hat{\mathcal{E}}(s)$ recovered by the four empowerment baselines of Section 5.1. None recovers a comparable map. **DADS-Empowerment** saturates at its ceiling of $\log K$ almost everywhere; **CF-Empowerment** learns a reasonable estimate, but is noisy and only captures the coarser trends; **CF-FlowBC-Empowerment** also fails. **DV-Empowerment**, whose bound is known to suffer from high variance (Song & Ermon, 2020), is virtually unbounded.

6.3 Online Exploration

Fig. 4 compares our empowerment bonus against CRL, CRL + RND, and SUPE on every task and dataset (Section 5.2), and also shows the bonus combined with RND and SUPE. Across *cube-single-*

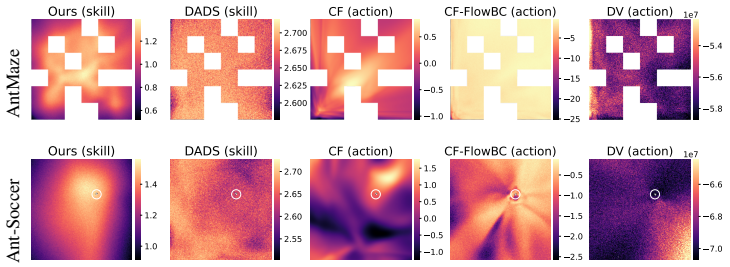


Figure 3: Empowerment maps for our method and the four baselines of Section 5.1, on *antmaze-medium-navigate* (top) and *antsoccer-arena-navigate* with the ball (circled in white) fixed off-center (bottom). DADS is skill-empowerment; CF, CF-FlowBC, and DV are action-empowerment.

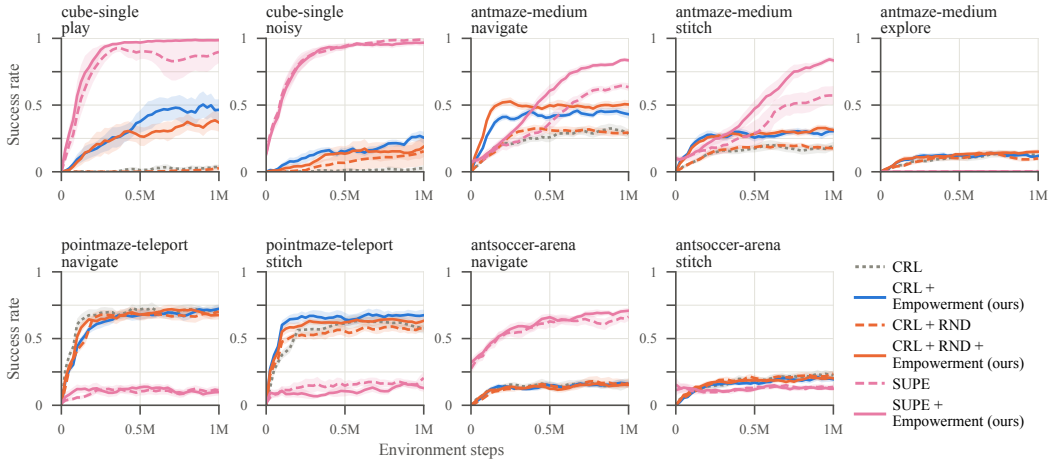


Figure 4: Online success rate over 1M steps on each task (top line of each panel title) with each offline dataset (bottom line), for CRL, RND, and SUPE with and without our empowerment bonus. Dashed lines show the baselines on their own and solid lines of the same color show them with the empowerment bonus added. Lines show the mean over 5 seeds with 95% confidence intervals.

play, *cube-single-noisy*, *antmaze-medium-navigate*, *antmaze-medium-stitch*, and *pointmaze-teleport-stitch*, adding the empowerment bonus improves over baseline CRL, enabling faster learning and higher performance. On the other environments, our method still performs on par with CRL and does not hurt performance. Our method can also be combined with existing exploration methods. Adding the empowerment to CRL + RND leads to improved performance on *cube-single*, *antmaze-medium*, and *pointmaze-teleport* over RND alone. Adding it to SUPE on *cube-single-play*, *antmaze-medium-navigate*, *antmaze-medium-stitch*, and *antsoccer-arena-navigate* speeds up learning.

Ablations. Section A.2 ablates the action noise scale σ_a and behavioral constraint coefficient α_{bc} of our estimator, and the weight β of the online empowerment bonus.

7 Conclusion

In this work we introduced a method for estimating empowerment purely from offline data, by jointly learning an empowerment-maximizing policy together with its discounted state-occupancy measure. Unlike prior methods, which rely on online rollouts of a skill-conditioned policy, our estimator trains on unlabeled offline trajectories and recovers an intuitive empowerment landscape across OGBench environments. Our results show that the estimator can be used to improve online exploration on sparse, goal-conditioned RL tasks. Beyond exploration, our estimator is general, and we hope future work will apply it to other problems and domains.

Reproducibility Statement

We conduct all experiments in the publicly released OGBench environments and offline datasets (Park et al., 2025). The online evaluation tasks, including their goal configurations and episode horizons, are detailed in Appendix A.3.1. Our offline empowerment estimator is described in full in Section 4, with Algorithm 1 providing an overview of the training procedure. Further details on the action-noise scheme and the training loss are provided in Section 4.4 and Appendix A.5, respectively. The estimator's hyperparameters are listed in Appendix A.1, while Appendix A.6 describes the model architectures, training procedures, and hyperparameters used for all four empowerment baselines. For online exploration, Section 5.2 and Appendix A.3 provide a complete description of the evaluated agents, including the integration of our empowerment bonus with CRL, RND, and SUPE, as well as the hyperparameter sweeps and common settings used across methods. We report the mean and 95% confidence intervals across five random seeds for all online experiments, and evaluate the sensitivity of our approach to its primary hyperparameters in Appendix A.2. Details of the computational resources used for our experiments are provided in Appendix A.4.

