
XQCfD: Accelerating Fast Actor-Critic Algorithms with Prior Data and Prior Policies

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 For reinforcement learning in the real world, online exploration is expensive. A
2 common practice in robotic reinforcement learning is to incorporate additional
3 data to improve sample efficiency. Expert demonstration data is often crucial for
4 solving hard exploration tasks with sparse rewards. While prior data is used to
5 augment experience and pre-train models, we show that the design of existing
6 algorithms fails to achieve the sample efficiency that is possible in this setting
7 due to a failure to use pretrained policies effectively. We propose XQCfD, which
8 extends the sample-efficient XQC actor-critic to learn from demonstrations, using
9 augmented replay buffers, pre-trained policies and stationary policy architectures,
10 designed to avoid rapidly ‘unlearning’ the strong initial policy like prior works.
11 We show our stationary network architecture enables policy improvement out-of-
12 distribution better than standard network architectures due to its higher entropy
13 predictions. XQCfD achieves state of the art performance across a range of complex
14 manipulation tasks with sparse rewards from the popular Adroit, Robomimic and
15 MimicGen benchmarks — notably, with a low update-to-data ratio and no ensemble
16 networks.

17 1 Introduction

18 Reinforcement learning (RL) provides a general framework for sequential decision-making, but
19 high sample complexity remains the central barrier to real-world deployment [39]. Model-free deep
20 actor-critic methods have made remarkable strides in sample efficiency through careful algorithmic
21 and architectural choices: batch normalization, weight normalization, categorical critics, high update-
22 to-data ratios, culminating in methods such as CROSSQ [4] and XQC [25, 24]. In many practical
23 settings, prior data is readily available before online interaction begins: expert demonstrations,
24 previously collected offline datasets, or trajectories from related tasks. Leveraging such data promises
25 to accelerate online learning significantly. Several research threads have combined RL with this
26 additional data: *offline-to-online RL* pre-trains a policy offline and finetunes it online; *RL from*
27 *demonstrations* (RLfD, [41, 30]) incorporates expert data into the agent’s experience; *RL from prior*
28 *data*, e.g., RLPD [2], mixes sub-optimal offline and online data in the agent’s replay buffer. *Imitation-*
29 *bootstrapping* [12] interleaves on-policy and behavioral cloning actions during data collection to
30 incorporate prior knowledge.

31 A recurring limitation in RLfD is failing to properly leverage the performance of *behavioral*
32 *cloning* (BC [28]) at initialization [2, 12]. Many current RLfD algorithms require millions of
33 interactions to match BC performance, even though they eventually surpass it. This phenomenon
34 occurs because the pre-trained policy is rapidly unlearned due to the learning dynamics of deep
35 actor-critic algorithms and their use of function approximation. The BC policy is not optimal for the
36 randomly-initialized critic, and therefore the BC policy is rapidly unlearned in favor of a policy that

37 is. This phenomenon has been observed in inverse reinforcement learning, where it is more severe
38 due to the additional learned reward [23, 42].

39 We propose $XQCfD$ (XQC from demonstrations), which extends the XQC actor-critic implementation
40 with a set of algorithmic changes to achieve state-of-the art sample efficiency for RLfD:

- 41 • We perform policy pretraining on the demonstration data to start learning at the BC-level of
42 performance, rather than learn from scratch.
- 43 • We also perform critic pretraining to ensure the initial actor and critic are coherent.
- 44 • To mitigate unlearning due to stochastic optimization, we use KL regularization between the
45 policy and BC policy as an additional regularizer, instead of a maximum entropy one.
- 46 • To bridge KL and maximum entropy regularization, we adopt a ‘stationary’ parametric poli-
47 cies which returns a constant, maximum entropy action distribution when out of distribution.
48 This property enables exploration and policy improvement in states outside of the expert
49 demonstration data, so the KL regularization does not limit policy improvement over BC.

50 While none of these design decisions are independently novel, this combination has not been used
51 for RLfD, and we show empirically across a range of sparse manipulation tasks that this approach
52 prevents BC unlearning and therefore enables dramatic sample efficiency compared to baselines
53 when combined with XQC . In summary, our contributions are

- 54 1. We combine the actor-critic algorithm XQC with prior data and pre-trained policies for a
55 sample-efficient RLfD algorithm, highlighting the crucial implementation details.
- 56 2. We motivate and demonstrate the utility of stationary network architectures in RLfD.
- 57 3. We provide thorough ablation experiments to validate our design decisions.

58 2 Related Work

59 Reinforcement learning from demonstrations or prior data is a long-established idea, especially for
60 domains where exploration is challenging and expensive. Currently, using prior data comes in three
61 distinct settings, based on how the data is used and what is pre-trained.

62 **Finetuning behavioral cloning with reinforcement learning.** For challenging domains such as
63 real-world robotics and strategy games, a practical solution for sample efficiency is to finetune BC
64 policies [35, 26, 14, 16, 37]. More modern practices combine an on- or off-policy deep RL algorithm
65 with an additional BC-based loss or policy regularization in the actor objective to prevent unlearning
66 [30, 33]. We build on this work by incorporating a more sample-efficient off-policy algorithm that is
67 able to finetune a BC policy without a significant performance drop, which is observed when using
68 randomly initialized critics, e.g., the RLFT baseline in [12].

69 **Online reinforcement learning with additional data.** With the advent of replay buffers in deep
70 reinforcement learning, prior data can be easily integrated by merely augmenting the replay buffer.
71 Algorithm design decisions concern whether the replay buffer is simply augmented or multiplexed¹
72 [41, 2]. Imitation-bootstrapped RL [12] improves sample efficiency by incorporating BC action
73 during interactions, and identifying dropout in the policy as an effective implementation detail and
74 REDQ [6] as an effective RL algorithm in this setting. However, by not utilizing BC initialization
75 effectively, buffer-based methods forfeit potentially significant sample efficiency gains, which we
76 show in our experimental results.

