Flattening Hierarchies with Policy Bootstrapping

John L. Zhou, Jonathan C. Kao
`john.ly.zhou@gmail.com, kao@seas.ucla.edu`

University of California, Los Angeles

Abstract

Offline goal-conditioned reinforcement learning (GCRL) is a promising approach for pretraining generalist policies on large datasets of reward-free trajectories, akin to the self-supervised objectives used to train foundation models for computer vision and natural language processing. However, scaling GCRL to longer horizons remains challenging due to the combination of sparse rewards and discounting, which obscures the comparative advantages of primitive actions with respect to distant goals. Hierarchical GCRL methods achieve strong empirical results on long-horizon goal-reaching tasks, but their reliance on modular, timescale-specific policies and subgoal generation introduces significant additional complexity and hinders scaling to high-dimensional goal spaces. In this work, we introduce an algorithm to train a flat (non-hierarchical) goal-conditioned policy by bootstrapping on subgoal-conditioned policies with advantage-weighted importance sampling. Our approach eliminates the need for a generative model over the (sub)goal space, which we find is key for scaling to high-dimensional control in large state spaces. We further show that existing hierarchical and bootstrapping-based approaches correspond to specific design choices within our derivation. Across a comprehensive suite of state- and pixel-based locomotion and manipulation benchmarks, our method matches or surpasses state-of-the-art offline GCRL algorithms and scales to complex, long-horizon tasks where prior approaches fail.

1 Introduction

Goal-conditioned reinforcement learning (GCRL) specifies tasks by desired outcomes, alleviating the burden of defining reward functions over the state-space and enabling the training of general policies capable of achieving a wide range of goals. Offline GCRL extends this paradigm to leverage existing datasets of reward-free trajectories and has been likened to the simple self-supervised objectives that have been successful in training foundation models for other areas of machine learning (Yang et al., 2023; Park et al., 2024a). However, the conceptual simplicity of GCRL belies practical challenges in learning accurate value functions and, consequently, effective policies for goals requiring complex, long-horizon behaviors. These limitations call into question its applicability as a general and scalable objective for learning *foundation policies* (Black et al., 2024; Park et al., 2024c; Physical Intelligence et al., 2025) that can be efficiently adapted to a diverse array of control tasks.

Hierarchical reinforcement learning (HRL) is commonly used to address these challenges and is particularly well-suited to the recursive subgoal structure of goal-reaching tasks, where reaching distant goals entails first passing through intermediate subgoal states. Goal-conditioned HRL exploits this structure by learning a hierarchy composed of multiple levels: one or more high-level policies, tasked with generating intermediate subgoals between the current state and the goal; and a low-level actor, which operates over the primitive action space to achieve the assigned subgoals. These approaches have achieved state-of-the-art results in both online (Nachum et al., 2018; Levy et al., 2018) and offline GCRL (Park et al., 2023), and are especially effective in long-horizon tasks.

However, despite the strong empirical performance of HRL, it suffers from major limitations as a scalable pretraining strategy. In particular, the modularity of hierarchical policy architectures, fixed to specific levels of temporal abstraction, precludes unified task representations and necessitates learning a generative model over the subgoal space to interface between policy levels.

Learning to predict intermediate goals in a space that may be as high-dimensional as the raw observations poses a difficult generative modeling problem. To ensure that subgoals are physically realistic and reachable in the allotted time, previous work often implements additional processing and verification of proposed subgoals (Zhang et al., 2020; Czechowski et al., 2021; Zawalski et al., 2022; Hatch et al., 2024). An alternative is to instead predict in a compact learned latent subgoal space, but simultaneously optimizing subgoal representations and policies results in a nonstationary input distribution to the low-level actor, which can slow and destabilize training (Vezhnevets et al., 2017; Levy et al., 2018). The choice of objective for learning such representations, ranging from autoregressive prediction (Seo et al., 2022; Zeng et al., 2023) to metric learning (Tian et al., 2020; Nair et al., 2023; Ma et al., 2023), remains an open question and adds significant complexity to the design and tuning of hierarchical methods.

Following the tantalizing promise that flat, one-step policies can be optimal in fully observable, Markovian settings (Puterman, 2005), this work aims to isolate the core advantages of hierarchies for offline GCRL and distill them into a simpler training recipe for a single, unified policy. We begin our empirical analysis by revisiting a state-of-the-art hierarchical method for offline GCRL that significantly outperforms previous approaches on a range of long-horizon goal-reaching tasks. Beyond the original explanation based on improved value function signal-to-noise ratio, we find that separately training a low-level policy on nearby subgoals improves sampling efficiency. We reframe this hierarchical approach as a form of implicit test-time bootstrapping on subgoal-conditioned policies, revealing a conceptual connection to earlier methods that learn subgoal generators and bootstrap directly from subgoal-conditioned policies to train a flat, unified goal-conditioned policy.

Building on these insights, we present an inference-based theoretical framework that unifies these ideas and yields **Subgoal Advantage-Weighted Policy Bootstrapping (SAW)**, a novel policy extraction objective for offline GCRL. SAW uses advantage-weighted importance sampling to bootstrap on subgoals sampled directly from data, capturing the long-horizon strengths of hierarchies in a single, flat policy *without* requiring a generative subgoal model. In evaluations across 20 state- and pixel-based offline GCRL datasets, our method matches or surpasses all baselines in diverse locomotion and manipulation tasks and scales especially well to complex, long-horizon tasks, being the only existing approach to achieve nontrivial success in the `humanoidmaze-giant` environment.

2 Related Work

Our work builds on a rich body of literature encompassing goal-conditioned RL (Kaelbling, 1993), offline RL (Lange et al., 2012; Levine et al., 2020), and hierarchical RL (Dayan & Hinton, 1992; Sutton et al., 1999; Jong et al., 2008; Bacon et al., 2017; Vezhnevets et al., 2017). The generality of the GCRL formulation enables powerful self-supervised training strategies such as hindsight relabeling (Andrychowicz et al., 2017; Ghosh et al., 2020) and state occupancy matching (Ma et al., 2022). These are often combined with approaches that exploit the recursive subgoal structure of GCRL: either implicitly via quasimetric learning (Wang et al., 2023), probabilistic interpretations (Hoang et al., 2021; Zhang et al., 2021b), and contrastive learning (Eysenbach et al., 2022; Zheng et al., 2023); or explicitly through hierarchical decomposition into subtasks (Nachum et al., 2018; Levy et al., 2018; Gupta et al., 2020; Park et al., 2023). Despite these advances, learning remains difficult for distant goals due to sparse rewards and discounting over time. Many methods use the key insight that actions which are effective for reaching an intermediate subgoal between the current state and the goal are also effective for reaching the final goal. Such subgoals are typically selected via planning (Huang et al., 2019; Zhang et al., 2021a; Hafner et al., 2022), searching within the replay buffer (Eysenbach et al., 2019), or, most commonly, sampling from generative models. Hierarchical

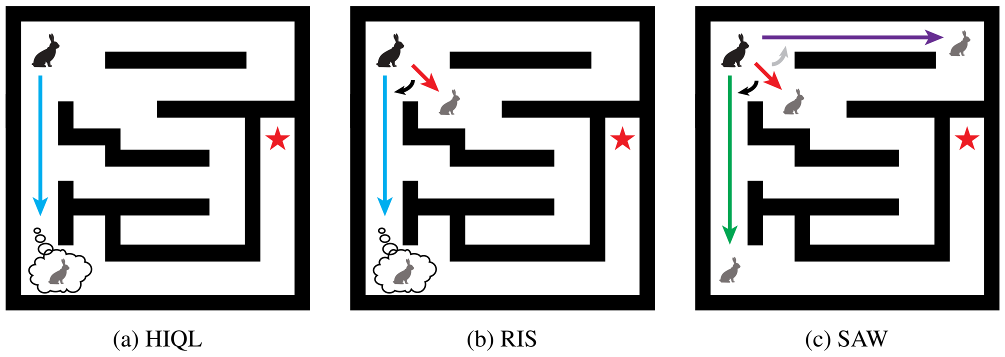

Figure 1: **Learning with subgoals**. Both HIQL and RIS "imagine" subgoals (thought bubbles) en route to the goal (red star) with generative models. However, HIQL samples actions directly from the subgoal-conditioned policy, while RIS regresses (**black** arrow) a flat goal-conditioned policy towards the subgoal-conditioned action distribution during training. SAW also performs regression but only uses "real" subgoals from the dataset $\mathcal{D}$, weighting the regression more heavily towards distributions conditioned on good subgoals and less (**gray** arrow) towards bad ones.

methods in particular generate subgoals during inference and use them to query "subpolicies" trained on shorter horizon goals, which are generally easier to learn (Strehl et al., 2009; Azar et al., 2017). Our method also leverages the ease of training subpolicies to effectively learn long-horizon behaviors, but aims to learn a flat, unified policy while avoiding the complexity of training generative models to synthesize new subgoals.

Policy bootstrapping. Our work is most closely related to Reinforcement learning with Imagined Subgoals (Chane-Sane et al., 2021, RIS), which, to our knowledge, is the only prior work that performs bootstrapping on *policies*, albeit in the online setting. Similar to goal-conditioned hierarchies, RIS learns a generative model to synthesize "imagined" subgoals that lie between the current state and the goal. Unlike HRL approaches, however, it regresses the full-goal-conditioned policy towards the subgoal-conditioned target, treating the latter as a prior to guide learning and exploration [Figure 1]. While RIS yields a flat policy for inference, it still requires the full complexity of a hierarchical policy, including a generative model over the goal space. In contrast, our work extends the core benefits of subgoal-based bootstrapping to *offline* GCRL with an advantage-based importance weight on subgoals sampled from dataset trajectories, eliminating the need for a subgoal generator altogether.

