OFFLINE QUALITY-DIVERSITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline reinforcement learning methods learn a single policy from a static dataset, but in doing so fail to leverage the behavioral diversity the dataset may contain. Meanwhile, Quality-Diversity (QD) methods produce rich archives of diverse high-performing behaviors, but require online environment interactions, which poses a fundamental barrier when trial-and-error is costly or unsafe. In this paper, we try to reach the best of both worlds: the same dataset that yields a single offline RL policy could, in principle, yield an entire repertoire that QD methods could find. We introduce Offline Quality-Diversity (Offline QD), a new problem setting in which the goal is to recover a full behavioral repertoire over a given descriptor space from a static dataset, without any environment interaction during training. We propose three independent methods that each solve this differently: successor features that act as a proxy for a policy's achieved descriptor; a learned world model that lets a standard QD algorithm search entirely in imagined rollouts; and a behavior-conditioned decision transformer that recovers an archive through sequence modeling alone. Experiments show that Offline QD recovers a substantial fraction of online QD performance on tasks where exploration is easy, and substantially surpasses online QD on tasks where random exploration is hard but the offline dataset provides more diversity. We further show that offline archives serve as a strong initialization for online search, turning environments where online QD struggles from scratch into tractable ones. Our results suggest that offline datasets already contain far more behavioral diversity than standard offline RL methods extract, opening a new direction for QD research beyond online exploration.

1 INTRODUCTION

Quality-Diversity (QD) algorithms search for archives of solutions that are not just high-performing but behaviorally diverse, discovering many distinct ways to solve a task rather than converging to a single optimum. This diversity is valuable as it yields robustness to unforeseen conditions (Cully et al., 2015), an archive of behaviors to select from at deployment time (Cully & Demiris, 2018), and a backup when any single solution fails. Every existing QD method, however, shares the same underlying assumption: producing this archive requires online interaction with the environment, evaluating and re-evaluating candidate behaviors during the search. Improvements over the years, whether through better sample efficiency (Nilsson & Cully, 2021; Lim et al., 2022b; 2023), large-scale parallelization (Lim et al., 2022a), or archive distillation (Rusu et al., 2015; Faldor et al., 2023), have all operated within this assumption rather than removing it. In many of the settings where behavioral diversity is most valuable, robotics (Cully et al., 2015; Chatzilygeroudis et al., 2021), healthcare (Gottesman et al., 2019), or any domain where trial-and-error is costly, slow, or unsafe, online interaction is exactly what is hardest to get. At the same time, large offline datasets of pre-collected experience already exist in these domains, and offline Reinforcement Learning (RL) has shown that useful policies can be extracted from such data without any further interaction. But offline RL, as typically framed, discards the diversity a QD archive is built to preserve: it only optimizes a single reward-maximizing policy from the dataset. Yet the reward-maximizing behavior is only one of potentially many behaviors present in the data. So the same dataset that yields a single offline RL policy could, in principle, yield an entire repertoire, one that the standard single-policy objective of offline RL never aims to recover.

In this paper, we first formalize this idea as Offline Quality-Diversity (Offline QD): given a static dataset, a fitness function, and a behavioral descriptor space, one needs to recover an approximation

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.

of the optimal QD archive without any environment interaction during training. The central obstacle to offline QD is that verifying whether a policy achieves a target behavior requires rolling it in the environment, which is forbidden in the offline setting. Without online interaction, there is no direct way to tell whether a candidate policy complies with its intended behavior from the dataset alone. Building on this formulation, we then explore three independent ways of supplying that missing signal, each grounded in a different paradigm for learning from offline data: Successor Features (SF) that act as a proxy for a policy's achieved descriptor; a learned world model that lets a standard QD algorithm search entirely inside imagined rollouts; and a behavior-conditioned decision transformer that recovers an archive through sequence modeling over the trajectories of a dataset. Across six environments spanning locomotion, navigation, and manipulation, we find that Offline QD is consistently tractable. Our methods recover a substantial fraction of the online QD ceiling on tasks where exploration is easy, and exceed online QD by nearly $30\times$ on tasks where exploration is hard and the offline dataset already captures relevant diversity. We further show that offline archives are a strong initialization for subsequent online search, turning environments where online QD struggles from scratch into ones it solves efficiently.

Our contributions are the following:

- We formalize the **Offline QD problem**: given a static dataset, a fitness function, and a behavioral descriptor, recover an approximation of the optimal QD archive without any environment interaction during training.
- We explore **three independent approaches** to this problem, each grounded in a different paradigm for learning from offline data: (i) SF as an offline proxy for behavioral compliance; (ii) model-based search inside a learned world model; and (iii) sequence modeling over the trajectories of the dataset.
- We conduct **extensive experiments** across six environments and 15 datasets of varying quality, showing that Offline QD performance tracks and, in hard-exploration settings, substantially exceeds online QD.
- We show that offline archives, even when imperfect on their own, serve as a strong initialization for subsequent online search.

2 Related work

QD algorithms recover archives of diverse, high-performing behaviors, tracing back to novelty search (Lehman & Stanley, 2011a;b) and the foundational MAP-Elites (ME) (Mouret & Clune, 2015), through methods that improve sample efficiency (Lim et al., 2023; Faldor et al., 2023; Nilsson & Cully, 2021; Grillotti et al., 2024; Fontaine et al., 2020; Fontaine & Nikolaidis, 2021; Pierrot et al., 2022) and scale (Chalumeau et al., 2024; Lim et al., 2022a; Tjanaka et al., 2023). Every existing QD method, however, requires environment interaction at some point during training. Offline RL (Levine et al., 2020; Prudencio et al., 2023) extracts policies from static datasets without further interaction. Existing methods (Kumar et al., 2020; Fujimoto & Gu, 2021; Fujimoto et al., 2019; Kumar et al., 2019a; Kostrikov et al., 2022; Peng et al., 2019; Nair et al., 2020; Chen et al., 2021) optimize a single policy for a single reward, discarding any behavioral diversity latent in the data. We combine both settings: recovering a QD archive from a static dataset with no environment interaction. Some works also adapt unsupervised skill discovery to the offline setting to recover multiple behaviors from a dataset (Mao et al., 2024; Osa & Harada, 2024), but without a specified behavioral descriptor space as in QD. In our case, we draw on three distinct paradigms for learning from offline data. SF, originally introduced for multi-task transfer (Barreto et al., 2017; Borsa et al., 2019) and previously used to enforce behavioral compliance online (Grillotti et al., 2024), can be adapted to act as an offline proxy for behavioral compliance. Joint-Embedding Predictive Architectures (JEPA) (LeCun et al., 2022; Assran et al., 2023; Bardes et al., 2024) allow search to be run inside a learned latent-space model rather than the real environment. The Decision Transformer (Chen et al., 2021; Janner et al., 2021; Zheng et al., 2022) and the QD-Transformer (Macé et al., 2023) show that both return-conditioning and descriptor-conditioning can be cast as sequence modeling, which we combine into a single objective. We discuss each line of work in more depth in Appendix B.

3 PROBLEM FORMULATION

Let $\mathbb{S}$ be a state space, $\mathbb{A}$ an action space, and $\mathcal{Z} \subset \mathbb{R}^d$ a behavioral descriptor space. We assume that all descriptors in $\mathcal{Z}$ are reachable. A policy $\pi : \mathbb{S} \to \Delta(\mathbb{A})$ induces a distribution over trajectories $\tau = (s_0, a_0, s_1, a_1, \ldots) \in \mathcal{T}$, where actions are sampled from π and transitions follow the environment dynamics. The trajectory fitness $f : \mathcal{T} \to \mathbb{R}$ induces a policy fitness $f(\pi) = \mathbb{E}_{\tau\sim\pi}[f(\tau)]$, and $\phi : \mathbb{S} \to \mathbb{R}^d$ is a state feature map. The behavioral descriptor of a policy is defined as

$$\mathrm{BD}(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\mathrm{BD}(\tau)\right] \in \mathcal{Z}, \tag{1}$$

where $\mathrm{BD}(\tau)$ maps a full trajectory to a descriptor in $\mathcal{Z}$. $\mathrm{BD}(\tau)$ can take any functional form; two common instances in the literature are $\mathrm{BD}(\tau) = \phi(s_T)$ (the final-state feature) and $\mathrm{BD}(\tau) = \frac{1}{T}\sum_{t=0}^{T-1}\phi(s_t)$ (the mean feature along the trajectory) (Mouret & Clune, 2015; Chalumeau et al., 2024; Faldor et al., 2023), but the problem formulation is not restricted to either.

QD Archive. A QD archive is a mapping $\mathcal{A} : \mathcal{Z} \to \Pi$ that associates each descriptor $z \in \mathcal{Z}$ with a policy π_z satisfying the behavioral constraint $\mathrm{BD}(\pi_z) = z$. *The optimal archive* $\mathcal{A}^*$ is the archive that, in every cell, contains the highest-fitness policy among all valid archives:

$$f(\mathcal{A}^*(z)) \geq f(\mathcal{A}(z)) \qquad \forall z \in \mathcal{Z},\ \forall \mathcal{A} \text{ satisfying the definition above.} \tag{2}$$

That is, $\mathcal{A}^*$ associates each descriptor $z \in \mathcal{Z}$ with the fitness-maximizing policy that achieves it:

$$\mathcal{A}^*(z) = \arg\max_{\pi\in\Pi} f(\pi) \quad \text{s.t.} \quad \mathrm{BD}(\pi) = z \qquad \forall z \in \mathcal{Z}. \tag{3}$$

The goal of QD is to find $\mathcal{A}^*$, simultaneously maximizing the coverage over the descriptor space $\mathcal{Z}$ and maximizing fitness in every filled cell. Let $\mathcal{D} = \{\tau_j\}_{j=1}^{M}$ be a static dataset of trajectories $\tau_j = (s_0, a_0, s_1, \ldots)$ collected by an unknown behavior policy μ; where convenient, we also refer to its flattened set of transitions $\mathcal{D} = \{(s_i, a_i, s'_i, r_i)\}_{i=1}^{N}$.

Offline QD Problem. Given a static dataset $\mathcal{D}$, a fitness function $f : \mathcal{T} \to \mathbb{R}$, and a behavioral descriptor $\mathrm{BD} : \mathcal{T} \to \mathcal{Z}$, recover an approximation $\hat{\mathcal{A}} \approx \mathcal{A}^*$ with no environment interactions during training.

What makes this problem challenging is that Offline QD inherits the core challenge of offline RL: the learned policy inevitably visits state-action pairs outside the support of the dataset, where value estimates become unreliable and overestimated, requiring conservative policy extraction. This challenge is compounded in Offline QD, which must recover an entire family of policies $\{\pi_z\}_{z\in\mathcal{Z}}$ from the same fixed dataset $\mathcal{D}$, rather than the single policy standard offline RL extracts. Offline QD faces a further difficulty specific to the QD objective: computing $\mathrm{BD}(\pi)$ requires rolling out π in the environment, which the offline setting forbids.

4 APPROACHES TO OFFLINE QD

We now explore three directions as initial proofs of tractability and first stepping stones to solve the Offline QD problem.

4.1 OFFLINE QD VIA SUCCESSOR FEATURES

A first direction is to learn a proxy that approximates the behavioral descriptor of a given policy. As established in Section 3, Offline QD requires recovering a family of policies $\{\pi_z\}_{z\in\mathcal{Z}}$, each satisfying a distinct behavioral constraint. For that, we train a single behavior-conditioned policy $\pi_\theta(\cdot|z) : \mathbb{S} \times \mathcal{Z} \to \Delta(\mathbb{A})$ that takes the target descriptor z as input. At evaluation, this policy acts as a continuous implicit archive $\pi \approx \mathcal{A}^*$: any $z \in \mathcal{Z}$ can be queried to retrieve the corresponding behavior. This policy must jointly satisfy two objectives: maximize fitness $f(\pi_\theta(\cdot|z))$, and satisfy $\mathrm{BD}(\pi_\theta(\cdot|z)) = z,\ \forall z \in \mathcal{Z}$. The first objective is a standard offline RL problem. The second is the core challenge: enforcing $\mathrm{BD}(\pi_\theta(\cdot|z)) = z$ requires rolling out $\pi_\theta(\cdot|z)$ in the environment, which is forbidden in our setting. We therefore need **(i)** a compliance signal computable from $\mathcal{D}$ alone to replace the online rollout, and **(ii)** a way to combine it with reward maximization.

A compliance signal from offline data. To enforce BD compliance without rollouts, we use SF. For a policy $\pi_\theta(\cdot|z)$, the SF function $\psi : \mathbb{S} \times \mathbb{A} \times \mathcal{Z} \to \mathbb{R}^d$ is defined as the expected discounted cumulative state features:

$$\psi^{\pi_\theta}(s, a, z) = \mathbb{E}_{\pi_\theta(\cdot|z)} \left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) \,\middle|\, s_0 = s,\, a_0 = a \right]. \tag{4}$$

Unlike BD(π), the SF satisfies a Bellman equation and can therefore be trained offline from $\mathcal{D}$ alone:

$$\mathcal{L}_\psi = \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\left\| \psi(s, a, z) - \phi(s) - \gamma \bar{\psi}(s', \pi_\theta(s'|z), z) \right\|_2^2 \right], \tag{5}$$

with $\bar{\psi}$ an Exponential Moving Average (EMA) target. Following Grillotti et al. (2024), when the behavioral descriptor takes the additive form $\text{BD}(\tau) = \frac{1}{T}\sum_{t=0}^{T-1} \phi(s_t)$, ψ provides an upper bound on the BD compliance error (Proposition 1, proof in Appendix A.1):

$$\|\text{BD}(\pi_\theta(\cdot|z)) - z\|_2 \leq \mathbb{E}_{\substack{s\sim\rho_z^{\pi_\theta} \\ a\sim\pi_\theta(\cdot|s,z)}} \left[\|(1-\gamma)\psi(s, a, z) - z\|_2 \right], \tag{6}$$

where $\rho_z^{\pi_\theta}$ denotes the stationary state distribution induced by $\pi_\theta(.|z)$ itself. Computing this expectation requires rolling out $\pi_\theta(.|z)$ in the environment, which the offline setting forbids. We rather approximate it by training on states $s \sim \mathcal{D}$. Minimizing this surrogate is therefore an offline proxy for BD compliance, computable entirely from $\mathcal{D}$. For the reward-maximization side, we learn a critic $Q(s, a, z)$ conditioned on the target descriptor z, trained from $\mathcal{D}$ using Implicit Q-Learning (IQL) (Kostrikov et al., 2022), a conservative offline RL method that avoids overestimating out-of-distribution actions.

Final Objective. Naively combining reward maximization and BD compliance into a weighted sum leads to reward exploitation at the cost of compliance. We resolve this by gating the reward gradient on whether the policy is already close to its target descriptor:

$$J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\mathcal{D} \\ z\sim\mathcal{Z}}} \left[-\mathbf{1}\left[\ell(s, z) < \delta\right] \mathbb{E}_{a\sim\pi_\theta(\cdot|s,z)}\big[Q(s, a, z)\big] + \underbrace{\mathbb{E}_{a\sim\pi_\theta(\cdot|s,z)}\Big[\big\|(1-\gamma)\psi(s, a, z) - z\big\|_2\Big]}_{\ell(s,z)} \right] \tag{7}$$

where $\delta > 0$ is a compliance threshold. When $\ell \geq \delta$, only the compliance term is active; otherwise, the policy is sufficiently close to its target descriptor, and the reward gradient is enabled. We refer to this method as **Offline QDAC**.

4.2 Model-Based Offline QD: World Models for Imagined Rollouts

A second direction shifts the burden of environment interaction onto a learned world model. Instead of approximating the behavioral descriptor directly, as in Section 4.1, we can learn the environment dynamics themselves and run a standard QD algorithm entirely in imagination.

