Challenges and Opportunities in Offline Reinforcement Learning from Visual Observations

Cong Lu[*] University of Oxford

Philip J. Ball[*] University of Oxford

Tim G. J. Rudner University of Oxford

Jack Parker-Holder University of Oxford

Michael A. Osborne University of Oxford

Yee Whye Teh University of Oxford

Abstract

Offline reinforcement learning has shown great promise in leveraging large pre-collected datasets for policy learning, allowing agents to forgo often-expensive online data collection. However, to date, offline reinforcement learning from *visual observations* has been relatively under-explored, and there is a lack of understanding of where the remaining challenges lie. In this paper, we seek to establish simple baselines for continuous control in the visual domain. We show that simple modifications to two state-of-the-art vision-based online reinforcement learning algorithms, DreamerV2 and DrQ-v2, suffice to outperform prior work and establish a competitive baseline. We rigorously evaluate these algorithms on both existing offline datasets and a new testbed for offline reinforcement learning from visual observations that better represents the data distributions present in real-world offline reinforcement learning problems, and open-source our code and data to facilitate progress in this important domain. Finally, we present and analyze several key desiderata unique to offline RL from visual observations, including visual distractions and visually identifiable changes in dynamics.

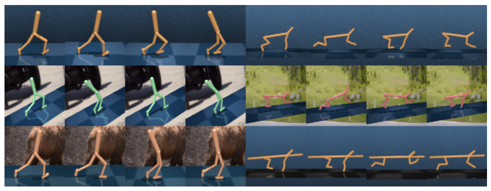
1 Introduction

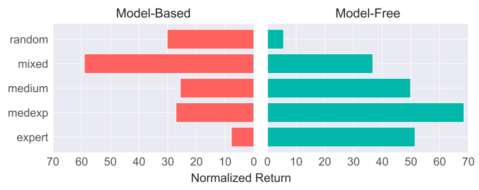
Many prominent recent successes in deep learning have come from training models on large datasets [6, 10, 30]. By contrast, the reinforcement learning (RL, Sutton and Barto [54]) paradigm is typically characterized by agents learning from *online interaction*, which is often expensive or potentially even impractical to collect in abundant quantities. *Offline* (or batch) reinforcement learning [11, 38] aims to address these issues by leveraging pre-collected datasets to train and deploy autonomous agents without requiring any online interaction with an environment.

While offline reinforcement learning algorithms, both model-based [2, 26, 62] and model-free [14, 16, 28, 31, 32, 56], have mastered challenging continuous control tasks, most prior works have relied on access to proprioceptive states [13]. By contrast, there exists only a single study of offline reinforcement learning from *visual observations* [47] for continuous control tasks, a problem setting for which there exist neither well-designed benchmarking tasks, nor any carefully evaluated baselines.

Training agents offline from visual observations provides an opportunity to make reinforcement learning more widely applicable to real-world settings, where we have access to vast quantities of visual observations of desirable behaviors. For example, in autonomous driving [25], large quantities of visual offline data already exist but have not been fully utilized [41, 61]. Similarly, in robotics, data collection is expensive due to costs associated with set-up and human supervision. Effective,

[*]Equal contribution. Correspondence to `cong.lu@stats.ox.ac.uk` and `ball@robots.ox.ac.uk`.


(a) Sampled images from V-D4RL.



(b) Model-based vs. Model-free on V-D4RL.

Figure 1: We introduce V-D4RL, a benchmarking suite for offline reinforcement learning from visual observations, which includes a comprehensive set of D4RL-style datasets and modalities unique to learning from visual observations.

transferable offline reinforcement learning could allow us to reuse datasets gathered previously for different tasks or settings for unseen new problems [8]. Unlocking this potential would represent significant progress towards learning general-purpose agents for realistic problems.

To enable the development of effective, robust, and adaptive algorithms for offline RL from visual observations, we present a suite of carefully designed datasets and benchmarking tasks for this burgeoning domain. We use these tasks to establish simple performance baselines, to study how the composition of vision-based offline datasets affects performance of different types of RL algorithms, and to evaluate the extent to which algorithms for offline RL from visual observations satisfy a set of *desiderata*, including robustness to visual distractions, generalization across environment dynamics, and improved performance at scale. Our evaluation identifies clear failure modes of the baseline methods and highlights opportunities and open problems for future work which can be tackled with our benchmark.

Recent progress in offline reinforcement learning from proprioceptive observations has been driven by well-designed and easy-to-use evaluation testbeds and baselines. We hope that V-D4RL and the analysis in this paper will help facilitate the development of robust RL agents that leverage large, diverse, and often imperfect offline datasets of visual observations across tasks and deployment settings. We open-source our code and data at `https://github.com/conglu1997/v-d4rl`.

The core contributions of this paper are as follows:

1. We present a benchmark for offline RL from visual observations of DMCONTROL SUITE (DMC) tasks [55]. This benchmark, **Vision Datasets for Deep Data-Driven RL** (V-D4RL), follows the design principles of the popular D4RL benchmark [13].
2. We identify **three key desiderata** for realistic offline RL from visual observations: robustness to distractions [53], generalization across dynamics [64], and improved performance for offline reinforcement learning at scale. We present a suite of evaluation protocols designed to test whether offline RL algorithms satisfy these desiderata.
3. We establish model-based and model-free baselines for offline RL from visual observations. We do so by modifying state-of-the-art online RL algorithms, **DreamerV2** [21] and **DrQ-v2** [58], which showcase the relative strengths of model-based and model-free algorithms for offline RL from visual observations. We use these algorithms to provide simple baselines for the aforementioned desiderata to serve as a measure of progress for future advances in this domain.

2 Preliminaries

We model the environment as a Markov Decision Process (MDP), defined as a tuple $M = (\mathcal{S}, \mathcal{A}, P, R, \rho_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces respectively, $P(s'|s, a)$ the transition dynamics, $R(s, a)$ the reward function, ρ_0 the initial state distribution, and $\gamma \in (0, 1)$ the discount factor. The standard goal in online reinforcement learning is to optimize a policy $\pi(a|s)$ that maximizes the expected discounted return $\mathbb{E}_{\pi,P,\rho_0}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$ through interactions with the environment.

In *offline reinforcement learning*, the policy is not deployed in the environment until test time. Instead, the algorithm has access to a fixed dataset $\mathcal{D}_{\text{env}} = \{(s_i, a_i, r_i, s_{i+1})\}_{i=1}^{N}$, collected by one or more behavioral policies π_b. Following Yu et al. [62], we refer to the distribution from which $\mathcal{D}_{\text{env}}$ was sampled as the *behavioral distribution*.

We first describe recent advancements in offline RL and RL from visual observations through the lens of model-based and model-free methods.

2.1 Offline Reinforcement Learning Paradigms

Model-based. A central problem in offline reinforcement learning is over-estimation of the value function [54] due to incomplete data [31]. Model-based methods in offline RL provide a natural solution to this problem by penalizing the reward from model-rollouts by a suitable measure of uncertainty. Yu et al. [62] provide theoretical justification for this approach by constructing a pessimistic MDP (P-MDP) by lower-bounding the expected true return, $\eta_M(\pi)$, using the error between the estimated and true model dynamics.

However, this quantity is usually not available without access to the true environment dynamics, so algorithms such as MOPO and MOReL in practice [26, 62] penalize reward with a surrogate measure of uncertainty. Those algorithms train an ensemble of K probabilistic dynamics models [44], and define a heuristic based on the ensemble predictions. Recent work [40] has shown that a more optimal choice to approximate true dynamics error is the standard deviation of the ensemble's mixture distribution, as proposed by Lakshminarayanan et al. [35].

Model-free. In the model-free paradigm, we lose the natural measure of uncertainty provided by the model. In lieu of this, algorithms such as CQL [32] attempt to avoid catastrophic over-estimation by penalizing actions outside the support of the offline dataset with a wide sampling distribution over the action bounds. Recently, Fujimoto and Gu [14] have shown that a minimal approach to offline reinforcement learning works in proprioceptive settings, where offline policy learning with TD3 [15] can be stabilized by augmenting the loss with a behavioral cloning term.

2.2 Reinforcement Learning from Visual Observations

Recent advances in reinforcement learning from visual observations have been driven by use of data augmentation, contrastive learning and learning powerful recurrent world models of the environment. We describe the current state-of-the-art in model-based and model-free methods.

Model-based (DreamerV2, Hafner et al. [21]). DreamerV2 learns a powerful model of the environment using a Recurrent State Space Model (RSSM, Hafner et al. [19, 20], and predicts ahead using compact model latent states. The particular instantiation used in DreamerV2 uses model states s_t containing a deterministic component h_t, implemented as the recurrent state of a Gated Recurrent Unit (GRU, [9]), and a stochastic component z_t with categorical distribution. The actor and critic are trained from imagined trajectories of latent states, starting at encoded states of previously encountered sequences.

Model-free (DrQ-v2, Yarats et al. [59]). DrQ-v2 is an off-policy algorithm for vision-based continuous control, which uses data-augmentation [36, 60] of the state and next state observations. The base policy optimizer is DDPG [39], and the algorithm uses a convolutional neural network (CNN) encoder to learn a low-dimensional feature representation.