77 **Offline-to-online reinforcement learning.** A related setting where the prior data is suboptimal
78 considers continuing offline RL with online interactions and learning [21]. This setting is challenging
79 due to the difficulty of performing continual learning with function approximation with severe
80 performance degradation, characterized as ‘catastrophic forgetting’ or a lack of network ‘plasticity’.
81 Solutions include BC regularization [43], careful sampling of offline and online samples from the
82 replay buffer [15] and calibrating the Q values during offline learning to mitigate policy unlearning
83 during online interactions [22]. Our RLfD setting corresponds to offline-to-online RL with expert
84 offline data, and we do not consider using suboptimal data as we desire strong initial BC policies.

¹Multiplexing refers to combining samples from several replay buffers within a single minibatch.

85 **Kullback-Leibler (KL)-regularized RL.** While XQC builds on SAC (soft actor-critic, [10]),
 86 XQcFD uses a KL-regularized realization rather than a maximum entropy one, where the BC policy is
 87 a prior. Siegel et al. [36] use the same regularization for offline RL, using MPO [1] as the actor-critic
 88 algorithm. Rudner et al. [33] use this regularization for RLfD with SAC, but use a nonparametric
 89 Gaussian process as the BC policy and a standard network architecture for the RL policy and min-
 90 imizes their KL during pre-training. Watson et al. [42] use this formulation with SAC, but in an
 91 inverse RL setting.

92 3 Sample-Efficient Actor-Critic with Prior Data and Prior Policies

93 We consider the standard reinforcement learning setting, using a Markov decision process (MDP),
 94 defined by the tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$ where \mathcal{S} is the state space, \mathcal{A} is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow$
 95 $[0, 1]$ are the transition dynamics, $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, and $\gamma \in [0, 1]$ is the discount
 96 factor. At each timestep h , an agent observes state $\mathbf{s}_h \in \mathcal{S}$, selects action $\mathbf{a}_h \sim \pi(\cdot | \mathbf{s}_h)$ according
 97 to a stochastic policy $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$, transitions to $\mathbf{s}_{h+1} \sim P(\cdot | \mathbf{s}_h, \mathbf{a}_h)$, and receives reward
 98 $r_h = r(\mathbf{s}_h, \mathbf{a}_h)$. The objective is to find a policy π^* that maximizes the expected discounted return
 99 $\mathcal{J}(\pi) = \mathbb{E}[\sum_{h=0}^{\infty} \gamma^h r_h]$. A central quantity in reinforcement learning is the state-action value
 100 function $Q^\pi(\mathbf{s}, \mathbf{a}) = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k r_{h+k} | \mathbf{s}_h = \mathbf{s}, \mathbf{a}_h = \mathbf{a}]$, which satisfies the Bellman equation
 101 $Q^\pi(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}' \sim P(\cdot | \mathbf{s}, \mathbf{a}), \mathbf{a}' \sim \pi(\cdot | \mathbf{s}')} [Q^\pi(\mathbf{s}', \mathbf{a}')]$. Actor-critic algorithms maintain two
 102 function approximators: an actor π_ϕ parameterized by ϕ that represents the policy, and a critic Q_θ
 103 parameterized by θ that estimates the value function. Deep actor-critic algorithms maintain a replay
 104 buffer \mathcal{B} of past interactions. When learning from demonstrations, we also have access to a dataset of
 105 demonstrations \mathcal{D} of state-action trajectories capturing expert behavior.

106 Given a sample-efficient RL algorithm, incorporating prior expert data is a way of alleviating the
 107 pressure to explore and therefore achieve greater sample efficiency. While prior RLfD works adopt
 108 effective RL algorithms with Q ensembles, layer norm, and high update-to-data ratios, the recently
 109 proposed XQC [25] has matched or outperformed these architectures w.r.t. parameter count, sample
 110 efficiency and wall-clock speed by combining batch norm [13], weight norm [17] and a distributional
 111 critic [3]. This combination ensures the critic’s ‘effective’ learning rate [40] does not decrease and
 112 the critic’s loss landscape is smooth and well-conditioned [34, 9], which enables sample-efficient
 113 learning without any great computational overhead [25]. To leverage prior data, we multiplex the
 114 replay buffers during online learning and pre-train the policy using BC. The open research question
 115 concerns the tension between preventing unlearning and facilitating finetuning. Can we finetune a
 116 parametric policy without unlearning the strong initialization? To tackle this challenge, we use a
 117 policy architecture that has maximum entropy action predictions, as we adopt the entropy-regularized
 118 RL formulation, which regularizes the policy against a prior distribution using the KL divergence.
 119 While online RL uses a uniform action prior for exploration² with temperature $\alpha \geq 0$, in the RLfD
 120 setting we replace this uninformative prior with the pre-trained BC policy $\pi_{\text{BC}}(\mathbf{a} | \mathbf{s})$, which also
 121 initializes $\pi_\phi(\mathbf{a} | \mathbf{s})$, resulting in actor loss

$$\phi_\star = \arg \max_\phi \mathcal{L}_\pi(\phi) = \mathbb{E}_{\mathbf{s} \sim \mathcal{B}} [\mathbb{E}_{\mathbf{a} \sim \pi_\phi(\cdot | \mathbf{s})} [Q_\theta(\mathbf{s}, \mathbf{a})] - \alpha \mathbb{D}_{\text{KL}}[\pi_\phi(\mathbf{a} | \mathbf{s}) || \pi_{\text{BC}}(\mathbf{a} | \mathbf{s})]]. \quad (1)$$

122 Adopting the KL divergence provides strong policy improvement guarantees. The solution to Equation
 123 1 has monotonic improvement guarantees over the BC policy (Lemma 1).