We adopt JEPA (LeCun et al., 2022), which allows us to predict transitions in a compact latent space rather than the observation space. We train four components from the static dataset $\mathcal{D}$: an encoder $E_\theta : \mathbb{S} \to \mathcal{H}$ mapping states into the latent space $\mathcal{H}$, a latent predictor $P_\theta : \mathcal{H} \times \mathbb{A} \to \mathcal{H}$, a reward head $R_\theta : \mathcal{H} \times \mathbb{A} \to \mathbb{R}$, and a state feature head $\Phi_\theta : \mathcal{H} \to \mathbb{R}^d$ predicting the per-state feature $\phi(s)$. We use an EMA target encoder E_ξ (Grill et al., 2020; He et al., 2020) combined with a stop-gradient on the target branch to prevent representation collapse (Chen & He, 2021; Caron et al., 2021).

Once trained, $(E_\theta, P_\theta, R_\theta, \Phi_\theta)$ together define a surrogate environment operating entirely in latent space. A QD algorithm, typically ME, is then run entirely on this surrogate: starting from an initial real state $s_0 \sim \mathcal{D}$ encoded as $h_0 = E_\theta(s_0)$, candidate policies are evaluated by rolling out full episodes in imagination, generating a predicted latent trajectory $\hat{\tau} = (h_0, a_0, \hat{h}_1, a_1, \ldots, \hat{h}_T)$ with $\hat{h}_{t+1} = P_\theta(\hat{h}_t, a_t)$, and computing predicted fitness f and behavioral descriptor BD via R_θ and Φ_θ applied to $\hat{h}_t$. We refer to this method as **JEPA-ME**.

Compounding Error. This approach inherits a well-known failure mode of model-based RL: since each predicted latent $\hat{h}_{t+1}$ is generated from the previous predicted latent $\hat{h}_t$ rather than from

the true environment, one-step prediction errors compound over the length of the rollout. This is a substantially harder regime than in most world-model planning approaches, which replan over short horizons against a freshly observed real state at every step. Our setting affords no such correction: the QD search process evaluates full episodes entirely in imagination, without any real environment interactions. To partially mitigate this, P_θ is trained with a k-step rollout-consistency objective rather than a purely 1-step loss:

$$\mathcal{L}_{\text{rollout}} = \frac{1}{k}\sum_{j=1}^{k}\left\|\hat{h}_j - \text{sg}\big(E_\xi(s_{t+j})\big)\right\|_2^2, \qquad \hat{h}_j = P_\theta(\hat{h}_{j-1}, a_{t+j-1}),\ \ \hat{h}_0 := E_\theta(s_t),$$

directly optimizing it to remain well-behaved when consuming its own prior predictions (see Appendix D.1.3 for more details).

4.3 Sequence Modeling Offline QD: BD-Conditioned Decision Transformers

An alternative to value estimation or world models is to cast Offline QD as a sequence modeling problem (Kumar et al., 2019b; Schmidhuber, 2019; Emmons et al., 2022). We treat the dataset $\mathcal{D}$ as a distribution of achieved behaviors and train a model to autoregressively predict actions conditioned on desired behaviors. The Decision Transformer (Chen et al., 2021) introduced this idea for standard offline RL: a causal transformer (Vaswani et al., 2017) is trained to predict the action a_t at each timestep from the preceding sequence of states, actions, and a single scalar return-to-go. QD-Transformer (Macé et al., 2023) also uses a similar idea to distill an archive into a transformer by conditioning it on target behavioral descriptors (without rewards). We extend both formulations by conditioning the transformer jointly on a target return and a target behavioral descriptor, so that the generated trajectory is steered toward both high fitness and a desired behavior. We refer to this method as **BD-DT**. BD-DT therefore conditions every timestep on two targets simultaneously: the trajectory's behavioral descriptor $z_\tau = \text{BD}(\tau)$, and the return-to-go $\hat{R}_t = \sum_{t'=t}^{T}\gamma^{t'-t}r_{t'}$. Given a sequence window of length K ending at timestep t, we interleave four tokens per timestep (the descriptor, the return-to-go, the state, and the action) into a single sequence: $\tau_K = \Big(z_\tau, \hat{R}_{t-K+1}, s_{t-K+1}, a_{t-K+1}, \ldots, z_\tau, \hat{R}_t, s_t, a_t\Big)$. The descriptor z_τ is repeated at every timestep rather than supplied once at the start of the window. Similar to prior work, we train a causal transformer π_θ to predict a_t from the preceding context:

$$\mathcal{L}_{\text{BD-DT}} = -\mathbb{E}_{\tau\sim\mathcal{D}}\left[\sum_{t=0}^{T-1}\log\pi_\theta(a_t \mid \tau_{<t})\right], \tag{8}$$

where $\tau_{<t}$ is causal context (the descriptor, return-to-go, state, and action tokens for all timesteps preceding t including $\{z_t, R_t, s_t\}$). During evaluation, to query the archive for a target descriptor $z \in \mathcal{Z}$, the model is prompted with z and an initial return-to-go $\hat{R}_0(z)$, set to the 90th percentile of returns among dataset trajectories whose descriptor lies near z (Appendix D.1.4). The rollout then proceeds autoregressively in the real environment: at each step, $\hat{R}_t$ is updated as $\hat{R}_{t+1} = (\hat{R}_t - r_t)/\gamma$ before the next action is queried, steering the model toward trajectories that are simultaneously high-return and on-target. See Appendix D.1.4 for more details.

5 Experiments

In this section, we empirically validate the effectiveness of the three proposed approaches in solving the Offline QD problem. We structure our analysis around the following questions:

1. Can any of the three approaches recover a meaningful QD archive from offline data alone, and how close do they get to online QD (Section 5.1)?
2. Which approach performs best on which environment, and why (Section 5.2)?
3. Does an offline QD archive provide a useful initialization for further online search, and if so, in which regimes (Section 5.3)?

Table 1: **QD score across environments and datasets.** Mean ± std over 5 seeds. Online methods are dataset-independent and shown once per environment. **Bold** marks the best offline method per row. ME (online) is colored green when it beats the best offline method, red when it does not.

Environment	Dataset	BD-DT	JEPA-ME	Offline QDAC	QDAC (online)	ME (online)
Pointmass (last state)	Uniform	1532±162	**2228**±295	2042±302	2135±271	2321±254
Pointmass (range)	Uniform	1764±279	**1926**±273	487±46	418±42	2019±202
Halfcheetah	RND	86±14	10±2	**231**±32	2134±227	3221±415
	Replay	447±85	22±3	**2411**±245		
	Expert	974±156	83±16	**2461**±203		
Walker	RND	49±6	7±1	**59**±10	156±26	361±53
	Replay	60±7	3±1	**91**±12		
	Expert	75±14	9±2	**109**±14		
Ant_uni	RND	16±2	9±1	**23**±4	42±6	51±7
	Replay	31±5	12±2	**39**±7		
	Expert	32±5	13±2	**38**±6		
Antmaze-medium	Explore	482±48	0±0	**532**±58	12±2	26±4
	Stitch	**778**±132	0±0	696±136		
	Navigate	448±59	0±0	**638**±102		
Cube-single	Noisy	75±15	0±0	**226**±42	18±3	15±3
	Play	112±13	0±0	**297**±39		

Experimental Setup

Environments. We evaluate on six environments of increasing complexity. **Pointmass**: a pointmass navigation task with two descriptor variants, **last-state** (BD is the agent's final position) and **range** (BD is the extent of positions visited, i.e. the max minus the min along each axis); **Halfcheetah**, **Walker**, and **Ant_uni** from QDax (Chalumeau et al., 2024): continuous locomotion tasks where the BD encodes foot contact ratios and fitness is forward velocity; **Antmaze-medium** from OGBench (Park et al., 2025): a navigation task where the BD is the final (x, y) position of the ant and fitness measures energy efficiency; **Cube-single** from OGBench: a contact-rich manipulation task where the BD is the final position and height of a cube and fitness measures energy efficiency. Together, these environments span locomotion, navigation, and manipulation, covering a broad range of offline QD challenges (See Appendix C for more details about the benchmark). The horizon length is 1000 for all environments, except 200 for Pointmass.

Datasets. For Pointmass, we generate a dataset with uniformly random actions; for QDax environments, we generate datasets at three quality levels, mirroring the taxonomy of D4RL (Fu et al., 2020): **RND** datasets are collected using Random Network Distillation (Burda et al., 2018); **Expert** datasets are obtained by running ME to convergence, then generating trajectories from each policy in the final archive; **Replay** datasets consist of trajectories collected throughout the ME run; For **Antmaze-medium** and **Cube-single**, we use the public OGBench datasets.

Metrics. We report the **QD score** as our primary metric: the sum of fitness across all archive cells, counting only *compliant* skills, those for which the achieved descriptor satisfies $\|\text{BD}(\pi(\cdot|z)) - z\| < \delta_{\text{eval}}$. Non-compliant solutions contribute neither fitness nor coverage to the score, so QD score implicitly penalizes both low fitness and poor behavioral compliance. In Section 5.1, we also report the **skill distance** $\|\text{BD}(\pi(\cdot|z)) - z\|$ to evaluate compliance independently from fitness.

5.1 Main Results

We want to establish whether the proposed approaches recover a meaningful behavioral repertoire from offline data and how close they come to online QD methods with online environment access.

Experiment. We train Offline QDAC, JEPA-ME, and BD-DT on every environment and dataset in the experimental setup, and report the QD score. We also include online QDAC and ME as references. Results are averaged over 5 seeds, with standard deviation reported alongside the mean.

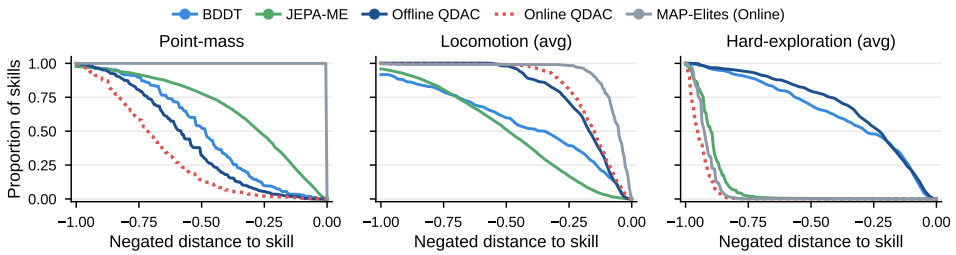

Figure 1: **Behavioral compliance across methods.** Negated skill distance $-\|\mathrm{BD}(\pi(\cdot|z)) - z\|$ over a fixed grid of queried descriptors z; further right is better compliance.

To verify the behavioral compliance of each method, we additionally query a fixed grid of target descriptors z spanning $\mathcal{Z}$, roll out the resulting policy in the environment, and record the skill distance $\|\mathrm{BD}(\pi(.|z)) - z\|$, following Grillotti et al. (2024). We report distance profiles (Figure 1) showing, for each distance threshold d, the proportion of queried skills achieving $\|\mathrm{BD}(\pi(.|z)) - z\| < d$.

Results. Table 1 shows that offline QD is tractable across most of the domains we tested, with at least one of the three proposed methods constructing a diverse and high-performing repertoire in each environment. In the low-dimensional Pointmass environments, the offline approaches achieve near-parity with online methods: JEPA-ME recovers 96% of the ME ceiling on Pointmass (last state). Similarly, on Pointmass (range), BD-DT and JEPA-ME achieve scores closely matching the ME performance.

On the more complex locomotion tasks (Halfcheetah, Walker and Ant_uni), the offline approaches still recover a significant fraction of the online performance ceiling: Offline QDAC reaches 2461 on Halfcheetah Expert, recovering 76% of ME (3221) and surpassing online QDAC (2134).

Conversely, in hard-exploration domains (Antmaze-medium and Cube-single), offline methods drastically outperform online baselines: ME scores only 26 on Antmaze-medium against BD-DT's 778 (nearly $30\times$), and 15 on Cube-single against Offline QDAC's 297. This performance gap illustrates a key advantage of offline QD: rather than being bottlenecked by online exploration in hard-exploration domains, offline methods can leverage the diverse trajectories already latent in static datasets.

Figure 1 shows that compliance varies substantially across methods and environments. Offline QDAC achieves the closest compliance on locomotion tasks (Halfcheetah, Walker), though it remains behind online methods there. On Antmaze-medium and Cube-single, offline methods match or exceed online compliance. JEPA-ME achieves strong compliance only on Pointmass, degrading sharply everywhere else. BD-DT achieves strong compliance on Antmaze-medium but degrades on locomotion. We discuss the per-method causes behind these compliance patterns in Section 5.2.

Takeaway: Offline QD is tractable across most tasks tested, achieving near-parity on simple tasks, recovering up to 76% of the online ceiling in locomotion, and substantially outperforming online baselines in hard-exploration domains by leveraging latent dataset diversity.

In Appendix E.1 we compare against three stronger online QD methods and confirm the same conclusions hold.

5.2 Comparing the Three Approaches

Offline QDAC. Offline QDAC achieves the highest QD score on Halfcheetah, Walker, Ant_uni, Cube-single, and two of three Antmaze-medium datasets (Table 1). This follows from SF's compliance guarantee (Eq. 6), which holds for additive behavioral descriptors, $\mathrm{BD}(\tau) = \frac{1}{T}\sum_{t=0}^{T-1}\phi(s_t)$. Halfcheetah, Walker, and Ant_uni all use the foot-contact ratio as their descriptor, which satisfies this form. Antmaze's final position and Cube-single's position/height descriptors are not additive, yet Offline QDAC remains strong on both: we hypothesize this is because minimizing the SF compliance loss for a final-state target implicitly rewards reaching z and remaining there, since arriving late or leaving early dilutes the cumulative feature away from z. The clearest failure case for Offline QDAC is Pointmass (range), a genuinely non-additive descriptor (487 vs. $1764/1926$ for the other

two methods, Table 1). Online QDAC also fails on this task, confirming a representational limit of the SF formulation rather than an offline-specific one.

JEPA-ME. JEPA-ME performs best on Pointmass, reaching nearly 96% of ME on both variants: because the environment dynamics are simple and short horizon, the latent predictor P_θ stays accurate over the full imagined rollout. On the higher-dimensional locomotion tasks (Halfcheetah, Walker, Ant_uni), JEPA-ME only recovers a small fraction of ME and scores 0 on Antmaze-medium and Cube-single. These environments combine long horizons (1000 steps) with contact-rich dynamics, where prediction error compounds with no mechanism to correct drift outside the training distribution. We verify this in Appendix E.7, showing QD score decreases monotonically as imagined rollout length increases.

BD-DT. BD-DT is trained by imitation on trajectories labeled with their own realized descriptor, with no bootstrapped value estimate and no dynamics model to extrapolate beyond the dataset. Its output quality therefore depends strongly on the data quality: on Halfcheetah, QD score rises $86 \rightarrow 447 \rightarrow 974$ from RND to Replay to Expert, a steeper dependence than Offline QDAC's ($231 \rightarrow 2411 \rightarrow 2461$). This means that BD-DT requires only that the target descriptor be realized, or approximately reconstructable, by some trajectory in the dataset, making it the structurally simplest of the three approaches at the cost of an inability to generalize beyond dataset coverage. Consistent with this trade-off, BD-DT is rarely the best performer (it wins only on Antmaze-medium/stitch), but remains decent everywhere else (not failing like JEPA-ME). This shows that offline QD can be achieved through sequence modeling alone.

Takeaway:

- **Offline QDAC** ✓ performs best when the behavioral descriptor is additive; ✗ it fails specifically on genuinely non-additive descriptors (Pointmass range). This is a representational limit of SF.
- **JEPA-ME** ✓ performs best on short-horizon environments with simple, low-dimensional dynamics (Pointmass); ✗ it fails on long-horizon, contact-rich dynamics, because the prediction error compounds with no mechanism for correction.
- **BD-DT** ✓ performs best with high-quality datasets and ✗ degrades on low-quality data, as no value bootstrap or dynamics model helps extrapolate beyond what the data already contains.