3 Preliminaries

Problem setting: We consider the problem of offline goal-conditioned RL, described by a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $\mathcal{R} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ the goal-conditioned reward function (where we assume that the goal space $\mathcal{G}$ is equivalent to the state space $\mathcal{S}$), and $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ the transition function. In the offline setting, we are given a dataset $\mathcal{D}$ of trajectories $\tau = (s_0, a_0, s_1, a_1, \dots, s_T)$ previously collected by some arbitrary policy (or multiple policies), and must learn a policy that can reach a specified goal state g from an initial state $s_0 \in \mathcal{S}$ without further interaction in the environment, maximizing the objective

$$J(\pi) = \mathbb{E}_{g \sim p(g), \tau \sim p^{\pi}(\tau)} \left[\sum_{t=0}^{\infty} \gamma^t r\left(s_t, g\right) \right], \tag{1}$$

where $p(g)$ is the goal distribution and $p^{\pi}(\tau)$ is the distribution of trajectories generated by the policy π and the transition function $\mathcal{P}$ during (online) evaluation.

Offline value learning: We use a goal-conditioned, action-free variant of implicit Q-learning (Kostrikov et al., 2021, IQL) referred to as goal-conditioned implicit value learning (Park et al., 2023, GCIVL). The original IQL formulation modifies standard value iteration for offline RL by replacing the $\max$ operator with an expectile regression, in order to avoid value overestimation for out-of-distribution actions. GCIVL replaces the state-action value function with a value-only estimator

$$\min_{\psi} \mathcal{L}_{\mathrm{GCIVL}}(\psi) = \mathbb{E}_{s,a,g\sim p^{\mathcal{D}}} \left[\ell_{\tau}^{2} \left(r(s,g) + \gamma \bar{V}\left(s', g\right) - V(s,g) \right) \right], \tag{2}$$

where $\ell_{\tau}^{2}(x) = |\tau - \mathbb{1}(x < 0)|x^{2}$ is the expectile loss parameterized by $\tau \in [0.5, 1)$ and $\bar{V}(\cdot)$ denotes a target value function. Note that GCIVL is optimistically biased in stochastic environments, since it directly regresses towards high-value transitions without using Q-values to marginalize over action-independent stochasticity.

Offline policy extraction: To learn a target subpolicy, we use Advantage-Weighted Regression (Peng et al., 2019, AWR) to extract a policy from a learned value function. AWR reweights state-action pairs according to their exponentiated advantage with an inverse temperature hyperparameter α, via the objective

$$\max_{\pi} \mathcal{J}_{\mathrm{AWR}}(\pi) = \mathbb{E}_{s,a,g\sim\mathcal{D}} \left[e^{\alpha(Q(s,a,g) - V(s,g))} \log \pi(a \mid s, g) \right], \tag{3}$$

thus remaining within the support of the data without requiring an additional behavior cloning penalty.

4 Understanding Hierarchies in Offline GCRL

In this section, we seek to identify the core reasons behind the empirical success of hierarchies in offline GCRL that can be used to guide the design of a simpler training objective for a flat policy. We first review previous explanations for the benefits of HRL and propose an initial algorithm that seeks to capture these benefits in a flat policy, but find that it still fails to close the performance gap to Hierarchical Implicit Q-Learning (Park et al., 2023, HIQL), a state-of-the art method. We then identify an additional practical benefit of hierarchical training schemes and show how HIQL exploits this from a policy bootstrapping perspective.

4.1 Hierarchies in online and offline GCRL

Previous investigations into the benefits of hierarchical RL in the online setting attribute their success to improved exploration (Jong et al., 2008) and training value functions with multi-step rewards (Nachum et al., 2019). They demonstrate that augmenting non-hierarchical agents in this manner can largely close the performance gap to hierarchical policies. However, the superior performance of hierarchical methods in the *offline* GCRL setting, where there is no exploration, calls this conventional wisdom into question.

HIQL is a state-of-the-art hierarchical offline GCRL method that extracts a high-level policy over subgoals and a low-level policy over primitive actions from a single goal-conditioned value function trained using standard *one-step* temporal difference learning. HIQL achieves significant performance gains across a number of complex, long-horizon navigation tasks purely through improvement on the policy extraction side, without needing multi-step rewards to train the value function as done in Nachum et al. (2019). While this does not preclude the potential benefits of multi-step rewards for offline GCRL, it does demonstrate that the advantages of hierarchies are not limited to temporally extended value learning, in line with previous claims that the primary bottleneck in offline RL is policy extraction and not value learning (Park et al., 2024b).

4.2 Value signal-to-noise ratio in offline GCRL

Instead, HIQL addresses a separate “signal-to-noise ratio” (SNR) issue in value functions conditioned on distant goals, where a combination of sparse rewards and discounting makes it nearly impossible to accurately determine the advantage of one primitive action over another with respect to distant goals. By separating policy extraction into two levels, the low-level actor can instead evaluate the relative advantage of actions with respect to nearby subgoals and the high-level policy can utilize multi-step advantage estimates to get a clearer learning signal with respect to distant goals.

To test whether improved SNR in advantage estimates with respect to distant goals is indeed the key to HIQL's superior performance, we propose to utilize subgoals to directly improve advantage estimates in a simple baseline method we term **goal-conditioned waypoint advantage estimation (GCWAE)**. Briefly, we use the advantage of actions with respect to subgoals generated by a high-level policy as an estimator of the undiscounted advantage with respect to the true goal

$$\tilde{A}(s_t, a_t, g) \approx A\left(s_t, a_t, w\right), \tag{4}$$

where $w \sim \pi^h_{\text{sg}}(w \mid s_t, g)$ is a subgoal sampled from a high-level policy π^h, and the sg subscript indicates a stop-gradient operator. Apart from using this advantage to directly train a flat policy with AWR, we use the same architectures, sampling distributions, and training objective for π^h as HIQL. Despite large gains over one-step policy learning objectives in several navigation tasks, GCWAE still underperforms its hierarchical counterpart, achieving a 75% success rate on `antmaze-large-navigate` compared to 90% for HIQL without subgoal representations and 16% for GCIVL with AWR [Appendix H].

4.3 It's easier to find good (dataset) actions for closer goals

While diagnosing this discrepancy, we observed that training statistics for the two methods were largely identical except for a striking difference in the mean action advantage $A(s_t, a_t, w)$. The advantage was significantly lower for GCWAE, which samples "imagined" subgoals $w \sim \pi^h(\cdot \mid s, g)$ from a high-level policy, than HIQL, which samples directly from the k-step future state distribution $w \sim p^{\mathcal{D}}(s_{t+k} \mid s_t)$ of the dataset [Appendix H]. This hints at an obvious but important benefit of training on real trajectories: in most cases, dataset actions are simply *better* with respect to subgoals sampled from the same trajectory than "imagined" subgoals generated by a high-level policy. Similarly, the dataset is far more likely to contain high-advantage actions for goals sampled at the ends of short subsequences, whereas optimal state-action pairs for more distant goals along the trajectory are much rarer, due to the combinatorial explosion of possible goal states as the goal-sampling horizon increases.

The practical benefits of being able to easily sample high-advantage state-action-goal tuples are hinted at in Park et al. (2024a), who pose the question "*Why can't we use random goals when training policies*?" after finding that offline GCRL algorithms empirically perform better when only sampling (policy) goals from future states in the same trajectory as the initial state. While their comparison focuses on in-trajectory versus random goals instead of nearer versus farther in-trajectory goals, we hypothesize both observations are driven by similar explanations.

4.4 Hierarchies perform test-time policy bootstrapping

Our observations suggest that training policies on nearby goals benefits both from better value SNR in advantage estimates and the ease of sampling good state-action-goal combinations. For brevity, we will refer to such policies trained only on goals of a restricted horizon length as "subpolicies," denoted by π^{sub} and analogous to the low-level policies π^{ℓ} in hierarchies. The remainder of this work seeks to answer the question: how do hierarchical methods take advantage of the relative ease of training subpolicies to reach distant goals, and can we use similar strategies to train flat policies?

HIQL separately trains a low-level subpolicy on goals sampled from states at most k steps into the future, reaping all the benefits of policy training with nearby goals. Similar to other goal-conditioned hierarchical methods (Nachum et al., 2018; Levy et al., 2018), it then uses the high-level policy to predict optimal subgoals between the current state and the goal at *test* time, and "bootstraps" by using the subgoal-conditioned action distribution as an estimate for the full goal-conditioned policy.

5 Subgoal Advantage-Weighted Policy Bootstrapping

We now seek to unify the above insights into an objective to learn a single, flat goal-reaching policy *without* the additional complexity of HRL. Following the bootstrapping perspective, a direct analogue to hierarchies would use the subpolicy to construct *training* targets, regressing the full goal-conditioned policy towards a target subpolicy π^{sub} conditioned on the output of a subgoal generator $\pi^{\text{sub}}\left(a \mid s, \pi^h(w \mid s, g)\right)$. This approach, taken by RIS [Figure 1], still inherits the full complexity of hierarchical policies and then some: it requires learning a subgoal generator, a subpolicy, and an additional flat policy.

5.1 Hierarchical RL as inference

To eliminate this additional machinery, we adopt the view of GCRL as probabilistic inference (Levine, 2018). In this framing, the bilevel objectives for HIQL's hierarchical policy and the KL bootstrapping term for RIS's flat policy can be derived from the same inference problem with different choices of variational posterior. Our main insight is that the expectation over generated subgoals can be expressed as an expectation over the dataset distribution with an advantage-based importance weight, yielding our SAW objective. We present an abridged version below and leave the full derivation to Supplementary Section D.

Similar to previous work (Abdolmaleki et al., 2018), we cast the infinite-horizon, discounted GCRL formulation as an inference problem by constructing a probabilistic model via the likelihood function $p\left(U=1 \mid \tau, \{w\}, g\right) \propto \exp\left(\beta \sum_{t=0}^{\infty} \gamma^t A\left(s_t, w_t, g\right)\right)$, where the binary variable U can be interpreted as the event of reaching the goal g as quickly as possible from state s_t by passing through subgoal state w. The subgoal advantage is defined as $A(s_t, w, g) = -V(s_t, g) + \gamma^k V(w, g) + \sum_{t'=t}^{k-1} r(s_{t'}, g)$. In practice, we follow HIQL and simplify the advantage estimate to $V(w, g) - V(s_t, g)$, i.e., the progress towards the goal achieved by reaching w.