124 **Lemma 1.** *Monotonic policy improvement [32]. Given a prior policy $p(\mathbf{a} | \mathbf{s})$, the pseudo-posterior*
 125 *policy update $\pi(\mathbf{a} | \mathbf{s})$ improves on the prior in a monotonic fashion for critic Q due to the change-of-*
 126 *measure inequality [8],*

$$\mathbb{E}_{\mathbf{a} \sim \pi(\cdot | \mathbf{s})} [Q(\mathbf{s}, \mathbf{a})] \geq \mathbb{E}_{\mathbf{a} \sim p(\cdot | \mathbf{s})} [Q(\mathbf{s}, \mathbf{a})] \quad \forall \mathbf{s} \in \mathcal{S}, \quad \pi(\mathbf{a} | \mathbf{s}) \propto \exp\left(\frac{1}{\alpha} Q(\mathbf{s}, \mathbf{a})\right) p(\mathbf{a} | \mathbf{s}). \quad (2)$$

127 However, in moving our objective from maximum entropy regularization with a uniform prior to
 128 KL regularization with the BC prior, we may have compromised our deep reinforcement learning
 129 algorithm. In states outside of the demonstration distribution where we wish to perform online
 130 RL, $\pi_{\text{BC}}(\mathbf{a} | \mathbf{s})$ may be very different from the uniform prior $\mathcal{U}(\mathbf{a})$, especially as the predictive
 131 behavior of neural networks outside of the data distribution (OOD) is undefined, as illustrated in
 132 Figure 1. Therefore, during RL finetuning in out-of-distribution states, Equation 1 may encourage the

²A maximum entropy objective is equivalent to a KL minimization objective against a uniform distribution.

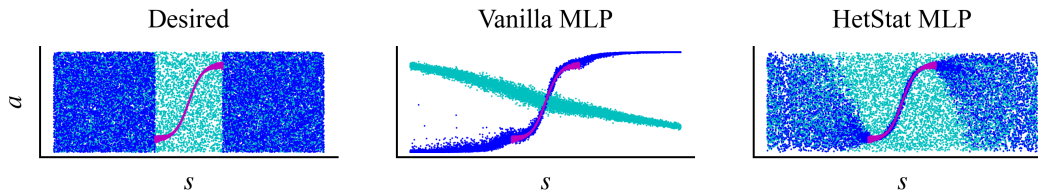


Figure 1: (Left) In the entropy-regularized RLfD setting, we desire a BC policy that fits the data (●) while reflecting a maximum entropy prior out of distribution. (Centre) The typical multi-layer perceptron (MLP) architecture used in deep actor-critic methods does not have this property. The out-of-distribution predictive behavior is undefined. Due to this behavior, KL regularization encourages the policy to stay close to suboptimal actions. (Right) By incorporating stationarity into the policy architecture, the policy is maximum entropy at initialization (●) and maximum entropy out-of-distribution after training (●). In practice, there is a smooth transition between modeling the data and maximum entropy prior due to the smoothness of the function approximation.

133 agent to perform suboptimal actions rather than random exploration, and reducing α as a result also
 134 encourages unlearning the expert behavior. To fix this issue, we desire a policy network that predicts
 135 a uniform action distribution when out of distribution. Fortunately, this is possible by using policies
 136 with *stationary* feature spaces.

137 3.1 Maximum Entropy Policies using Stationary Features

138 To finetune the BC policy in out-of-distribution states with maximum entropy RL using Equation 1,
 139 we require a BC policy with the following characteristic,

$$\pi_{\text{BC}}(\mathbf{a} | \mathbf{s}) = \begin{cases} p_{\mathcal{D}}(\mathbf{a} | \mathbf{s}) & \text{if } \mathbf{s} \in \mathcal{D}, \text{ where } p_{\mathcal{D}} \text{ is a conditional generative model of } \mathcal{D}, \\ \mathcal{U}(\mathbf{a}), & \text{the uniform prior, otherwise.} \end{cases} \quad (3)$$

140 A consequence of this characteristic is that, in the absence of any demonstration data, we desire a
 141 policy architecture where $\pi_{\text{BC}}(\mathbf{a} | \mathbf{s}) = \mathcal{U}(\mathbf{a}) \forall \mathbf{s} \in \mathcal{S}$. Following soft actor-critic [10], we use a
 142 Gaussian policy with a tanh transformation to bound the support of the action distribution to match
 143 the uniform prior. Therefore, this property is approximated by a latent action $\mathbf{a} = \tanh(\mathbf{z})$ with fixed
 144 zero-mean Gaussian predictions. The desired out-of-distribution behavior described above can be
 145 achieved with a latent Gaussian model with stationary features φ . Stationary features have constant
 146 self-similarity: the inner product $\varphi_{\theta}(s')^{\top} \varphi_{\theta}(s)$ is a function of a distance metric between s' and
 147 s , which means that the variance is a constant when $s' = s$, $p(z | s) = \mathcal{N}(0, \sigma^2)$ as desired when
 148 $p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$ due to the predictive distribution of Gaussian models shown below. Our latent
 149 Gaussian policy is implemented using stationary last layer features, on top of MLP features, with
 150 Gaussian last layer weights \mathbf{w} ,

$$\mathbf{z} = \mathbf{w}^{\top} \varphi_{\theta}(s), \quad p(\mathbf{w}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad p(z | s) = \mathcal{N}(\boldsymbol{\mu}^{\top} \varphi_{\theta}(s), \varphi_{\theta}(s)^{\top} \boldsymbol{\Sigma} \varphi_{\theta}(s)).$$

151 This predictive behavior is preserved for out-of-distribution states after training, subject to the
 152 smoothness of the features. Unfortunately, this stationary property is only achieved exactly when
 153 $\varphi(s)$ is an infinitely large feature space and $\varphi_{\theta}(s)^{\top} \varphi_{\theta}(s')$ is expressed through its tractable kernel
 154 function $k(s, s')$ [31]. Fortunately, we can construct a *finite* approximate to these feature spaces
 155 using *random Fourier features* [29] \mathbf{f}_k for kernel $k(\cdot, \cdot)$,

$$\varphi_{\theta, \mathbf{V}}(s) = \mathbf{f}_k(\mathbf{V}s), \quad V_{ij} \sim \mathcal{N}(0, 1), \quad (4)$$