5.3 Online Search After Offline QD

The archives recovered in Section 5.1 are trained offline, but nothing prevents using them as a starting point for further online improvement when environment access eventually becomes available. A problem arises because Offline QDAC and BD-DT learn a behavior-conditioned policy $\pi_\theta(\cdot|z)$, that cannot be handed directly to a standard QD algorithm, which operates on a population of discrete, independently mutable networks rather than a single conditional network.

We therefore distill each offline policy into such a population: for a fixed grid of descriptors z, we distill $\pi_\theta(\cdot|z)$ on states from $\mathcal{D}$ into an independent policy π_{θ_z} per cell: $\mathcal{L}(\hat{\pi}_{\theta_z}) = \mathbb{E}_{s\sim\mathcal{D}}\left[\|\hat{\pi}_{\theta_z}(s) - \pi_\theta(s \mid z)\|_2^2\right]$. This population becomes the initial generation of a standard ME run, allowing us to isolate the contribution of the offline archive from what online search can achieve unassisted with the same budget.

Table 2: **QD score before and after online search.** Reported as *offline* $\rightarrow$ *after online*, mean $\pm$ std over 5 seeds.

Env	Dataset	ME (ref.)	BD-DT	Offline QDAC
Halfcheetah	RND	$3221_{\pm 415}$	$86_{\pm 14} \rightarrow 3231_{\pm 163}$	$231_{\pm 32} \rightarrow \mathbf{3247}_{\pm 244}$
	Replay		$447_{\pm 85} \rightarrow 3275_{\pm 245}$	$2411_{\pm 245} \rightarrow \mathbf{3428}_{\pm 194}$
	Expert		$974_{\pm 156} \rightarrow 3288_{\pm 188}$	$2461_{\pm 203} \rightarrow \mathbf{3442}_{\pm 210}$
Walker	RND	$361_{\pm 53}$	$49_{\pm 6} \rightarrow 374_{\pm 27}$	$59_{\pm 10} \rightarrow \mathbf{375}_{\pm 23}$
	Replay		$60_{\pm 7} \rightarrow 371_{\pm 20}$	$91_{\pm 12} \rightarrow \mathbf{381}_{\pm 26}$
	Expert		$75_{\pm 14} \rightarrow 382_{\pm 22}$	$109_{\pm 14} \rightarrow \mathbf{383}_{\pm 36}$
Antmaze	Explore	$26_{\pm 4}$	$482_{\pm 48} \rightarrow 546_{\pm 53}$	$532_{\pm 58} \rightarrow \mathbf{605}_{\pm 31}$
	Stitch		$778_{\pm 132} \rightarrow \mathbf{825}_{\pm 70}$	$696_{\pm 136} \rightarrow 797_{\pm 63}$
	Navigate		$448_{\pm 59} \rightarrow 517_{\pm 49}$	$638_{\pm 102} \rightarrow \mathbf{734}_{\pm 61}$
Cube	Noisy	$15_{\pm 3}$	$75_{\pm 15} \rightarrow 83_{\pm 6}$	$226_{\pm 42} \rightarrow \mathbf{257}_{\pm 22}$
	Play		$112_{\pm 13} \rightarrow 124_{\pm 9}$	$297_{\pm 39} \rightarrow \mathbf{337}_{\pm 29}$

Experiment. We distill the offline policies from BD-DT and Offline QDAC as described above until convergence, then run ME on the resulting population with the same budget used for ME in Table 1, and report the QD scores in Table 2.

Results. Two regimes emerge. In Halfcheetah and Walker, online search adds a modest 3.8% on average on top of the ME performance. This is expected given the good head start provided by the offline phase. In Antmaze and Cube, the gain is 12.7% on average relative to the offline starting point, while the fine-tuned archive reaches 5.5–$31.7\times$ the ME reference. This is significant because ME alone performs poorly in these environments. Initializing from the offline archive instead yields a substantial improvement. This shows that the failure of ME from scratch is not a failure of the algorithm, but a consequence of an intractably large search space. A strong initial population addresses this directly: it anchors the search around promising regions, effectively reducing the region ME must explore.

Takeaway: Offline initialization can substantially improve online QD when exploration from scratch is difficult, suggesting that the main bottleneck is discovering promising regions of the behavior space rather than improving the local search procedure itself.

5.4 Additional Experiments

Beyond the main experiments above, we conduct several additional experiments in Appendix E: **Stronger online QD baselines:** Offline QD still substantially outperforms stronger online QD methods on the hard-exploration tasks (E.1). **Offline goal-conditioned RL:** Offline QDAC and BD-DT achieve competitive behavioral compliance with a dedicated offline GCRL baseline, while additionally optimizing fitness (E.2). **Fitness vs. diversity:** Interpolating between fitness-only and diversity-only datasets shows that QD score peaks at an intermediate mixture, highlighting the need for both trajectory quality and behavioral coverage (E.3). We further analyze coverage beyond the dataset support (E.4), ablation of Offline QDAC's components (E.5), ablation of BD-DT's components (E.6), JEPA-ME horizon and capacity (E.7), the compliance threshold (E.8), and the empirical validity of the SF compliance bound (E.9).

6 Limitations

Collecting a dataset diverse enough to make Offline QD worthwhile is itself a nontrivial problem that we do not address here: our experiments assume such data is already available, and Appendix E.3 shows how performance degrades if this assumption is not met. Given such data, each method still inherits a specific weakness from the offline signal it substitutes for a rollout: Offline QDAC cannot represent non-additive behavioral descriptors and has no guarantee for final-state behavioral descriptors, JEPA-ME's imagined rollouts degrade with horizon, and BD-DT cannot generalize past what some trajectory in the dataset already reached. More broadly, all three remain bounded by what $\mathcal{D}$ demonstrates, though this bound is not strict, as shown in Appendix E.4. Moreover, none of the three methods carries any notion of certainty in its compliance, so each returns a policy for a queried z with no signal to flag when the returned behavior is likely to be unreliable, which matters most in safety-critical settings. Finally, no single method dominates across every setting we test, and all environments here are simulated with low-dimensional descriptor spaces (four at most), leaving real-world data and higher-dimensional descriptors untested.

7 Conclusion

In this paper, we introduced Offline Quality-Diversity: recovering a full behavioral repertoire from a static dataset with no environment interaction during training. The core difficulty is that behavioral compliance requires rolling out a policy inside the environment, which the offline setting forbids. We proposed three independent approaches: Offline QDAC that uses Successor Features, JEPA-ME that uses imagined rollouts inside a learned world model, and BD-DT using behavior-conditioned sequence modeling. Empirically, Offline QDAC and BD-DT recover a substantial part of online QD performance on easy-exploration tasks and surpass online QD when exploration is hard but the dataset already reflects the relevant diversity; JEPA-ME is competitive only on short-horizon, low-dimensional dynamics. Each method fails along a different predictable axis, suggesting that the right choice depends on the descriptor space, the environment dynamics, and the quality of the dataset. We also show that offline archives are a strong initialization for further online search, turning environments that are intractable for online QD into solvable ones.

AI USE STATEMENT

In this work, we did not use generative AI tools for any of the tasks requiring mandatory disclosure under the ICLR AI Policy. We used generative AI tools for several tasks with recommended disclosure: correcting writing errors and improving readability, and assisting with the literature review. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken several steps to ensure reproducibility. The full proof of the SF compliance bound (Eq. 6) is provided in Appendix A.1, including the stated assumptions. Appendix C details the environments, behavioral descriptors, fitness functions, and dataset generation procedure. Appendix D.1 describes the architectures, training procedures, and evaluation protocol for all proposed methods, and Appendix D.2 lists the complete hyperparameters used for each method and environment. Compute resources used are reported in Appendix D.3. Code will be released upon acceptance.

REFERENCES

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5048–5058, 2017.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, pp. 4055–4065, 2017.

André Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Žídek, and Rémi Munos. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *International Conference on Machine Learning*, pp. 501–510. PMLR, 2018.

André Barreto, Shaobo Hou, Diana Borsa, David Silver, and Doina Precup. Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences*, 117 (48):30079–30087, 2020.

Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies–a comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

Diana Borsa, André Barreto, John Quan, Daniel Mankowitz, Rémi Munos, Hado van Hasselt, David Silver, and Tom Schaul. Universal successor features approximators. *International Conference on Learning Representations*, 2019.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9650–9660, 2021.

Felix Chalumeau, Bryan Lim, Raphael Boige, Maxime Allard, Luca Grillotti, Manon Flageat, Valentin Macé, Guillaume Richard, Arthur Flajolet, Thomas Pierrot, et al. QDAX: A library for quality-diversity and population-based algorithms with hardware acceleration. *Journal of Machine Learning Research*, 25(108):1–16, 2024.

Konstantinos Chatzilygeroudis, Antoine Cully, Vassilis Vassiliades, and Jean-Baptiste Mouret. Quality-diversity optimization: a novel branch of stochastic optimization. In *Black Box Optimization, Machine Learning, and No-Free Lunch Theorems*, pp. 109–135. Springer, 2021.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758, 2021.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, volume 31, pp. 4754–4765, 2018.

Antoine Cully and Yiannis Demiris. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation*, 22(2):245–259, 2018.

Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline rl via supervised learning? *International Conference on Learning Representations*, 2022.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *International Conference on Learning Representations*, 2019.

Maxence Faldor, Félix Chalumeau, Manon Flageat, and Antoine Cully. Map-elites with descriptor-conditioned gradients and archive distillation into a single policy. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 138–146, 2023.

Matthew Fontaine and Stefanos Nikolaidis. Differentiable quality diversity. *Advances in Neural Information Processing Systems*, 34:10040–10052, 2021.

Matthew C Fontaine, Julian Togelius, Stefanos Nikolaidis, and Amy K Hoover. Covariance matrix adaptation for the rapid illumination of behavior space. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 94–102, 2020.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Omer Gottesman, Fredrik Johansson, Matthieu Komorowski, Aldo Faisal, David Sontag, Finale Doshi-Velez, and Leo Anthony Celi. Guidelines for reinforcement learning in healthcare. *Nature Medicine*, 25(1):16–18, 2019.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284, 2020.

Luca Grillotti, Maxence Faldor, Borja G. León, and Antoine Cully. Quality-diversity actor-critic: Learning high-performing and diverse behaviors via value and successor features critics. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16416–16459. PMLR, 21–27 Jul 2024.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *International Conference on Learning Representations*, 2020.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *International Conference on Learning Representations*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*. PMLR, 2022.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in Neural Information Processing Systems*, 34:1273–1286, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *International Conference on Learning Representations*, 2022.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32:11761–11771, 2019a.

Aviral Kumar, Xue Bin Peng, and Sergey Levine. Reward-conditioned policies. *arXiv preprint arXiv:1912.13465*, 2019b.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Yann LeCun et al. A path towards autonomous machine intelligence version 0.9.2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Joel Lehman and Kenneth O Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223, 2011a.

Joel Lehman and Kenneth O Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 211–218, 2011b.

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

Bryan Lim, Maxime Allard, Luca Grillotti, and Antoine Cully. Accelerated quality-diversity through massive parallelism. *Transactions on Machine Learning Research*, 2022a.

Bryan Lim, Luca Grillotti, Lorenzo Bernasconi, and Antoine Cully. Dynamics-aware quality-diversity for efficient learning of skill repertoires. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 5360–5366. IEEE, 2022b.

Bryan Lim, Manon Flageat, and Antoine Cully. Understanding the synergies between quality-diversity and deep reinforcement learning. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1212–1220, 2023.

Valentin Macé, Raphaël Boige, Felix Chalumeau, Thomas Pierrot, Guillaume Richard, and Nicolas Perrin-Gilbert. The quality-diversity transformer: Generating behavior-conditioned trajectories with decision transformers. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1221–1229, 2023.

Yihuan Mao, Chengjie Wu, Xi Chen, Hao Hu, Ji Jiang, Tianze Zhou, Tangjie Lv, Changjie Fan, Zhipeng Hu, Yi Wu, et al. Stylized offline reinforcement learning: Extracting diverse high-quality behaviors from heterogeneous datasets. In *International Conference on Learning Representations*, volume 2024, pp. 51217–51237, 2024.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *IEEE International Conference on Robotics and Automation*, pp. 7559–7566. IEEE, 2018.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Olle Nilsson and Antoine Cully. Policy gradient assisted map-elites. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 866–875, 2021.

Takayuki Osa and Tatsuya Harada. Discovering multiple solutions from a single task in offline reinforcement learning. *arXiv preprint arXiv:2406.05993*, 2024.

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline rl? *Advances in Neural Information Processing Systems*, 37:79029–79056, 2024.

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.

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

Thomas Pierrot, Valentin Macé, Felix Chalumeau, Arthur Flajolet, Geoffrey Cideron, Karim Beguir, Antoine Cully, Olivier Sigaud, and Nicolas Perrin-Gilbert. Diversity policy gradient for sample efficient quality-diversity optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1075–1083, 2022.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems*, 1:305–313, 1988.

Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):10237–10257, 2023.

Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 661–668, 2010.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635, 2011.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

Jürgen Schmidhuber. Reinforcement learning upside down: Don't predict rewards–just map them to actions. *arXiv preprint arXiv:1912.02875*, 2019.

Bryon Tjanaka, Matthew C Fontaine, David H Lee, Yulun Zhang, Nivedit Reddy Balam, Nathaniel Dennler, Sujay S Garlanka, Nikitas Dimitri Klapsis, and Stefanos Nikolaidis. Pyribs: A bare-bones python library for quality diversity optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 220–229, 2023.

Peter Vamplew, Richard Dazeley, Adam Berry, Rustam Issabekov, and Evan Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine learning*, 84 (1):51–80, 2011.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008, 2017.

Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *International Conference on Machine Learning*, pp. 27042–27059. PMLR, 2022.