Without loss of generality (since we can represent any flat Markovian policy simply by setting $\pi^h(\cdot \mid s, g)$ to a point distribution on g), we use an inductive bias on the subgoal structure of GCRL to consider policies π of a factored *hierarchical* form

$$p_\pi(\tau \mid g) = p(s_0) \prod_{t=0}^{\infty} p(s_{t+1} \mid s_t, a_t) \pi^\ell(a_t \mid s_t, w_t) \pi^h(w_t \mid s_t, g).$$

The distinctions between hierarchical approaches like HIQL and non-hierarchical approaches such as RIS and SAW begin with our choice of variational posterior. For the former, we would consider similarly factored distributions, whereas for the latter, we use a *flat* posterior $q^f(\tau \mid g)$ that factors as

$$q^f(\tau \mid g) = p(s_0) \prod_{t=0}^{\infty} p(s_{t+1} \mid s_t, a_t) q^f(a_t \mid s_t, g),$$

assuming that the dataset policies are Markovian. We also introduce a variational posterior $q^h(\{w\} \mid g)$ which factors over a sequence of waypoints $\{w\} = \{w_0, w_1, \ldots\}$ as

$$q^h(\{w\} \mid g) = p(s_0) \prod_{t=0}^{\infty} p(s_{t+1} \mid s_t, a_t) \pi^{\text{sub}}(a_t \mid s_t, w_t) q^h(w_t \mid s_t, g).$$

where we treat the target subpolicy π^{sub} as fixed. Using these definitions, we define the evidence lower bound (ELBO) on the optimality likelihood $p_\pi(U=1)$ for policy π and goal distribution $p(g)$

$$\begin{aligned}\log p_\pi(U=1) &= \log \int p(g) p_\pi(\tau, \{w\} \mid g) p(U=1 \mid \tau, \{w\}, g) d\{w\}\, d\tau\, dg \\ &\geq \mathbb{E}_{q^f(\tau \mid g), q^h(\{w\} \mid g), p(g)} \log \left[\frac{p(U=1, \tau, \{w\} \mid g)}{q^f(\tau \mid g) q^h(\{w\} \mid g)}\right] = \mathcal{J}(q, \pi).\end{aligned}$$

Algorithm 1 Subgoal Advantage-Weighted Policy Bootstrapping (SAW)

```
1: **Input**: offline dataset $\mathcal{D}$, goal distribution $p(g)$.
2: Initialize value function $V_\phi$, target subpolicy $\pi_\psi$, and policy $\pi_\theta$.
3: **while** not converged **do**
4:     Train value function: $\phi \leftarrow \phi - \lambda \nabla_\phi \mathcal{L}_{\mathrm{GCIVL}}(\phi)$ with $(s_t, s_{t+1}) \sim p^{\mathcal{D}}, g \sim p(g)$ [Equation 2]
5: **end while**
6: **while** not converged **do**
7:     Train target subpolicy: $\omega \leftarrow \omega - \lambda \nabla_\omega \mathcal{J}_{\mathrm{AWR}}(\omega)$ with $(s_t, a, w) \sim p^{\mathcal{D}}$ [Equation 3]
8: **end while**
9: **while** not converged **do**
10:     Train policy: $\theta \leftarrow \theta - \lambda \nabla_\theta \mathcal{J}_{\mathrm{SAW}}(\theta)$ with $(s_t, a, w) \sim p^{\mathcal{D}}, g \sim p(g)$ [Equation 7]
11: **end while**
```

Here, q^f and q^h serve as auxiliary distributions that allow us to optimize their corresponding parametric policies π_θ and π^h via an iterative expectation-maximization (EM) procedure. Expanding distributions according to their factorizations, dropping terms that are independent of the variationals, and rewriting the discounted sum over time as an expectation over the (unnormalized) discounted stationary state distribution $\mu_\pi(s) = \sum_{t=0}^{\infty} \gamma^t p(s_t = s \mid \pi)$ results in the final objective

$$\begin{aligned}\mathcal{J}(q,\pi) = \mathbb{E}_{\mu(s),p(g)}\big[\mathbb{E}_{q^h(w\mid s,g)}\left[A(s,w,g)\right] - \mathbb{E}_{q^f(a\mid s,g)}\left[D_{\mathrm{KL}}(q^h(w \mid s,g)\|\pi^h(w \mid s,g))\right] \\ - \mathbb{E}_{q^h(w\mid s,g)}\left[D_{\mathrm{KL}}(q^f(a \mid s,g)\|\pi^\ell(a \mid s,w))\right]\big], \quad (5)\end{aligned}$$

where we optimize an approximation of $\mathcal{J}$ by sampling from the dataset distribution $\mu_{\mathcal{D}}(s)$ (Schulman et al., 2015; Abdolmaleki et al., 2018; Peng et al., 2019).

5.2 Eliminating the subgoal generator

Both RIS and HIQL iteratively minimize the KL divergence between the parametric generative model $\pi^h_\psi(w \mid s,g)$ and the optimal sample-based distribution $q^h(w \mid s,g) \propto \pi^h_\psi(w \mid s,g)\exp(A(s,w,g))$, which is the closed-form solution to the first line of Equation 5. Then, RIS trains a flat policy π_θ using the remaining KL divergence term in the second line of Equation 5, which minimizes the divergence between the flat goal-conditioned policy posterior q^f and the subgoal-conditioned policy π^ℓ, in expectation over subgoals sampled from π_ψ, which approximates the optimal subgoal distribution q^h.

Instead of learning a generative model over the potentially high-dimensional space of subgoals, we can directly estimate this expectation from dataset samples using a simple application of Bayes' rule:

$$\begin{aligned}p(w \mid s,g,U=1) &\propto p^{\mathcal{D}}(w \mid s)p(U=1 \mid s,w,g) \\ &\propto p^{\mathcal{D}}(w \mid s)\exp(A(s,w,g)),\end{aligned}$$

which replaces the expectation over q to yield our subgoal advantage-weighted bootstrapping term

$$\mathbb{E}_{\mu(s),p^{\mathcal{D}}(w\mid s),p(g)}[\exp(A(s,w,g))D_{\mathrm{KL}}(\pi_\theta(a \mid s,g)\|\pi^\ell(a \mid s,w))]. \quad (6)$$

We learn an approximation to $\pi^\ell(a \mid s,w)$ by training a separate (sub)goal-conditioned policy with AWR in a similar fashion to HIQL, whereas RIS uses an exponential moving average of the parameters of its full goal-conditioned policy as a target. Since our regression target is a parametric distribution, this conveniently allows us to directly model q^f with another parametric policy $\pi_\theta(a \mid s,g)$.

While approximating $p(w \mid s, U=1)$ with q directly and using our importance weight on the dataset distribution are mathematically equivalent, the latter does introduce sampling-based limitations, which we discuss in Appendix A. However, we show empirically that the benefits from lifting the burden of learning a distribution over a high-dimensional subgoal space far outweigh these drawbacks, especially in large state spaces with high intrinsic dimensionality.

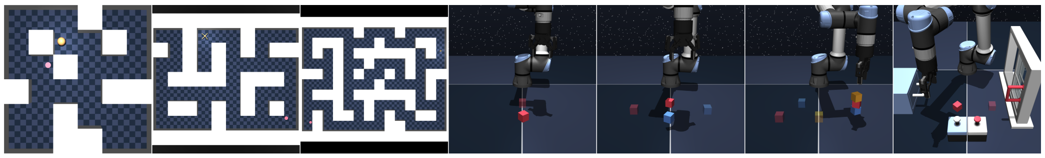

Figure 2: **OGBench tasks**. We train SAW on 20 datasets collected from 7 different environments (pictured above) and perform evaluations across 5 state-goal pairs for each dataset.

5.3 The SAW objective

The importance weight in Equation 6 allows the policy to bootstrap from subgoals sampled directly from dataset trajectories by ensuring that only subpolicies conditioned on high-advantage subgoals influence the direction of the goal-conditioned policy. We combine our bootstrapping term with an additional learning signal from a (one-step) policy extraction objective utilizing the value function, which improves performance in stitching-heavy environments [Supplementary Section I]. Here, we use one-step AWR [Equation 3], yielding the full SAW objective:

$$\mathcal{J}(\theta) = \mathbb{E}_{p^{\mathcal{D}}(s,a,w),p(g)}\left[e^{\alpha A(s,a,g)} \log \pi_\theta(a \mid s, g) - e^{\beta A(s,w,g)} D_{\mathrm{KL}}\left(\pi_\theta(a \mid s, g) \| \pi^{\mathrm{sub}}(a \mid s, w)\right)\right] \tag{7}$$

where α and β are inverse temperature hyperparameters. This objective provides a convenient dynamic balance between its two terms: as the goal horizon increases, the differences in action values and therefore the contribution of one-step term decreases. This, in turn, downweights the noisier value-based learning signal and shifts emphasis toward the policy bootstrapping term. Finally, we use GCIVL to learn V, resulting in the full training scheme outlined in Algorithm 1.

6 Experiments

To assess SAW's ability to reason over long horizons and handle high-dimensional observations, we conduct experiments across 20 datasets corresponding to 7 locomotion and manipulation environments [Figure 2] with both state- and pixel-based observation spaces. We report performance averaged over 5 state-goal pairs for each dataset, yielding **100** total evaluation tasks. Implementation details and hyperparameter settings are discussed in Supplementary Sections F and G, respectively.

6.1 Experimental setup

We select several environments and their corresponding datasets from the recently released OGBench suite (Park et al., 2024a), a comprehensive benchmark specifically designed for offline GCRL. OGBench provides multiple state-goal pairs for evaluation and datasets tailored to evaluate desirable properties of offline GCRL algorithms, such as the ability to reason over long horizons and stitch across multiple trajectories or combinatorial goal sequences. We use the baselines from the original OGBench paper, which include both one-step and hierarchical state-of-the-art offline GCRL methods. We briefly describe each category of tasks below and baseline algorithms in Appendix C, and encourage readers to refer to original OGBench work for further details.