156 where \mathbf{f}_k are periodic functions required to approximate stationary kernel k and \mathbf{V} is a fixed random
 157 weight matrix that constructs the random features with \mathbf{f}_k . In practice, we apply the random
 158 features to neural network features rather than s for greater function approximation capability, i.e.,
 159 $\mathbf{f}_k(\mathbf{V} \varphi_{\text{NN}}(s))$, which preserves stationarity [20]. We refer to this architecture as HetStat, as it is
 160 heteroscedastic³ and stationary. HetStat policies were first proposed by Watson et al. [42] in an
 161 inverse RL setting, combining it with SAC and the same actor objective (Equation 1). While this
 162 policy architecture was required for their learned reward, they also benefited from regularization
 163 properties of stationary priors observed by Rudner et al. [33]. However, while Rudner et al. [33] used
 164 a (stationary) homoscedastic, nonparametric Gaussian process BC prior and an MLP policy, we and
 165 Watson et al. [42] adopt a stationary network policy for both to benefit from parameter initialization
 166 and avoid the scaling issues that arise with nonparametric GPs [31].

³Heteroscedastic predictions have input-varying variance, whereas homoscedastic predictions have constant variance.

Algorithm 1: XQC from Demonstrations (XQCFD).

Data: Expert demonstrations \mathcal{D} , temperature α , parametric policy class $\pi_\phi(\mathbf{a} \mid \mathbf{s})$, parametric critic class $Q_\theta(\mathbf{s}, a)$, total steps T , update-to-data ratio N , policy delay D
Result: $\pi_{\phi_T}(\mathbf{a} \mid \mathbf{s})$, matching or improving the initial policy $\pi_{\phi_1}(\mathbf{a} \mid \mathbf{s})$ // pre-train policy and critic

Train initial policy from demonstrations using Equation 5, $\phi_1 = \arg \min_\phi \mathcal{L}_{\text{BC}}(\phi)$.
Pre-train critic on \mathcal{D} using ϕ_1 and Equation 6.

```
for  $h = 1 \rightarrow T$  do // finetune policy with reinforcement learning
  Interact with environment  $s_{h+1} \sim P(\cdot \mid s_h, \mathbf{a}_h)$ ,  $\mathbf{a}_h \sim \pi_{\phi_h}(\cdot \mid s_h)$ , store in buffer  $\mathcal{B}$ ;
  for  $n = 1 \rightarrow N$  do
    Update critic using  $\mathcal{L}_Q(\theta)$  (Equation 6) on minibatch from  $\mathcal{B}$  and  $\mathcal{D}$ ;
    if  $i \% D = 0$  then
      Update policy optimizing  $\mathcal{L}_\pi(\phi)$  (Equation 1) on minibatch from  $\mathcal{B}$  and  $\mathcal{D}$ ;
    end
  end
end
end
```

167 3.2 Off-policy reinforcement learning from demonstrations

168 We now outline the design of XQC and our extensions for XQCFD.

169 **Actor-critic algorithm.** XQC builds on SAC and adopts three design decisions: batch norm, weight
170 norm, and a categorical critic. Following [4], during critic learning we apply batch norm to the states
171 and next states combined, i.e., computing $\mathbf{q}, \mathbf{q}' = Q_\theta([\mathbf{s}, \mathbf{s}'], [\mathbf{a}, \mathbf{a}'])$ to ensure the batch statistics
172 correctly capture the state-action distribution seen by the critic. In practice a target network is used to
173 predict \mathbf{q}' . Secondly, we apply weight norm to the critic network to constrain them to the unit sphere,
174 i.e., $\tilde{\mathbf{W}} = \mathbf{W} / \|\mathbf{W}\|_2$. This normalization keeps the effective learning rate constant [17], which
175 describes the phenomena where learning slows down as the weight norms of the parameters grow
176 due to gradients scaling inversely proportional to the weights when batch norm is used [24]. The
177 categorical critic has 101 atoms and a cross entropy loss following the C51 architecture (Appendix
178 B, Bellemare et al. [3]). For the actor, applying BN and WN is optional, but often beneficial and
179 therefore used in our experiments.

180 For XQCFD, we make three algorithmic changes: initializing the policy via pre-training, replacing
181 the entropy term in the actor loss with KL regularization (Equation 1), and maintaining both the
182 demonstration and replay buffer separately. The algorithm is summarized in Algorithm 1.

183 **Model pre-training.** For BC pre-training, we use a loss function that combines mean-squared error
184 (MSE) and negative log likelihood (NLLH) metrics. As our stochastic policy predicts both mean
185 and variance, it is a heteroscedastic regression model. In deep learning, these models can suffer
186 from ambiguity between signal and noise, resulting in regression fits that under-fits in the mean and
187 overestimates the variance rather than accurately modeling complex functions. The *faithful* loss [38]
188 resolves this issue by training the mean prediction $\boldsymbol{\mu}(\mathbf{s}) = \mathbf{f}_\mu(\varphi(\mathbf{s}))$ with the MSE objective, while
189 training only the variance head with the NLLH. Crucially, gradients are cut between the variance
190 head and the shared feature space $\varphi(\mathbf{s})$, so that $\boldsymbol{\Sigma}(\mathbf{s}) = \mathbf{f}_\Sigma(\text{sg}(\varphi(\mathbf{s})))$, where $\text{sg}(\cdot)$ denotes the
191 stop-gradient operator, to ensure the features are optimized for the mean fit rather than uncertainty
192 quantification. The resulting loss is

$$\mathcal{L}_{\text{BC}}(\phi) = \mathbb{E}_{\mathbf{a}, \mathbf{s} \sim \mathcal{D}} [\|\mathbf{a} - \mathbf{l}(\boldsymbol{\mu}_\phi(\mathbf{s}))\|_2^2 - \log p(\mathbf{z} \mid \boldsymbol{\mu}(\mathbf{s}), \boldsymbol{\Sigma}(\mathbf{s})^2) - \log |\nabla_{\mathbf{z}} \mathbf{l}(\mathbf{z})|], \quad (5)$$

193 where $\mathbf{l}(\cdot)$ is the action transform, e.g., \tanh , of the latent Gaussian action $\mathbf{z} \sim p(\cdot \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$. We also
194 pre-train the critic on the demonstration data using the same critic objective used for RL.