AI Disclosure

In this work, we did not use generative AI tools for developing the method or its theoretical framework, formulating or proving mathematical claims, proposing or refining hypotheses, designing experiments, implementing methods, or interpreting the results. Generating synthetic data, translation, dataset cleaning, and qualitative data analysis are not applicable to this work. We used generative AI tools for tasks with recommended disclosure: writing and editing the scripts that generate our figures, drafting parts of the paper that the authors then rewrote, editing the text for readability and concision, identifying and formatting references, and transcribing hyperparameter values from our experiment configurations into tables. We have reviewed all AI-assisted work: the authors read and substantially revised every passage, checked all figures against experiment logs, and verified all references. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

References

Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. OPAL: Offline Primitive Discovery for Accelerating Offline Reinforcement Learning. In *International Conference on Learning Representations*, 2021.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. aXiv:1607.06450, 2016. URL https://arxiv.org/abs/1607.06450.

Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient Online Reinforcement Learning With Offline Data. In *International Conference on Machine Learning*, pp. 1577–1594. PMLR, 2023.

David Barber and Felix Agakov. The IM Algorithm: A Variational Approach to Information Maximization. In *International Conference on Neural Information Processing Systems*, NIPS'03, pp. 201–208. MIT Press, 2003.

Philip Becker-Ehmck, Maximilian Karl, Jan Peters, and Patrick van der Smagt. Exploration via Empowerment Gain: Combining Novelty, Surprise and Learning Progress. In *ICML 2021 Workshop on Unsupervised Reinforcement Learning*, 2021.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual Information Neural Estimation. In *International Conference on Machine Learning*, pp. 531–540. PMLR, 2018.

Homanga Bharadhwaj, Mohammad Babaeizadeh, Dumitru Erhan, and Sergey Levine. Information Prioritization Through Empowerment in Visual Model-Based RL. In *International Conference on Learning Representations*, 2022.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by Random Network Distillation. In *International Conference on Learning Representations*, 2019.

Jongwook Choi, Archit Sharma, Honglak Lee, Sergey Levine, and Shixiang Shane Gu. Variational Empowerment as Representation Learning for Goal-Based Reinforcement Learning. In *Inter-*

national Conference on Machine Learning, volume 139 of *Proceedings of Machine Learning Research*. PMLR, 2021.

Peter Dayan. Improving Generalization for Temporal Difference Learning: The Successor Representation. *Neural Computation*, 5(4):613–624, 1993.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity Is All You Need: Learning Skills Without a Reward Function. In *International Conference on Learning Representations*, 2019.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov. Contrastive Learning as Goal-Conditioned Reinforcement Learning. In *Neural Information Processing Systems*, volume 35, pp. 35603–35620, 2022.

Scott Fujimoto and Shixiang Shane Gu. A Minimalist Approach to Offline Reinforcement Learning. In *Neural Information Processing Systems*, volume 34, pp. 20132–20145, 2021.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing Function Approximation Error in Actor-Critic Methods. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1587–1596. PMLR, 2018.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational Intrinsic Control. In *International Conference on Learning Representations*, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning With a Stochastic Actor. In *International Conference on Machine Learning*, pp. 1861–1870. Pmlr, 2018.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic Algorithms and Applications. aXiv:1812.05905, 2019. URL https://arxiv.org/abs/1812.05905.

Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. EMI: Exploration With Mutual Information. In *International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3360–3369. PMLR, 2019.

Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. Empowerment: A Universal Agent-Centric Measure of Control. In *IEEE Congress on Evolutionary Computation*, volume 1, pp. 128–135. IEEE, 2005.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline Reinforcement Learning With Implicit Q-Learning. In *International Conference on Learning Representations*, 2022.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-Learning for Offline Reinforcement Learning. In *Neural Information Processing Systems*, volume 33, 2020.

Navneet Madhu Kumar. Empowerment-Driven Exploration Using Mutual Information Estimation. arXiv:1810.05533, 2018.

Qiyang Li, Jason Zhang, Dibya Ghosh, Amy Zhang, and Sergey Levine. Accelerating Exploration With Unlabeled Prior Data. In *Neural Information Processing Systems*, volume 36, pp. 67434–67458, 2023.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous Control With Deep Reinforcement Learning. In *International Conference on Learning Representations*, 2016.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations*, 2023.

Shakir Mohamed and Danilo Jimenez Rezende. Variational Information Maximisation for Intrinsically Motivated Reinforcement Learning. In *Neural Information Processing Systems*, volume 28, 2015.

Vivek Myers, Evan Ellis, Sergey Levine, Benjamin Eysenbach, and Anca Dragan. Learning to Assist Humans Without Inferring Rewards. In *Neural Information Processing Systems*, volume 37, pp. 71540–71567, 2024.

Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-QL: Calibrated Offline RL Pre-Training for Efficient Online Fine-Tuning. In *Neural Information Processing Systems*, volume 36, pp. 62244–62269, 2023.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking Offline Goal-Conditioned RL. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025.

Ahmad Parsian and Syed N. U. A. Kirmani. Estimation Under LINEX Loss Function. In Aman Ullah, A. T. K. Wan, and Anoop Chaturvedi (eds.), *Handbook of Applied Econometrics and Statistical Inference*, pp. 53–76. Marcel Dekker, 2002.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On Variational Bounds of Mutual Information. In *International Conference on Machine Learning*, pp. 5171–5180. PMLR, 2019.