Locomotion: Locomotion tasks require the agent to control a simulated robot to navigate through a maze and reach a designated goal. The agent embodiment varies from a simple 2D point mass with two-dimensional action and observation spaces to a humanoid robot with 21 degrees of freedom and a 69-dimensional state space. In the `visual` variants, the agent receives a third-person, egocentric $64 \times 64 \times 3$ pixel-based observations, with its location within the maze indicated by the floor color. Maze layouts range from `medium` to `giant`, where tasks in the `humanoidmaze` version of the latter require up to 3000 environment steps to complete.

Environment	Dataset	GCBC	GCIVL	GCIQL	QRL	CRL	HIQL	SAW
	`pointmaze-medium-navigate-v0`	9 ±6	63 ±6	53 ±8	82 ±5	29 ±7	79 ±5	**97** ±2
`pointmaze`	`pointmaze-large-navigate-v0`	29 ±6	45 ±5	34 ±3	**86** ±9	39 ±7	58 ±5	**85** ±10
	`pointmaze-giant-navigate-v0`	1 ±2	0 ±0	0 ±0	**68** ±7	27 ±10	46 ±9	**68** ±8
	`antmaze-medium-navigate-v0`	29 ±4	72 ±8	71 ±4	88 ±3	**95** ±1	**96** ±1	**97** ±1
`antmaze`	`antmaze-large-navigate-v0`	24 ±2	16 ±5	34 ±4	75 ±6	83 ±4	**91** ±2	**90** ±3
	`antmaze-giant-navigate-v0`	0 ±0	0 ±0	0 ±0	14 ±3	16 ±3	65 ±5	**73** ±4
	`humanoidmaze-medium-navigate-v0`	8 ±2	24 ±2	27 ±2	21 ±8	60 ±4	**89** ±2	**88** ±3
`humanoidmaze`	`humanoidmaze-large-navigate-v0`	1 ±0	2 ±1	2 ±1	5 ±1	24 ±4	**49** ±4	46 ±4
	`humanoidmaze-giant-navigate-v0`	0 ±0	0 ±0	0 ±0	1 ±0	3 ±2	12 ±4	**35** ±4
	`cube-single-play-v0`	6 ±2	53 ±4	68 ±6	5 ±1	19 ±2	15 ±3	**72*** ±5
`cube`	`cube-double-play-v0`	1 ±1	36 ±3	**40** ±5	1 ±0	10 ±2	6 ±2	**40** ±7
	`cube-triple-play-v0`	1 ±1	1 ±0	3 ±1	0 ±0	**4** ±1	3 ±1	**4** ±2
`scene`	`scene-play-v0`	5 ±1	42 ±4	51 ±4	5 ±1	19 ±2	38 ±3	**63** ±6
	`visual-antmaze-medium-navigate-v0`	11 ±2	22 ±2	11 ±1	0 ±0	**94** ±1	**93** ±4	**95** ±0
`visual-antmaze`	`visual-antmaze-large-navigate-v0`	4 ±0	5 ±1	4 ±1	0 ±0	**84** ±1	53 ±9	**82** ±4
	`visual-antmaze-giant-navigate-v0`	0 ±0	1 ±1	0 ±0	0 ±0	**47** ±2	6 ±4	10 ±2
	`visual-cube-single-play-v0`	5 ±1	60 ±5	30 ±5	41 ±15	31 ±15	**89** ±0	**88** ±3
`visual-cube`	`visual-cube-double-play-v0`	1 ±1	10 ±2	1 ±1	5 ±0	2 ±1	**39** ±2	**40** ±3
	`visual-cube-triple-play-v0`	15 ±2	14 ±2	15 ±1	16 ±1	17 ±2	**21** ±0	**20** ±1
`visual-scene`	`visual-scene-play-v0`	12 ±2	25 ±3	12 ±2	10 ±1	11 ±2	**49** ±4	**47** ±6

Table 1: **Evaluating SAW on state- and pixel-based offline goal-conditioned RL tasks.** We compare our method's average (binary) success rate (%) against the numbers reported in Park et al. (2024a) across the five test-time goals for each environment, averaged over 8 seeds (4 seeds for pixel-based `visual` tasks) with standard deviations after the $\pm$ sign. Numbers within 5% of the best value in the row are in **bold**. Results with an asterisk (*) use different value learning hyperparameters and are discussed further in Section 6.3.

Manipulation: Manipulation tasks use a 6-DoF UR5e robot arm to manipulate object(s), including up to four cubes and a more diverse `scene` environment that includes buttons, windows, and drawers. The multi-cube and scene environments are designed to test an agent's ability to perform sequential, long-horizon goal stitching and compose together multiple atomic behaviors. The `visual` variants replace the ground truth state with a $64 \times 64 \times 3$ pixel-based observation from a fixed third-person perspective, where parts of the environment and arm are made semitransparent to ease state estimation.

6.2 Locomotion results

State-based locomotion: As a method designed for long-horizon reasoning, SAW excels in all variants of the state-based locomotion tasks. It scales particularly well to long horizons, exhibiting the best performance of **73**% across all tasks in `antmaze-giant-navigate` and is the first method to achieve non-trivial success in `humanoidmaze-giant-navigate`, reaching **35**% success compared to the previous state-of-the-art of 12% (Park et al., 2024a). We demonstrate that training subpolicies with subgoal representations scales poorly to the `giant` maze environments [Figure 3] but are critical to HIQL's performance, emphasizing a fundamental tradeoff in hierarchical methods: subgoal representations are essential for making high-level policy prediction tractable, but those same representations can constrain policy expressiveness and limit overall performance. While other subgoal representation learning objectives may perform better than those derived from the value function, as in HIQL, this highlights the additional design complexity and tuning required for HRL methods. We also implement an offline variant of RIS [Appendix C] and find that it performs significantly worse than SAW with subgoal representations, which we suspect can be explained by our insights in Section 4.3.

Pixel-based locomotion: SAW maintains strong performance when given $64 \times 64 \times 3$ visual observations and scales much better to `visual-antmaze-large` than does its hierarchical counterpart. However, we do see a significant performance drop in the `giant` variant relative to the results in the state-based observation space. As a possible explanation for this discrepancy, we observed that value function training diverged for HIQL and SAW in `visual-antmaze-giant` as well as all `visual-humanoidmaze` sizes (omitted since no method achieved non-trivial performance). This occurred even without shared policy gradients, suggesting that additional work is

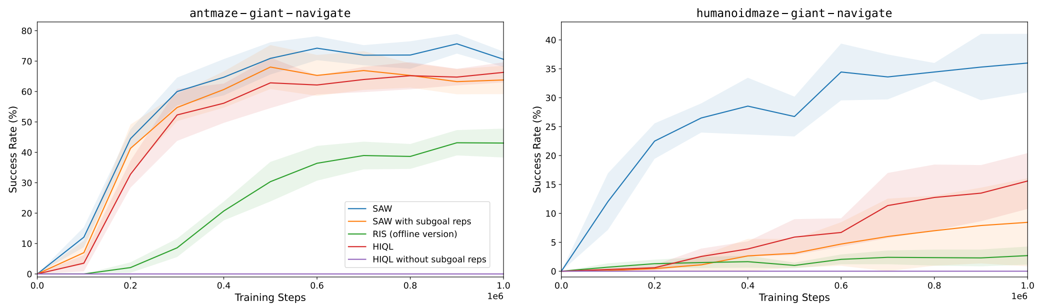

Figure 3: **Subgoal representations scale poorly to high-dimensional control in large state spaces**. Using HIQL's subgoal representations (taken from an intermediate layer of the value function) for SAW's target subpolicy harms performance compared to training directly on observations. However, HIQL fails to learn meaningful behaviors when predicting subgoals directly in the raw observation space. RIS, which bootstraps on generated subgoals at every step, performs the worst of the three.

needed to scale offline value learning objectives to very long-horizon tasks with high-dimensional visual observations.

6.3 Manipulation results

State-based manipulation: SAW consistently matches state-of-the-art performance in `cube` environments and significantly outperforms existing methods in the 5 `scene` tasks, which require extended compositional reasoning. Interestingly, we found that methods which use expectile regression-based offline value learning methods (GCIVL, GCIQL, HIQL, and SAW) are highly sensitive to value learning hyperparameters in the `cube-single` environment. Indeed, SAW performs more than twice as well on `cube-single-play-v0` with settings of $\tau = 0.9$ and $\beta = 0.3$, reaching state-of-the-art performance (**72%** ± 5 vs. 32% ± 4 with $\tau = 0.7$). While SAW is agnostic to the choice of value learning objective, we make special mention of these changes since they depart from the OGBench convention of fixing value learning hyperparameters for each method across all datasets.

Pixel-based manipulation: In contrast to the state-based environments, SAW and HIQL achieve near-equivalent performance in visual manipulation. This suggests that representation learning, and not long-horizon reasoning or goal stitching, is the primary bottleneck in the visual manipulation environments. While we do not claim any representation learning innovations for this paper, our results nonetheless demonstrate that SAW is able to utilize similar encoder-sharing tricks as HIQL to scale to high-dimensional observation spaces.

7 Discussion

We presented Subgoal Advantage-Weighted Policy Bootstrapping (SAW), a simple yet effective policy extraction objective that leverages the subgoal structure of goal-conditioned tasks to scale to long-horizon tasks, without learning generative subgoal models. SAW consistently matches or surpasses current state-of-the-art methods across a wide variety of locomotion and manipulation tasks that require different timescales of control, whereas existing methods tend to specialize in particular task categories. Our method especially distinguishes itself in long-horizon reasoning, excelling in the most difficult locomotion tasks and scene-based manipulation. While the simplicity of our objective does introduce some practical limitations related to subgoal sampling, which we discuss in Appendix A, we find that avoiding explicit subgoal prediction is crucial for maintaining performance in large state spaces. By demonstrating a scalable approach to train unified policies for offline GCRL, we believe that SAW takes a step toward realizing the full potential of robotic foundation models in addressing the long-horizon, high-dimensional challenges of real-world control.