195 **Policy regularization.** For the KL divergence between policies in Equation 1, we use the closed-form
196 KL between latent Gaussian actions, as the KL divergence between the tanh-squashed Gaussian
197 has no closed-form expression and would require a Monte Carlo KL approximation. Since the KL
198 divergence is invariant under shared bijections [27], the closed-form Gaussian KL is in fact equal to
199 our desired quantity. For the KL scaling term α , we find a constant value is sufficient to regularize the
200 update [42] and did not jointly optimize the temperature like Haarnoja et al. [10]. The BC parameters
201 initialize the RL policy, so the KL starts at zero.

202 **Experience replay.** When combining data buffers, we combine the minibatches equally throughout
 203 training, coined as ‘symmetric sampling’ in Ball et al. [2]. We also use the expert demonstration data
 204 to pre-compute the reward statistics for normalization and avoid online estimators.

205 4 Experiments

206 We now evaluate the empirical performance of XQCfD across a diverse set of sparse-reward manipula-
 207 tion tasks drawn from three established benchmarks, comparing XQCfD with HetStat and MLP policy
 208 architectures. This section provides an overview of the experimental setup, including environments,
 209 implementation details and evaluation protocol. Section 4.1 presents our main results, comparing
 210 against relevant baselines in terms of sample efficiency and asymptotic performance. We then ablate
 211 key design decisions to isolate their individual contributions (Section 4.3).

212 **Environments.** We evaluate XQCfD on a total of 11 sparse-reward manipulation tasks spanning
 213 3 popular benchmarks. From Adroit [30], we use four dexterous hand tasks: pen, door, hammer,
 214 relocate. From Robomimic [18], we consider three tasks: Lift, NutAssemblySquare,
 215 and PickPlaceCan. From MimicGen [19], we include four tasks: StackThree, Coffee,
 216 HammerCleanup, MugCleanup. These environments are characterized by high-dimensional
 217 action spaces, contact-rich dynamics, and binary sparse rewards, making learning particularly chal-
 218 lenging for agents relying solely on online exploration.

219 **Demonstration data.** For each task, we assume access to the standard dataset of expert demon-
 220 strations. For most environments this is 200 demonstrations per task, apart from StackThree_D0
 221 which provides 1000 demonstrations to variety of initial cube positions. This data regime is reason-
 222 able to be collected manually by experts and provides performant BC policies. We use the same
 223 expert datasets for BC pre-training, replay buffer augmentation and reward normalization statistics.

224 **Baselines.** We compare XQCfD against several baselines, including BC [28], RLPD [2], XQC [25],
 225 and SAC [10]. As a means of separating the contribution of the actor-critic algorithm from the RLfD,
 226 we consider three variants of XQC, reflecting several RLfD approaches. These are (i) using BC
 227 pre-training (XQC + BC), (ii) combining expert offline data with the online replay buffer (XQC + OD)
 228 like RLPD, and (iii) combining (i) and (ii) (XQC + BC + OD). For SAC, we use a HetStat policy.

229 **Hypotheses.** Our experimental design is built upon the following hypotheses.

- 230 1. XQCfD will improve upon BC performance without unlearning.
- 231 2. The sample efficiency of XQCfD will depend on the performance of the BC policy.
- 232 3. XQC will still outperform SAC and LayerNorm (RLPD) in the sparse RLfD setting.

233 If these three hypotheses are true, XQCfD will be an effective RLfD implementation, subject to the
 234 quality of the demonstration data.

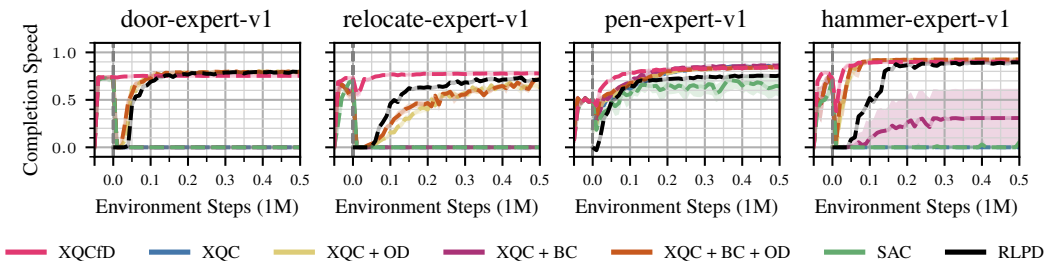


Figure 2: Performance of XQCfD and baselines on Adroit over 10 seeds, showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals. BC pretraining performance is shown before the vertical dashed line. Notably, in two of the tasks, BC achieves expert performance. The benefit of improving upon BC is clear in this scenario, and baselines then require 100K–1M interactions to recover BC-level performance after unlearning. In the pen task, where BC is only ~60% successful, XQCfD only achieves a speed-up of 50K interactions, which supports our Hypothesis 2.