3 Baselines for Offline Reinforcement Learning from Visual Observations

In this section, we begin by motivating our creation of a new benchmark (V-D4RL), introduce our new simple baselines combining recent advances in offline RL and vision-based online RL, and present a comparative evaluation of current methods on V-D4RL. Comprehensive and rigorous benchmarks are crucial to progress in nascent fields. To our knowledge, the only prior work that trains vision-based offline RL agents on continuous control tasks is LOMPO [47]. We analyze their datasets in Section 3.3.1 and find they do not conform to standard D4RL convention.

3.1 Adopting D4RL Design Principles

In this section, we outline how to generate D4RL-like vision-based datasets for V-D4RL.

To generate offline datasets of visual observations, we consider the following three DMControl Suite (DMC) environments:

- **walker-walk**: a planar walker is rewarded for being upright and staying close to a target velocity.
- **cheetah-run**: a planar biped agent is rewarded linearly proportional to its forward velocity.
- **humanoid-walk**: a 21-jointed humanoid is rewarded for staying close to a target velocity. Due to the huge range of motion styles possible, this environment is *extremely challenging* with many local minima and is included as a stretch goal.

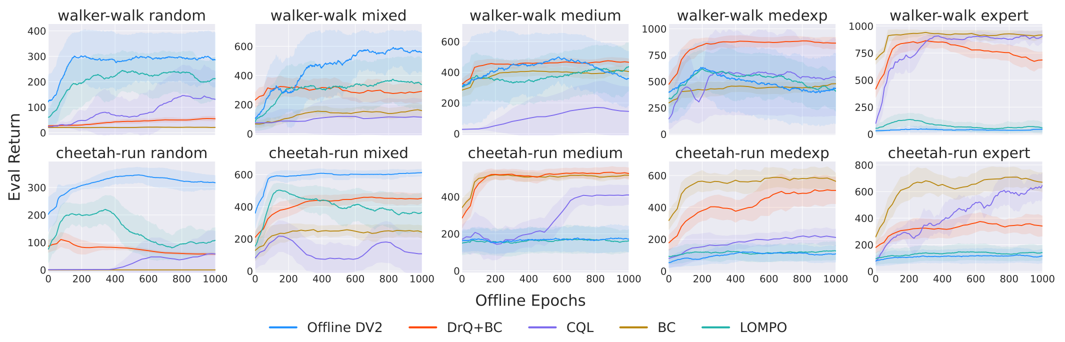


Figure 2: Rigorous comparison on datasets from the V-D4RL benchmark, each setting is averaged over 6 seeds with error bar showing one standard deviation. Total gradient steps are normalized under epochs, and we plot the un-normalized evaluated return. We note that the model-free and BC baselines are far more stable than the model-based.

From these environments, we follow a D4RL-style procedure in considering five different behavioral policies for gathering the data. As in D4RL, the base policy used to gather the data is Soft Actor–Critic (SAC, Haarnoja et al. [18]) on the proprioceptive states.

- **random**: Uniform samples from the action space.
- **medium-replay (mixed)**: The initial segment of the replay buffer until the SAC agent reaches medium-level performance.
- **medium**: Rollouts of a fixed medium-performance policy.
- **expert**: Rollouts of a fixed expert-level policy.
- **medium-expert (medexp)**: Concatenation of medium and expert datasets above.

By default, each dataset consists of 100,000 total transitions (often $10\times$ less than in D4RL) in order to respect the memory demands of vision-based tasks. The cheetah and humanoid medium-replay datasets consist of 200,000 and 600,000 transitions respectively due to the increased number of samples required to train policies on these environments. Full statistics of each dataset are given in Appendix A. Further details on data generation are given in Appendix B.

3.2 Baselines

We show that for the two state-of-the-art online vision-based RL algorithms described in Section 2.2, simple adjustments from the proprioceptive literature suffice to transfer them to the offline setting. In Section 3.3, we demonstrate that these baselines provide a new frontier on our benchmark and on prior datasets. Additional details and hyperparameters for our algorithms are given in Appendix C.

Model-based. For DreamerV2, the latent stochastic states are sampled from an ensemble, which allows us to naturally define a penalty for the reward based on the dynamics uncertainty as in [62]. We adopt a similar approach to that studied in [40] and use the mean-disagreement of the ensemble. Thus, the reward at each step becomes:

$$\tilde{r}(s,a) = r(s,a) - \lambda \sum_{k=1}^{K} (\mu_\phi^k(s,a) - \mu^*(s,a))^2, \tag{1}$$

where λ is a normalization constant and $\mu^*(s,a) = \frac{1}{K}\sum_{k=1}^{K} \mu_\phi^k(s,a)$ is the mean over the dynamics ensemble. Instead of interleaving model-training steps and policy optimization steps, we simply perform one phase of each. We refer to this algorithm as **Offline DV2**.

Model-free. For DrQ-v2, we note that the base policy optimizer shares similarities with TD3 [15], which has recently been applied effectively in offline settings from proprioceptive states by simply adding a regularizing behavioral-cloning term to the policy loss, resulting in the algorithm TD3+BC [14]. Concretely, the policy objective becomes:

$$\pi = \underset{\pi}{\operatorname{argmax}} \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{env}}} \left[\lambda Q(s,\pi(s)) - (\pi(s) - a)^2 \right], \tag{2}$$

where λ is a normalization term, Q is the learned value function and π is the learned policy. We apply the same regularization to DrQ-v2, and call this algorithm: **DrQ+BC**.

Table 1: Final performance on V-D4RL averaged over 6 seeds, with one standard deviation given as error for full transparency. Evaluated return is mapped from $[0, 1000]$ to $[0, 100]$. Our model-based method does best on the diverse low-reward datasets, model-free on the diverse high-reward datasets and behavioral cloning on the narrow expert data. We list the 'average rank' for each algorithm computed over each individual dataset.

Environment		Offline DV2	DrQ+BC	CQL	BC	LOMPO
walker-walk	random	**28.7** ±**13.0**	5.5 ±0.9	14.4 ±12.4	2.0 ±0.2	**21.9** ±**8.1**
	mixed	**56.5** ±**18.1**	28.7 ±6.9	11.4 ±12.4	16.5 ±4.3	34.7 ±19.7
	medium	34.1 ±19.7	**46.8** ±**2.3**	14.8 ±16.1	40.9 ±3.1	43.4 ±11.1
	medexp	43.9 ±34.4	**86.4** ±**5.6**	56.4 ±38.4	47.7 ±3.9	39.2 ±19.5
	expert	4.8 ±0.6	68.4 ±7.5	**89.6** ±**6.0**	**91.5** ±**3.9**	5.3 ±7.7
cheetah-run	random	**31.7** ±**2.7**	5.8 ±0.6	5.9 ±8.4	0.0 ±0.0	11.4 ±5.1
	mixed	**61.6** ±**1.0**	44.8 ±3.6	10.7 ±12.8	25.0 ±3.6	36.3 ±13.6
	medium	17.2 ±3.5	**53.0** ±**3.0**	40.9 ±5.1	**51.6** ±**1.4**	16.4 ±8.3
	medexp	10.4 ±3.5	50.6 ±8.2	20.9 ±5.5	**57.5** ±**6.3**	11.9 ±1.9
	expert	10.9 ±3.2	34.5 ±8.3	**61.5** ±**4.3**	**67.4** ±**6.8**	14.0 ±3.8
humanoid-walk	random	0.1 ±0.0	0.1 ±0.0	0.2 ±0.1	0.1 ±0.0	0.1 ±0.0
	mixed	0.2 ±0.1	**15.9** ±**3.8**	0.1 ±0.0	**18.8** ±**4.2**	0.2 ±0.0
	medium	0.2 ±0.1	6.2 ±2.4	0.1 ±0.0	**13.5** ±**4.1**	0.1 ±0.0
	medexp	0.1 ±0.0	7.0 ±2.3	0.1 ±0.0	**17.2** ±**4.7**	0.2 ±0.0
	expert	0.2 ±0.1	2.7 ±0.9	1.6 ±0.5	**6.1** ±**3.7**	0.1 ±0.0

Prior work. Since DrQ-v2 is an actor-critic algorithm, we may also use it to readily implement the **CQL** [32] algorithm by adding the CQL regularizers to the Q-function update. We additionally compare against **LOMPO** [47], and behavioral cloning (**BC**, [3, 5]), where we apply supervised learning to mimic the behavioral policy. Offline DV2 is closely related to LOMPO as both use an RSSM [19, 20] as the fundamental model, however Offline DV2 is based on the newer discrete RSSM with an uncertainty penalty more closely resembling the ensemble penalties in supervised learning [35] and uses KL balancing during training [21].

3.3 Comparative Evaluation

We now evaluate the five algorithms described in Section 3.2 on a total of fifteen datasets. To provide a fair evaluation, we provide full training curves for each algorithm in Figure 2 and summarize final performance with error in Table 1. Since no online data collection is required, we measure progress through training via an "offline epochs" metric which we define in Appendix D.

Table 1 shows a clear trend: Offline DV2 is the strongest on the random and mixed datasets, consisting of lower-quality but diverse data, DrQ+BC is the best on datasets with higher-quality but still widely-distributed data and pure BC outperforms on the high-quality narrowly-distributed expert data. We see from Table 1 and Figure 2 that DrQ+BC is extremely stable across seeds and training steps and has the highest overall performance. CQL is also a strong baseline, especially on expert data, but requires significant hyperparameter tuning per dataset, often has high variance across seeds, and is also far slower than DrQ+BC to train. Finally, no algorithm achieves strong performance on the challenging humanoid datasets, mirroring the online RL challenges [21, 58], with only the supervised BC and, by extension, DrQ+BC showing marginal positive returns (for more graphs, see Appendix E.1).