APPENDIX TABLE OF CONTENTS

A Proofs **16**

A.1 Proof of the Successor Features Compliance Bound (Eq. 6) 16

B Extended Related Work **19**

B.1 Quality-Diversity Algorithms 19

B.2 Offline Reinforcement Learning 20

B.3 World Models and Latent-Space Planning 20

B.4 Successor Features and Successor Representations 22

C Environments and Tasks **23**

D Implementation details **25**

D.1 Architecture, Training, and Evaluation Protocol 25

D.1.1 Shared Experimental Protocol 25

D.1.2 Offline QDAC 25

D.1.3 JEPA-ME 26

D.1.4 BD-DT 27

D.2 Hyperparameters 28

D.2.1 Offline QDAC 28

D.2.2 JEPA-ME 29

D.2.3 BD-DT 30

D.2.4 Online Baselines (QDAC, ME) 30

D.3 Compute 30

E Additional Results **31**

E.1 Comparison Against Stronger Online QD Baselines 31

E.2 Comparison to Offline Goal-Conditioned RL 32

E.3 Dataset Quality: Fitness and Diversity 33

E.4 Dataset Quality: Coverage and Generalization 34

E.5 Ablating the Compliance Signal and Offline RL Backbone 35

E.6 Ablating Descriptor and Return Conditioning in BD-DT 35

E.7 World Model Horizon and Capacity Sensitivity 36

E.8 Ablation on the Compliance Threshold 37

E.9 Empirical Validity of the SF Compliance Bound 38

A PROOFS

A.1 PROOF OF THE SUCCESSOR FEATURES COMPLIANCE BOUND (EQ. 6)

Proposition 1. Consider an infinite-horizon finite MDP with observable features $\phi(s) \in \mathbb{R}^d$, and let the behavioral descriptor be the long-run average feature under the policy's stationary distribution, $\mathrm{BD}(\pi_\theta(\cdot|z)) := \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \phi(s_t)$. Let $\psi(s, a, z)$ be the corresponding Successor Features (SF), satisfying the Bellman equation:

$$\psi(s_t, a_t, z) = \phi(s_t) + \gamma \mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t) \\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}} \big[\psi(s_{t+1}, a_{t+1}, z)\big]. \tag{9}$$

Let $\rho_z^{\pi_\theta}(s) = \lim_{t\to\infty} P(s_t = s \mid s_0, \pi_\theta(\cdot \mid z))$ denote the stationary state distribution induced by $\pi_\theta(\cdot \mid z)$, assumed to exist and be independent of s_0. Then, for all $z \in \mathcal{Z}$,

$$\big\|\mathrm{BD}(\pi_\theta(\cdot|z)) - z\big\|_2 \;\le\; \mathbb{E}_{\substack{s\sim\rho_z^{\pi_\theta} \\ a\sim\pi_\theta(\cdot|s,z)}} \Big[\big\|(1-\gamma)\psi(s, a, z) - z\big\|_2\Big]. \tag{10}$$

Proof. For all skills $z \in \mathcal{Z}$, and for all trajectories $(s_0, a_0, \ldots, s_{T-1}, a_{T-1})$ sampled from $\pi_\theta(\cdot \mid z)$, we have:

$$\begin{aligned}
&\left\|\frac{1}{T}\sum_{t=0}^{T-1}\phi(s_t) - z\right\|_2 \\
&= \left\|\frac{1}{T}\sum_{t=0}^{T-1}\left(\psi(s_t, a_t, z) - \gamma\mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t) \\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}[\psi(s_{t+1}, a_{t+1}, z)] - z\right)\right\|_2 \qquad \text{(Eq. 9)} \\
&= \left\|\frac{1}{T}\sum_{t=0}^{T-1}((1-\gamma)\psi(s_t, a_t, z) - z) + \frac{\gamma}{T}\sum_{t=0}^{T-1}\left(\psi(s_t, a_t, z) - \mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t) \\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}[\psi(s_{t+1}, a_{t+1}, z)]\right)\right\|_2 \\
&\le \left\|\frac{1}{T}\sum_{t=0}^{T-1}(1-\gamma)\psi(s_t, a_t, z) - z\right\|_2 \\
&\qquad + \gamma\left\|\frac{1}{T}\sum_{t=0}^{T-1}\left(\psi(s_t, a_t, z) - \mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t) \\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}[\psi(s_{t+1}, a_{t+1}, z)]\right)\right\|_2
\end{aligned} \tag{11}$$

Under $\pi_\theta(\cdot \mid z)$, once we reach stationarity, (s_t, a_t) is distributed according to $\mu_z(s, a) := \rho_z^{\pi_\theta}(s)\,\pi_\theta(a \mid s, z)$. Drawing $s_{t+1} \sim p(\cdot \mid s_t, a_t)$ from $(s_t, a_t) \sim \mu_z$ and then $a_{t+1} \sim \pi_\theta(\cdot \mid s_{t+1}, z)$ yields $(s_{t+1}, a_{t+1}) \sim \mu_z$ as well, by definition of $\rho_z^{\pi_\theta}$ as a stationary distribution. Therefore,

$$\begin{aligned}
\mathbb{E}_{(s_t,a_t)\sim\mu_z}\big[\psi(s_t, a_t, z)\big] &= \mathbb{E}_{(s_t,a_t)\sim\mu_z}\,\mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t) \\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}\big[\psi(s_{t+1}, a_{t+1}, z)\big] \\
&= \mathbb{E}_{(s_{t+1},a_{t+1})\sim\mu_z}\big[\psi(s_{t+1}, a_{t+1}, z)\big].
\end{aligned} \tag{12}$$

Since $\mathcal{S}$ and $\mathcal{A}$ are finite, $(s_t, a_t)_{t\ge0}$ is a finite-state Markov chain, and the assumption that $\rho_z^{\pi_\theta}$ exists and is independent of s_0 means this chain is ergodic, with unique stationary distribution μ_z. By the Ergodic Theorem for finite Markov chains, for any $g : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$, $\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} g(s_t, a_t) = \mathbb{E}_{(s,a)\sim\mu_z}[g(s, a)]$ almost surely. Taking the second term of Eq. 11 to the limit, applying the Ergodic

Theorem above term-by-term, and using Eq. 12:

$$\begin{aligned}
&\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\Big(\psi(s_t,a_t,z)-\mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t)\\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}\big[\psi(s_{t+1},a_{t+1},z)\big]\Big)\\
&=\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\psi(s_t,a_t,z)-\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t)\\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}\big[\psi(s_{t+1},a_{t+1},z)\big]\\
&=\mathbb{E}_{(s_t,a_t)\sim\mu_z}\big[\psi(s_t,a_t,z)\big]-\mathbb{E}_{(s_t,a_t)\sim\mu_z}\mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t)\\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}\big[\psi(s_{t+1},a_{t+1},z)\big]\\
&=\mathbb{E}_{(s,a)\sim\mu_z}\big[\psi(s,a,z)\big]-\mathbb{E}_{(s,a)\sim\mu_z}\big[\psi(s,a,z)\big]\\
&=0. \qquad (13)
\end{aligned}$$

Consequently,

$$\lim_{T\to\infty}\gamma\left\|\frac{1}{T}\sum_{t=0}^{T-1}\Big(\psi(s_t,a_t,z)-\mathbb{E}_{\substack{s_{t+1}\sim p(\cdot|s_t,a_t)\\ a_{t+1}\sim\pi_\theta(\cdot|s_{t+1},z)}}\big[\psi(s_{t+1},a_{t+1},z)\big]\Big)\right\|_2=\|\mathbf{0}\|_2=0. \qquad (14)$$

By the Ergodic Theorem above, the first term of Eq. 11 becomes:

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\big((1-\gamma)\psi(s_t,a_t,z)-z\big)=(1-\gamma)\,\mathbb{E}_{(s,a)\sim\mu_z}\big[\psi(s,a,z)\big]-z. \qquad (15)$$

Finally, by taking Eq. 11 to the limit as $T\to\infty$, and substituting Eq. 14 and Eq. 15:

$$\begin{aligned}
\|\mathrm{BD}(\pi_\theta(\cdot|z))-z\|_2 &\le \big\|(1-\gamma)\,\mathbb{E}_{(s,a)\sim\mu_z}[\psi(s,a,z)]-z\big\|_2+0\\
&\le \mathbb{E}_{(s,a)\sim\mu_z}\Big[\big\|(1-\gamma)\psi(s,a,z)-z\big\|_2\Big] \qquad \text{(Jensen's inequality)}\\
&= \mathbb{E}_{\substack{s\sim\rho_z^{\pi_\theta}\\ a\sim\pi_\theta(\cdot|s,z)}}\Big[\big\|(1-\gamma)\psi(s,a,z)-z\big\|_2\Big], \qquad (16)
\end{aligned}$$

where the last line uses $\mu_z(s,a)=\rho_z^{\pi_\theta}(s)\,\pi_\theta(a\mid s,z)$. This establishes Eq. 10. □

Assumption 1 (Concentrability)**.** Let $d_{\mathcal{D}}$ denote the state distribution induced by the offline dataset $\mathcal{D}$. For a given skill $z\in\mathcal{Z}$, assume $d_{\mathcal{D}}$ has full support over the recurrent class of $\rho_z^{\pi_\theta}$, and define the concentrability coefficient

$$C_z \;:=\; \sup_{s\,:\,\rho_z^{\pi_\theta}(s)>0}\frac{\rho_z^{\pi_\theta}(s)}{d_{\mathcal{D}}(s)} \;<\; \infty. \qquad (17)$$

Corollary 1 (Dataset-relative SF Compliance Bound)**.** Under the setting of Proposition 1 and Assumption 1, for all $z\in\mathcal{Z}$,

$$\big\|\mathrm{BD}(\pi_\theta(\cdot|z))-z\big\|_2 \;\le\; C_z\,\mathbb{E}_{\substack{s\sim\mathcal{D}\\ a\sim\pi_\theta(\cdot|s,z)}}\Big[\big\|(1-\gamma)\psi(s,a,z)-z\big\|_2\Big]. \qquad (18)$$

Proof. Write $h_z(s):=\mathbb{E}_{a\sim\pi_\theta(\cdot|s,z)}\big[\|(1-\gamma)\psi(s,a,z)-z\|_2\big]\ge 0$. By Assumption 1,

$$\begin{aligned}
\mathbb{E}_{s\sim\rho_z^{\pi_\theta}}\big[h_z(s)\big] &= \sum_s \rho_z^{\pi_\theta}(s)\,h_z(s)\\
&= \sum_s d_{\mathcal{D}}(s)\,\frac{\rho_z^{\pi_\theta}(s)}{d_{\mathcal{D}}(s)}\,h_z(s)\\
&\le C_z\sum_s d_{\mathcal{D}}(s)\,h_z(s)\\
&= C_z\,\mathbb{E}_{s\sim\mathcal{D}}\big[h_z(s)\big]. \qquad (19)
\end{aligned}$$

Chaining Eq. 19 with Proposition 1 gives Eq. 18. □

Remark 1. Proposition 1 bounds compliance error in terms of an expectation over $s \sim \rho_z^{\pi_\theta}$, the stationary state distribution of $\pi_\theta(\cdot \mid z)$ itself. Evaluating this expectation would require rolling out $\pi_\theta(\cdot \mid z)$ in the environment, which the offline setting forbids. In Offline QDAC (Eq. 7), we substitute $s \sim \mathcal{D}$ in place of $s \sim \rho_z^{\pi_\theta}$; Corollary 1 makes this substitution formal under Assumption 1. When $\mathcal{D}$'s state coverage matches $\rho_z^{\pi_\theta}$'s support closely, the substitution is nearly free. We do not estimate C_z in practice; Appendix E.9 instead evaluates the surrogate's empirical fidelity directly.

B Extended Related Work

This appendix expands on the related work summarized in Section 2, linking each of our three approaches to its respective literature.

B.1 Quality-Diversity Algorithms

QD formalizes the search for a set of solutions that jointly cover a space of behaviors $\mathcal{Z}$ while remaining individually near-optimal within each behavior, rather than collapsing the search onto a single global optimum. This objective is useful because it does not rely on fitness maximization alone, but instead seeks high fitness across many distinct behaviors spanning the behavioral space. This is useful when the deployment environment differs from the training environment in ways not anticipated at training time, or when a single high-fitness solution is fragile to perturbation. A substantial body of work has developed around this objective, differing primarily in how candidate solutions are proposed and how efficiently the search converges to the optimal archive $\mathcal{A}^*$ of Section 3. QD's emphasis on illumination rather than optimization traces back to novelty search (Lehman & Stanley, 2011a), later combined with a local competition mechanism (Lehman & Stanley, 2011b). This is a different notion of diversity than unsupervised skill discovery methods such as DIAYN (Eysenbach et al., 2019), which learn distinguishable behaviors without a pre-specified behavioral space at all, at the cost of no explicit control over what each discovered skill represents. See (Pugh et al., 2016; Cully & Demiris, 2018; Chatzilygeroudis et al., 2021) for broader surveys of the field.

MAP-Elites (ME) (Mouret & Clune, 2015) is the foundational QD approach. It maintains an archive $\mathcal{A} : \mathcal{Z} \to \Pi$ over a discretization of the descriptor space $\mathcal{Z}$ and repeats three steps: a parent π_θ is sampled uniformly from the occupied cells of $\mathcal{A}$; a candidate $\pi_{\theta'}$ is obtained by mutating θ, typically by isotropic Gaussian perturbation (Beyer & Schwefel, 2002), or by adapting the mutation's covariance directly as in CMA-ES (Hansen & Ostermeier, 2001) and its QD variant CMA-ME (Fontaine et al., 2020); and $\pi_{\theta'}$ is evaluated by a full environment rollout, yielding $(f(\pi_{\theta'}), \mathrm{BD}(\pi_{\theta'}))$. The candidate is inserted into the cell $z = \mathrm{BD}(\pi_{\theta'})$ if and only if it improves on the cell's current occupant,

$$\mathcal{A}(z) \leftarrow \pi_{\theta'} \quad \text{if} \quad f(\pi_{\theta'}) > f(\mathcal{A}(z)) \text{ or cell } z \text{ is empty.} \tag{20}$$

When iterated to convergence, each cell converges to the highest-fitness solution found for that region of $\mathcal{Z}$, and the only mechanism available to discover a policy's descriptor is to do an online rollout, which is what Offline QD removes.

One problem is that random noise is an inefficient mutation, as it perturbs θ without using any gradient signal, which becomes increasingly sample-inefficient as its size grows. PGA-ME (Nilsson & Cully, 2021) replaces a fraction of the mutation operators with policy-gradient steps computed against a critic trained on a replay buffer of previously evaluated transitions, so that some offspring move in a direction informed by an estimated value function rather than at random. DCG-ME (Faldor et al., 2023) conditions this actor directly on the target descriptor, training a single network $\pi_\theta(\cdot \mid z)$ with a descriptor-conditioned policy gradient. Related gradient-based directions include QD-PG (Pierrot et al., 2022), which adds an explicit diversity-seeking policy gradient alongside the reward gradient rather than relying on mutation alone, and MEGA (Fontaine & Nikolaidis, 2021), which back-propagates through a differentiable quality and diversity objective to directly adapt a search distribution over parameters. The specific form used by DCG-ME is

$$\nabla_\theta J(\theta) = \mathbb{E}_{s,z}\left[\nabla_\theta \pi_\theta(s \mid z)\, \nabla_a Q(s, a, z)\big|_{a=\pi_\theta(s|z)}\right]. \tag{21}$$

A reward gradient of this form has no mechanism to keep $\pi_\theta(\cdot \mid z)$ tied to its assigned target, as nothing in (Eq. 21) penalizes the policy for drifting toward whichever behavior maximizes Q regardless of z, so descriptor-conditioned actors trained this way tend to collapse onto a small number of high-reward behaviors rather than spreading across $\mathcal{Z}$. QDAC (Grillotti et al., 2024) addresses this directly by introducing a second, descriptor-compliance critic alongside the reward critic and gating the reward gradient on whether the policy is already close to its target, so that compliance is enforced before fitness is optimized rather than being left to emerge as a side effect of reward maximization.

B.2 Offline Reinforcement Learning

Offline RL extracts a policy from a fixed dataset $\mathcal{D} = \{(s_i, a_i, s'_i, r_i)\}_{i=1}^N$ collected by an unknown behavior policy μ, without further access to the environment (Levine et al., 2020). The central obstacle is distributional shift: a policy trained to maximize a learned Q-function drifts toward actions rarely or never observed under μ, and because Q is fit only to transitions generated by μ, its estimates for these out-of-distribution actions are unconstrained and tend to be optimistic. This overestimation makes standard off-policy actor-critic methods diverge under purely offline training. To address this overestimation problem, multiple methods have been proposed in the Offline RL literature:

- Conservative Q-Learning (CQL) (Kumar et al., 2020) controls overestimation at the value level by augmenting the standard Bellman error with a regularizer that pushes Q-values down for out-of-distribution actions (including actions from the current policy), yielding a value function that lower-bounds the true Q and is therefore safe to optimize against without querying unseen actions directly.
- BCQ (Fujimoto et al., 2019) instead constrains the action space itself: a generative model of μ proposes a small set of candidate actions similar to those in $\mathcal{D}$, and the policy is restricted to selecting among these candidates rather than optimizing freely over $\mathcal{A}$.
- BEAR (Kumar et al., 2019a) similarly constrains the policy's support, but does so with an explicit maximum mean discrepancy penalty against μ rather than BCQ's generative candidate-sampling scheme, allowing more flexibility in how far the policy may deviate from the behavior distribution.
- TD3+BC (Fujimoto & Gu, 2021) shows that a much simpler recipe consisting of standard TD3 with a behavioral-cloning term added directly to the policy loss, matches or exceeds these more elaborate constructions on standard benchmarks.
- AWAC (Nair et al., 2020) and Advantage-Weighted Regression (Peng et al., 2019) instead avoid off-policy value maximization entirely, extracting the policy via return-weighted supervised regression.
- IQL (Kostrikov et al., 2022) uses an expectile regression to fit a state-value function $V(s)$ that approximates $\max_a Q(s, a)$ using only actions already present in $\mathcal{D}$:

$$\mathcal{L}_V = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[L_2^\tau\big(Q(s,a) - V(s)\big)\right], \qquad L_2^\tau(u) = |\tau - \mathbf{1}(u < 0)|\, u^2, \tag{22}$$

  with expectile $\tau > 0.5$ biasing V toward the upper portion of the Q-values already observed at s without extrapolating beyond them. Because Q and V are never evaluated at an action absent from $\mathcal{D}$, IQL sidesteps distributional shift structurally rather than constraining it, which is the property Section 4.1 relies on in choosing IQL as the reward-maximizing backbone for Offline QDAC.