A Limitations

A theoretical limitation of our approach, which is common to all hierarchical methods as well as RIS, occurs in our assumption that the optimal policy can be represented in the factored form $\pi^\ell(a \mid s, w)\pi^h(w \mid s, g)$. While this is true in theory (since we could trivially set $\pi^h(w \mid s, g)$ to a point distribution at g), practical algorithms typically fix the distance of the subgoals to a shorter distance of k steps (or the midpoint in RIS), where subgoals s_{t+k} are sampled from the future state distribution $p^{\mathcal{D}}_{\text{traj}}(s_{t+k} \mid s_t)$. This introduces bias by limiting the space of subgoals to the support of the dataset k-step distribution, when more optimal subgoals can be proposed by taking into account policy improvement over the data collection policy. However, as discussed in Section 4.3, we find that subgoals sampled from the future state distribution work empirically well with respect to goals also sampled from the future state distribution, which is common in practice (Gupta et al., 2020; Ghosh et al., 2020; Yang et al., 2021; Eysenbach et al., 2022; Park et al., 2023). We suspect that subgoal generator-based methods may perform better in datasets that require a high degree of stitching: they can synthesize "imagined" subgoals on which to bootstrap, while our approach may require alternative subgoal-sampling strategies to reach the same level of performance.

B Planning Invariance

As an aside, we note that the discussions in this paper are closely related to the recently introduced concept of *planning invariance* (Myers et al., 2024), which describes a policy that takes similar actions when directed towards a goal as when directed towards an intermediate waypoint en route to that goal. In fact, we can say that subgoal-conditioned HRL methods achieve a form of planning invariance by construction, since they simply use the actions yielded by waypoint-conditioned policies to reach further goals. By minimizing the divergence between the full goal-conditioned policy and an associated subgoal-conditioned policy, both SAW and RIS can also be seen as implicitly enforcing planning invariance.

C Offline GCRL Baseline Algorithms

In this section, we briefly review the baseline algorithms referenced in Table 1. For more thorough implementation details, as well as goal-sampling distributions, interested readers may refer to Appendix C of Park et al. (2024a) as well as the original works.

Goal-conditioned behavioral cloning (GCBC): GCBC is an imitation learning approach that clones behaviors using hindsight goal relabeling on future states in the same trajectory.

Goal-conditioned implicit {Q, V}-learning (GCIQL & GCIVL): GCIQL is a goal-conditioned variant of implicit Q-learning (Kostrikov et al., 2021), which performs policy iteration with an expectile regression to avoid querying the learned Q-value function for out-of-distribution actions. Park et al. (2023) introduced a V-only variant that directly regresses towards high-value transitions [Equation 2], using $r(s, g) + \gamma \bar{V}(s', g)$ as an estimator of $Q(s, a, g)$. Since it does not learn Q-values and therefore cannot marginalize over non-causal factors, it is optimistically biased in stochastic environments.

Although both baselines are value learning methods that can be used with multiple policy extraction objectives (including our own, which uses GCIVL), the OGBench implementations are paired with following objectives: Deep Deterministic Policy Gradient with a behavior cloning penalty term (Fujimoto & Gu, 2021, DDPG+BC) for GCIQL, and AWR [Equation 3] for GCIVL.

Quasimetric RL (QRL): QRL (Wang et al., 2023) is a non-traditional value learning algorithm that uses Interval Quasimetric Embeddings (Wang & Isola, 2022, IQE) to enforce quasimetric properties (namely, the triangle inequality and identity of indiscernibles) of the goal-conditioned value function on distances between representations. It uses a constrained "maximal spreading" objective to estimate the shortest paths between states, then learns a one-step dynamics model combined with DDPG+BC to extract a policy from the learned representations.

Contrastive RL (CRL): CRL (Eysenbach et al., 2022) is a representation learning algorithm which uses contrastive learning to enforce that the inner product between the learned representations of a state-action pair and the goal state corresponds to the discounted future state occupancy measure of the goal state, which is estimated directly from data using Monte Carlo sampling. CRL then performs one-step policy improvement by choosing actions that maximize the future occupancy of the desired goal state.

Hierarchical implicit Q-learning (HIQL): HIQL (Park et al., 2023) is a policy extraction method that learns two levels of hierarchical policy from the same goal-conditioned value function. The low-level policy π^ℓ is trained using standard AWR, and the high-level policy π^h is trained using an action-free, multi-step variant of AWR that treats (latent) subgoal states as "actions."

Reinforcement learning with imagined subgoals (RIS): RIS (Chane-Sane et al., 2021) is a policy extraction method originally designed for the online GCRL setting, which learns a subgoal generator and a flat, goal-conditioned policy. Unlike SAW, RIS uses a fixed coefficient on the KL term, instead learning a subgoal generator and bootstrapping directly on a target policy (parameterized by an exponential moving average of online policy parameters rather than a separately learned subpolicy) conditioned on "imagined" subgoals. It also incorporates a value-based policy learning objective similar to our approach, but learns a Q-function and differentiates directly through the policy with DDPG.

To modify RIS for the offline setting in our implementation for Figure 3, we fixed the coefficient on the KL term to $\beta = 3.0$, trained a subgoal generator identical to the one in HIQL, and replaced the dataset subgoals in SAW with "imagined" subgoals. Otherwise, for fairness of comparison, our offline RIS implementation used the same hyperparameters and architectures as SAW, including a separate target subpolicy network instead of a soft copy of the online policy, subgoals at a fixed distance instead of at midpoints, and AWR instead of DDPG+BC for the policy extraction objective, which we found to perform better in locomotion environments.

References

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Neural Information Processing Systems*, 2017.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *International Conference on Machine Learning*, 2017.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI Conference on Artificial Intelligence*, 2017.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. arXiv, 2410.24164 [cs], 2024.

Elliot Chane-Sane, Cordelia Schmid, and Ivan Laptev. Goal-conditioned reinforcement learning with imagined subgoals. In *International Conference on Machine Learning*, 2021.

Konrad Czechowski, Tomasz Odrzygóźdź, Marek Zbysiński, Michał Zawalski, Krzysztof Olejnik, Yuhuai Wu, Lukasz Kuciński, and Piotr Miłoś. Subgoal search for complex reasoning tasks. In *Neural Information Processing Systems*, 2021.

Peter Dayan and Geoffrey E Hinton. Feudal reinforcement learning. In *Neural Information Processing Systems*, 1992.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. Search on the replay buffer: Bridging planning and reinforcement learning. In *Neural Information Processing Systems*, 2019.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Ruslan Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. In *Neural Information Processing Systems*, 2022.

Scott Fujimoto and Shixiang Gu. A minimalist approach to offline reinforcement learning. In *Neural Information Processing Systems*, 2021.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Manon Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. In *International Conference on Learning Representations*, 2020.

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In *Conference on Robot Learning*, 2020.

Danijar Hafner, Kuang-Huei Lee, Ian Fischer, and Pieter Abbeel. Deep hierarchical planning from pixels. In *Neural Information Processing Systems*, 2022.

Kyle Beltran Hatch, Ashwin Balakrishna, Oier Mees, Suraj Nair, Seohong Park, Blake Wulfe, Masha Itkina, Benjamin Eysenbach, Sergey Levine, Thomas Kollar, and Benjamin Burchfiel. GHIL-glue: Hierarchical control with filtered subgoal images. In *CoRL Workshop on Mastering Robot Manipulation in a World of Abundant Data*, 2024.

Christopher Hoang, Sungryull Sohn, Jongwook Choi, Wilka Carvalho, and Honglak Lee. Successor feature landmarks for long-horizon goal-conditioned reinforcement learning. In *Neural Information Processing Systems*, 2021.

Zhiao Huang, Fangchen Liu, and Hao Su. Mapping state space using landmarks for universal goal reaching. In *Neural Information Processing Systems*, 2019.

Nicholas K. Jong, Todd Hester, and Peter Stone. The utility of temporal abstraction in reinforcement learning. In *International Joint Conference on Autonomous Agents and Multiagent Systems*, 2008.

Leslie Pack Kaelbling. Learning to achieve goals. In *International Joint Conference on Artificial Intelligence*, 1993.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2021.

Sascha Lange, Thomas Gabel, and Martin Riedmiller. *Batch Reinforcement Learning*. Springer, 2012.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. arXiv, 1805.00909 [cs], 2018.

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

Andrew Levy, George Konidaris, Robert Platt, and Kate Saenko. Learning multi-level hierarchies with hindsight. In *International Conference on Learning Representations*, 2018.

Yecheng Jason Ma, Jason Yan, Dinesh Jayaraman, and Osbert Bastani. Offline goal-conditioned reinforcement learning via f-advantage regression. In *Advances in Neural Information Processing Systems*, 2022.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations*, 2023.

Vivek Myers, Catherine Ji, and Benjamin Eysenbach. Horizon generalization in reinforcement learning. In *International Conference on Learning Representations*, 2024.

Ofir Nachum, Shixiang (Shane) Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. In *Neural Information Processing Systems*, 2018.

Ofir Nachum, Haoran Tang, Xingyu Lu, Shixiang Gu, Honglak Lee, and Sergey Levine. Why does hierarchy (sometimes) work so well in reinforcement learning? In *NeurIPS DeepRL Workshop*, 2019.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, 2023.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. HIQL: Offline goal-conditioned RL with latent states as actions. In *Neural Information Processing Systems*, 2023.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *International Conference on Learning Representations*, 2024.

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline RL? In *Neural Information Processing Systems*, 2024.

Seohong Park, Tobias Kreiman, and Sergey Levine. Foundation policies with hilbert representations. In *International Conference on Machine Learning*, 2024.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. arXiv, 1910.00177, 2019.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. arXiv, 2504.16054 [cs], 2025.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, 2005.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, 2015.

Younggyo Seo, Kimin Lee, Stephen L. James, and Pieter Abbeel. Reinforcement learning with action-free pre-training from videos. In *International Conference on Machine Learning*, 2022.

Alexander L. Strehl, Lihong Li, and Michael L. Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10(84):2413–2444, 2009.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, 1999.

Stephen Tian, Suraj Nair, Frederik Ebert, Sudeep Dasari, Benjamin Eysenbach, Chelsea Finn, and Sergey Levine. Model-based visual planning with self-supervised functional distances. In *International Conference on Learning Representations*, 2020.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. FeUdal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, 2017.