Perhaps surprisingly, Offline DV2 learns mid-level policies from random data on DMC environments. Furthermore, the random data are more challenging than their D4RL equivalents because there is no early termination, and thus mostly consists of uninformative failed states; this shows the strength of model-based methods in extracting signal from large quantities of suboptimal data. On the other hand, Offline DV2 is considerably weaker on the expert datasets that have narrow data distributions. For these environments, we find the uncertainty penalty is uninformative, as discussed in Appendix F.2.

Taking all these findings into consideration leads us to our first open problem, which we believe continued research using our benchmark can help to answer:

Open Problem 1: Can a single algorithm outperform both the model-free and model-based baselines, and produce strong performance across *all* offline datasets?

Table 2: We confirm that our simple baselines outperform LOMPO on the original walker-walk data provided by Rafailov et al. [47]. We report final performance mapped from $[0, 1000]$ to $[0, 100]$ averaged over 6 seeds.[1] We show our baselines are more performant than LOMPO on their benchmark. CQL numbers taken from [47], we were unable to verify these results using V-D4RL-CQL.

LOMPO Dataset	LOMPO	Offline DV2	DrQ+BC	CQL
medium-replay	61.3 ±9.1	**76.3 ±3.1**	31.1 ±3.7	14.7
medium-expert	69.0 ±24.1	72.3 ±20.1	**73.3 ±3.5**	45.1
expert	52.4 ±35.7	59.4 ±26.6	**90.8 ±2.2**	40.3

3.3.1 Comparison to the LOMPO Benchmark

For a fair comparison to LOMPO, we also benchmark on the data used in Rafailov et al. [47] on the DMC Walker-Walk task. In the LOMPO benchmark, the datasets are limited to three types: {medium-replay, medium-expert and expert}. We provide final scores in Table 2. While LOMPO struggles on V-D4RL, it performs reasonably on its own benchmark. However, LOMPO is still outperformed by Offline DV2 on all datasets, whereas DrQ+BC is the best on two datasets.

We may explain the relative strength of LOMPO on this benchmark by noting that the medium-expert dataset used by Rafailov et al. [47] is described as consisting of the second half of the replay buffer after the agent reaches medium-level performance, thus containing far more diverse data than a bimodal D4RL-style concatenation of two datasets. Furthermore, the expert data is far more widely distributed than that of a standard SAC expert, as we confirm in the statistics in Table 6 of Appendix A.

4 Desiderata for Offline Reinforcement Learning from Visual Observations

A key ingredient in recent advances in deep learning is the use of large and diverse datasets, which are particularly prevalent in vision, to train models. Such datasets enable the learning of *general* features that can be successfully transferred to downstream tasks, even when the original task bears little immediate similarity with the transferred task [46, 57]. This is a clear rationale for adopting visual observations in offline RL; by leveraging large quantities of diverse high-dimensional inputs, we should be able to learn generalizable features and agents for control. However, combining rich visual datasets with RL presents its own unique challenges. In this section, we present important desiderata that highlight this, and conclude each with an open problem that requires further research.[2]

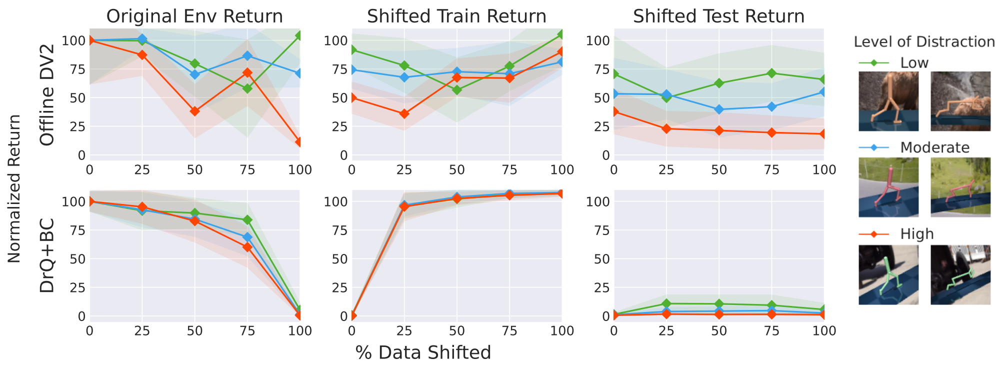


Figure 3: Both DrQ+BC and Offline DV2 readily support training datasets with different distractions (mixture of original and shifted train). Offline DV2 additionally shows the ability to generalize to unseen distractions (shifted test) whereas DrQ+BC is more brittle. Return is normalized against unshifted performance without distractions from Table 1 and averaged over 6 seeds. Unnormalized returns are provided in Tables 10 and 11.

4.1 Robustness to Visual Distractions

One desideratum for offline RL from visual observations is the ability to learn a policy from data collected under multiple different settings. For example, quadrupedal robots deployed at different times of day (e.g., morning and night) will likely gather data having significantly different visual properties, such as brightness and range of visibility. Although the robot may produce similar

[1]It is unclear how the original scores in Rafailov et al. [47] were normalized.

[2]As CQL is quite sensitive to hyperparameters per environment, in the following sections we use the more robust Offline DV2 and DrQ+BC algorithms.

behaviors in both deployments, superficial differences in visual outputs present a host of opportunities for the agent to learn spurious correlations that prevent generalization [48, 52].

A key opportunity that arises is the potential to *disentangle* sources of distractions through training on multiple settings, facilitating the learning of general features. By contrast, proprioceptive observations do not generally have distractions, as they are typically low-dimensional and engineered for the task of interest [37]. This also limits their ability to transfer, as it is unclear how to incorporate features learned under one set of agent geometries to another.

To test a simplified version of this challenge, we train our baseline agents using newly created datasets featuring varying visual augmentations from the Distracting Control Suite [53]. This suite provides three levels of distractions (i.e., low, moderate, high), and each distraction represents a shift in the data distribution. We subsequently refer to the level of distraction as the "shift severity" [50]. The offline datasets are then constructed as mixtures of samples from the original environment *without distractions* and samples from an environment with a *fixed distraction* level. Further details of how we generate this data are given in Appendix B. The learned policies are then evaluated on test environments featuring unseen distractions of the same shift severity.

We compare the baseline algorithms, Offline DV2 and DrQ+BC on datasets that they excel on in Section 3 and Section 4.3: walker-walk random with 100,000 datapoints and cheetah-run medium-expert with 1 million respectively. The returns normalized by unshifted performance are shown in Figure 3. Offline DV2 is able to accommodate datasets with mixed distractions and generalizes reasonably well to unseen test distractions, especially when trained with 'low' and 'moderate' levels of shift severity. Similarly, DrQ+BC is robust to multiple training distractions, with little to no degradation in performance. However, the policy learned is brittle with respect to unseen distractions, and performs significantly worse on the test environments.

Overall, both Offline DV2 and DrQ+BC represent strong baselines for mixed datasets. Interestingly, Offline DV2 demonstrates strong generalization to unseen distractions. This can be explained by generalization that occurs in the trained world-model, which uses a self-supervised loss; we discuss reasons behind this in Appendix F.3. This could be improved even further with recent reconstruction-free variants of DreamerV2 [43, 45] which have shown robustness to distractions. On the other hand, we observe DrQ+BC generalizes poorly to unseen distractions, presenting a direction for future work using our datasets to learn robust *model-free* agents. This directly leads us to our next open problem:

Open Problem 2: How can we improve the generalization performance of offline model-free methods to unseen visual distractions?

4.2 Generalization Across Environment Dynamics

Another desideratum of offline RL from visual observations is learning policies that can generalize across multiple dynamics. This challenge manifests in three clear ways. Firstly, we will likely collect data from multiple agents that each have different dynamics, and must therefore learn a policy that can perform well when deployed on any robot that gathered the data (i.e., train time robustness). Secondly, we may be provided with asymmetric data, featuring scarce coverage of particular dynamics, and therefore require the ability to leverage data from more abundant sources (i.e., transferability). Thirdly, we may be presented with *unseen* dynamics at deployment time, and must therefore learn a policy that is robust to these changes (i.e., test time robustness).

A key opportunity that arises in visual observations is the improved richness of the underlying dataset compared to proprioceptive data. For instance, *some dynamics changes may be visually obvious* (e.g., changed limb sizes, broken actuators), whereas in the proprioceptive setting such information may not be available. Without this information, we must turn to meta-RL [49, 65] or HiPMDP [27, 64] approaches that try to infer the missing information from gathered trajectories, adding complexity to the RL process. In contrast, this information can be contained explicitly in visual observations, and should allow adaption to a range of downstream tasks without complex inference methods.

To test this hypothesis, we consider two settings from the MTEnv benchmark [51] which adapts DMC: cheetah-run with modified torso length and walker-walk with modified leg length. We follow an analogous approach to [64] where we consider eight different settings {A - H} ordered in terms of limb length, and construct new offline datasets using {B, C, F, G} in equal proportions as our training data. The settings {A, H} are considered the *extrapolation* generalization environments and {D, E} *interpolation* generalization. Data generation details are provided in Appendix B.