A few recent works go beyond a single policy and recover several behaviors from one unlabeled offline dataset. SORL (Mao et al., 2024) clusters trajectories into a small, fixed number of styles and trains one advantage-weighted policy per cluster, while DiveOff (Osa & Harada, 2024) learns a latent-conditioned policy whose latent space is shaped by a mutual-information regularizer. In both cases, there is no predefined descriptor space. The behavior associated with a latent vector z is only known after training and evaluation. Offline QD differs in that the desired behavioral space is given through the behavioral descriptor BD, and the goal is a compliant, high-fitness policy for every $z \in \mathcal{Z}$. This also makes a direct comparison ill-posed. Since these methods never observe BD, their QD score only reflects how well the modes of the data happen to align with $\mathcal{Z}$. SORL can fill at most as many cells as it has clusters, and DiveOff's latent representation must first be mapped to $\mathcal{Z}$ through rollouts, so the method is no longer purely offline. Conversely, their diversity objective rewards distinguishable behaviors rather than coverage of a descriptor space.

B.3 World Models and Latent-Space Planning

A world model is a learned approximation $\hat{T}(s' \mid s, a)$ of the environment's transition dynamics, typically paired with a learned reward or fitness head $\hat{R}(s, a)$, trained so that an agent can be optimized against the model instead of the real environment. Early instantiations (Ha & Schmidhuber, 2018; Nagabandi et al., 2018) learn this model directly in observation space, or reconstruct observations

Table 3: Comparison between Offline QD, online QD, and prior offline methods that recover multiple behaviors from a single dataset.

	Offline	Multiple behaviors	Predefined behavior space
Online QD (Mouret & Clune, 2015)	✗	✓	✓
SORL (Mao et al., 2024)	✓	✓	✗
DiveOff (Osa & Harada, 2024)	✓	✓	✗
Offline QD (ours)	✓	✓	✓

from a learned latent as an auxiliary training signal (Hafner et al., 2019; 2020; 2021; 2023), coupling model capacity to the complexity of the observation itself rather than to the underlying structure governing state transitions. JEPA-ME (Section 4.2) instead builds on Joint-Embedding Predictive Architectures (JEPA) (LeCun et al., 2022; Assran et al., 2023; Bardes et al., 2024), which drop reconstruction entirely: an encoder E_θ and predictor P_θ are trained so that $P_\theta(E_\theta(s), a)$ matches the representation $E_\xi(s')$ of the true next state, with no decoder required in the objective. The encoder is then free to discard whatever is not predictable from (s, a) pairs, and to spend its capacity on what is useful for the prediction.

Avoiding representation collapse. A purely predictive, decoder-free objective introduces a failure mode absent from reconstruction-based training: since nothing in the loss depends on the content of the representation, $E_\theta(s) = c$ for a constant c trivially minimizes prediction error while carrying no information. Non-contrastive self-supervised methods such as BYOL (Grill et al., 2020), SimSiam (Chen & He, 2021), and DINO (Caron et al., 2021) avoid this with an asymmetric architecture and a stop-gradient, EMA-updated target encoder E_ξ on one branch (He et al., 2020),

$$\mathcal{L}_{\text{pred}} = \left\| P_\theta\big(E_\theta(s), a\big) - \text{sg}\big(E_\xi(s')\big) \right\|_2^2, \qquad \xi \leftarrow \tau_E \xi + (1 - \tau_E)\theta, \tag{23}$$

with τ_E close to 1. Because the target branch receives no gradient and updates slowly, it cannot collapse in response to the predictor's own outputs. This is the mechanism Section 4.2 uses for E_ξ and the stop-gradient in $\mathcal{L}_{\text{rollout}}$.

Compounding error over long rollouts. Generating a full imagined trajectory rather than a single-step prediction introduces a second failure mode: each predicted latent $\hat{z}_{t+1} = P_\theta(\hat{z}_t, a_t)$ is generated from the previous prediction rather than a true state, so any one-step error made at time t is compounded, not corrected, by every subsequent step. The same argument appears in imitation learning, where a policy trained on expert trajectories accumulates error at a rate that grows with rollout length because it must act from its own, slightly off-distribution states (Ross & Bagnell, 2010; Ross et al., 2011); it applies to any autoregressive model consuming its own outputs, world models included. If the one-step prediction error is bounded by ϵ and the dynamics are Lipschitz in the state with constant L, the worst-case error after H steps of self-conditioned rollout grows as

$$\|\hat{z}_H - z_H\| \lesssim \epsilon \sum_{h=0}^{H-1} L^h, \tag{24}$$

geometric in horizon whenever $L > 1$, regardless of how accurate the one-step model is. This is the reason behind the claim in Section 4.2 that JEPA-ME operates in *a substantially harder regime than most world-model planning work,* and behind the horizon sweep ablation in Appendix E.7, where QD score decays with rollout length regardless of predictor capacity, consistent with this bound.

Standard model-based RL practice sidesteps this bound by never unrolling the model over long, uncorrected horizons. MBPO (Janner et al., 2019) generates only short rollouts from real states, keeping H small enough that compounding error stays negligible. MPC-style planners (Nagabandi et al., 2018; Chua et al., 2018; Hansen et al., 2022) go further, replanning at every decision step where only the first action is executed in the real environment, and the plan is recomputed from the newly observed real state. Both strategies rely on periodic access to a true environment state that resets accumulated model error to zero, and neither is available in Offline QD.

B.4 Successor Features and Successor Representations

A recurring idea in RL is to decouple *how an agent expects to move through state space under its current policy* from *how much reward it collects*, so that a single learned quantity capturing the former can be reused across different reward specifications without relearning anything about the dynamics or the policy's behavior.

Successor Representation. The successor representation (SR) (Dayan, 1993) is defined, for a fixed policy π and discrete state space, as the expected discounted future occupancy of every state s' starting from s:

$$M^\pi(s, s') = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \, \mathbb{1}[s_t = s'] \;\middle|\; s_0 = s \right]. \tag{25}$$

Given any reward function $R(s')$ defined over states, the value function decomposes as

$$V^\pi(s) = \sum_{s'} M^\pi(s, s') \, R(s'), \tag{26}$$

which follows directly from linearity of expectation applied to $V^\pi(s) = \mathbb{E}_\pi[\sum_t \gamma^t R(s_t) \mid s_0 = s]$. The dynamics- and policy-dependent term M^π and the reward-dependent term R are factored: if R changes but π does not, V^π can be recomputed by a single dot product, with no relearning of M^π required. Like a value function, M^π satisfies its own Bellman equation, $M^\pi(s, s') = \mathbb{1}[s = s'] + \gamma \mathbb{E}_{s_1 \sim T(\cdot|s,\pi(s))}[M^\pi(s_1, s')]$, and is therefore learnable by temporal-difference methods from transitions alone, similar to a value function.

Successor Features. The tabular SR requires enumerating every state s', which is infeasible once the state space is continuous or high-dimensional. Successor features (SF) (Barreto et al., 2017) generalize SR by replacing the indicator $\mathbb{1}[s_t = s']$ with a fixed feature map $\phi : \mathcal{S} \to \mathbb{R}^d$, defining

$$\psi^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) \;\middle|\; s_0 = s, \, a_0 = a \right] \in \mathbb{R}^d. \tag{27}$$

Under the assumption that the reward is linear in the same features, $r(s) = \phi(s)^\top w$ for some task vector w, the same factoring trick applies to the action-value function:

$$Q^\pi(s, a) = \psi^\pi(s, a)^\top w, \tag{28}$$

again by linearity of expectation. This factorization underlies Generalized Policy Improvement (Barreto et al., 2017; 2018; 2020): given the SFs of a set of previously learned policies $\{\pi_i\}$, a newly specified task w can be evaluated against every π_i instantly via $\psi^{\pi_i}(s, a)^\top w$, without any additional environment interaction, and the best action across the whole set can be selected directly. Extending this idea to a continuous space of tasks, Universal Successor Feature Approximators (Borsa et al., 2019) learn a single SF network conditioned directly on the task vector w, the same conditioning strategy Section 4.1 uses for the target descriptor z. As with the SR, ψ^π satisfies its own Bellman equation:

$$\psi^\pi(s, a) = \phi(s) + \gamma \, \mathbb{E}_{\substack{s' \sim T(\cdot|s,a) \\ a' \sim \pi(\cdot|s')}} \left[\psi^\pi(s', a') \right]. \tag{29}$$

In our work, Section 4.1 repurposes SF for a different purpose than reward transfer: Rather than conditioning ψ on a task vector w, Offline QDAC conditions it on a target behavioral descriptor z, defining $\psi^{\pi_\theta}(s, a, z)$ for the descriptor-conditioned policy $\pi_\theta(\cdot|z)$. The quantity is then used not for policy improvement, but as an offline-computable proxy for the policy's own achieved descriptor $\mathrm{BD}(\pi_\theta(\cdot|z))$.

C Environments and Tasks

This appendix details the behavioral descriptor, fitness functions, and datasets used in each environment evaluated in Section 5.

Pointmass.

A standard pointmass navigation task (no walls), with state dimension 4 and action dimension 2. The state feature $\phi(s)$ is the 2D position (x, y) of the point mass. We evaluate two instances of the general descriptor form introduced in Section 3: **Pointmass (last state)** uses the final-state descriptor $\text{BD}(\tau) = \phi(s_T)$, the point mass's position at the end of the episode; **Pointmass (range)** instead uses a range descriptor over the trajectory, $\text{BD}(\tau) = \max_t \phi(s_t) - \min_t \phi(s_t)$, the extent of positions visited, which is not expressible as a mean of ϕ and is therefore non-additive in the sense of Section 5.2. Fitness is energy efficiency (negative control cost), penalizing large actions: $f_t = (\sqrt{2} - \|a_t\|_2)/\sqrt{2}$. For this environment, we use a single random dataset, collected with a random policy that selects uniformly random actions. The dataset contains 1M transitions.

Halfcheetah.

A planar 2D robot from QDax (Chalumeau et al., 2024), with state dimension 17 and action dimension 6. The state feature $\phi(s) \in \{0, 1\}^2$ indicates, per foot, whether it is in contact with the ground: $\phi(s) = [0, 0]$ if neither foot touches the ground, $[1, 0]$ or $[0, 1]$ if only the back or front foot touches, respectively, and $[1, 1]$ if both do. The behavioral descriptor is the fraction of the episode each foot spends in contact with the ground, i.e. the mean of $\phi(s)$ over the trajectory. Fitness is forward velocity, as used in QDax. Concerning the datasets, we generate three datasets at different quality levels, mirroring the taxonomy introduced by D4RL (Fu et al., 2020):

- **RND**: collected by a Random Network Distillation (Burda et al., 2018) exploration policy. We use the implementation from the URLB codebase (Laskin et al., 2021). The dataset contains 1.5M transitions.
- **Replay**: trajectories logged over a full ME training run, mixing low- and high-quality behaviors as the archive improves over time. The dataset contains 1.5M transitions.
- **Expert**: obtained by taking the final converged ME archive and re-rolling out every stored policy, giving high-fitness, high-diversity trajectories. The dataset contains 1.5M transitions.

The same three-level protocol is used for Walker and Ant_uni below.

We construct these datasets to mirror the dataset-quality variants commonly used in Offline RL (Fujimoto et al., 2019), while accounting for the additional need for behavioral coverage in Offline QD. The RND, Replay, and Expert datasets provide progressively stronger sources of trajectories: RND emphasizes exploration, Replay mixes behaviors collected throughout ME training, and Expert contains high-fitness trajectories from a converged ME archive. This gives us controlled regimes for studying how Offline QD depends on trajectory quality and behavioral coverage.

Walker.

A bipedal walker from QDax, with state dimension 17 and action dimension 6. As with Halfcheetah, $\phi(s) \in \{0, 1\}^2$ is the per-foot ground-contact indicator, the behavioral descriptor is its time-average over the episode, and fitness is forward velocity. Datasets are as described for Halfcheetah.

Ant_uni.

A quadrupedal locomotion task from QDax, with state dimension 27 and action dimension 8. As with Halfcheetah, $\phi(s) \in \{0, 1\}^4$ is the per-leg ground-contact indicator (one entry per leg), the behavioral descriptor is its time-average over the episode, and fitness is forward velocity. Datasets are as described for Halfcheetah.

ANTMAZE-MEDIUM.

A long-horizon navigation task from OGBench (Park et al., 2025), in which a quadrupedal Ant robot with state dimension 29 and action dimension 8 must traverse a medium-sized maze. The state feature $\phi(s)$ is the (x, y) position of the ant in the maze, whereas the behavioral descriptor is the state feature of the last state $\text{BD}(\tau) = \phi(s_{\text{final}})$. Fitness is **energy efficiency**, the per-episode mean over the trajectory of

$$f_t = \sqrt{d_A} - \|a_t\|_2, \qquad d_A = 8, \tag{30}$$

the negative ℓ_2 norm of the action vector, offset by $\sqrt{d_A}$ so that fitness is non-negative; actions lie in $[-1, 1]^{d_A}$, so $f_t \in [0, \sqrt{d_A}]$; d_A is the action dimension. Concerning the datasets, we use three public OGBench (Park et al., 2025) datasets, which differ in how the data-collection policy trades off trajectory quality against state coverage:

- **Navigate**: collected by a noisy expert policy that repeatedly navigates to randomly sampled goals. The dataset contains 1M transitions, 1,000 episodes of 1,000 steps each.
- **Stitch**: short trajectory segments that individually cover only local regions of the maze. The dataset contains 1M transitions, 5,000 episodes of 200 steps each.
- **Explore**: collected by a policy with a large amount of injected action noise, yielding extremely low-quality but high-coverage data that tests recoverability from near-random exploration. The dataset contains 5M transitions, 10,000 episodes of 500 steps each.

CUBE-SINGLE.

A contact-rich manipulation task from OGBench, using a 6-DoF robotic arm with state dimension 28 and a 5-dimensional action space ($\Delta x, \Delta y, \Delta z, \Delta$yaw, gripper speed). The state feature $\phi(s)$ is the cube's x coordinate and height, whereas the behavioral descriptor is the state feature of the last state $\text{BD}(\tau) = \phi(s_{\text{final}})$. Fitness is energy efficiency, identical in form to Antmaze-medium (with $d_A = 5$). **Datasets.** We use two public OGBench datasets for this manipulation task:

- **Play**: collected by a non-Markovian scripted policy with temporally correlated noise, producing natural, realistic pick-and-place trajectories. 1M transitions.
- **Noisy**: collected by a Markovian policy with larger, uncorrelated Gaussian noise, trading naturalism for higher state coverage. 1M transitions.

Table 4: Summary of environments, behavioral descriptors, and fitness functions used in Section 5.