Moritz Schneider, Robert Krug, Narunas Vaskevicius, Luigi Palmieri, Michael Volpp, and Joschka Boedecker. Information-Theoretic Policy Pre-Training With Empowerment. arXiv:2510.05996, 2025.

Archit Sharma, Michael Ahn, Sergey Levine, Vikash Kumar, Karol Hausman, and Shixiang Gu. Emergent Real-World Robotic Skills via Unsupervised Off-Policy Reinforcement Learning. In *Robotics: Science and Systems*, 2020a.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-Aware Unsupervised Discovery of Skills. In *International Conference on Learning Representations*, 2020b.

Jiaming Song and Stefano Ermon. Understanding the Limitations of Variational Mutual Information Estimators. In *International Conference on Learning Representations*, 2020.

Jinyeop Song, Jeff Gore, and Max Kleiman-Weiner. Estimating the Empowerment of Language Model Agents. In *International Conference on Machine Learning*, 2026.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. aXiv:1611.04717, 2017. URL https://arxiv.org/abs/1611.04717.

Ikechukwu Uchendu, Ted Xiao, Yao Lu, Banghua Zhu, Mengyuan Yan, Joséphine Simon, Matthew Bennice, Chuyuan Fu, Cong Ma, Jiantao Jiao, Sergey Levine, and Karol Hausman. Jump-Start Reinforcement Learning. In *International Conference on Machine Learning*, volume 202, pp. 34556–34583, 2023.

Max Wilcoxson, Qiyang Li, Kevin Frans, and Sergey Levine. Leveraging Skills From Unlabeled Prior Data for Efficient Online Exploration. In *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 66833–66860. PMLR, 2025.

Rui Zhao, Yang Gao, Pieter Abbeel, Volker Tresp, and Wei Xu. Mutual Information State Intrinsic Control. In *International Conference on Learning Representations*, 2021.

Chongyi Zheng, Jens Tuyls, Joanne Peng, and Benjamin Eysenbach. Can a MISL Fly? Analysis and Ingredients for Mutual Information Skill Learning. In *International Conference on Learning Representations*, volume 2025, pp. 31876–31903, 2025.

A Appendix

A.1 Hyperparameters

This section shows hyperparameters for our offline empowerment estimator (Section 4). Hyperparameters for the four baselines are given alongside their algorithm descriptions in Section A.6.

Offline Empowerment (Ours). Table 1 lists hyperparameters for the offline empowerment estimator and skill-conditioned policy of Section 4.

Table 1: Hyperparameters for offline empowerment pretraining (ours).

Hyperparameter	Value
Number of skills K	50
Q/V network width / depth	512 / 3
Policy network width / depth	512 / 3
Value latent dimension d	256
Layer normalization	yes
Learning rate (Q/V and policy)	3×10^{-4}
Batch size	1024
Discount γ	0.99
Target network update rate τ	0.005
Clipped linexp threshold t	5
Action noise std σ_a	0.01
Behavioral constraint coefficient α_{bc}	0.001
Future-state samples per update	1
Training steps	10^6

A.2 Ablations

This section ablates three hyperparameters of our method: the policy's action noise scale σ_a (Sections 3 and 4.4), the behavioral constraint coefficient α_{bc} (Section 4.2), and the weight β of the online empowerment bonus (Section 5.2). For σ_a and α_{bc}, we sweep the hyperparameter, holding everything else at the values in Table 1, and track the validation empowerment estimate during training.

A.2.1 Action Noise Scale

Fig. 5 sweeps σ_a over three orders of magnitude, $\sigma_a \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$, on *antmaze-medium-navigate-v0*. The converged empowerment estimate is highly robust to the action noise scale.

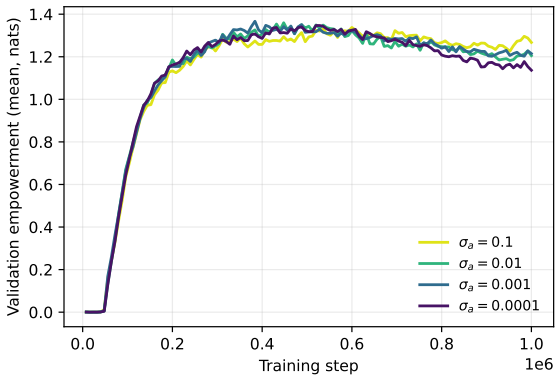


Figure 5: Validation empowerment over training for four action noise scales σ_a, on *antmaze-medium-navigate-v0*. The converged empowerment is robust to σ_a; all other hyperparameters are fixed at the values in Table 1.

A.2.2 Behavioral Constraint Coefficient

Fig. 6 sweeps the behavioral constraint coefficient $\alpha_{\text{bc}} \in \{3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}, 10^{-2}\}$ on *antsoccer-arena-stitch-v0*. Because α_{bc} directly weights how strongly the policy is pulled toward the offline behavior policy, a value that is too large prevents the policy from ever departing from the dataset's own behavior far enough to realize the higher-empowerment behaviors. Conversely, α_{bc} cannot be too low as it must regularize the skill-conditioned policy to the offline data.

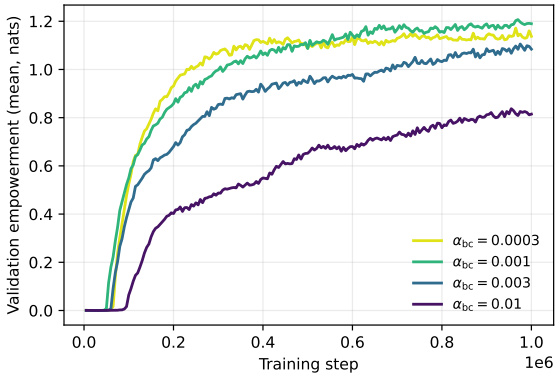


Figure 6: Validation empowerment over training for four behavioral constraint coefficients α_{bc}, on *antsoccer-arena-stitch-v0*. Empowerment is roughly flat until $\alpha_{\text{bc}} = 10^{-2}$, where it collapses.