We evaluate Offline DV2 and DrQ+BC on medium-expert datasets of size 100,000 and 1 million respectively and show the results in Table 3. We see that DrQ+BC learns policies that are suitable for transfer across multiple tasks in both walker and cheetah, and maintains that performance on the interpolation test environments. For the extrapolation environments, we see an average drop of around 30%. While this may represent adequate performance, especially compared to a medium policy, it is a striking drop when compared to performance on in-distribution dynamics. This suggests there is a dynamics generalization gap that remains for model-free methods when extrapolating, and represents clear opportunities for further research.

Table 3: Evaluation on the DMC-Multitask benchmark using *medexp* data for Offline DV2, DrQ+BC and BC. Normalized performance from [0, 1000] to [0, 100] over 6 seeds. Our algorithms learn multitask policies from visual observations, with a slight generalization gap for extrapolation tasks.

Algorithm	**Environment**	**Eval. Return**		
		Train Tasks	**Test Interp.**	**Test Extrap.**
DrQ+BC	walker	90.8	91.4	65.1
	cheetah	71.6	65.1	43.2
BC	walker	61.2	61.4	47.2
	cheetah	69.7	61.3	39.6
Offline DV2	walker	23.2	16.5	19.8
	cheetah	8.2	7.2	9.6

Offline DV2 displays similar trends (results on random datasets are in Appendix E.3). On the random data, Offline DV2 learns a similar quality policy to that on the base environment, but experiences no deterioration in performance on the test environments in walker or cheetah. Thus, we demonstrate the sufficiency of both Offline DV2 and DrQ+BC as baselines in multitask offline RL *without any modification*; model-based approaches further admit opportunities for zero-shot generalization [4].

We now contrast our work to that of [64], where a multitask policy was trained using a total of 3.2 million timesteps of online data collection. Whilst it is hard to compare offline and online results, our DrQ+BC algorithm uses less data, with 1 million total timesteps, and obtains similar extrapolation return on the walker environments. This supports a similar conclusion reached by [34] who show that our approach, simply minimizing the sum of the task losses, is drastically underestimated in the literature. As noted before, we suffer a comparatively larger drop in performance, lending further evidence that closing this generalization gap should be prioritized.

In conclusion, we believe there are many further avenues for future research using these benchmarks; an immediate open problem we have identified is as follows:

Open Problem 3: How can we improve generalization to new dynamics that are not contained in the offline dataset?

4.3 Improved Performance with Scale

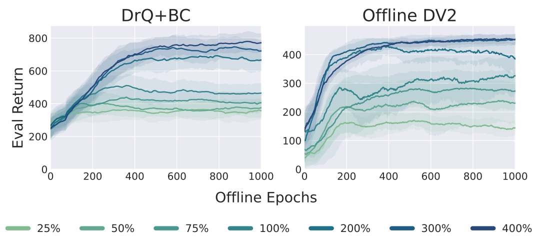


Figure 4: Sensitivity analysis on dataset size for both Offline DV2 (walker-walk random) and DrQ+BC (cheetah-run medexp). Both methods scale with more data, but receive diminishing returns past 2× the original size. Performance averaged over 4 seeds.

Learning from large datasets presents huge opportunities for learning general agents for control. To make use of them, we need to understand how our baselines scale with dataset size. We analyze our base choice of 100,000 observations for V-D4RL in Figure 4, where we vary the size of the walker-walk random dataset for Offline DV2 and the cheetah-run medexp dataset for DrQ+BC in the range of $\{0.25\times, \ldots, 4\times\}$ the size of the original dataset. We observe a monotonic increase in the performance of both Offline DV2 and DrQ+BC with increasing dataset size but hit diminishing returns past 2× the original size. We note, perhaps surprisingly, that Offline DV2 can reach ≈500 total return from random data that average 10× less.

For the walker and cheetah datasets with fixed distributions—random, medium, medium-expert and expert—Table 4 shows an average increase of 42.1% for Offline DV2 and 32.5% for DrQ+BC compared to 10.8% for BC when we scale the dataset size to 500,000, showing that the reinforcement learning algorithms make far better use of increased data than supervised behavioral cloning. However, a crucial difference between Offline DV2 and DrQ+BC is that DrQ+BC handles larger offline datasets far more readily. DrQ+BC policy training for the same number of epochs on 500,000 and 100,000 observations takes 8 and 1.6 hours respectively on a V100 GPU. This is significantly quicker than Offline DV2, which takes 10 hours to train on 100,000 observations; we discuss this further in Appendix F.2. This significant computational discrepancy leads to a clear open problem:

Open Problem 4: How can we scale model-based methods to larger datasets?

Table 4: The reinforcement learning algorithms readily scale to higher dataset sizes, compared to supervised behavioral cloning, with a particular benefit to the *medexp* and *expert* datasets for DrQ+BC and the *random* and *medium* datasets for Offline DV2. Results are averaged over 6 seeds, with one standard deviation given as error. The evaluated return is mapped from $[0, 1000]$ to $[0, 100]$, and the *mixed* dataset is excluded. The *medexp* dataset is a concatenation of the *medium* and *expert* datasets.

Environment		**Offline DV2**		**DrQ+BC**		**BC**	
		100K	**500K**	**100K**	**500K**	**100K**	**500K**
walker	random	28.7 ±13.0	49.9 ±1.6	5.5 ±0.9	3.5 ±0.6	2.0 ±0.2	2.1 ±0.3
	medium	34.1 ±19.7	61.3 ±10.9	46.8 ±2.3	51.0 ±1.1	40.9 ±3.1	40.9 ±3.0
	medexp	43.9 ±34.4	38.9 ±28.1	86.4 ±5.6	94.1 ±2.0	47.7 ±3.9	48.8 ±5.3
	expert	4.8 ±0.6	7.1 ±5.3	68.4 ±7.5	94.2 ±2.3	91.5 ±3.9	95.1 ±2.5
cheetah	random	31.7 ±2.7	40.8 ±4.2	5.8 ±0.6	10.6 ±0.7	0.0 ±0.0	0.0 ±0.0
	medium	17.2 ±3.5	39.2 ±14.4	53.0 ±3.0	57.3 ±1.2	51.6 ±1.4	52.9 ±1.3
	medexp	10.4 ±3.5	9.7 ±5.0	50.6 ±8.2	79.1 ±5.6	57.5 ±6.3	69.6 ±10.6
	expert	10.9 ±3.2	11.3 ±4.7	34.5 ±8.3	75.3 ±7.5	67.4 ±6.8	87.8 ±1.9
Average Overall		22.7 ±10.1	32.3 ±9.3	43.9 ±4.6	58.1 ±2.6	44.8 ±3.2	49.7 ±3.1
Percentage Gain		+42.1%		+32.5%		+10.8%	

5 Related Work

There has been significant recent progress in offline RL, giving rise to many benchmarks. We list several here; to our knowledge, no contemporary works feature tasks related to distractions or changed dynamics.

Benchmarks for continuous control on states. D4RL [13] is the most prominent benchmark for continuous control with proprioceptive states. The large variety of data-distributions have allowed for comprehensive benchmarking [26, 29, 32, 62, 63] and understanding the strengths and weaknesses of offline algorithms. Our work aims to establish a similar benchmark for vision-based tasks. RL Unplugged [17] also provides data on the DMControl Suite, but focuses on proprioceptive states.

Analysis on characteristics of offline datasets. Recent work [12] has sought to understand when offline RL algorithms outperform behavioral cloning in the proprioceptive setting. Kumar et al. [33] recommend BC for many settings but showed theoretically that offline RL was preferable in settings combining expert and suboptimal data, which we confirm in Tables 1 and 4.

Vision-based discrete control datasets. Whilst there has been a lack of suitable benchmarks for vision-based offline continuous control, vision-based datasets for discrete control have been created for Atari [1]. However, at 50M samples per environment, this poses significant challenges; we believe that V-D4RL's 100K benchmark represents a far more approachable challenge for practitioners.

Offline vision-based robotics. Offline vision-based learning is an active area of robotics research. In [24], policies are fine-tuned on real interactions, after offline pre-training on visual data of robot interactions. [8] follows a similar setup, and explores the representation learning for controllers. Our work complements this, is not focused on goal-conditioned learning, and is open source.

6 Conclusion

In this paper, we took the first steps towards establishing benchmarking tasks and competitive baselines for offline reinforcement learning from visual observations. Until now, work in this space has been nascent, with ad-hoc analyses leading to unclear comparisons. To address the lack of meaningful evaluations and comparative analyses in this space, we provided a set of straightforward and standardized benchmarking tasks that follow popular low-dimensional equivalent experiment setups and presented competitive model-based and model-free baselines. We analyzed key factors that help explain the performance of these approaches, while also demonstrating their ability to generalize in more challenging settings that are unique to visual observations. We hope this work will be useful to practitioners and researchers alike and that it will provide a springboard for developing offline reinforcement learning methods for real-world continuous-control problems and spark further progress in this space.

Acknowledgments and Disclosure of Funding

Cong Lu and Tim G. J. Rudner are funded by the Engineering and Physical Sciences Research Council (EPSRC). Philip J. Ball is funded through the Willowgrove Studentship. Tim G. J. Rudner is also funded by the Rhodes Trust and by a Qualcomm Innovation Fellowship. We are grateful to Rafael Rafailov for sharing the data used with LOMPO and Edoardo Cetin for his NumPy Array replay buffer implementation. We thank Samuel Daulton, Clare Lyle, Muhammad Faaiz Taufiq for detailed feedback on an early draft. The authors would also like to thank the anonymous ICML 2022 reviewers for constructive feedback which helped improve the paper.