Environment	State/Action dim.	Behavioral Descriptor	Fitness Function	Descriptor dim.
Pointmass (last state)	4 / 2	Final position $\phi(s_T)$	Energy efficiency	2
Pointmass (range)	4 / 2	Position range $\max_t \phi(s_t) - \min_t \phi(s_t)$	Energy efficiency	2
Halfcheetah	17 / 6	Foot contact pattern	Forward velocity	2
Walker	17 / 6	Foot contact pattern	Forward velocity	2
Ant_uni	27 / 8	Foot contact pattern	Forward velocity	4
Antmaze-medium	29 / 8	(x, y) position	Energy efficiency (d_A=8)	2
Cube-single	28 / 5	Cube x coordinate & height	Energy efficiency (d_A=5)	2

D IMPLEMENTATION DETAILS

D.1 ARCHITECTURE, TRAINING, AND EVALUATION PROTOCOL

This section describes the network architectures, training procedures, and evaluation protocol behind the hyperparameters reported in Appendix D.2. Appendix D.1.1 covers choices shared across all three methods, and the following sections detail each method individually.

D.1.1 SHARED EXPERIMENTAL PROTOCOL

Descriptor normalization. Every behavioral descriptor is rescaled to $[0, 1]^d$ before being used as a network input, a query target, or an archive coordinate. All descriptor-space quantities reported in this paper, including QD score, the skill-distance profiles of Figure 1, and every queried z at evaluation, are computed in this normalized space.

Compliance threshold for QD score. Section 5 defines QD score as counting only descriptors for which $\|\mathrm{BD}(\pi(\cdot|z)) - z\| < \delta$. We denote this evaluation-time threshold δ_{eval}, to distinguish it from Offline QDAC's training-time compliance threshold δ (Eq. 7). We set $\delta_{\mathrm{eval}} = 0.1$, in normalized descriptor units, for every environment and method.

Evaluation grid. We discretize $\mathcal{Z}$ into a 50×50 grid ($5 \times 5 \times 5 \times 5$ for Ant_uni) for both training and evaluation. Evaluation queries a fixed grid of target descriptors spanning $\mathcal{Z}$, with 5 independent rollouts per queried z; reported skill distance and fitness are the mean over these 5 rollouts.

D.1.2 OFFLINE QDAC

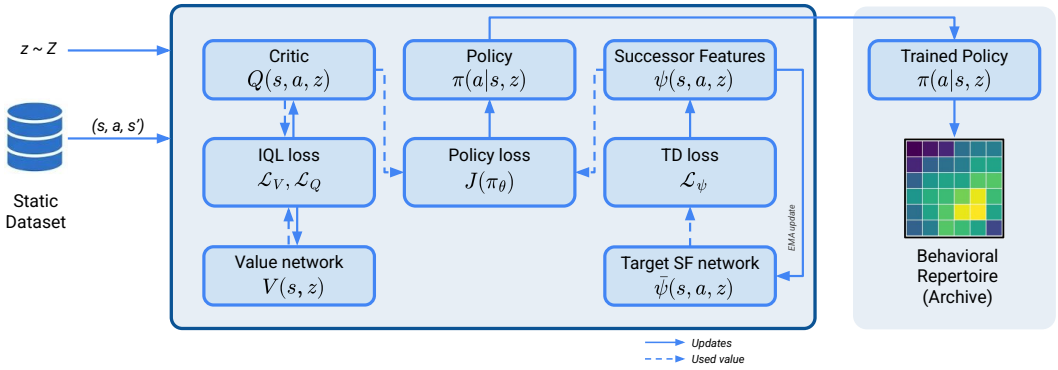

Figure 2: **Offline QDAC architecture.** Policy, critic, value, and successor-feature networks are trained jointly from $\mathcal{D}$ via IQL losses and the SF-based compliance objective (Eq. 7); the trained policy $\pi(\cdot|z)$ acts as the archive at evaluation.

Architecture. The actor π_θ, critic Q, value function V, and successor-feature network ψ (Section 4.1) are each 2-hidden-layer MLPs with 256 units per layer and ReLU activations (Table 5), taking the state and target descriptor as a single concatenated input. The critic and the successor-feature network each maintain two independently initialized heads sharing the same trunk. Following IQL, the value target uses the minimum of the two critic heads, and the actor's policy gradient instead backpropagates through their mean. Both the successor-feature Bellman target and the descriptor estimate used at query time use the mean, not the minimum, of the two successor-feature heads. The actor outputs the mean and log-standard-deviation of a Gaussian; sampled actions are passed through $\tanh$.

Training. At each gradient step, the target descriptor z is resampled independently and uniformly from $[0, 1]^d$ for every transition in the batch: the policy is never trained only on descriptors realized by the dataset's own trajectories. Networks are updated in a fixed order every step (value, critic, SF, actor), each with its own Adam optimizer (Kingma & Ba, 2015). The critic and successor-feature target networks are updated by an exponential moving average at rate τ, applied at every step. For

Antmaze and Cube, we use an additional behavioral cloning term with coefficient 0.3 to ground the actions similar to Fujimoto & Gu (2021); Park et al. (2024).

D.1.3 JEPA-ME

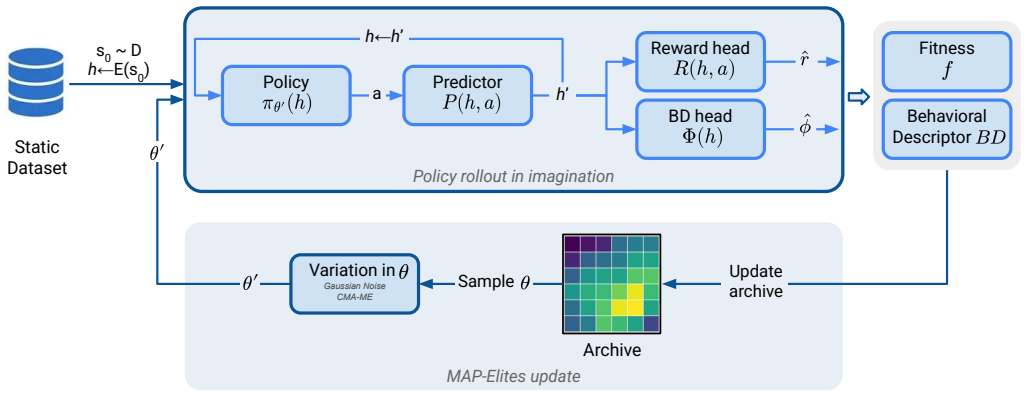

Figure 3: **JEPA-ME architecture.** A world model is trained from $\mathcal{D}$, then MAP-Elites searches entirely inside imagined rollouts generated by the learned predictor, reward head, and BD head.

World-model architecture. E_θ, E_ξ, and P_θ (Section 4.2) are feedforward networks with ReLU activations and no normalization layers; P_θ takes the concatenation of h and a as input. R_θ and Φ_θ each have two hidden layers and take, respectively, (h, a) and h as input.

World-model training. Section 4.2 defines $\mathcal{L}_{\text{rollout}}$ over unnormalized latents. We implement it with both branches ℓ_2-normalized before the squared distance, writing $\bar{v} \doteq v/\|v\|_2$:

$$\mathcal{L}_{\text{rollout}} = \frac{1}{k}\sum_{j=1}^{k} \left\| \bar{\hat{h}}_j - \overline{\text{sg}\big(E_\xi(s_{t+j})\big)} \right\|_2^2, \tag{31}$$

matching BYOL-style non-contrastive objectives; this is the only latent-prediction loss used, and there is no separate 1-step-only loss. The reward and feature heads are trained against the *imagined* latent at every horizon in the window, not a fresh encoding of the corresponding real state:

$$\mathcal{L}_{\text{reward}} = \frac{1}{k}\sum_{j=1}^{k} \big(R_\theta(\hat{h}_{j-1}, a_{t+j-1}) - \tilde{r}_{t+j-1}\big)^2, \qquad \mathcal{L}_{\text{BD}} = \frac{1}{k}\sum_{j=1}^{k} \big(\Phi_\theta(\hat{h}_{j-1}) - \tilde{\phi}_{t+j-1}\big)^2, \tag{32}$$

where $\tilde{r}$ and $\tilde{\phi}$ are the dataset reward and state feature rescaled to zero mean and unit variance using statistics computed once over the full training dataset, and $\hat{h}_0 \doteq E_\theta(s_t)$. The total loss is

$$\mathcal{L}_{\text{WM}} = \alpha_{\text{JEPA}}\,\mathcal{L}_{\text{rollout}} + \alpha_{\text{reward}}\,\mathcal{L}_{\text{reward}} + \alpha_{\text{BD}}\,\mathcal{L}_{\text{BD}}, \tag{33}$$

with weights as in Table 6. Windows of $k{+}1$ consecutive states are drawn by sampling a trajectory and a start time independently and uniformly at random.

Imagined-rollout search: solution architecture. A candidate policy is a feedforward network with two 64-unit ReLU hidden layers and a $\tanh$ output, mapping the latent state directly to an action. It has the same architecture as the ME solution networks, except its input dimension is $\dim(h)$ rather than $\dim(s)$.

Imagined environment. At the start of an episode, a real dataset state is sampled uniformly at random from initial states and encoded once via E_θ; every subsequent step is a latent transition through P_θ, with reward and state features from R_θ and Φ_θ applied to the latent *before* that step's transition. The real environment is never queried for the rest of the episode.

Numerical safeguards. Three safeguards bound how far an imagined rollout can diverge over its horizon (up to 1000 steps):

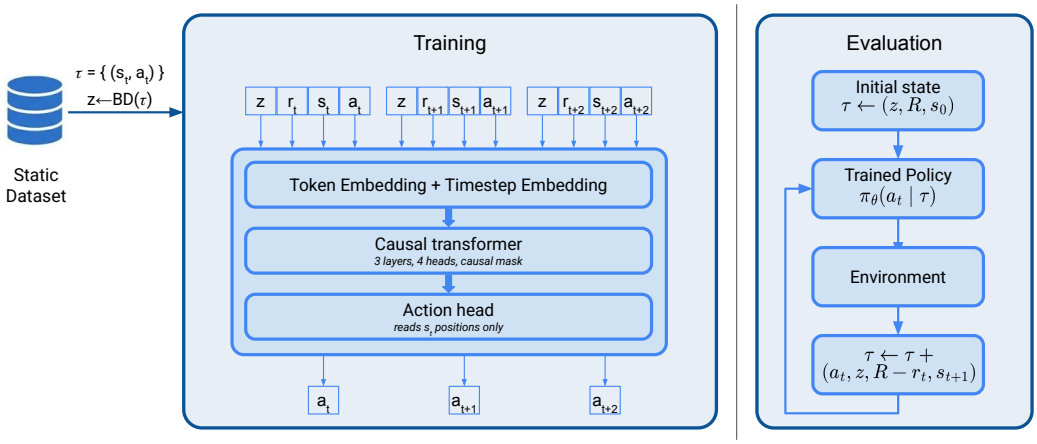

Figure 4: **BD-DT architecture. Training:** trajectories are tokenized into interleaved $(z, \hat{R}, s, a)$ sequences for causal action prediction. **Evaluation:** the policy is prompted with a target z and return-to-go, then rolled out autoregressively in the real environment.

1. The latent is clipped per-dimension to $\text{mean} \pm 6\,\text{std}$, computed from E_θ applied to the dataset's states.
2. The predicted reward and state feature are each clipped to the dataset's empirical $[\min, \max]$ range; for the foot-contact descriptor (Halfcheetah, Walker, Ant_uni), the clipped, continuous predicted feature is further thresholded at 0.5 and binarized before being averaged into the episode-level descriptor, matching the true environment's binary contact signal.
3. Any infinite or `NaN` values produced by a diverged rollout are replaced with 0.

D.1.4 BD-DT

Architecture. Each of the four token types, z, $\hat{R}_t$, s_t, a_t, is projected to the embedding dimension h by its own linear layer. A single learned timestep embedding, shared across all four modalities at a given step, is added to each projection before the tokens are interleaved into the length-$4K$ sequence of Section 4.3 and passed through LayerNorm. The sequence is processed by a standard causal transformer encoder (Table 8). Actions are predicted by a linear layer followed by $\tanh$, taking only the hidden states at the s_t positions. We use a deterministic regression head, not a stochastic policy.

Training. The target descriptor token at every step of a training window is the dataset descriptor $z_\tau = \text{BD}(\tau)$ of the trajectory the window is drawn from. The log-likelihood in (Eq. 8) is implemented as mean-squared error between predicted and dataset actions, computed only at non-padded positions. Return-to-go targets are the raw discounted sum of raw rewards, with no return scaling; windows that extend before a trajectory's start are left-padded with zeros and excluded from the loss via the padding mask.

Evaluation. The initial return-to-go is conditioned on the queried descriptor z: writing $\hat{R}_0(\tau)$ for a trajectory's total discounted return,

$$\hat{R}_0(z) = \begin{cases} P_{90}\big(\{\hat{R}_0(\tau) : \tau \in \mathcal{D},\ \|\text{BD}(\tau) - z\|_2 < \varepsilon_{\text{RTG}}\}\big) & \text{if at least 5 such } \tau \text{ exist,} \\ P_{90}\big(\{\hat{R}_0(\tau) : \tau \in \mathcal{D}\}\big) & \text{otherwise,} \end{cases} \tag{34}$$

where P_{90} denotes the 90th percentile and $\varepsilon_{\text{RTG}} = 0.1$ is a neighborhood radius in normalized descriptor space (unrelated to δ_{eval} from Appendix D.1.1 despite sharing its default). During rollout, z is held fixed and $\hat{R}_t$ is updated as $\hat{R}_{t+1} = (\hat{R}_t - r_t)/\gamma$ at every step over a sliding window of the K most recent steps.

D.2 HYPERPARAMETERS

We report the hyperparameters used for each of the three Offline QD methods, and for the online QDAC and ME baselines, as run in Section 5. Unless noted otherwise, hyperparameters are shared across all environments and datasets.

D.2.1 OFFLINE QDAC

Table 5 reports the hyperparameters for Offline QDAC (Section 4.1). The actor, critic, value function, and successor-feature networks are 2-layer MLPs with 256 hidden units per layer and ReLU activations; all four are optimized jointly with Adam. Online QDAC (Table 1) uses *identical* network architecture, optimizer, and loss hyperparameters, replacing only the offline dataset with an online replay buffer; no hyperparameter differs between the offline and online variants.

Table 5: Offline QDAC hyperparameters.

Hyperparameter	**Value**
Hidden layer sizes (actor, critic, value, ψ)	$(256, 256)$
Optimizer	Adam
Learning rate	1×10^{-4}
Batch size (transitions)	512
Discount factor γ	0.99
Target network update rate ρ_{target}	0.001
IQL expectile τ_{IQL}	0.9
Compliance threshold δ (Eq. 7)	0.01
Reward scaling	1.0
Total gradient steps	1×10^{6}

D.2.2 JEPA-ME

JEPA-ME is trained in two stages: first the JEPA world model, then ME search inside the trained model. Table 6 reports world-model training hyperparameters; Table 7 reports the imagined-rollout QD search hyperparameters. The encoder, the predictor, the reward and feature heads are 2-layer MLPs with 256 hidden units. World-model training uses substantially fewer gradient steps than Offline QDAC or BD-DT (2×10^5 vs. 10^6). We verified training loss plateaus well before this budget is exhausted on all environments, so this difference reflects convergence rather than an under-trained model.

Table 6: JEPA world model hyperparameters.

Hyperparameter	**Value**
Latent dimension	64
Encoder hidden dim / layers	256 / 2
Predictor hidden dim / layers	256 / 2
Reward & feature dim / layers	256 / 2
Target encoder EMA rate	0.995
Rollout-consistency horizon k	10
JEPA loss weight α_{JEPA}	1.0
Reward loss weight α_{reward}	1.0
BD loss weight α_{BD}	1.0
Optimizer	Adam (Kingma & Ba, 2015)
Learning rate	1×10^{-3}
Batch size (windows)	512
Total gradient steps	2×10^5

Table 7: JEPA-ME imagined-rollout QD search hyperparameters.