Qualitative failure mode. Fig. 7 shows the empowerment map for two checkpoints on *antsoccer-arena-stitch-v0* that differ only in α_{bc}. At $\alpha_{\text{bc}} = 10^{-3}$, empowerment peaks tightly around the ball, matching the desired pattern in Fig. 1a. At $\alpha_{\text{bc}} = 10^{-2}$, the map instead shows a distinctive *hole* exactly at the ball. This is the collapse from Fig. 6 made spatially concrete: on the stitch dataset, the offline trajectories that pass near the ball are dominated by a single behavior—dribbling it—so a heavily BC-regularized policy is pulled toward dribbling near the ball regardless of skill, and never discovers the alternative high-empowerment behaviors (e.g. deliberately moving away from the ball) that would make the skill distinguishable. In general, we find that there is around a single order of magnitude range in which the BC coefficient should lie to give good empowerment estimates.

Notably, this failure mode is specific to the stitch dataset; since the *navigate* dataset contains trajectories of the ant doing many behaviors near the ball, a much wider range of BC values works well. On low-coverage data such as *stitch*, α_{bc} must be kept low enough so that the policy can generalize beyond the offline data and learn empowerment-maximizing skills.

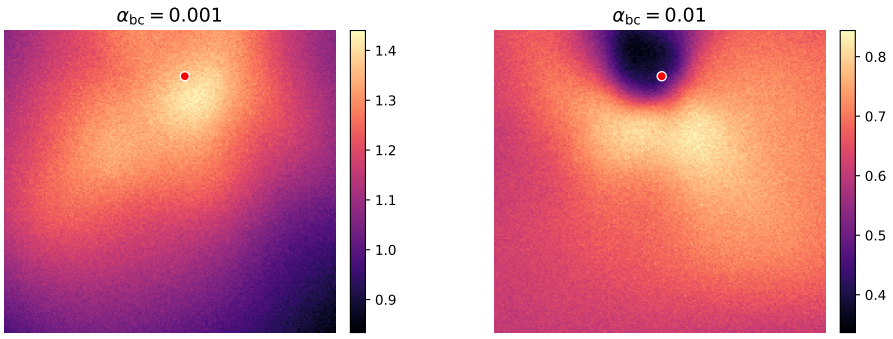


Figure 7: Empowerment maps on *antsoccer-arena-stitch-v0* at matched ball position, for $\alpha_{\text{bc}} = 10^{-3}$ (left) vs. $\alpha_{\text{bc}} = 10^{-2}$ (right). Heavy BC regularization produces a hole in empowerment exactly at the ball, where the offline data is dominated by a single (dribbling) behavior.

A.2.3 Empowerment Bonus Weight

Fig. 8 varies the weight β of the empowerment bonus in the policy objective of Eq. (17), comparing with CRL ($\beta = 0$). We sweep $\beta \in \{3, 10, 30\}$ on *cube-single* (*play*) and *antmaze-medium* (*navigate*). On *cube-single*, performance is non-monotonic in β: $\beta = 10$ performs best, while $\beta = 3$ improves only slightly over CRL. On *antmaze-medium*, every β improves over CRL. In general, like other exploration methods, β must be chosen to encourage the policy to visit high-empowerment states without dominating the policy objective.

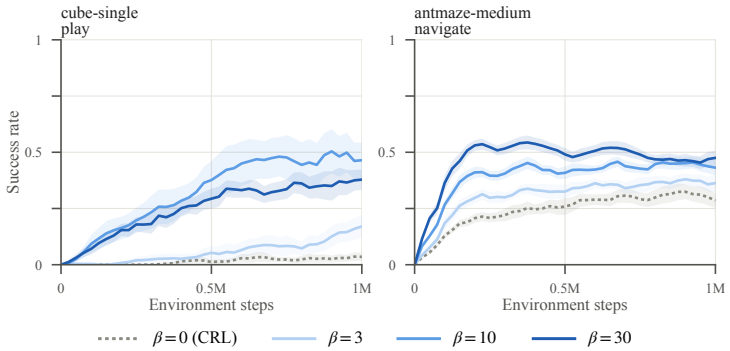


Figure 8: Online success rate of CRL + Empowerment for different weights β of the empowerment bonus, on each task (top line of each panel title) with each offline dataset (bottom line); $\beta = 0$ corresponds to CRL. Lines show the mean over 5 seeds and shaded regions show 95% confidence intervals, smoothed as in Fig. 4.

A.3 Online Exploration Details

This section gives implementation details for the online experiments of Section 5.2.

A.3.1 Online Environments

All online tasks are goal-conditioned variants of OGBench environments (Park et al., 2025) with a single fixed start state and a fixed set of goals. At the start of every episode, one goal is drawn uniformly from the set and given to the agent, and neither the start nor the goal is perturbed by noise. The reward is sparse: the agent receives no signal until it reaches the goal, and an evaluation episode counts as a success if the agent reaches the goal within the episode horizon. Table 2 summarizes the tasks, and the offline datasets used for RLPD and for pretraining the empowerment estimator. In Fig. 4, each curve's mean and 95% confidence interval are smoothed with a trailing moving average over 4 consecutive evaluations.