Kevin Wang, Ishaan Javali, Michał Bortkiewicz, Tomasz Trzciński, and Benjamin Eysenbach. 1000 layer networks for self-supervised RL: Scaling depth can enable new goal-reaching capabilities. arXiv, 2503.14858 [cs], 2025.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, 2020.

Tongzhou Wang and Phillip Isola. Improved representation of asymmetrical distances with interval quasimetric embeddings. In *NeurIPS Workshop on Symmetry and Geometry in Neural Representations*, 2022.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*, 2023.

Rui Yang, Yiming Lu, Wenzhe Li, Hao Sun, Meng Fang, Yali Du, Xiu Li, Lei Han, and Chongjie Zhang. Rethinking goal-conditioned supervised learning and its connection to offline RL. In *International Conference on Learning Representations*, 2021.

Rui Yang, Lin Yong, Xiaoteng Ma, Hao Hu, Chongjie Zhang, and Tong Zhang. What is essential for unseen goal generalization of offline goal-conditioned RL? In *International Conference on Machine Learning*, 2023.

Michał Zawalski, Michał Tyrolski, Konrad Czechowski, Tomasz Odrzygóźdź, Damian Stachura, Piotr Piękos, Yuhuai Wu, Lukasz Kuciński, and Piotr Miłoś. Fast and precise: Adjusting planning horizon with adaptive subgoal search. In *International Conference on Learning Representations*, 2022.

Zilai Zeng, Ce Zhang, Shijie Wang, and Chen Sun. Goal-conditioned predictive coding for offline reinforcement learning. In *Neural Information Processing Systems*, 2023.

Lunjun Zhang, Ge Yang, and Bradly C. Stadie. World model as a graph: Learning latent landmarks for planning. In *International Conference on Machine Learning*, 2021.

Tianjun Zhang, Benjamin Eysenbach, Ruslan Salakhutdinov, Sergey Levine, and Joseph E. Gonzalez. C-planning: An automatic curriculum for learning goal-reaching tasks. In *International Conference on Learning Representations*, 2021.

Tianren Zhang, Shangqi Guo, Tian Tan, Xiaolin Hu, and Feng Chen. Generating adjacency-constrained subgoals in hierarchical reinforcement learning. In *Neural Information Processing Systems*, 2020.

Chongyi Zheng, Ruslan Salakhutdinov, and Benjamin Eysenbach. Contrastive difference predictive coding. In *International Conference on Learning Representations*, 2023.

Supplementary Materials

The following content was not necessarily subject to peer review.

D Derivations of HIQL, RIS, and SAW Objectives

We cast the infinite-horizon, discounted GCRL formulation as an inference problem by constructing a probabilistic model via the likelihood function

$$p\left(U_t = 1 \mid \tau, \{w\}, g\right) \propto \exp\left(\beta \sum_{t=0}^{\infty} \gamma^t A\left(s_t, w_t, g\right)\right),$$

where β is an inverse temperature parameter and the binary variable U can be intuitively understood as the event of reaching the goal g as quickly as possible by passing through a subgoal w, or passing through a subgoal w which is on the shortest path between s_t and g.

We consider policies π^ℓ and π^h of a factored *hierarchical* form

$$p_\pi(\tau \mid g) = p(s_0) \prod_{t=0}^{\infty} p(s_{t+1} \mid s_t, a_t)\, \pi_\theta^\ell(a \mid s_t, w_t)\, \pi_\psi^h(w_t \mid s_t, g).$$

where the low-level policy over actions π^ℓ and high-level policy over subgoals π^h are parameterized by θ and ψ, respectively. However, for the sake of clarity, we drop the parameters from the notation in the following derivations.

D.1 HIQL derivation

For HIQL's hierarchical policy, we use variational posteriors q^ℓ and q^h of the same form

$$q(\tau \mid g) = p(s_0) \prod_{t=0}^{\infty} p(s_{t+1} \mid s_t, a_t)\, q^\ell(a_t \mid s_t, w_t)\, q^h(w_t \mid s_t, g).$$

To incorporate training of the low-level policy, we also construct an additional probabilistic model for the optimality of primitive actions towards a waypoint w (note that this can also be done to incorporate target policy training into the SAW objective, but we leave it out for brevity)

$$p\left(O_t = 1 \mid \tau, \{w\}\right) \propto \exp\left(\alpha \sum_{t=0}^{\infty} \gamma^t A\left(s_t, a_t, w_t\right)\right).$$

With these definitions, we define the evidence lower bound (ELBO) on the joint optimality likelihood $p_\pi(O = 1, U = 1)$ for policy π and posterior q as

$$
\begin{aligned}
\log p_\pi(O=1,U=1) &= \log \int p(g)\, p(O=1,U=1,\tau,\{w\} \mid g) d\{w\}\, d\tau\, dg \\
&= \log \int p(g) q^\ell(\tau \mid g) q^h(\{w\} \mid g) \frac{p(O=1,U=1,\tau,\{w\} \mid g)}{q^\ell(\tau \mid g)\, q^h(\{w\} \mid g)} d\{w\}\, d\tau\, dg \\
&= \log \mathbb{E}_{q^\ell(\tau|g), q^h(\{w\}|g), p(g)} \left[\frac{p(O=1,U=1,\tau,\{w\} \mid g)}{q^\ell(\tau \mid g) q^h(\{w\} \mid g)} \right] \\
&\geq \mathbb{E}_{q^\ell(\tau|g), q^h(\{w\}|g), p(g)} \log \left[\frac{p(O=1,U=1,\tau,\{w\} \mid g)}{q^\ell(\tau \mid g) q^h(\{w\} \mid g)} \right] = \mathcal{J}(q,\pi).
\end{aligned}
$$

Expanding the fraction, moving the $\log$ inside, and dropping the start state distribution $p(s_0)$ and transition distributions $p(s_{t+1} \mid s_t, a_t)$, which are fixed with respect to θ and ψ, gives us

$$
\begin{aligned}
\mathbb{E}_{q^\ell(\tau|g),\, q^h(\{w\}|g),\, p(g)} \Bigg[& \alpha \sum_{t=0}^{\infty} \gamma^t A(s_t, a_t, w_t) + \beta \sum_{t=0}^{\infty} \gamma^t A(s_t, w_t, g) \\
& + \sum_{t=0}^{\infty} \log \left(\frac{\pi^\ell(a_t \mid s_t, w_t)\, \pi^h(w_t \mid s_t, g)}{q^\ell(a_t \mid s_t, g)\, q^h(w_t \mid s_t, g)} \right) \Bigg]
\end{aligned}
$$

We rewrite the discounted sum over time as an expectation over the (unnormalized) discounted stationary state distribution $\mu_\pi(s) = \sum_{t=0}^{\infty} \gamma^t p(s_t = s \mid \pi)$ induced by policy π. In practice, however, we optimize an approximation of $\mathcal{J}(q,\theta,\psi)$ by sampling from the dataset distribution over states $\mu_{\mathcal{D}}$. For brevity, we omit the conditionals in the expectations below, defining $q^\ell(a) := q^\ell(a \mid s, g)$ and $q^h(w) := q^h(w \mid s, g)$ in the expectations below

$$
\begin{aligned}
& \mathbb{E}_{\mu(s), q^\ell(a), q^h(w), p(g)} \left[\alpha A(s,a,w) + \log \left[\frac{\pi^\ell(a \mid s, w)}{q^\ell(a \mid s, w)} \right] \right] \\
& \quad + \mathbb{E}_{\mu(s), q^h(w), p(g)} \left[\beta A(s,w,g) + \log \left[\frac{\pi^h(w \mid s, g)}{q^h(w \mid s, g)} \right] \right] \\
& = \mathbb{E}_{\mu(s), q^\ell(a), q^h(w), p(g)} \left[\alpha A(s,a,w) \right] - \mathbb{E}_{\mu(s), q^h(w), p(g)} \left[D_{\mathrm{KL}} \left[q^\ell(a \mid s, w) \| \pi^\ell(a \mid s, w) \right] \right] \\
& \quad + \mathbb{E}_{\mu(s), q^h(w), p(g)} \left[\beta A(s,w,g) \right] - \mathbb{E}_{\mu(s), p(g)} \left[D_{\mathrm{KL}} \left[q^h(w \mid s, g) \| \pi^h(w \mid s, g) \right] \right]. \qquad (8)
\end{aligned}
$$

HIQL separately optimizes the two summation terms, which correspond to the low- and high-level policies, respectively. Forming the Lagrangian with the normalization condition and solving for the optimal low- and high-level policies, as done in Abdolmaleki et al. (2018) and Peng et al. (2019), yields the optimal sample-based q^ℓ and q^h distributions:

$$
\begin{aligned}
q^\ell(a \mid s, g) &\propto \pi^\ell(a \mid s, w) \exp(A(s,a,w)) \\
q^h(a \mid s, g) &\propto \pi^h(w \mid s, g) \exp(A(s,w,g)).
\end{aligned}
$$

Minimizing the KL divergence between the optimal (non-parametric) posteriors and their respective parametric policies π_θ^ℓ and π_ψ^h yields the bilevel AWR policy extraction objectives for HIQL

$$
\begin{aligned}
\mathcal{J}^\ell(\theta) &= \mathbb{E}_{\mu(s), q^\ell(a), q^h(w), p(g)} \left[\exp(A(s,a,w)) \log \pi_\theta^\ell(a \mid s, w) \right] \\
\mathcal{J}^h(\psi) &= \mathbb{E}_{\mu(s), q^h(w), p(g)} \left[\exp(A(s,w,g)) \log \pi_\psi^h(w \mid s, g) \right].
\end{aligned}
$$

While our derivation produces an on-policy expectation over actions and subgoals, the sampling distribution over states, actions, subgoals, and goals in the offline setting varies in practice (see Appendix C of Park et al. (2024a) for commonly used goal distributions).