Hyperparameter	**Value**
Population size	200
Number of generations	500
Number of emitters	5
Mutation	Isotropic Gaussian / CMA-ES mix
Mutation σ_0	1.0
Imagined rollout horizon	1000 (200 for Pointmass)
Latent safety clip	mean $\pm\, 6\,\sigma$

D.2.3 BD-DT

Table 8 reports the hyperparameters for the BD-conditioned Decision Transformer (Section 4.3). The architecture follows the standard causal transformer encoder used by Chen et al. (2021), with four token types (descriptor, return-to-go, state, action) interleaved per timestep and a linear action-prediction head.

Table 8: BD-DT hyperparameters.

Hyperparameter	**Value**
Context length K	20
Embedding dimension h	128
Feedforward dimension	512
Number of layers	3
Number of attention heads	4
Dropout	0.1
Discount factor γ (return-to-go)	0.99
Gradient clipping norm	0.25
Optimizer	Adam (Kingma & Ba, 2015)
Learning rate	1×10^{-4}
Batch size (windows)	64
Total gradient steps	1×10^{6}

D.2.4 Online Baselines (QDAC, ME)

Online QDAC uses the network architecture, optimizer, and loss hyperparameters of Table 5 without modification (Appendix D.2.1). Table 9 reports the hyperparameters for the ME baseline.

Table 9: ME hyperparameters.

Hyperparameter	**Value**
Population size	200
Number of generations	500
Number of emitters	5
Mutation	Isotropic Gaussian / CMA-ES mix
Mutation σ_0	1.0

D.3 Compute

All experiments were run on one H100 GPU. Table 10 reports the approximate compute cost per method, averaged over environments and datasets.

Table 10: Approximate compute cost per method, per seed.

Method	**GPU-hours (per run)**
Offline QDAC	0.76
JEPA-ME	0.46
BD-DT	1.13
QDAC (online)	0.96
ME (online)	0.62

E ADDITIONAL RESULTS

We provide additional experiments that isolate the specific factors behind each method's performance in Section 5.

E.1 COMPARISON AGAINST STRONGER ONLINE QD BASELINES

The hard-exploration benchmark in Section 5.1 compares offline methods with ME, which explores purely through isotropic mutation and CMA-ES. To check whether the gap reported there reflects a genuine limitation of online QD under hard exploration, rather than the online baseline we chose, we repeat the comparison with three stronger online QD methods already discussed in Appendix B.1: PGA-ME (Nilsson & Cully, 2021), DCG-ME (Faldor et al., 2023), and QD-PG (Pierrot et al., 2022), which adds an explicit diversity-seeking gradient.

Experiment. We evaluate PGA-ME, DCG-ME, and QD-PG as additional online baselines, alongside ME. All three use the same population size, number of generations, and archive resolution as ME (Table 9). For PGA-ME and DCG-ME, we use the same critic architecture and policy-gradient formulation as QDAC's critic (Appendix D.1.2). For QD-PG, we add the diversity-seeking gradient on top of the standard reward gradient (offspring are generated by 33% isotropic Gaussian mutation, 33% reward-critic policy-gradient steps, and 33% diversity-critic policy-gradient steps). We report QD score for all four online baselines, next to the best offline method per row from Table 1.

Table 11: **QD score against stronger online QD baselines.** Mean ± std over 5 seeds, alongside the best offline method (from Table 1) and ME. **Bold** marks the best online method per row. Best Offline is colored green when it beats every online baseline shown, red when it does not.

Environment	Dataset	Best Offline	ME	PGA-ME	DCG-ME	QD-PG
Halfcheetah	RND	231±32 (QDAC)	**3221**±415	2762±356	2975±384	617±21
	Replay	2411±245 (QDAC)				
	Expert	2461±203 (QDAC)				
Walker	RND	59±10 (QDAC)	361±53	**406**±60	396±58	45±12
	Replay	91±12 (QDAC)				
	Expert	109±14 (QDAC)				
Antmaze-medium	Explore	532±58 (QDAC)	26±4	32±6	30±7	**52**±5
	Stitch	778±132 (BD-DT)				
	Navigate	638±102 (QDAC)				
Cube-single	Noisy	226±42 (QDAC)	15±3	21±4	24±5	**28**±6
	Play	297±39 (QDAC)				

Results. On the **locomotion tasks** (Halfcheetah, Walker), where exploration is easy, ME remains the strongest online method on Halfcheetah (3221), with PGA-ME and DCG-ME close behind (2762 and 2975). On Walker, both PGA-ME and DCG-ME surpass ME (406 and 396 vs. 361), confirming that policy-gradient updates can improve performance. QD-PG fails on both locomotion tasks: its diversity-seeking gradient is designed to work with final-state descriptors, whereas Halfcheetah and Walker use a mean foot-contact-ratio descriptor accumulated over the whole trajectory, a setting QD-PG is not built for (Pierrot et al., 2022). On both tasks, the best offline method (Offline QDAC) is still behind most online baselines (except for QD-PG, mainly with Replay and Expert datasets).

On the **hard-exploration tasks** (Antmaze-medium and Cube-single), every online baseline, including QD-PG, remains well behind the best offline method: QD-PG reaches only 52 on Antmaze-medium and 28 on Cube-single, against offline scores of 532–778 and 226–297 respectively. PGA-ME and DCG-ME are only marginally better than ME on these tasks. Notably, QD-PG's explicit diversity-seeking gradient yields greater improvements over ME in both environments, yet it is still behind: adding a goal-directed or diversity-seeking mechanism narrows the gap, but it is still insufficient to solve hard-exploration environments.

Takeaway: Offline QD still surpasses online QD methods equipped with diversity-seeking mechanisms suited for hard-exploration tasks. This highlights the limits of online exploration in such settings and positions Offline QD as a promising alternative for tackling them.

E.2 Comparison to Offline Goal-Conditioned RL

A natural question is how Offline QD relates to offline goal-conditioned RL (GCRL). Offline GCRL learns a policy $\pi(\cdot \mid g)$ from a static dataset that reaches a specified goal g, typically by optimizing a sparse reward $R(s, a) = \mathbf{1}[s = g]$ via hindsight relabeling: any visited state along a trajectory can serve as a goal label for the states that preceded it, turning a reward-free dataset into goal-reaching supervision for free.

Directly comparing GCRL against Offline QD is not straightforward, because the two problems are different. Offline QD optimizes two objectives jointly: reaching a target descriptor z, similar to GCRL's single goal-reaching objective, and maximizing fitness within that constraint, for which GCRL has no counterpart. Combining them into one scalar reward, $R_g(s, a) = \mathbf{1}[s = g] + \text{Fitness}(s, a)$ is possible, but likely to produce suboptimal results (Vamplew et al., 2011). The mismatch is even sharper on Halfcheetah and Walker, where the descriptor is not a single terminal state but a trajectory-averaged quantity (mean foot-contact ratio): there is no single state to relabel as a goal, and forcing one, e.g. by relabeling every state with its trajectory's realized descriptor, tests GCRL outside the setting it is designed for.

Given this mismatch in the settings of Offline QD and Offline GCRL, we restrict the comparison to Antmaze-medium and Cube-single, where $\text{BD}(\tau) = \phi(s_{\text{final}})$ is itself a natural goal, and we compare skill-distance profiles only, not QD score. This isolates one part of what Offline QD solves, while leaving aside the fitness component, which GCRL does not address by construction.

Experiment. We train GCIQL on Antmaze-medium and Cube-single by hindsight relabeling (Andrychowicz et al., 2017). We use the same network architecture, optimizer, and IQL hyperparameters as Offline QDAC (Table 5). At evaluation, we query GCIQL on the same fixed grid of target descriptors z used in Section 5.1, treating z directly as the goal, and record the skill distance $\|\text{BD}(\pi(\cdot|g{=}z)) - z\|$ over 5 rollouts per queried z, following the same protocol as Figure 1. We report the resulting distance profile alongside Offline QDAC, BD-DT, and JEPA-ME on both environments.

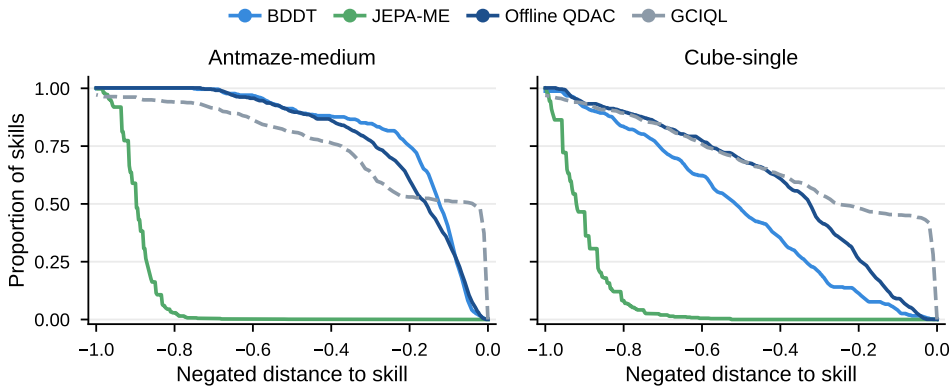

Figure 5: **Behavioral compliance vs. a goal-conditioned RL baseline.** Negated skill distance $-\|\text{BD}(\pi(\cdot|z)) - z\|$ over the same fixed grid of queried descriptors z, on Antmaze-medium and Cube-single; further right is better compliance.

Results. Figure 5 shows that GCIQL achieves the best compliance at tight distance thresholds on both environments, reaching 0.51 and 0.43 proportion of skills at distance 0.1 on Antmaze-medium and Cube-single, respectively, higher than both Offline QDAC and BD-DT. This is expected given the asymmetry noted above: GCIQL optimizes goal-reaching exclusively, while BD-DT and Offline QDAC must satisfy compliance jointly with a fitness objective.

Interestingly, at looser thresholds (0.3–0.4), Offline QDAC and BD-DT track GCIQL closely and in several regions match or exceed it, despite solving the strictly harder joint problem. This reveals a difference in failure pattern: GCIQL's advantage at tight thresholds does not persist at looser ones, meaning a fraction of its skills remain far off target even under a wide distance threshold. GCIQL is precise on the skills it reaches, but performs poorly on the rest. Offline QDAC and BD-DT exhibit the opposite pattern: less precise at tight thresholds, but without a comparable fraction of skills performing poorly, so their more moderate, consistent errors match GCIQL's proportion once the threshold increases.

Takeaway: Offline QD is a more general problem than offline GCRL: compliance is only one of its two objectives, alongside fitness, which GCRL does not optimize at all. Despite this, Offline QDAC and BD-DT reach compliance comparable to a dedicated GCRL baseline over most of the distance range.

E.3 Dataset Quality: Fitness and Diversity

The RND and Expert datasets used elsewhere in this work vary along two axes at once, fitness and behavioral coverage. They are meant to reflect realistic offline data rather than isolate a single factor. To disentangle which axis drives the gains reported in Section 5.1, we construct two additional datasets that each vary only one axis, and interpolate between them directly.

Experiment. We collect a **diversity-only** dataset by running ME with the fitness objective fixed to zero, so cell occupancy is driven purely by exploration, and a **fitness-only** dataset using standard fitness maximization with no diversity term. We construct mixtures at fixed ratios (100% fitness-only, 75/25, 50/50, 25/75, and 100% diversity-only) while keeping total dataset size constant, train Offline QDAC and BD-DT on every mixture, and report QD score, max fitness, and coverage as a function of the mixture ratio.

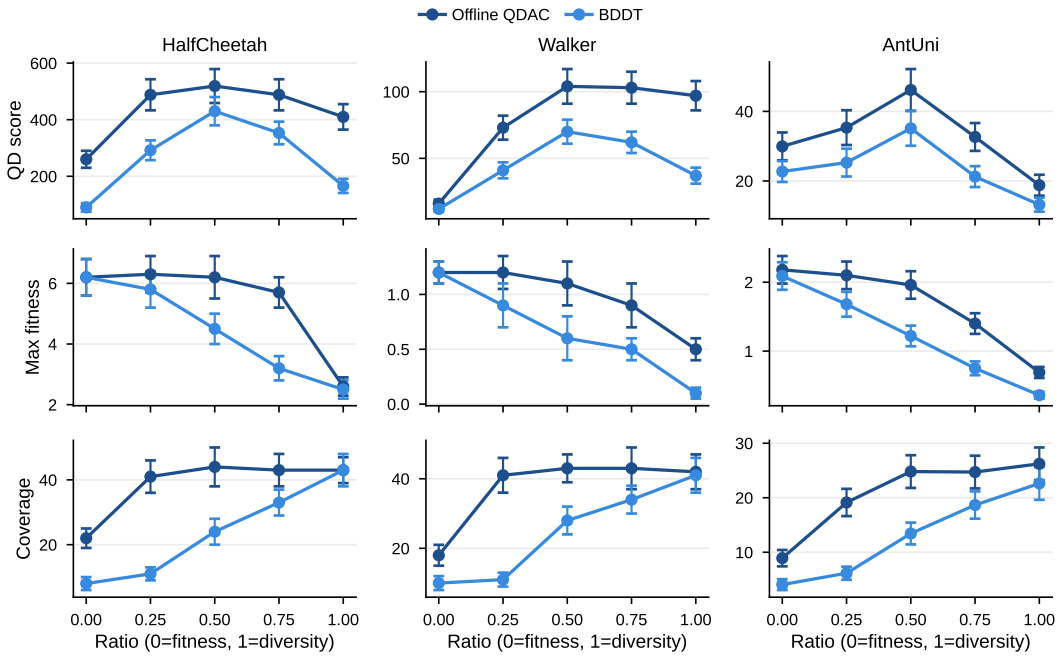

Figure 6: **Fitness vs. diversity mixture.** QD score, max fitness, and coverage vs. mixture ratio (0 = fitness-only, 1 = diversity-only dataset), for Offline QDAC and BD-DT.

Results. Figure 6 shows that QD score peaks at an intermediate ratio for both methods, not at either extreme. Max fitness generally declines as the diversity fraction increases, while coverage rises and saturates. Since QD score sums fitness only over compliant cells, neither pole maximizes

it: fitness-only data has too little coverage to accumulate a high sum, diversity-only data has too little per-cell fitness to be worth counting. The intermediate ratio balances both.

Offline QDAC saturates coverage earlier than BD-DT, consistent with Section 5.2: SF generalize compliance across z, so a modest amount of diverse data unlocks most of the descriptor space, while BD-DT can only reproduce descriptors it was directly shown.

Takeaway: QD score is maximized by mixing fitness and diversity, not by either pure dataset: fitness-only data lacks coverage, diversity-only data lacks fitness. Offline QDAC needs less diversity data than BD-DT to saturate coverage: BD-DT can only comply with a BD it has seen demonstrated in the data, while Offline QDAC's SF signal can generalize to BD values beyond what was directly demonstrated.

E.4 Dataset Quality: Coverage and Generalization

Recoverability in Offline QD is bounded by what is reachable in the dataset. What remains unclear is how each method behaves at the boundary of that support: whether skill compliance degrades smoothly as queried descriptors move away from covered regions, or fails abruptly once outside them.

Experiment. We evaluate on Pointmass, Halfcheetah, and Antmaze, spanning the range of difficulty established in Section 5.2. For each environment, we split the behavioral descriptor space along the midpoint of each axis and retain only trajectories falling in the bottom-left quadrant, discarding the remaining three quarters of the descriptor space. Each method is trained on the truncated dataset, then queried on a fixed grid of target descriptors z spanning the full descriptor space, measuring skill distance $\|\mathrm{BD}(\pi(\cdot|z)) - z\|$ as a function of the distance of z from the boundary of the covered quadrant.