References

[1] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, 2020.

[2] Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=OMNB1G5xzd4.

[3] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129, 1995.

[4] Philip J Ball, Cong Lu, Jack Parker-Holder, and Stephen Roberts. Augmented world models facilitate zero-shot dynamics generalization from a single offline environment. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 619–629. PMLR, 2021. URL http://proceedings.mlr.press/v139/ball21a.html.

[5] Ivan Bratko, Tanja Urbančič, and Claude Sammut. Behavioural cloning: Phenomena, results and problems. *IFAC Proceedings Volumes*, 28(21):143–149, September 1995. ISSN 1474-6670. doi: 10.1016/s1474-6670(17)46716-4. URL https://doi.org/10.1016/s1474-6670(17)46716-4.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[7] Christopher P. Burgess, Irina Higgins, Arka Pal, Loïc Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae. *CoRR*, abs/1804.03599, 2018. URL http://arxiv.org/abs/1804.03599.

[8] Yevgen Chebotar, Karol Hausman, Yao Lu, Ted Xiao, Dmitry Kalashnikov, Jacob Varley, Alex Irpan, Benjamin Eysenbach, Ryan C Julian, Chelsea Finn, and Sergey Levine. Actionable models: Unsupervised offline reinforcement learning of robotic skills. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1518–1528. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/chebotar21a.html.

[9] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

[11] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 6(18):503–556, 2005. URL http://jmlr.org/papers/v6/ernst05a.html.

[12] Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=rif3a5NAxU6.

[13] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning, 2021.

[14] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[15] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR, 2018. URL https://proceedings.mlr.press/v80/fujimoto18a.html.

[16] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062, 2019.

[17] Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gómez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, Gabriel Dulac-Arnold, Jerry Li, Mohammad Norouzi, Matt Hoffman, Ofir Nachum, George Tucker, Nicolas Heess, and Nando deFreitas. Rl unplugged: Benchmarks for offline reinforcement learning, 2020.

[18] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018.

[19] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, pages 2555–2565, 2019.

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

[21] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[22] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[23] Arnav Kumar Jain, Shiva Kanth Sujit, Shruti Joshi, Vincent Michalski, Danijar Hafner, and Samira Ebrahimi Kahou. Learning robust dynamics through variational sparse gating. In *Deep RL Workshop NeurIPS 2021*, 2021. URL https://openreview.net/forum?id=46OhxFeWzyr.

[24] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In Aude Billard,

Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 651–673. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/kalashnikov18a.html.

[25] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day, 2018.

[26] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL: Model-based offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21810–21823. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/f7efa4f864ae9b88d43527f4b14f750f-Paper.pdf.

[27] Taylor W Killian, Samuel Daulton, George Konidaris, and Finale Doshi-Velez. Robust and efficient transfer learning with hidden parameter markov decision processes. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/2227d753dc18505031869d44673728e2-Paper.pdf.

[28] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning, 2021.

[29] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning, 2021.

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25:*, pages 1106–1114, 2012.

[31] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/c2073ffa77b5357a498057413bb09d3a-Paper.pdf.

[32] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1179–1191. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf.

[33] Aviral Kumar, Joey Hong, Anikait Singh, and Sergey Levine. Should i run offline reinforcement learning or behavioral cloning? In *Deep RL Workshop NeurIPS 2021*, 2021. URL https://openreview.net/forum?id=iQibgAN7mT.

[34] Vitaly Kurin, Alessandro De Palma, Ilya Kostrikov, Shimon Whiteson, and M. Pawan Kumar. In defense of the unitary scalarization for deep multi-task learning, 2022.

[35] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6405–6416, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[36] Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19884–19895. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/e615c82aba461681ade82da2da38004a-Paper.pdf.

[37] Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Francois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, December 2018. ISSN 0893-6080. doi: 10.1016/j.neunet.2018.07.006. URL https://doi.org/10.1016/j.neunet.2018.07.006.

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

[39] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[40] Cong Lu, Philip J. Ball, Jack Parker-Holder, Michael A. Osborne, and Stephen J. Roberts. Revisiting design choices in offline model based reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zz9hXVhf40.

[41] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford RobotCar dataset. *The International Journal of Robotics Research*, 36(1):3–15, November 2016. ISSN 0278-3649, 1741-3176. doi: 10.1177/0278364916679498. URL https://doi.org/10.1177/0278364916679498.

[42] Emile Mathieu, Tom Rainforth, N Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4402–4412. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/mathieu19a.html.

[43] Tung D Nguyen, Rui Shu, Tuan Pham, Hung Bui, and Stefano Ermon. Temporal predictive coding for model-based planning in latent space. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8130–8139. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/nguyen21h.html.

[44] D.A. Nix and A.S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pages 55–60 vol.1. IEEE, 1994. doi: 10.1109/icnn.1994.374138. URL https://doi.org/10.1109/icnn.1994.374138.

[45] Masashi Okada and Tadahiro Taniguchi. Dreaming: Model-based reinforcement learning by latent imagination without reconstruction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4209–4215, 2021. doi: 10.1109/ICRA48506.2021.9560734.

[46] Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/w19-4302. URL https://doi.org/10.18653/v1/w19-4302.

[47] Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. In *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pages 1154–1168. PMLR, 2021.

[48] Roberta Raileanu and Rob Fergus. Decoupling value and policy for generalization in reinforcement learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8787–8798. PMLR, 2021. URL https://proceedings.mlr.press/v139/raileanu21a.html.

[49] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97, pages 5331–5340. PMLR, 2019.

[50] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *arXiv preprint arXiv:2006.16971*, 2020.

[51] Shagun Sodhani, Ludovic Denoyer, Pierre-Alexandre Kamienny, and Olivier Delalleau. MTEnv - environment interface for mulit-task reinforcement learning. Github, 2021. URL https://github.com/facebookresearch/mtenv.

[52] Xingyou Song, Yiding Jiang, Stephen Tu, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. In *International Conference on Learning Representations*, 2020.

[53] Austin Stone, Oscar Ramirez, Kurt Konolige, and Rico Jonschkowski. The distracting control suite – a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

[54] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning*. Springer US, second edition, 1992. ISBN 9781461366089, 9781461536185. doi: 10.1007/978-1-4615-3618-5. URL https://doi.org/10.1007/978-1-4615-3618-5.

[55] Yuval Tassa, Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, and Nicolas Heess. dm_control: Software and tasks for continuous control, 2020.

[56] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning, 2019.

[57] Yiting Xie and David Richmond. Pre-training on grayscale ImageNet improves medical image classification. In Laura Leal-Taixé and Stefan Roth, editors, *Computer Vision – ECCV 2018 Workshops*, pages 476–484, Cham, 2019. Springer International Publishing. ISBN 978-3-030-11024-6.

[58] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

[59] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning, 2021.

[60] Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=GY6-6sTvGaf.

[61] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving dataset for heterogeneous multitask learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2020. doi: 10.1109/cvpr42600.2020.00271. URL https://doi.org/10.1109/cvpr42600.2020.00271.

[62] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14129–14142. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/a322852ce0df73e204b7e67cbbef0d0a-Paper.pdf.

[63] Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. COMBO: Conservative offline model-based policy optimization, 2021.

[64] Amy Zhang, Shagun Sodhani, Khimya Khetarpal, and Joelle Pineau. Learning robust state abstractions for hidden-parameter block MDPs. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=fmOOI2a3tQP.

[65] Luisa Zintgraf, Sebastian Schulze, Cong Lu, Leo Feng, Maximilian Igl, Kyriacos Shiarlis, Yarin Gal, Katja Hofmann, and Shimon Whiteson. VariBAD: Variational Bayes-adaptive deep RL via meta-learning. *Journal of Machine Learning Research (JMLR)*, 22(289):1–39, 2021. URL http://jmlr.org/papers/v22/21-0657.html.

Supplementary Material

Table of Contents

A Offline Dataset Characteristics **17**

B Offline Data Generation Details **18**

B.1 Standard Datasets 18

B.2 Visually Distracted Datasets 18

B.3 Multitask Datasets 18

B.4 Choice of Behavioral Policy for Offline Data 18

B.5 Broader Issues with Visual Data 18

C Algorithmic Details **18**

C.1 Offline DV2 19

C.2 DrQ+BC 19

C.3 CQL 19

D Hyperparameter and Experiment Setup **19**

D.1 Offline DV2 19

D.2 DrQ+BC 20

D.3 Behavioral Cloning 20

D.4 CQL 21

D.5 LOMPO 21

D.6 Computational Cost 21

E Further Tabular Results and Training Curves **21**

E.1 Training Curves for the Humanoid Environment 21

E.2 Full Tabular Results for Experiments with Distractions 22

E.3 Further Offline DV2 Results on Multitask Datasets 23

F Further Ablation Studies **23**

F.1 DrQ+BC Random-Expert Ablation Studies 23

F.2 Offline DV2 Training Time and Uncertainty Penalty 23

F.3 Understanding Model-Based Extrapolation 24

A Offline Dataset Characteristics

We provide explicit statistics on the returns of each episode for the datasets used in our main evaluation. This provides a reasonable proxy to how diverse each dataset is.