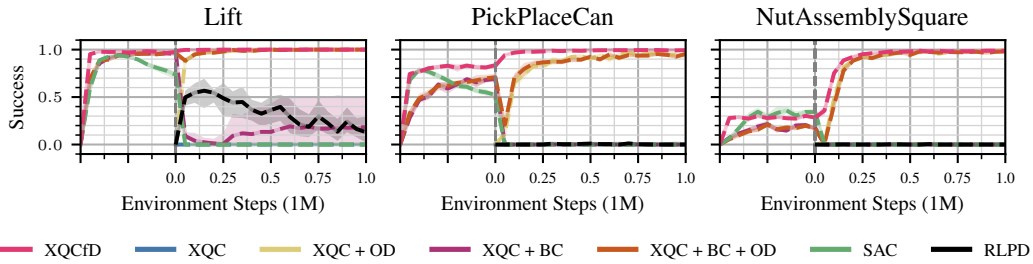


Figure 3: Performance results on Robomimic over 10 seeds showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals. BC pretraining performance is shown before the vertical dashed line. OD refers to adding expert offline data. Robomimic includes three sparse manipulation tasks that are challenging to solve without leveraging demonstrations. For `Lift` and `PickPlaceCan`, the benefit of XQcFD is clear, and BC performs close-to-expert performance and therefore XQcFD is dramatically more stable than baselines. However, we see in the `Lift` task (simply lifting a block) XQC + offline data learns extremely rapidly, and XQC + offline data + BC unlearns very little. However, for the harder `PickPlaceCan` task, unlearning is much more severe for our strong XQC-based baselines and XQcFD achieves a sample efficiency of 1M steps. For the harder `NutAssemblySquare` task, BC performs worse, therefore the performance gap between XQcFD and XQC + offline data is initially small and rapidly negligible, matching our Hypothesis 2.

235 4.1 Experimental results

236 In Figures 2, 3, 4, we benchmark XQcFD against baselines on Adroit, Robomimic and MimicGen
 237 respectively, looking at 11 tasks in total. Across all tasks, Hypothesis 1 holds, with little observable un-
 238 learning at the transition to BC. Improving on BC rather than starting from scratch yields notable gains
 239 in sample efficiency. In `PickPlaceCan` the strongest baselines require $\sim 1M$ additional samples to
 240 reach XQcFD performance and $\sim 500K$ in `relocate-expert-v1` and `HammerCleanup-D0`.
 241 Moreover, XQcFD is the only method to achieve non-zero success rate in the harder MimicGen tasks
 242 `StackThree-D0` and `Coffee-D0`. For tasks `NutAssemblySquare` and `MugCleanup-D0`,
 243 the BC success rate was only in the region of 20-30%, and there was a much smaller performance gap
 244 between XQcFD and XQC + OD. We find a lower temperature was better in these environments, as the
 245 agent benefited from a greater trust region to improve on the worse-performing BC policy. However,
 246 this result confirms our Hypothesis 2 that the value in XQcFD relies on initial BC performance. We
 247 found that this temperature parameter was also highly coupled to the unlearning phenomena, as
 248 `NutAssemblySquare` and `MugCleanup-D0` both exhibit a small drop in performance due to a
 249 lack of regularization. Figure 7 shows that a lower temperature can prevent premature convergence.

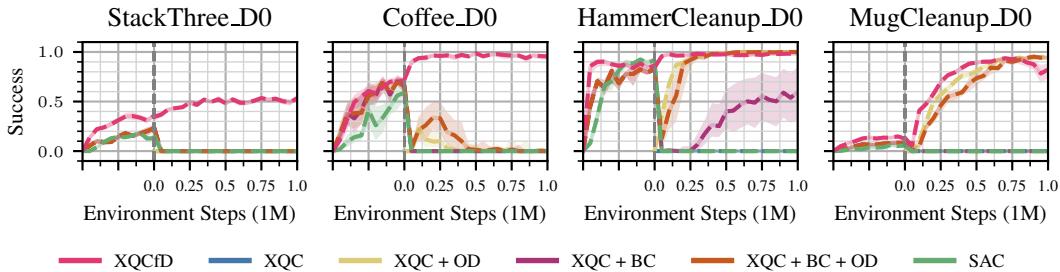


Figure 4: Performance results of XQcFD and baselines on MimicGen over 10 seeds, showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals. BC pretraining performance is shown before the vertical dashed line. OD refers to adding expert offline data. Compared to Robomimic, these tasks are slightly more complex due to a broader initial state distribution and longer task horizon. For this suite, XQcFD excels in tasks where the initial BC policy performs well. In the `Mug` task, where the BC policy achieves a low success rate, XQcFD is comparable to baselines without the benefit of pretraining. For the hardest task, `StackThree`, XQcFD is the only algorithm to achieve greater-than-zero success, but only improve the BC policy by 10%. This suggests there are still improvements to be made before harder tasks can be mastered.

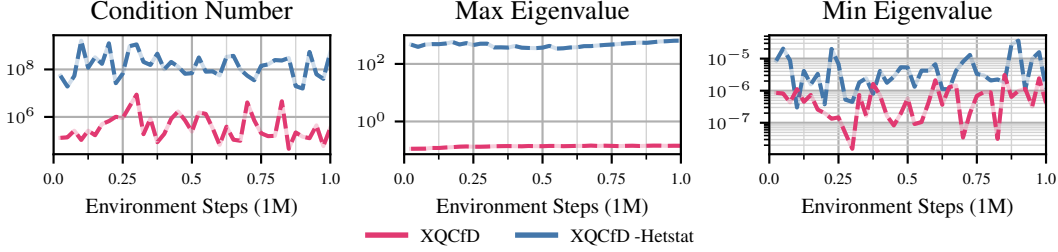


Figure 5: An empirical analysis of the optimization landscape of the actor with and without the HetStat policy for `PickPlaceCan`. The HetStat policy improves the condition number by around $100\times$, which we attribute to the inherent regularization from the stationarity of the HetStat policy [33]. The condition number analysis technique follows Behrooz et al. [9] and Palenicek et al. [25].