- **antmaze-medium:** The ant starts in the bottom-left corner of OGBench's medium maze, and the goal is the center of any of the 25 other free cells of the maze, so most goals require navigating several corridors and turns.
- **pointmaze-teleport:** A point mass starts in the bottom-left corner of the teleport maze, and the goal is the center of any of 41 other free cells. The maze contains teleporters that send the agent to a random exit; we exclude the two teleporter entrances, where the agent cannot remain inside the goal region, and one walled-off pocket that can only be entered by a lucky teleport and cannot be left.
- **antsoccer-arena:** The ant starts in the open arena about three units behind the ball, so it must first walk to the ball and then push it to one of eight goals, each one to three units beyond the ball's start on the side away from the ant. The episode horizon is 500 steps.
- **cube-single:** A single cube starts at a fixed position on the table, and the goal is one of 30 positions on a 5×6 grid covering the table. The closest goals are just outside the success radius of the start, so every task requires picking up or pushing the cube. The episode horizon is 200 steps.

Table 2: Online tasks and the offline datasets $\mathcal{D}$ paired with them.

Task	Goals	Horizon	Offline datasets
antmaze-medium	25	1000	navigate, stitch, explore
pointmaze-teleport	41	1000	navigate, stitch
antsoccer-arena	8	500	navigate, stitch
cube-single	30	200	play, noisy

A.3.2 AGENTS

Training loop. All online agents interact with a single environment. After an initial warm-up period, the agent collects a fixed number of environment steps and then performs a fixed number of gradient updates per environment step, on batches drawn from the online replay buffer and the offline dataset $\mathcal{D}$ in equal proportion (RLPD). For both sources, CRL goals are future states of the same trajectory, sampled at a geometrically distributed offset.

Empowerment bonus. The estimator $\hat{\mathcal{E}}$ is the pretrained, frozen model of Section 4.3, trained on the same dataset $\mathcal{D}$ that RLPD uses. We evaluate $\hat{\mathcal{E}}(s)$ once for every state of $\mathcal{D}$ before training, and once for every new online state in a single batched call before each round of updates. The regression target in Eq. (16) is shifted by the mean of $\hat{\mathcal{E}}$ over $\mathcal{D}$; this shifts B_ω by a constant and leaves the policy gradient unchanged, but keeps the regression well conditioned. B_ω has two heads, both trained with Eq. (16), and the policy uses their minimum. Transitions of an online episode that has not yet ended are excluded from the regression until the episode ends and its target is known; they are used by the CRL critic and the policy as usual.

RND. The target and predictor networks share one architecture and take the next state s' as input, normalized by a running mean and standard deviation of online states and clipped to $[-5, 5]$. The intrinsic reward is the squared error between the predictor and target outputs, divided by a running standard deviation of the discounted intrinsic return. Q_{RND} is trained with a twin-critic Bellman backup (Fujimoto et al., 2018) that does not stop at episode ends, as in Burda et al. (2019). The predictor, the running statistics, Q_{RND}, and the policy's RND bonus all use online transitions only.

CRL + RND + Empowerment. The two bonuses are trained independently exactly as above, with their own weights β and β_{RND}.

SUPE. We follow the implementation of Wilcoxson et al. (2025). The skill model is an OPAL-style VAE (Ajay et al., 2021) with a continuous latent, trained on $\mathcal{D}$ over segments of $H = 4$ steps (8 for soccer) and then frozen. The high-level policy is a tanh-Gaussian SAC actor over the latent, with an ensemble of 10 layer-normalized (Ba et al., 2016) critics. Each offline segment inside a single trajectory is labeled with the posterior mean of its latent. Following SUPE's treatment of sparse $-1/0$ tasks, at every update its reward is set to the dataset minimum plus an RND bonus, and its termination is set to the prediction of a reward model trained on online data. We match the gradient budget of the CRL agents.

SUPE + Empowerment. $B_\omega(s, u)$ has the same architecture as for CRL, takes the high-level state and skill u as input, and is regressed onto the same targets as in Eq. (16). The high-level actor loss becomes $\alpha_{\text{ent}} \log \pi(u \mid s, g) - \bar{Q}(s, u, g) - \beta\, B_\omega(s, u)$, where $\bar{Q}$ is the mean over the critic ensemble.

Hyperparameter selection. For each task and dataset, we perform a sweep over the weight of each exploration bonus and report the best-performing setting: $\beta \in \{3, 10, 30\}$ for the empowerment bonus and $\beta_{\text{RND}} \in \{0.3, 1, 3\}$ for RND. For SUPE + Empowerment, we simply use the best empowerment weight identified in the CRL sweep. For CRL + RND + Empowerment, we evaluate two configurations: the best weights from the individual sweeps and both weights halved. Table 5 lists the weights used for each task; all other hyperparameters are shared across tasks (Tables 3 and 4).

Table 3: Hyperparameters shared by all online agents (CRL + RLPD).

Hyperparameter	Value
Actor network width / depth	512 / 3
Critic network width / depth	512 / 3
Critic latent dimension	512
Layer normalization	yes
Learning rate	3×10^{-4}
Batch size	1024
Offline fraction of each batch (RLPD)	0.5
Discount γ	0.99
Target entropy	$-\frac{1}{2}\dim(\mathcal{A})$
Environment steps per update round	50
Updates per environment step	1
Warm-up environment steps	1000
Replay buffer size	10^6
Total environment steps	10^6

Table 4: Hyperparameters of the exploration bonuses.