Table 5: Full summary statistics of per-episode return in the V-D4RL benchmark.

Dataset		Timesteps	Mean	Std. Dev.	Min.	P25	Median	P75	Max.
walker	random	100K	42.3	8.7	30.0	34.9	41.3	46.8	74.6
	mixed	100K	144.5	155.9	10.9	44.3	69.4	162.4	604.9
	medium	100K	439.6	48.4	176.2	423.1	445.5	466.7	538.0
	medexp	200K	704.1	267.7	176.2	445.5	538.0	969.1	990.6
	expert	100K	969.8	12.4	909.2	963.9	969.1	979.5	990.6
cheetah	random	100K	6.6	2.6	1.1	4.7	6.3	8.4	16.3
	mixed	200K	191.2	144.6	2.5	48.3	191.9	303.4	473.8
	medium	100K	523.8	25.5	325.3	509.1	524.2	538.3	578.3
	medexp	200K	707.0	184.9	325.3	524.2	578.3	894.1	905.7
	expert	100K	891.1	11.2	843.0	886.9	894.2	898.5	905.7
humanoid	random	100K	1.1	0.8	0.0	0.5	1.0	1.5	5.7
	mixed	600K	275.7	176.1	0.0	93.7	341.7	423.1	529.0
	medium	100K	573.0	16.7	526.5	560.6	572.9	584.9	609.4
	medexp	200K	715.9	146.1	526.5	572.9	620.6	877.4	889.8
	expert	100K	858.1	42.4	631.8	846.4	877.6	885.5	889.8

We compare this to the LOMPO datasets and find that they are more widely distributed, due to the differing data-collection method.

Table 6: Summary statistics of per-episode return in the LOMPO DMC walker-walk datasets.

Dataset		Timesteps	Mean	Std. Dev.	Min.	P25	Median	P75	Max.
walker	mixed	100K	208.9	144.1	33.3	75.1	172.4	340.5	496.7
	medexp	100K	674.4	92.9	501.3	596.8	679.8	752.5	869.1
	expert	100K	920.6	81.6	17.9	905.9	950.3	957.9	987.0

As we see in Table 6 and Figure 5, the LOMPO walker-walk expert dataset has a standard deviation roughly 8x higher than our expert dataset and has an extremely wide [min, max] range. Furthermore, whilst our medexp dataset is bimodal, the LOMPO medexp dataset's returns are a continuous progression. This reflects that the LOMPO data is sampled from the second half of a replay buffer after medium-level performance is attained, akin to the medium-replay (mixed) datasets.

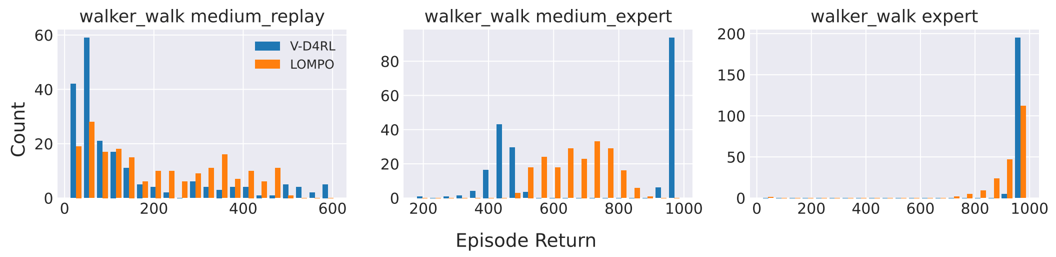


Figure 5: Comparison of the episodic returns from the LOMPO and V-D4RL. We see LOMPO has significantly more diversity in the medium-expert and expert datasets. Note that there is a single episode in the LOMPO expert dataset which has a low return of 17.9.

B Offline Data Generation Details

B.1 Standard Datasets

For the standard offline medium and expert datasets, we first train SAC [18] policies on the proprioceptive states until convergence, taking checkpoints every 10,000 frames of interaction. We use a frame skip of 2, the default in other state-of-the-art vision RL algorithms [21, 58]. We define expert policies as those that have converged in the limit. For DMControl tasks, this typically means near 1,000 reward on the environment they were trained on. We define medium policies as the first saved agent during training that is able to consistently achieve above 500 reward in the environment. We confirm these thresholds are reasonable, as we observe a noticeable gap between the behavior of the medium and expert level policies.

In order to generate the offline visual observations, we deploy the proprioceptive agents in the environment, and save the visual observation rendered from the simulator instead of the proprioceptive state. This provides us with the flexibility to *generate observations of any size* without having to retrain for that resolution (e.g., 84×84 or 64×64). As was done in D4RL, we generate data using a stochastic actor, which involves sampling actions from the Gaussian posterior distribution of the SAC policy, featuring a parameterized variance head that determines the amount of action stochasticity at each state.

For the mixed datasets, we simply store the replay buffer of the medium agent when it finishes training, and convert the proprioceptive observations into visual observations. The cheetah-run mixed dataset is larger because the SAC agent takes longer to reach medium-level performance.

B.2 Visually Distracted Datasets

The visual distractions from the Distracting Control Suite do not manifest themselves in the proprioceptive states, so we can naturally generate the shifted visual observations by simply rendering the distractions on top of the existing visual observations. Thus, we can simply use the same "medium" and "expert" level policies as before, trained on proprioceptive states.

B.3 Multitask Datasets

For the multitask dataset, we follow the same procedure described in Appendix B.1 but instead train a SAC on the proprioceptive states from the modified tasks (e.g., $\{$A-H$\}$). The medium and expert policies are defined in the same way.

B.4 Choice of Behavioral Policy for Offline Data

Interestingly, we found that using online DrQ-v2 as the offline behavioral policy made the tasks significantly easier to learn for all agents. This suggests that the proprioceptive agent may learn behavior modes that are less biased towards being easy under vision-based methods; for instance, DrQ-v2 may be biased towards behavior modes that induce fewer visual occlusions compared to proprioceptive SAC.

B.5 Broader Issues with Visual Data

Large datasets consisting of images often contain systematic biases, which can damage generalization. The datasets constructed in this paper are all synthetic from simulated reinforcement learning environments. However, as we move towards applying offline RL from visual observations to real-world tasks, it is important to take these potential dangers into account and extend existing work in algorithmic fairness from computer vision to our setting.

C Algorithmic Details

We provide additional details for both algorithms here and indicate where our modifications have been made.

C.1 Offline DV2

In the offline setting, it suffices to simply perform one phase of model training and one phase of policy training for the DreamerV2 [21] algorithm. Each episode in the offline dataset is ordered sequentially to facilitate sequence learning. To this instantiation of DreamerV2, we simply add a reward penalty corresponding to the mean disagreement of the dynamics ensemble. During standard DreamerV2 policy training, imagined latent trajectories $\{(s_\tau, a_\tau)\}_{\tau=t}^{t+H}$ are assigned reward $r_\tau = \mathrm{E}\left[q_\theta\left(\cdot \mid s_\tau\right)\right]$ according to the mean output of the reward predictor. The imagined latent states s_t consist of a deterministic component h_t, implemented as the recurrent state of a GRU, and a stochastic component z_t with categorical distribution. The logits of the categorical distribution are computed from an ensemble (with input h_t) over which we compute the mean disagreement.

C.2 DrQ+BC

Here, we simply modify the policy loss term in DrQ-v2 [58] to match the loss given in Fujimoto and Gu [14]. Following the notation from Yarats et al. [58], the DrQ-v2 actor π_ϕ is trained with the following loss:

$$\mathcal{L}_\phi(\mathcal{D}) = -\mathbb{E}_{\boldsymbol{s}_t \sim \mathcal{D}}\left[Q_\theta\left(\boldsymbol{h}_t, \boldsymbol{a}_t\right)\right]$$

where $\boldsymbol{h}_t = f_\xi\left(\operatorname{aug}\left(\boldsymbol{s}_t\right)\right)$ is the encoded augmented visual observation, $\boldsymbol{a}_t = \pi_\phi\left(\boldsymbol{h}_t\right) + \epsilon$ is the action with clipped noise to smooth the targets $\epsilon \sim \operatorname{clip}\left(\mathcal{N}\left(0, \sigma^2\right), -c, c\right)$. Note that we also do not update encoder weights with the policy gradient. We also train an ensemble of two fully connected Q-networks, which both use features from a single encoder, and take their minimum when calculating target values and actor losses. The resultant algorithm can be viewed as TD3 [15], but with decaying smoothing noise parameters c.

In DrQ+BC, this loss becomes:

$$\mathcal{L}_\phi(\mathcal{D}) = -\mathbb{E}_{\boldsymbol{s}_t, \boldsymbol{a}_t \sim \mathcal{D}}\left[\lambda Q_\theta\left(\boldsymbol{h}_t, \boldsymbol{a}_t\right) - \left(\pi_\phi\left(\boldsymbol{h}_t\right) - \boldsymbol{a}_t\right)^2\right]$$

where $\lambda = \frac{\alpha}{\frac{1}{N}\sum_{(h_i, a_i)}\left|Q(h_i, a_i)\right|}$ is an adaptive normalization term computed over minibatches. α is a behavioral cloning weight, always set to 2.5 in Fujimoto and Gu [14], which we also adopt. We experimented performing an extensive grid search over α, but did not observe any noticeable benefit in our offline datasets by deviating away from the default value.