250 Regarding Hypothesis 3, Figure 2 shows that while SAC and RLPD can solve several tasks, similar
 251 XQC variants outperform them for all but one environment. This result reflects previous findings
 252 on the qualities of XQC [24, 25]. Moreover, Ball et al. [2] tune the hyperparameters of RLPD per
 253 environment. We were not able to tune RLPD to solve Robomimic, as observed by Biza et al. [5].

254 4.2 Actor conditioning analysis

255 Palenicek et al. [25] have shown that critic architectures with better conditioning can be linked to
 256 improved RL training performance. In Figure 5, we see that the HetStat policy improves the condition
 257 number of the actor loss by around $100\times$, supporting the insight that stationary policies provide a
 258 better loss landscape for learning due to their consistent predictions when out of distribution [33].

259 4.3 Ablation studies and hyperparameter sensitivity

260 In Figure 6, we perform an ablation study on the HetStat policy and KL regularization. For 6 out of
 261 11 environments, the HetStat policy has clear benefits over the standard MLP for stable finetuning.

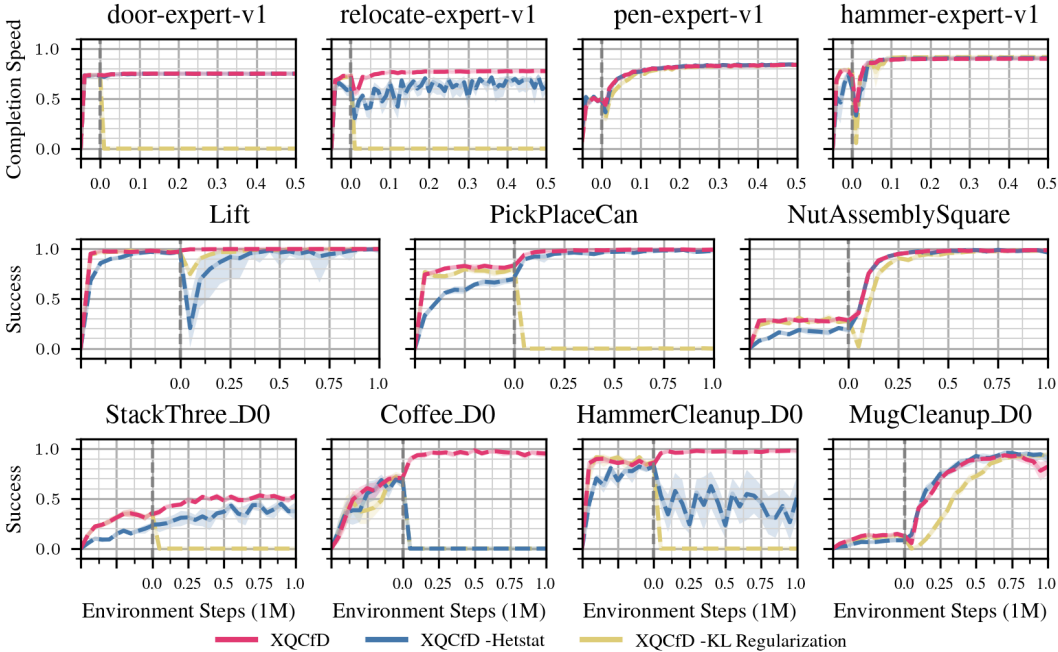


Figure 6: Ablation studies on XQcFD, replacing the HetStat MLP with a standard MLP and replacing KL regularization with standard entropy regularization for the Adroit (top), Robomimic (middle) and MimicGen (bottom) environments over 10 seeds, showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals.

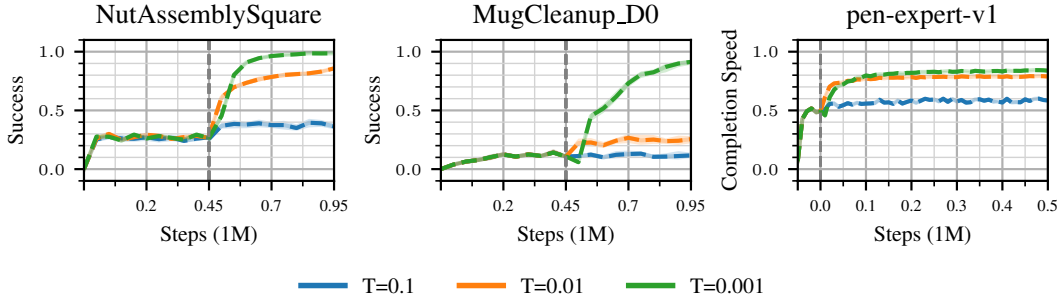


Figure 7: Sensitivity study of XQCFD’s temperature α that controls KL regularization against the BC policy. Evaluated over one task for Adroit, Robomimic and MimicGen over 10 seeds, showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals. Lower temperatures result in a larger performance drop on the transition from BC to RL due to unlearning, but lower temperatures also facilitate greater RL finetuning due to the smaller proximal regularization.

262 Regarding KL regularization, for 6 out of the 11 environments, removing KL regularization resulted
 263 in complete unlearning and failure to converge. For the 5 environments where KL regularization is
 264 in complete unlearning and failure to converge. For the 5 environments where KL regularization is
 265 not needed, these are environments where BC has a low success rate, so standard RL with auxiliary
 266 data is sufficient and unlearning does not need to be prevented. While HetStat policies and KL
 267 regularization are needed under the same conditions, only one environment (*Coffee*) required both.
 268 Our hypothesis is that the HetStat architecture is beneficial for stable finetuning, hence replacing it
 269 results in poor convergence (*relocate*, *Lift*, *PickPlaceCan*, *HammerCleanup*). Removing
 270 KL regularization appears to frequently trigger rapid unlearning, although it is unclear why the agent
 271 cannot recover with additional experience, as the baselines appear to do when learning from scratch.
 272 The baselines that do not have the additional data to aid exploration (SAC and XQC +BC) consistently
 fail after unlearning.