Hyperparameter	Value
Empowerment bonus (ours)	
B_ω network width / depth	512 / 3 (two heads)
Bonus weight β	per task (Table 5)
Future-state samples per skill for $\hat{\mathcal{E}}$	64
RND	
Target / predictor network width / depth	512 / 2
Embedding dimension	512
Predictor learning rate	10^{-4}
Q_{RND} network width / depth	512 / 3 (two heads)
Q_{RND} discount	0.99
Q_{RND} target update rate τ	0.005
Bonus weight β_{RND}	per task (Table 5)
SUPE	
Skill horizon H	4 (8 on antsoccer-arena)
Skill latent dimension	8
OPAL KL coefficient	0.1 (0.2 on cube-single)
Number of critics / critics in target minimum	10 / 1 (10 / 2 on cube-single)
Discount	0.995
RND coefficient	8
Warm-up environment steps	5000

Table 5: Bonus weights used for each task and dataset in Fig. 4.

Task	Dataset	CRL + Emp. β	CRL + RND β_{RND}	CRL + RND + Emp. β / β_{RND}	SUPE + Emp. β
cube-single	play	10	1	10 / 1	10
cube-single	noisy	30	1	30 / 1	30
antmaze-medium	navigate	10	3	10 / 3	10
antmaze-medium	stitch	10	3	10 / 3	10
antmaze-medium	explore	10	3	10 / 3	10
pointmaze-teleport	navigate	30	1	15 / 0.5	30
pointmaze-teleport	stitch	10	0.3	10 / 0.3	10
antsoccer-arena	navigate	10	1	10 / 1	10
antsoccer-arena	stitch	10	1	10 / 1	10

A.4 Compute Resources

All offline pretraining runs were trained on an internal cluster of NVIDIA A5000 GPUs, with a single GPU per run.

A.5 Clipped Linear Exponential Loss

Because Q and V operate in log-space (Section 4), we regress in the exponentiated domain for numerical stability. For a target y and prediction $\hat{y}$, we regress $e^{-\hat{y}}$ to κe^{-y} (for a target scale κ) using

$$\ell_{\text{linexp}}(\hat{y}, y;\, \kappa) \;=\; \begin{cases} \kappa\, e^{\hat{y}-y} \;-\; \hat{y}, & \hat{y} - y < t, \\ \big(\kappa e^{t} - 1\big)\big(\hat{y} - y - t\big) \;+\; \kappa e^{t} \;-\; (y + t), & \hat{y} - y \geq t. \end{cases} \tag{18}$$

The second branch is simply a linearization of the loss about the point $\hat{y} = y + t$, applied whenever $\hat{y} - y$ exceeds the threshold $t = 5$, which prevents the exponential term from producing exploding gradients.

A.6 Baseline Algorithm Details

This section gives full derivations, network architectures, and hyperparameters for the four baselines summarized in Section 5.1. All four baselines are trained on triples $(s, a, s^+) \sim \mathcal{D}$, where s^+ is a future state of s on the same dataset trajectory, drawn at a geometrically distributed offset as in the definition of the discounted state occupancy (Section 3). This differs from our method, which draws s^+ uniformly from $\mathcal{D}$ and recovers the dependence on (s, a) through TD backups. With the exception of DADS-Empowerment (Section A.6.4), the baselines estimate the *action empowerment*

$$I(s^+; a \mid s) \triangleq \mathbb{E}_{a\sim\pi_\beta(\cdot|s),\, s^+\sim p(\cdot|s,a)}\left[\log \frac{p(s^+ \mid s, a)}{p(s^+ \mid s)}\right], \tag{19}$$

that is, the mutual information between the primitive action $a \sim \pi_\beta(\cdot \mid s)$ actually taken at s under the behavior policy π_β and the resulting future state s^+, in place of the mutual information between a skill z and s^+ used by our method.

A.6.1 CF-Empowerment

Overview. CF-Empowerment estimates the action empowerment $I(s^+; a \mid s)$ by first fitting two contrastive critics that share a future-state encoder, then distilling the resulting pointwise mutual-information estimator into a small feedforward network that can be evaluated at any state without re-running the contrastive critics.

Architecture. Three MLP trunks map into a shared latent space of dimension d: an action-conditioned encoder $\phi(s, a)$, a state-only encoder $\phi'(s)$, and a future-state encoder $\psi(s^+)$, all mapping into $\mathbb{R}^d$. A separate empowerment network $E_\theta(s) \in \mathbb{R}$ is a feedforward MLP that only depends on the current state. There is no policy network.

Training objective. Given a minibatch of B triples $\{(s_i, a_i, s_i^+)\}_{i=1}^B$, define the bilinear logits

$$\ell^a_{ij} = \frac{\phi(s_i, a_i)^\top \psi(s_j^+)}{\sqrt{d}}, \qquad \ell^s_{ij} = \frac{\phi'(s_i)^\top \psi(s_j^+)}{\sqrt{d}}. \tag{20}$$

The critics are trained with a binary NCE objective (Eysenbach et al., 2022), treating s_i^+ as the positive for (s_i, a_i) (resp. s_i) and every other s_j^+ in the batch as a negative:

$$\mathcal{L}_{\text{contrastive}} = \sum_{i,j=1}^{B} \Big[-\mathbf{1}\{i=j\}\log\sigma(\ell^a_{ij}) - \mathbf{1}\{i\neq j\}\log\big(1-\sigma(\ell^a_{ij})\big)\Big] + \Big[\text{same expression with } \ell^s_{ij}\Big]. \tag{21}$$

At the optimum, $\ell^a(s, a, s^+) \approx \log p(s^+ \mid s, a) - \log p(s^+) + c$ and $\ell^s(s, s^+) \approx \log p(s^+ \mid s) - \log p(s^+) + c$ for the same constant c, so $T(s, a, s^+) \triangleq \ell^a(s, a, s^+) - \ell^s(s, s^+)$ estimates the conditional mutual-information integrand $\log p(s^+ \mid s, a) - \log p(s^+ \mid s)$. After a fixed number

of pretraining steps, the critics' parameters are frozen and the empowerment network is trained by regression onto T:

$$\mathcal{L}_{\text{distill}} = \mathbb{E}_{(s,a,s^+)\sim\mathcal{D}}\left[\left(E_\theta(s) - T(s,a,s^+)\right)^2\right]. \tag{22}$$

At convergence $E_\theta(s) \to \mathbb{E}_{(a,s^+)\sim p(a,s^+|s)}[T(s,a,s^+)] = I(s^+; a \mid s)$, i.e., the network converges to an estimate of the action empowerment at s.