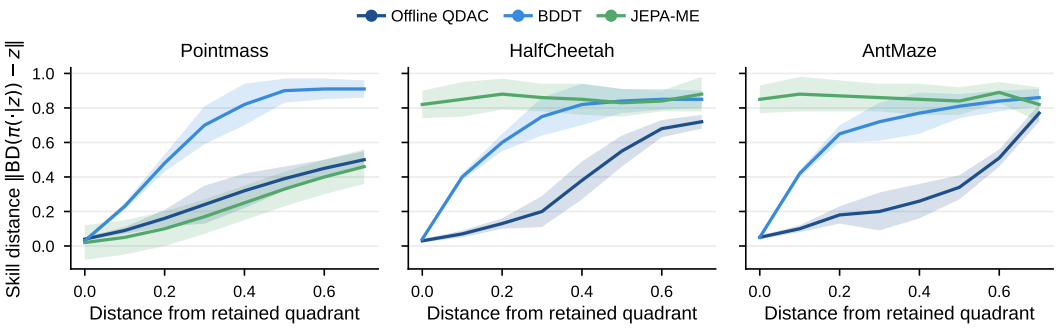

Figure 7: Skill distance $\|\mathrm{BD}(\pi(\cdot|z)) - z\|$ as a function of Euclidean distance from the queried descriptor z to the quadrant of $\mathcal{Z}$ retained during training (0 = inside the retained quadrant). Lines show the binned mean per method (shaded: std).

Results. Offline QDAC degrades most gradually over the three environments. This follows from how it is trained: it samples z from the full space $\mathcal{Z}$, so it is asked to handle out-of-distribution targets even though its SF network only sees covered states. BD-DT only ever trains on $z_\tau = \mathrm{BD}(\tau)$, the realized descriptor of a real trajectory, so it never sees an out-of-distribution target during training, only at evaluation. This is why it fails sharply: not poor extrapolation, but no exposure to extrapolate from. JEPA-ME depends on the environment: its performance degrades gradually in Pointmass, where its dynamics model is accurate; it is flat and near-maximal everywhere in Halfcheetah and Antmaze, meaning it fails uniformly rather than at the boundary.

Takeaway: Offline QDAC can extrapolate because it is trained on the full descriptor space; BD-DT cannot, because it only ever trains on targets a trajectory actually reached. Compliance at the edge of the data depends on whether training ever asked the policy to reach beyond it.

E.5 Ablating the Compliance Signal and Offline RL Backbone

Offline QDAC combines two components on top of a vanilla behavior-conditioned policy: an offline RL backbone for reward optimization, and the SF signal ψ for BD compliance. Each component involves a design choice: the offline RL backbone can be conservative (IQL) or not (TD3), and the compliance signal can be the Bellman-trained SF network or a naive per-state penalty. We isolate the contribution of each choice.

Experiment. We train four methods:

1. **BC**, a behavior-conditioned policy trained via behavioral cloning (Pomerleau, 1988) on transitions relabeled with $z = \text{BD}(\tau)$, with neither component.
2. **IQL+naive reward**, which removes SF, uses only IQL with a naive per-state reward $r'(s, z) = r(s) - \lambda\|\phi(s) - z\|^2$ to combine both fitness and skill compliance in one reward function. We set $\lambda = 1$, chosen by a small sweep over 0.1, 1.0, 10.0 to match the scale of $r(s)$.
3. **Offline QDAC (TD3)**, which replaces IQL with TD3 to remove conservatism, but still using SF compliance.
4. **Offline QDAC**, our full method.

We report the QD score for all four methods.

Table 12: QD score (mean $\pm$ std over 5 seeds) for the Offline QDAC component ablation. Results averaged over datasets.

Environment	BC	IQL+naive reward	Offline QDAC (TD3)	Offline QDAC
Halfcheetah	$464_{\pm 23}$	$95_{\pm 16}$	$1252_{\pm 54}$	$\mathbf{1701}_{\pm 48}$
Walker	$41_{\pm 5}$	$43_{\pm 10}$	$84_{\pm 24}$	$\mathbf{86}_{\pm 21}$
Antmaze-medium	$489_{\pm 51}$	$35_{\pm 7}$	$523_{\pm 65}$	$\mathbf{622}_{\pm 58}$
Cube-single	$94_{\pm 13}$	$130_{\pm 34}$	$226_{\pm 31}$	$\mathbf{261}_{\pm 36}$

Results. SF is the dominant component: Holding the offline RL backbone fixed at IQL, replacing the naive penalty with SF multiplies QD score by 2–18$\times$ across environments. Conservatism matters far less: holding SF fixed, replacing TD3 with IQL yields only a further 18% average improvement. IQL without SF is not simply weaker, it can be actively harmful: on Antmaze-medium and Halfcheetah, IQL with the naive penalty scores below plain behavioral cloning ($489 \to 35$ and $464 \to 95$), since matching z independently at each state conflicts with the structure needed to reach it, while SF with no conservatism at all **"Offline QDAC (TD3)"** still reaches 523 and 1252 on the same environments.

The low performance of IQL+naive reward is also explained by the fact that, on Halfcheetah and Walker, $\phi(s)$ is a binary per-step contact indicator, while z is a continuous fraction of the episode, so the per-step penalty $\|\phi(s) - z\|^2$ cannot reach zero at any single state even when the trajectory-averaged contact ratio matches z exactly. SF avoids this by accumulating $\phi(s)$ over the trajectory before comparing to z, matching the additive descriptor form rather than penalizing pointwise.

Takeaway: SF are the primary driver of Offline QDAC's performance: unlike the naive per-state penalty, SF look along the trajectory rather than matching z at each state in isolation, and this compliance signal helps regardless of which offline RL backbone it is paired with. Conservatism combined with SF yields a further, smaller improvement on average.

E.6 Ablating Descriptor and Return Conditioning in BD-DT

BD-DT conditions jointly on a target descriptor z and a target return-to-go $\hat{R}_t$ (Section 4.3). We isolate the contribution of each signal by comparing against two single-signal baselines sharing BD-DT's architecture and training budget: DT (Chen et al., 2021), conditioned on return-to-go only, and QDT (Macé et al., 2023), conditioned on a descriptor only.

Experiment. We train all three methods on every environment and dataset used for BD-DT in Table 1, excluding Pointmass and Ant_uni (omitted for compute reasons). QDT and BD-DT are queried on the same descriptor grid as in Section 5.1. DT has no descriptor input, so we instead roll it out $N = 20$ times at the same fixed return-to-go BD-DT uses, and report mean fitness over these rollouts (instead of the maximum fitness over the grid for QDT and BD-DT); coverage and QD score are not defined for DT and are left blank. All results are averaged over 5 seeds.

Table 13: Coverage, max fitness, and QD score. **Bold** marks the best value per row and metric. DT has no descriptor input and optimizes fitness alone, so coverage and QD score are undefined for it and left blank.

		Coverage (%)			Max fitness			QD score		
Env	**Dataset**	DT	QDT	BD-DT	DT	QDT	BD-DT	DT	QDT	BD-DT
Halfcheetah	RND	–	11.42	**13.27**	1.583	1.087	**2.317**	–	70	**86**
	Replay	–	**25.17**	21.64	7.412	4.329	**7.926**	–	360	**447**
	Expert	–	**33.89**	25.08	**11.114**	7.815	8.204	–	921	**974**
Walker	RND	–	12.63	**15.91**	**0.357**	0.104	0.311	–	25	**49**
	Replay	–	19.24	**20.33**	**0.481**	0.197	0.423	–	36	**60**
	Expert	–	**25.71**	23.75	0.524	0.412	**0.552**	–	69	**75**
Antmaze-medium	Explore	–	**16.55**	16.42	**2.638**	1.634	2.384	–	236	**482**
	Stitch	–	20.38	**22.19**	2.614	1.923	**2.743**	–	373	**778**
	Navigate	–	19.06	**20.87**	**2.809**	1.598	2.718	–	282	**448**
Cube-single	Noisy	–	**16.92**	14.53	**1.216**	0.397	0.914	–	40	**75**
	Play	–	15.14	**16.06**	**1.627**	0.508	1.427	–	58	**112**

Results. Table 13 shows BD-DT achieves the highest QD score in every setting. This is not because it maximizes both coverage and fitness: QDT has higher coverage in $5/11$ rows, BD-DT achieves higher QD score because it produces higher fitness among its compliant cells, due to being conditioned on both behaviors and returns.

QDT achieves the lowest max fitness in all 11 settings, since it conditions on no return signal and thus has no mechanism to recognize and produce high-return trajectories. In contrast, DT achieves the highest max fitness in $7/11$ settings: DT conditions on return alone, so it must average over the different behavioral modes that share a similar return.

Neither BD-DT nor QDT has consistently higher coverage, except on Halfcheetah and Walker Expert, where QDT is higher: the Expert data is collected directly from every archive cell, so descriptor-only imitation (QDT) can successfully distill most of the archive.

Takeaway: Return and descriptor conditioning are complementary rather than redundant: neither single-signal baseline matches BD-DT's QD score, confirming that joint conditioning is necessary to recover a repertoire that is both broad and high-fitness.

E.7 World Model Horizon and Capacity Sensitivity

Section 5.2 attributes JEPA-ME's failure on Halfcheetah and Cube-single to compounding prediction error, in contrast to its success on the low-dimensional dynamics of Pointmass. Two explanations remain unresolved: the predictor may lack the capacity to model these dynamics at all, or performance may decline with horizon regardless of capacity, which is a structural limitation of searching entirely in imagination.

Experiment. On Pointmass, Halfcheetah, and Cube-single, we run JEPA-ME at rollout horizons $h \in \{50, 100, 250, 500, 1000\}$ and three capacities of the latent predictor P_θ (Section 4.2): small (256 hidden units / 2 layers), medium (256 / 4), and large (512 / 4). We keep the encoder fixed. We separately run ME in the real environment truncated to the same horizons, at the same generation budget used elsewhere in the paper, and report JEPA-ME's QD score as a percentage of this reference at each horizon.

Results. Figure 8 shows that horizon, not capacity, drives JEPA-ME's performance. In every environment, increasing predictor capacity yields at most a modest gain at any fixed horizon, and this

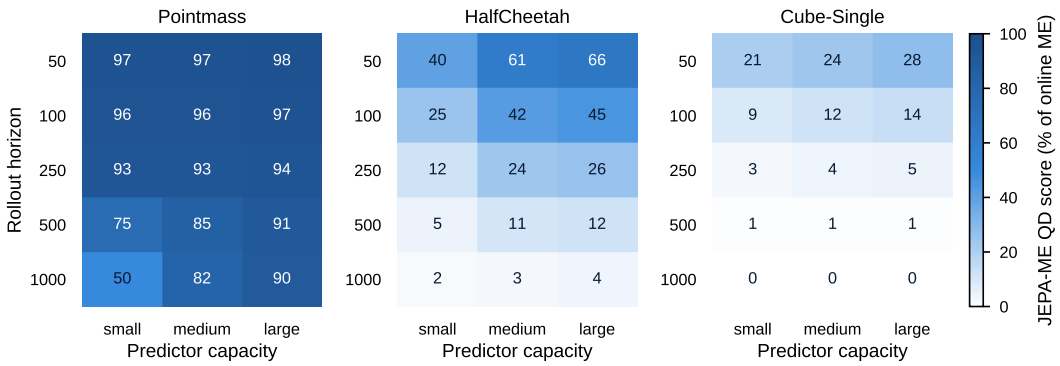

Figure 8: JEPA-ME QD score as a percentage of ME at the same rollout horizon, for three predictor capacities. Rows are rollout horizon; columns are predictor capacity.

gain shrinks as horizon grows: on Halfcheetah, the gap between small and large capacity narrows from 26% at $h = 50$ to 2% at $h = 1000$. No capacity setting prevents the collapse at long horizons. Pointmass is the exception, reaching 90% of the online reference even at $h = 1000$, consistent with its simple, low-dimensional dynamics.

Takeaway: JEPA-ME's failure on long-horizon tasks is a consequence of compounding prediction error, not insufficient model capacity. Scaling the predictor does not prevent performance degradation at long horizons, indicating a structural limitation of searching entirely inside imagination rather than a fixable modeling choice.

E.8 Ablation on the Compliance Threshold

Offline QDAC's final objective (Eq. 7) gates the reward gradient on a compliance threshold δ: the policy is optimized for fitness only once its SF compliance loss falls below δ, and for compliance alone otherwise. We study how sensitive Offline QDAC is to this threshold.

Experiment. We train Offline QDAC with $\delta \in \{0.001, 0.005, 0.01, 0.05, 0.1\}$ [1], holding every other hyperparameter fixed at the values in Appendix D.2.1, and report QD score averaged over datasets per environment.

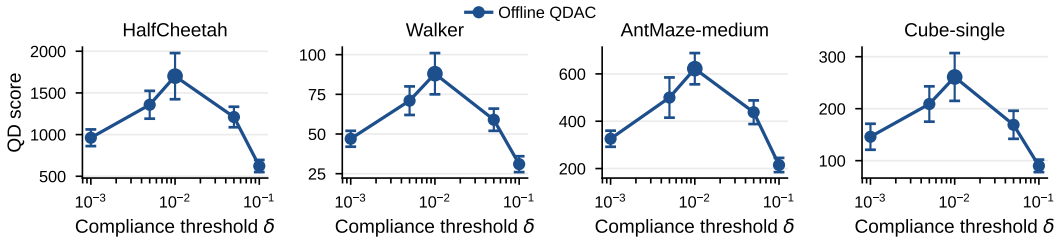

Figure 9: QD score (mean $\pm$ std over 5 seeds) as a function of the compliance threshold δ, averaged over datasets per environment.

Results. QD score peaks at $\delta = 0.01$ across every environment and degrades on both sides. A threshold that is too strict ($\delta = 0.001$) rarely allows the reward gradient to activate, so the policy optimizes almost exclusively for compliance. A threshold that is too loose ($\delta = 0.1$) allows the reward gradient to dominate too early, causing the policy to drift off-target. The default $\delta = 0.01$ used throughout the paper sits at this optimum for every environment tested.

[1] $\delta = 0.01$ is the default used elsewhere in the paper.

Takeaway: Offline QDAC is sensitive to the compliance threshold δ, with too strict or too loose a gate hurting QD score in opposite ways. The default value used throughout the paper is near-optimal across all environments tested.

E.9 Empirical Validity of the SF Compliance Bound

Proposition 1 upper bounds the true compliance error using a surrogate compliance error under $s \sim \rho_z^{\pi_\theta}$, but since we are in the Offline setting, we instead use $s \sim \mathcal{D}$ (Section 4.1, Remark 1). We check empirically whether the surrogate compliance error remains a good proxy for the true error under this substitution.

Experiment. On Halfcheetah Expert, we save policy checkpoints at regular intervals throughout Offline QDAC training and, for each checkpoint, query a grid of target descriptors z. For every descriptor z we compute the true compliance error $\|\mathrm{BD}(\pi_\theta(\cdot|z)) - z\|$ from real environment rollouts, and the offline surrogate compliance error $\mathbb{E}_{s\sim\mathcal{D}}\left[\|(1-\gamma)\psi(s, \pi_\theta(s|z), z) - z\|\right]$ computed on dataset states. We plot true error against the surrogate compliance error for every descriptor z, and report their correlation and the fraction of descriptors for which the surrogate underestimates the true error.



Figure 10: **True vs. surrogate compliance error.** Offline surrogate compliance error (Eq. 6) plotted against true compliance error ($\|\mathrm{BD}(\pi(\cdot|z)) - z\|$) for queried descriptors on Halfcheetah Expert.

Results. The offline surrogate compliance error correlates moderately with the true compliance error ($r = 0.69$). In 93% of queried descriptors, the surrogate is above the true error; in the remaining 7%, it underestimates the true error. These underestimation cases do not constitute violations of Proposition 1, since the proposition is stated under the policy-induced stationary distribution, whereas our practical surrogate is evaluated on states sampled from the offline dataset. Rather, they quantify the approximation introduced by replacing the on-policy distribution with offline data.

Takeaway: The offline surrogate compliance error is a moderately correlated proxy for the true compliance error, supporting its use as a training-time signal in Offline QDAC despite the offline setting preventing exact evaluation of Proposition 1.