C.3 CQL

Our CQL implementation was also built on top of the DrQv2 codebase for comparability. We use the CQL($\mathcal{H}$) objective with fixed weight, as we found this to be the most performant. This corresponds to choosing the KL-divergence to a uniform prior as the regularizer $\mathcal{R}(\mu)$ in Kumar et al. [32]. Concretely, the Q-function objective becomes

$$\min_Q \alpha_{\text{CQL}} \mathbb{E}_{\mathbf{s} \sim \mathcal{D}}\left[\log \sum_{\mathbf{a}} \exp(Q(\mathbf{s}, \mathbf{a})) - \mathbb{E}_{\mathbf{a} \sim \hat{\pi}_\beta(\mathbf{a} \mid \mathbf{s})}[Q(\mathbf{s}, \mathbf{a})]\right] + \frac{1}{2}\mathbb{E}_{\mathbf{s}, \mathbf{a}, \mathbf{s}' \sim \mathcal{D}}\left[\left(Q - \hat{\mathcal{B}}^{\pi_k} \hat{Q}^k\right)^2\right]$$

where α_{CQL} is a trade-off factor, $\hat{\pi}_\beta$ refers to the empirical behavioral policy and $\hat{\mathcal{B}}^{\pi_k}$ to the empirical Bellman operator that backs up a single sample. This is approximated by taking gradient steps and sampling actions from the given bounds.

D Hyperparameter and Experiment Setup

D.1 Offline DV2

Our Offline DV2 implementation was built on top of the official DreamerV2 repository at: `https://github.com/danijar/dreamerv2` with minor modifications. The code was released under the MIT License. Table 7 lists the hyperparameters used for Offline DV2. For other hyperparameter values, we used the default values in the DreamerV2 repository.

We found a default value of $\lambda = 10$ works for most settings. The only settings where this changes are $\lambda = 3$ for both random datasets and $\lambda = 8$ for the walker-walk mixed dataset.

Table 7: Offline DV2 hyperparameters.

Parameter	Value(s)
ensemble member count (K)	7
imagination horizon (H)	5
batch size	64
sequence length (L)	50
action repeat	2
observation size	[64, 64]
discount (γ)	0.99
optimizer	Adam
learning rate	{model = 3×10^{-4}, actor-critic = 8×10^{-5}}
model training epochs	800
agent training epochs	2,400
uncertainty penalty	mean disagreement
uncertainty weight (λ)	in [3, 10]

For the penalty choice, we chose mean disagreement of the ensemble because it comprises one half of the ensemble variance, which was shown to be an optimal choice for offline model-based reinforcement learning in Lu et al. [40]. We found that the other component of the ensemble variance, the average variance over the ensemble, was uninformative and so discarded it.

D.2 DrQ+BC

Our DrQ+BC implementation was built on top of the official DrQ-v2 repository at: `https://github.com/facebookresearch/drqv2`. The code was released under the MIT License. Table 8 lists the hyperparameters used for DrQ+BC. For other hyperparameter values, we used the default values in the DrQ-v2 repository. Due to the size of some of our offline datasets, we found the default replay buffer would not scale to the offline datasets. Thus, we used a NumPy [22] array implementation instead.

Table 8: DrQ+BC hyperparameters.

Parameter	Value
batch size	256
action repeat	2
observation size	[84, 84]
discount (γ)	0.99
optimizer	Adam
learning rate	1×10^{-4}
agent training epochs	256
n-step returns.	3
Exploration stddev. clip	0.3
Exploration stddev. schedule.	linear(1.0, 0.1, 500000)
BC Weight (α)	2.5

We tuned α within {1.5, 2.5, 3.5} but as Fujimoto and Gu [14] found, we did not observe any noticeable benefit from deviating away from the default value for α.

D.3 Behavioral Cloning

Our BC implementation shares the exact same policy network and hyperparameters in DrQ+BC but just minimizes MSE on the offline data. Consequently, we must also optimize the learned encoder using the supervised learning loss (unlike in DrQ+BC, where the TD-loss only contributes to the encoder representation learning, and not the policy loss).

D.4 CQL

Similarly to BC, our CQL implementation is also based on the same networks and hyperparameters in DrQ+BC. CQL introduces one extra hyperparameter, the trade-off factor α_{CQL}. We perform a hyperparameter sweep for this over the range: $\{0.5, 1, 2, 5, 10, 20\}$. We chose the following values per environment:

Table 9: CQL trade-off factor per environment for Walker and Cheetah. Humanoid omitted as all choices performed equally.

Dataset		**Trade-off Factor (α_{CQL})**
walker	random	0.5
	mixed	0.5
	medium	2
	medexp	2
	expert	5
cheetah	random	0.5
	mixed	0.5
	medium	10
	medexp	1
	expert	20

D.5 LOMPO

For our LOMPO evaluation, we used the official repository at: `https://github.com/rmrafailov/LOMPO`. The code was open-sourced without license. We perform a hyperparameter search over the uncertainty weight λ in the range $\{1, 5\}$. The default value used in Rafailov et al. [47] is $\lambda = 5$, but we found that $\lambda = 1$ worked better for random datasets. The accompanying data for the DMC Walker-Walk data from Rafailov et al. [47] was very kindly provided by the authors.

D.6 Computational Cost

The experiments in this paper were run on NVIDIA V100 GPUs. On the standard V-D4RL 100K datasets, DrQ+BC took 1.6 hours, Offline DV2 took 10 hours, and CQL took 12 hours.

Since we wish to compare several offline algorithms using the same dataset, we define a notion of "offline epoch" for all algorithms to show training performance over time. We simply normalize the total number of gradient steps, so training progress falls within $[0, 1000]$.

E Further Tabular Results and Training Curves

E.1 Training Curves for the Humanoid Environment

We include additional training curves for the Humanoid environment, as in Figure 2 for all our evaluated algorithms in Figure 6. As we noted previously, only the supervised BC and by extension DrQ+BC achieve meaningful return on this environment.

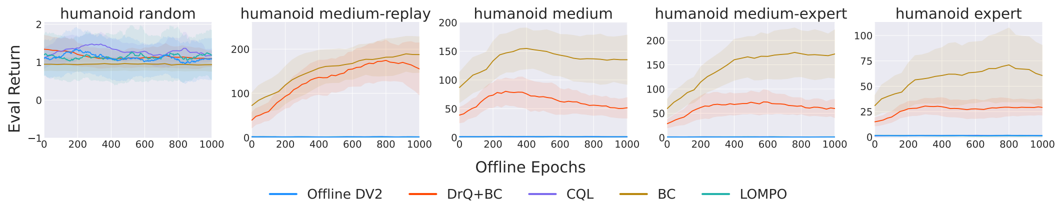


Figure 6: Rigorous comparison on the humanoid datasets from the V-D4RL benchmark, each setting is averaged over 6 seeds with error bar showing one standard deviation. Total gradient steps are normalized under epochs, and we plot the un-normalized evaluated return.

E.2 Full Tabular Results for Experiments with Distractions

We give the full un-normalized results for Section 4.1 in Table 10 and Table 11. The highlighted base results are the same as in Table 1 for Offline DV2 and Table 4 for DrQ+BC. The Offline DV2 results use 100K datapoints, and the DrQ+BC results use 1 million, in light of the discussion in Section 4.3.

Table 10: Offline DV2 shows surprisingly good generalization to unseen distractions and can handle multitask datasets. Final mean performance is averaged over 6 seeds and base undistracted performance is highlighted. The environment used is walker-walk random.

Shift Severity	**% Shifted**	**Eval. Return**		
		Original	**Dis. Train**	**Dis. Test**
low	0%	28.7	25.9	20.0
	25%	28.1	22.0	14.0
	50%	22.5	16.0	17.7
	75%	16.3	21.9	20.1
	100%	29.4	29.7	18.6
moderate	0%	28.7	21.0	15.1
	25%	28.6	19.1	14.9
	50%	19.8	20.5	11.2
	75%	24.4	20.0	11.9
	100%	20.1	22.9	15.5
high	0%	28.7	14.1	10.7
	25%	24.6	10.1	6.4
	50%	10.7	19.1	6.0
	75%	20.3	18.9	5.54
	100%	3.2	25.4	5.2

Table 11: DrQ+BC can adapt to multiple different distractions but is extremely brittle to settings it has not seen and struggles to generalize. Final mean performance is averaged over 6 seeds and base undistracted performance is highlighted. The environment used is cheetah-run medexp.

Shift Severity	**% Shifted**	**Eval. Return**		
		Original	**Dis. Train**	**Dis. Test**
low	0%	79.1	0.5	1.1
	25%	73.9	77.5	8.7
	50%	72.3	82.2	8.5
	75%	67.6	85.5	7.6
	100%	4.2	86.6	4.5
moderate	0%	79.1	0.2	0.9
	25%	74.6	77.9	3.1
	50%	68.0	83.6	3.4
	75%	55.4	86.1	3.7
	100%	0.9	86.7	2.0
high	0%	79.1	0.1	0.5
	25%	76.7	76.8	1.3
	50%	66.7	82.5	1.1
	75%	48.4	84.8	1.2
	100%	0.4	86.1	0.9

E.3 Further Offline DV2 Results on Multitask Datasets

On the random multitask data, Offline DV2 learns a similar quality policy to that on the base environment, but experiences no deterioration in performance on the test environments in walker or cheetah.