Offline adaptation. In the online setting, the negatives for the contrastive critics would be drawn by rolling out many actions from the same state; here they are instead the other (s_j, a_j, s_j^+) triples already present in the offline minibatch (in-batch negatives), and a ranges only over the fixed offline behavior policy rather than an arbitrary exploration policy.

Hyperparameters. Learning rate 10^{-3}; batch size 512; critic hidden dims $(256, 256)$ with layer normalization; latent dimension $d = 256$; empowerment network hidden dims $(256, 256)$; contrastive pretraining for 5×10^5 steps before switching to distillation.

A.6.2 CF-FlowBC-Empowerment

Overview. CF-FlowBC-Empowerment also estimates the action empowerment $I(s^+; a \mid s)$, but replaces the in-batch negatives of CF-Empowerment with negatives drawn from a learned behavior-cloned policy, and replaces the bilinear critic with a single scalar critic $T(s^+, a, s)$ trained with an explicit InfoNCE lower bound.

Architecture. Two flow-matching models (Lipman et al., 2023) are trained on the offline data: a behavior-cloned policy $\pi_{\text{bc}}(a \mid s)$ and a dynamics flow $f_{\text{dyn}}(s^+ \mid s, a)$ modeling the discounted future state. Both share the same velocity-field MLP architecture $v_\theta(x_t, t, \text{cond})$, conditioned on s (for π_{bc}) or on (s, a) (for f_{dyn}); samples are drawn by Euler-integrating the learned velocity field for 8 steps from Gaussian noise, with sampled actions clipped to the action bounds. A separate scalar critic $T(s^+, a, s)$, an MLP over the concatenation of its three inputs, provides the mutual-information estimate.

Training objective. Both flow models are trained with the standard conditional flow-matching loss,

$$\mathcal{L}_{\text{flow}}(\text{cond}, x) = \mathbb{E}_{x_0\sim\mathcal{N}(0,I),\, t\sim\text{Unif}(0,1)}\left[\left\|v_\theta\big((1-t)x_0 + tx,\, t,\, \text{cond}\big) - (x - x_0)\right\|^2\right], \tag{23}$$

instantiated with $(\text{cond}, x) = (s, a)$ for π_{bc} and $(\text{cond}, x) = ((s, a), s^+)$ for f_{dyn}. Given a positive triple (s, a, s^+), N negative actions $a_1, \ldots, a_N \overset{\text{iid}}{\sim} \pi_{\text{bc}}(\cdot \mid s)$ are sampled and the critic is trained to maximize the categorical InfoNCE bound

$$b(s, a, s^+) = \log(N{+}1) + \log \frac{e^{T(s^+,a,s)}}{e^{T(s^+,a,s)} + \sum_{i=1}^N e^{T(s^+,a_i,s)}}, \tag{24}$$

$$\mathcal{L}_{\text{critic}} = \log(N{+}1) - \mathbb{E}\left[b(s, a, s^+)\right]. \tag{25}$$

The reported estimate $\hat{I}(s^+; a \mid s) = \log(N{+}1) - \mathcal{L}_{\text{critic}}$ is therefore capped at $\log(N{+}1)$ nats, a known finite-sample ceiling of the InfoNCE bound (Poole et al., 2019). To evaluate the empowerment at an arbitrary state s (rather than at a specific offline triple), M joint samples $a \sim \pi_{\text{bc}}(\cdot \mid s)$, $s^+ \sim f_{\text{dyn}}(\cdot \mid s, a)$ are drawn and scored against fresh π_{bc} negatives via the same bound and averaged. Because this substitutes π_{bc} and f_{dyn} for the true behavior policy and environment dynamics, the resulting estimate is the mutual information under the *learned models* rather than the true environment, and errors in either model bias it.

Offline adaptation. π_{bc} substitutes for the ability to roll out arbitrary actions from a live policy, and f_{dyn} substitutes for stepping the environment; both are trained on the offline dataset with their own flow-matching losses, and no gradient from the critic flows into them.

Hyperparameters. Learning rate 3×10^{-4}; batch size 256; flow and critic hidden dims $(512, 512, 512)$ with layer normalization; 8 Euler integration steps; $N = 7$ negatives; 10^5 steps of π_{bc}/f_{dyn}-only pretraining before the critic begins training.

A.6.3 DV-Empowerment

Overview. DV-Empowerment is architecturally identical to CF-FlowBC-Empowerment—the same π_{bc}, f_{dyn}, and scalar critic $T(s^+, a, s)$—but replaces the categorical InfoNCE bound with the Donsker–

Varadhan (DV) lower bound (Belghazi et al., 2018), and replaces the per-element action negatives with full (action, next-state) negative pairs drawn from an explicit product-of-marginals construction.