D.2 RIS and SAW derivations

Unlike HIQL, RIS and SAW both seek to learn a unified flat policy, and therefore we choose a policy posterior $q^f(\tau)$ that factors as

$$q^f(\tau \mid g) = p(s_0) \prod_{t=0}^{\infty} p(s_{t+1} \mid s_t, a_t) q^f(a_t \mid s_t, g).$$

For RIS, which learns a generative subgoal policy identical to that of HIQL, we also use the same variational posterior q^h which factors over a sequence of waypoints $\{w\} = \{w_0, w_1, \ldots\}$ as

$$q^h(\{w\} \mid g) = p(s_0) \prod_{t=0}^{\infty} p(s_{t+1} \mid s_t, a_t) \pi^{\text{sub}}(a_t \mid s_t, w_t) q^h(w_t \mid s_t, g).$$

Rather than jointly learning low- and high-level policies, RIS bootstraps from a target subpolicy π^{sub}, which is treated as fixed. Using these definitions, we define the evidence lower bound (ELBO) on the likelihood of subgoal optimality $p_\pi(U = 1)$ for policy π

$$\begin{aligned}
\log p_\pi(U = 1) &= \log \int p(g) p(U = 1, \tau, \{w\} \mid g) d\tau d\{w\} dg \\
&= \log \int p(g) q^f(\tau \mid g) q^h(\{w\} \mid g) \frac{p(U = 1, \tau, \{w\} \mid g)}{q^f(\tau \mid g) q^h(\{w\} \mid g)} d\{w\}\, d\tau\, dg \\
&= \log \mathbb{E}_{q^f(\tau|g), q^h(\{w\}|g), p(g)} \left[\frac{p(U = 1, \tau, \{w\} \mid g)}{q^f(\tau \mid g) q^h(\{w\} \mid g)} \right] \\
&\geq \mathbb{E}_{q^f(\tau|g), q^h(\{w\}|g), p(g)} \log \left[\frac{p(U = 1, \tau, \{w\} \mid g)}{q^f(\tau \mid g) q^h(\{w\} \mid g)} \right] = \mathcal{J}(q, \pi).
\end{aligned}$$

Expanding the fraction, moving the log inside, and dropping the start state distribution $p(s_0)$, transition distributions $p(s_{t+1} \mid s_t, a_t)$, and target subpolicy $\pi^{\text{sub}}(a_t \mid s_t, w_t)$, which are fixed with respect to the variationals, leaves us with

$$\mathbb{E}_{q^f(\tau|g),\, q^h(\{w\}|g),\, p(g)} \left[\beta \sum_{t=0}^{\infty} \gamma^t A(s_t, w_t, g) + \sum_{t=0}^{\infty} \log \left[\frac{\pi^\ell(a_t \mid s_t, w_t)\, \pi^h(w_t \mid s_t, g)}{q^f(a_t \mid s_t, g)\, q^h(w_t \mid s_t, g)} \right] \right]$$

Once again, we express the discounted sum over time as an expectation over the discounted stationary state distribution $\mu(s)$ and omit the conditionals in the expectation over $q^f(a)$ and $q^h(w)$ for brevity. Simplifying gives us

$$\begin{aligned}
&\mathbb{E}_{\mu(s), q^f(a), q^h(w),\, p(g)} \left[\beta A(s, w, g) + \log \left[\frac{\pi^h(w \mid s, g)}{q^h(w \mid s, g)} \right] + \log \left[\frac{\pi^\ell(a \mid s, w)}{q^f(a \mid s, g)} \right] \right] \\
&= \mathbb{E}_{\mu(s), q^h(w),\, p(g)} \left[\beta A(s, w, g) \right] - \mathbb{E}_{\mu(s),\, p(g)} \left[D_{\text{KL}} \left(q^h(w \mid s, g) \| \pi^h\left(w \mid s, g\right) \right) \right] \\
&\quad + \mathbb{E}_{\mu(s), q^h(w),\, p(g)} \left[D_{\text{KL}} \left(q^f\left(a \mid s, g\right) \| \pi^\ell\left(a \mid s, w\right) \right) \right]
\end{aligned}$$

RIS: RIS partitions this objective into two parts and optimizes them separately. The first line is identical to the HIQL high-level policy objective and yields the same AWR-like objective for a parametric generative subgoal policy π_ψ^h.

$$\mathcal{J}^h(\psi) = \mathbb{E}_{\mu(s), q^h(w), p(g)} \left[\exp(A(s, w, g)) \log \pi_\psi^h(w \mid s, g)\right].$$

The remaining KL divergence term seeks to minimize the divergence between the flat goal-conditioned policy posterior q^f and the *subgoal*-conditioned policy $\pi^\ell(a \mid s, w)$, which is fit separately with the target subpolicy $\pi^{\text{sub}}(a \mid s, w)$ from earlier. Importantly, because $\pi^{\text{sub}}(a \mid s, w)$ is (presumably) a parametric distribution, we can directly optimize the KL divergence in the space of policies with another parametric policy $\pi_\theta(a \mid s, g)$, yielding the final bootstrapping objective

$$\mathcal{J}^f(\theta) = \mathbb{E}_{\mu(s), q^h(w), p(g)}[D_{\text{KL}}(\pi_\theta(a \mid s, g) \| \pi^{\text{sub}}(a \mid s, w))].$$

This divergence is minimized in expectation over optimal subgoals $q^h(w \mid s, g)$, which in turn is approximated by the parametric subgoal policy $\pi_\psi^h(w \mid s, g)$.

SAW: Instead of fitting a generative model $q^h(w \mid s, g)$ over the potentially high-dimensional space of subgoals, we can use a simple application of Bayes' rule to directly approximate the expectation over the optimality-conditioned distribution of subgoals $p(w \mid s, g, U = 1)$, where

$$\begin{aligned} p(w \mid s, g, U = 1) &\propto p^{\mathcal{D}}(w \mid s) p(U = 1 \mid s, w, g) \\ &\propto p^{\mathcal{D}}(w \mid s) \exp(A(s, w, g)). \end{aligned}$$

Although the proportionality constant in the first line is the $p_\pi(U = 1)$, which is the subject of our optimization, we note that approximating the expectation over subgoals w corresponds to the expectation step in a standard expectation-maximization (EM) procedure (Abdolmaleki et al., 2018). Because we are only seeking to fit the shape of the optimal variational posterior over subgoals for the purposes of approximating the expectation over q^h, and not maximizing $p_\pi(U = 1)$ (the M step), we can treat $p_\pi(U = 1)$ as constant with respect to q^h to get

$$\begin{aligned} \mathcal{J}^f(\theta) &= \mathbb{E}_{\mu(s), q^h(w), p(g)}[D_{\text{KL}}(\pi_\theta(a \mid s, g) \| \pi^\ell(a \mid s, w))] \\ &= \int \mu(s)\, p(g)\, q^h(w \mid s)[D_{\text{KL}}(\pi_\theta(a \mid s, g) \| \pi^\ell(a \mid s, w))] dw\, dg\, ds \\ &\propto \int \mu(s)\, p(g)\, p^{\mathcal{D}}(w \mid s) \exp(A(s, w, g))[D_{\text{KL}}(\pi_\theta(a \mid s, g) \| \pi^\ell(a \mid s, w))] dw\, dg\, ds \\ &= \mathbb{E}_{\mu(s), p^{\mathcal{D}}(w \mid s), p(g)} \exp(A(s, w, g))[D_{\text{KL}}(\pi_\theta(a \mid s, g) \| \pi^\ell(a \mid s, w))], \end{aligned}$$

giving us the final subgoal advantage-weighted bootstrapping term in Equation 6.

E Computational Resources

All experiments were conducted on a cluster consisting of Nvidia GeForce RTX 3090 GPUs with 24 GB of VRAM and Nvidia GeForce RTX 3070 GPUs with 8 GB of VRAM. State-based experiments take around 4 hours to run for the largest environments (`humanoidmaze-giant-navigate`) and visual experiments up to 12 hours.

F Implementation Details

Target policy: While Chane-Sane et al. (2021) use a exponential moving average (EMA) of the online policy parameters θ as the target policy prior $\pi_{\overline{\theta}}$, we instead simply train a smaller policy network parameterized separately by ψ on (sub)goals sampled from k steps into the future, where k is a hyperparameter. We find that this leads to faster training and convergence, albeit with a small increase in computational complexity.

Architecture: During our experiments, we observed that the choice of network architecture for both the value function and policy networks had a significant impact on performance in several environments. Instead of taking in the raw concatenated state and goal inputs, HIQL prepends a subgoal representation module consisting of an additional three-layer MLP followed by a bottleneck layer of dimension 10 and a length-normalizing layer that projects state-(sub)goal representations to the surface of a hypersphere with radius equal to the dimension of the input vector. The value and low-level policy networks receive this representation in place of the goal information, as well as the raw (in state-based environments) or encoded (in pixel-based environments) state information. We found that simply adding these additional layers (separately) to the value and actor network encoders significantly boosted performance in state-based locomotion tasks, with modifications to the former improving training stability in `pointmaze` and modifications to the latter being critical for good performance in the `antmaze` and `humanoidmaze` environments.

While we did not perform comprehensive architectural ablations due to computational limitations, we note that the desirable properties of the unit hypersphere as a representation space are well-studied in contrastive learning (Wang & Isola, 2020) and preliminary work by Wang et al. (2025) has explored the benefits of scaling network depth for GCRL (albeit with negative results for the offline setting). Further studying the properties of representations emerging from these architectural choices may inform future work in representation learning for offline GCRL.

G Hyperparameters

We find that our method is robust to hyperparameter selection for different horizon lengths and environment types in locomotion tasks, but is more sensitive to choices of the value learning expectile parameter τ and the temperature parameter β of the divergence term in manipulation tasks (see Supplementary Section I for training curves of different β settings). Unless otherwise stated in Table 2, all common hyperparameters are the same as specified in Park et al. (2024a) and state, subgoal, and goal-sampling distributions are identical to those for HIQL.