Table 12: Evaluation on the DMControl-Multitask benchmark using *random* data for Offline DV2. Normalized performance from [0, 1000] to [0, 100] is averaged over 6 seeds. Offline DV2 shows a strong ability to generalize to the extrapolation test environments.

Algorithm	**Environment**	**Eval. Return**		
		Train Tasks	**Test Interp.**	**Test Extrap.**
Offline DV2	walker	24.4	25.3	24.9
	cheetah	31.6	31.1	31.1

F Further Ablation Studies

F.1 DrQ+BC Random-Expert Ablation Studies

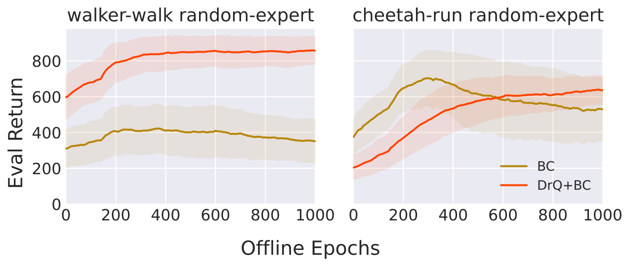


Figure 7: Comparison of DrQ+BC and BC on random-expert datasets with 500K of each datatype, results averaged over 6 seeds.

Analogously to Fujimoto and Gu [14], we continue to see that DrQ+BC does well on mixed datasets with high reward with experiments on the concatenated random-expert datasets. Surprisingly, we see behavioral cloning also does reasonably well on cheetah-run with random-expert data. This is likely to be because there is significant distribution shift between the states of random and expert trajectories for that environment. This would lead to minimal destructive interference between similar states which contain completely different actions, as these state distributions largely do not overlap. This is in contrast to the medium-expert dataset, which experiences higher state overlap between its two modes (i.e., states generated by a medium and an expert policy respectively), resulting in a marginal “average" action being learned which is likely suboptimal, as can be seen by the poor performance of the BC agent in Tables 1 and 4.

F.2 Offline DV2 Training Time and Uncertainty Penalty

One of the major factors preventing algorithms which use a RSSM like Offline DV2 and LOMPO from scaling to larger datasets is the time required for model training. The standard number of epochs of model training for our 100K datasets in Section 3 is 800 epochs, which takes around 6 hours on a V100 GPU. This scales linearly with number of training points if we maintain the same batch size. As we show in Figure 8, this is mandatory for performance and is a fundamental limitation of model-based methods. We can see that evaluated return increases and becomes more tightly distributed as the number of training epochs increases until 800.

In Table 1, we see that Offline DV2 is considerably weaker on the expert datasets that have narrow data distributions. On these expert datasets, we found that the model uncertainty often had lower variance across a trajectory compared to models trained on other datasets, and was thus uninformative. We give summary statistics for the uncertainty penalty on states generated by a random policy in the walker-walk environment in Table 13. Since we followed the author’s DreamerV2 implementation [21],

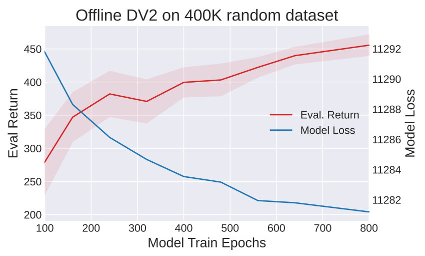


Figure 8: Evaluated return against model train epochs for Offline DV2 on a random dataset of size 400K. We train a single model up to 800 epochs and evaluate the model periodically on 3 seeds. We see that full model training, which scales linearly with training points, is mandatory for good performance.

only the stochastic portion of the latent state is predicted by the ensemble; this may mean crucial calibration is lost when ignoring the impact of the deterministic latent. Future work could involve investigating SVSG [23], an extension to DreamerV2 with a purely stochastic latent state.

Table 13: Mean and standard deviation of the uncertainty penalty computed over 1,024 states sampled from the 'random' dataset on the walker-walk environment. Note that the model trained on expert data reports considerably smaller and tighter uncertainty values (compared to 'medium' and 'medexp'), despite the large distribution shift that exists from 'random' to 'expert' data. Instead, we'd expect the 'expert' trained model to exhibit the largest mean uncertainty when tested on the 'random' data.

Dataset Type	**Mean**	**Std.**
random	0.223	0.040
mixed	0.226	0.035
medium	0.341	0.034
medexp	0.338	0.034
expert	0.262	0.020

F.3 Understanding Model-Based Extrapolation

We further investigate the reasons why our model-based baseline, Offline DV2, appears to generalize to unseen distractions. The RSSM includes a latent decoder for the visual observations, which is primarily used during training to provide a self-supervised reconstruction loss. During deployment, the policy solely relies on the latent states from the encoder, and the decoder is effectively discarded. However, we can still use the decoder at test-time in order to understand the information contained in the latent states. To do this, we take the latent states generated during rollouts in the extrapolation experiments, and 'translate' them into natural images by passing them through the decoder.

We first investigate the setting where each RSSM is trained only on data from a single distraction and then transferred. This setting is the most successful, as can be seen in Figure 3. Thus, in Figure 9 we first show the ground truth observation provided to the agent, then below this we show the decoder output reconstructed from the latent. In many cases, we can recover the original pose despite ending up in different visual surroundings. This is an indication that despite the decoder overfitting to the exogenous factors in the visual input (e.g., background, colors), the latent captures the salient *state* information of the agent (e.g., joint positions and angles), explaining the strong test-time transfer performance.

In Figure 3, we note that settings with a mixture of distractors often had worse test-time transfer performance than those with a single distractor. To explain this, we consider an RSSM trained on images with a combination of two fixed distractors, and examine the reconstruction of an episode under a third distraction. We see in Figure 10, that the RSSM latents are split between the two modes of the data and the reconstruction switches robot color and background midway. This significantly confuses the recovered pose of the walker and likely causes the degradation in performance. Disentangling the factors of variation [7, 42] represents important future work for offline RL from visual observations.

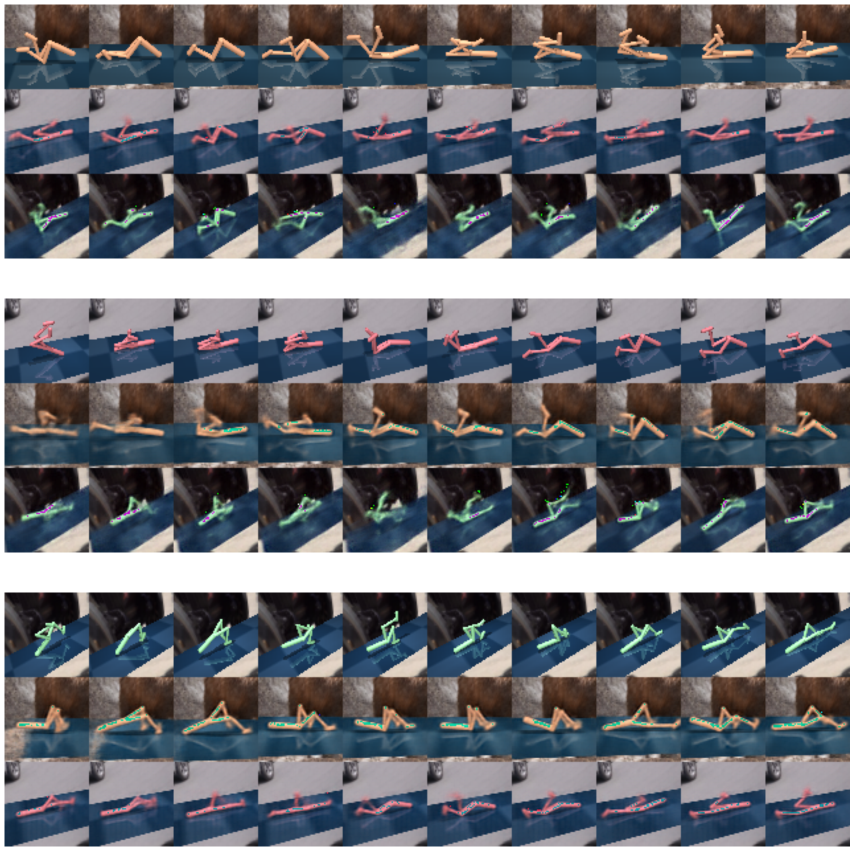


Figure 9: Reconstruction of random episodes shifted by a fixed distractor by models trained on data shifted by a different distractor. The top row shows the original ground truth, and the bottom two rows the model reconstruction for a different RSSM. We can see that in many cases, the original pose of the walker is still able to be recovered.

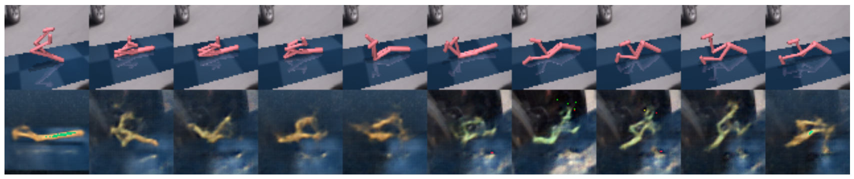


Figure 10: Reconstruction of a distracted random episode using an RSSM trained on a mixture of two differently distracted environments. We can see that the RSSM latents are split between the two modes of the data and the reconstruction switches robot color and background midway.