Training objective. Given a positive triple (s, a, s^+), N negative actions $a_1, \dots, a_N \overset{\text{iid}}{\sim} \pi_{\text{bc}}(\cdot \mid s)$ are sampled, each paired with its own simulated future state $s_i^+ \sim f_{\text{dyn}}(\cdot \mid s, a_i)$. To obtain valid samples from the product of marginals $\pi_{\text{bc}}(a \mid s) \times f_{\text{dyn}}(s^+ \mid s, a)$ rather than the joint, the future states are re-paired with a cyclic derangement σ (e.g. $\sigma(i) = (i \bmod N) + 1$), so that $(a_i, s^+_{\sigma(i)})$ never uses the future state that a_i itself generated. The DV objective is then

$$J(s, a, s^+) = T(s^+, a, s) - \log \frac{1}{N} \sum_{i=1}^{N} e^{T(s^+_{\sigma(i)},\, a_i,\, s)}, \qquad \mathcal{L}_{\text{critic}} = -\,\mathbb{E}\left[J(s, a, s^+)\right], \tag{26}$$

with reported estimate $\hat{I}(s^+; a \mid s) = \mathbb{E}[J(s, a, s^+)]$. As with CF-FlowBC-Empowerment, evaluating $\hat{\mathcal{E}}(s)$ at an arbitrary state draws M fresh joint samples from π_{bc} and f_{dyn} and reuses the same derangement-based negative construction.

Offline adaptation. Identical to CF-FlowBC-Empowerment: π_{bc} and f_{dyn}, both fit on the offline dataset, substitute for on-policy action and rollout sampling.

Hyperparameters. Identical to CF-FlowBC-Empowerment: learning rate 3×10^{-4}; batch size 256; flow and critic hidden dims $(512, 512, 512)$ with layer normalization; 8 Euler integration steps; $N = 7$ negatives; 10^5 steps of pretraining.

A.6.4 DADS-Empowerment

Overview. DADS-Empowerment is the only baseline that estimates the *skill* empowerment $I(s^+; z \mid s)$ for a small, fixed-size discrete skill alphabet, matching the formalism used by our method. Rather than fixing a skill-labeling scheme first, as in DIAYN/DADS-style skill discovery (Sharma et al., 2020b), and only then measuring the resulting mutual information, DADS-Empowerment learns the skill assignment *itself* so as to directly maximize the Barber–Agakov (BA) variational bound (Barber & Agakov, 2003) on offline (s, s^+) pairs.

Architecture. Three networks operate on a fixed number K of discrete, one-hot skills. An encoder $\rho(z \mid s, s^+)$ is an MLP that maps the concatenation of s and the observed transition delta $s^+ - s$ to a softmax distribution over the K skills; conditioning only on the endpoint pair (rather than any intermediate trajectory) ensures z cannot encode information that never manifests in s^+ itself. A transition channel $q(s^+ \mid s, z)$ is a diagonal-Gaussian mixture with 8 components conditioned on s and a one-hot z, modeling the delta $s^+ - s$. A skill marginal $m(z \mid s)$ is an MLP over s producing a softmax over the K skills. There is no action-conditioned network of any kind, and no policy: the primitive action never appears in this baseline's training objective.

Training objective. For every offline pair (s, s^+) and every skill $z \in \{1, \dots, K\}$ (the expectation over z is computed exactly, not by Monte Carlo, since K is small), define the BA bracket

$$\text{bracket}(z; s, s^+) = \log q(s^+ \mid s, z) - \log \sum_{z'=1}^{K} m(z' \mid s)\, q(s^+ \mid s, z'), \tag{27}$$

where the log-sum-exp over z' uses a skill marginal m. The encoder loss maximizes the bracket under its own soft skill assignment:

$$\mathcal{L}_{\text{bracket}} = -\,\mathbb{E}_{(s,s^+)\sim\mathcal{D}}\left[\textstyle\sum_{z=1}^{K} \rho(z \mid s, s^+)\, \text{bracket}(z; s, s^+)\right], \tag{28}$$

and the reported estimate is $\hat{I}(s^+; z \mid s) = -\mathcal{L}_{\text{bracket}}$. The skill marginal $m(z \mid s)$ is separately fit by cross-entropy toward the current (gradient-stopped) soft assignment,

$$\mathcal{L}_{\text{marginal}} = -\,\mathbb{E}_{(s,s^+)\sim\mathcal{D}}\left[\textstyle\sum_{z=1}^{K} \rho(z \mid s, s^+)\, \log m(z \mid s)\right], \tag{29}$$

so that m tracks ρ's marginal usage of each skill without corrupting ρ's own gradient. A skill-usage regularizer, weighted by a coefficient λ_{kl},

$$\mathcal{L}_{\text{usage}} = -\log K - \frac{1}{K} \sum_{z=1}^{K} \log \bar{\rho}_z = D_{\text{KL}}\big(\mathcal{U}_K \,\big\|\, \bar{\rho}\big), \qquad \bar{\rho}_z \triangleq \mathbb{E}_{(s,s^+)\sim\mathcal{D}}\left[\rho(z \mid s, s^+)\right], \tag{30}$$

where $\mathcal{U}_K$ is the uniform distribution over the K skills. This term penalizes deviation of the average skill usage from uniform, discouraging the assignment from collapsing onto a small subset of skills. The total loss is $\mathcal{L}_{\text{bracket}} + \mathcal{L}_{\text{marginal}} + \lambda_{\text{kl}}\mathcal{L}_{\text{usage}}$.

Offline adaptation. As with the other three baselines, (s, s^+) pairs are drawn from the offline dataset; no skill-conditioned rollouts, dynamics-model rollouts, or EM alternation are used.

Hyperparameters. Learning rate 3×10^{-4}; batch size 1024; mixture-density network with 8 components; encoder, channel, and marginal networks all use hidden dims $(512, 512, 512)$ with layer normalization. We use $\lambda_{\text{kl}} = 1$, including for the qualitative maps in Section 6.2.