Environment Type	Dataset	Expectile τ	AWR α	KLD β	Subgoal steps k
pointmaze	pointmaze-medium-navigate-v0	0.7	3.0	3.0	25
	pointmaze-large-navigate-v0	0.7	3.0	3.0	25
	pointmaze-giant-navigate-v0	0.7	3.0	3.0	25
antmaze	antmaze-medium-navigate-v0	0.7	3.0	3.0	25
	antmaze-large-navigate-v0	0.7	3.0	3.0	25
	antmaze-giant-navigate-v0	0.7	3.0	3.0	25
humanoidmaze	humanoidmaze-medium-navigate-v0	0.7	3.0	3.0	100
	humanoidmaze-large-navigate-v0	0.7	3.0	3.0	100
	humanoidmaze-giant-navigate-v0	0.7	3.0	3.0	100
visual-antmaze	visual-antmaze-medium-navigate-v0	0.7	3.0	3.0	25
	visual-antmaze-large-navigate-v0	0.7	3.0	3.0	25
	visual-antmaze-giant-navigate-v0	0.7	3.0	3.0	25
cube	cube-single-play-v0	0.9	3.0	0.3	10
	cube-double-play-v0	0.7	3.0	1.0	10
	cube-triple-play-v0	0.7	3.0	1.0	10
scene	scene-play-v0	0.7	3.0	1.0	10
visual-cube	visual-cube-single-play-v0	0.7	3.0	3.0	10
	visual-cube-double-play-v0	0.7	3.0	3.0	10
	visual-cube-triple-play-v0	0.7	3.0	3.0	10
visual-scene	visual-scene-play-v0	0.7	3.0	3.0	10

Table 2: **SAW hyperparameters.** Each cell indicates the hyperparameters for the corresponding environment and dataset. From left to right, these hyperparameters are: the expectile parameter τ for GCIVL, the one-step AWR temperature α (used for training both the target and policy networks), the temperature on the KL divergence term β, and the number of subgoal steps k.

H Waypoint Advantage Estimation

In this section, we report results for goal-conditioned waypoint advantage estimation (GCWAE), which uses a subgoal generator to produce an advantage estimator. Our implementation uses the exact same value and high-level policy training objective and architecture as HIQL, but instead evaluates the advantage of dataset actions with respect to “imagined” subgoals en route to the goal to provide a clearer signal for policy learning [Equation 4]. We show that this approach is able to achieve significantly better performance than its one-step counterpart (GCIVL) in long-horizon locomotion, but continues to lag in behind HIQL and struggles in manipulation tasks. In our experiments, we found that all training statistics were nearly identical to those of HIQL, except for the *action advantage*, as shown in the figures below.

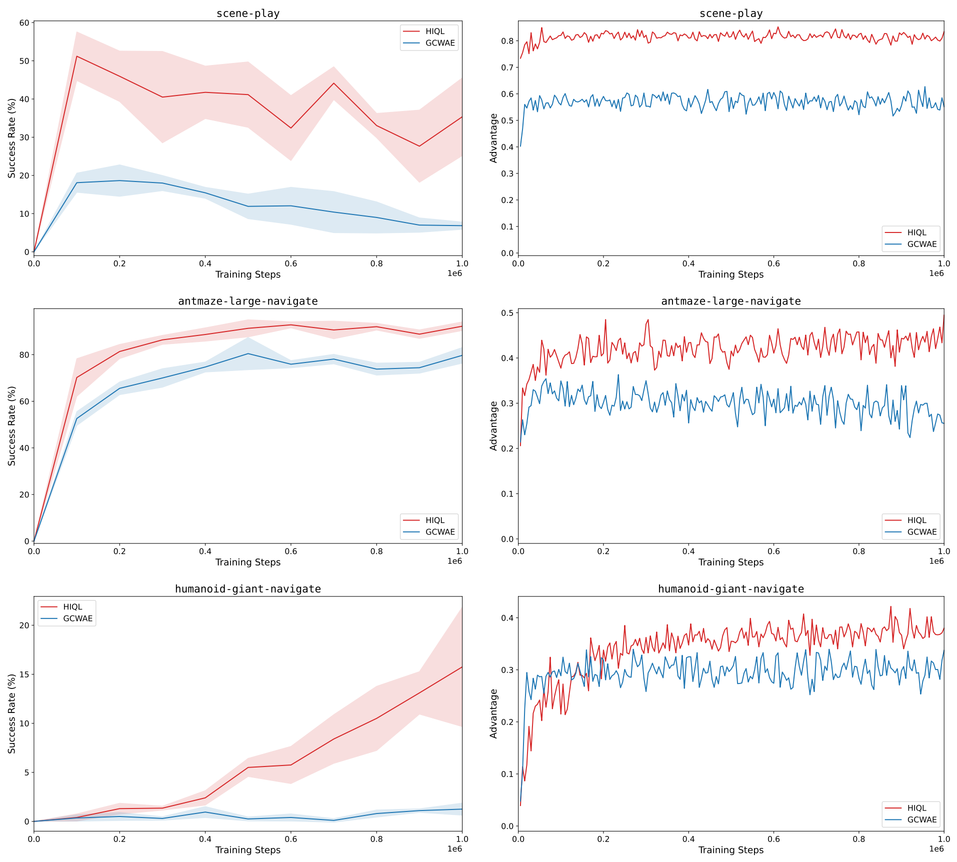

Figure 4: Training curves for `scene-play`, `antmaze-large-navigate`, and `humanoid-giant-navigate` on the left, and the mean one-step advantage over dataset actions on the right.

I Ablations

I.1 One-step AWR ablation

We ablate the one-step AWR term in our objective, which is akin to pure distillation from a target subpolicy. Note that ablating the bootstrapping term simply recovers the **GCIVL** baseline. We observe that ablations to the one-step term primarily affect performance in short-horizon, stitching-heavy manipulation environments. On the other hand, performance is largely unaffected in longer-horizon manipulation and locomotion tasks, confirming that the bulk of SAW's performance in more complex tasks is due to policy bootstrapping rather than the one-step policy extraction signal.

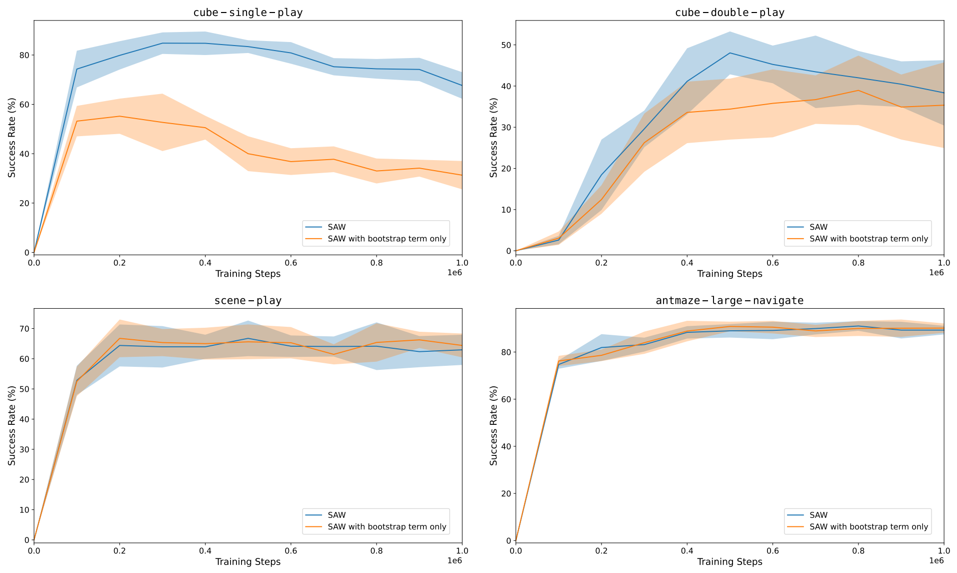

Figure 5: Training curves for `cube-single-play`, `cube-double-play`, `scene-play`, and `antmaze-large-navigate` with one-step ablations.

I.2 Hyperparameter sensitivity

Here, we investigate SAW's sensitivity to the β inverse temperature hyperparameter and run different settings of $\beta \in \{0.3, 1.0, 3.0, 10.0\}$ across selected state-based environments. We observe a similar pattern to the one-step AWR ablation experiments, where the simpler manipulation environments are much more sensitive to hyperparameter settings compared to more complex, long-horizon tasks.

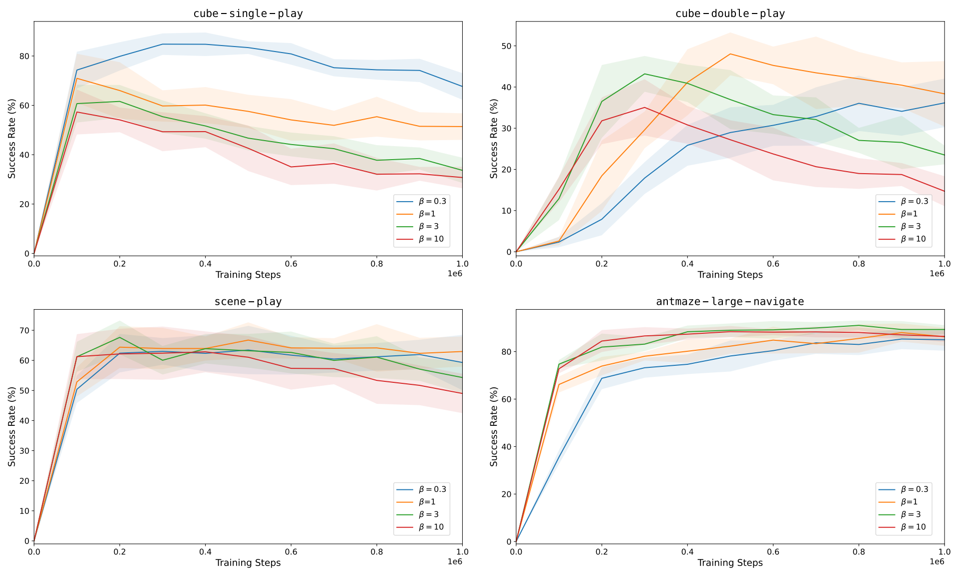

Figure 6: Training curves for `cube-single-play`, `cube-double-play`, `scene-play`, and `antmaze-large-navigate` with different values of β.