273 In Figure 7, we sweep across different values of the fixed temperature. Higher temperatures cor-
 274 respond to stronger BC regularization. When the BC policy has weak performance, online policy
 275 improvement struggles with stronger regularization, so lowering the temperature increases the policy
 276 improvement during online training.

277 Lastly, in Figures 8 to 10 in Appendix A, we provide a hyperparameter sensitivity experiment w.r.t.
 278 the number of expert demonstrations, which affects both the BC pretraining and auxiliary replay
 279 buffer. The trend we observe is that if fewer demonstrations result in a weaker BC policy, RL
 280 finetuning can be weaker, which suggests the KL regularization should be tuned to account for a
 281 suboptimal BC policy when using fewer demonstrations.

282 5 Conclusion

283 We present XQC from demonstrations (XQCFD), an effective RLfD implementation that combines a
 284 sample-efficient RL algorithm, BC pretraining and regularization, and stationary policies. XQCFD
 285 solves an persistent limitation of prior deep off-policy RLfD approaches that are incapable of
 286 efficiently finetuning BC policies due to the randomly initialized critic causing the policy performance
 287 to drop below BC. By finetuning BC policies instead of relearning from scratch, sample efficiency
 288 is significantly improved. Across 11 sparse-reward manipulation tasks from Adroit, Robomimic
 289 and MimicGen, XQCFD achieves state-of-the-art performance with a low update-to-data ratio and no
 290 Q-function ensembles. Our ablations confirm that our design decisions contribute to this performance
 291 in combination. For future work, we plan to extend this algorithm to the action chunking setting with
 292 generative policies, which has been shown to improve BC performance on complex tasks [7] but is
 293 currently not stationary or straightforward to finetune with off-policy RL.

294 **Limitations.** Our experimental scope is restricted to expert demonstrations; we do not evaluate
 295 on suboptimal offline data, although XQCFD could be applied to this setting with no adjustments.
 296 Another current limitation is the fixed KL temperature, since RL performance can be sensitive to
 297 the BC quality when balancing regularization against premature convergence. Future work could
 298 investigate adaptive temperature schedules that adjust regularization strength during training.

299 **References**

- 300 [1] Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and
301 Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on*
302 *Learning Representations (ICLR)*, 2018.
- 303 [2] Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement
304 learning with offline data. In *International Conference on Machine Learning (ICML)*, 2023.
- 305 [3] Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforce-
306 ment learning. In *International Conference on Machine Learning (ICML)*, 2017.
- 307 [4] Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas
308 Brox, and Jan Peters. CrossQ: Batch normalization in deep reinforcement learning for greater
309 sample efficiency and simplicity. *International Conference on Learning Representations (ICLR)*,
310 2024.
- 311 [5] Ondrej Biza, Thomas Weng, Lingfeng Sun, Karl Schmeckpeper, Tarik Kelestemur, Yecheng Ja-
312 son Ma, Robert Platt, Jan-Willem van de Meent, and Lawson LS Wong. On-robot reinforcement
313 learning with goal-contrastive rewards. In *IEEE International Conference on Robotics and*
314 *Automation (ICRA)*, 2025.
- 315 [6] Xinyue Chen, Che Wang, Zijian Zhou, and Keith W Ross. Randomized Ensembled Dou-
316 ble Q-Learning: Learning fast without a model. In *International Conference on Learning*
317 *Representations (ICLR)*, 2021.
- 318 [7] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ
319 Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion.
320 *The International Journal of Robotics Research (IJRR)*, 2025.
- 321 [8] Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain Markov
322 process expectations for large time. *Communications on Pure and Applied Mathematics*, 1983.
- 323 [9] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net opti-
324 mization via Hessian eigenvalue density. In *International Conference on Machine Learning*
325 *(ICML)*, 2019.
- 326 [10] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-
327 policy maximum entropy deep reinforcement learning with a stochastic actor. In *International*
328 *Conference on Machine Learning (ICML)*, 2018.
- 329 [11] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for
330 continuous control. In *International Conference on Learning Representations (ICLR)*, 2024.
- 331 [12] Hengyuan Hu, Suvir Mirchandani, and Dorsa Sadigh. Imitation bootstrapped reinforcement
332 learning. In *Robotics: Science and Systems (RSS)*, 2024.
- 333 [13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training
334 by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*,
335 2015.
- 336 [14] Jens Kober and Jan Peters. Policy search for motor primitives in robotics. *Advances in Neural*
337 *Information Processing Systems (NeurIPS)*, 2008.
- 338 [15] Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online
339 reinforcement learning via balanced replay and pessimistic Q-ensemble. In *Conference on*
340 *Robot Learning (CoRL)*, 2022.
- 341 [16] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep
342 visuomotor policies. *Journal of Machine Learning Research (JMLR)*, 2016.
- 343 [17] Clare Lyle, Zeyu Zheng, Khimya Khetarpal, James Martens, Hado van Hasselt, Razvan Pascanu,
344 and Will Dabney. Normalization and effective learning rates in reinforcement learning. In
345 *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

- 346 [18] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni,
347 Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning
348 from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*
349 (*CoRL*), 2021.
- 350 [19] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan,
351 Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning
352 using human demonstrations. In *Conference on Robot Learning (CoRL)*, 2023.
- 353 [20] Lassi Meronen, Martin Trapp, and Arno Solin. Periodic activation functions induce stationarity.
354 In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- 355 [21] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online
356 reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.
- 357 [22] Mitsuhiro Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn,
358 Aviral Kumar, and Sergey Levine. Cal-QL: Calibrated offline RL pre-training for efficient
359 online fine-tuning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- 360 [23] Manu Orsini, Anton Raichuk, Léonard Hussenot, Damien Vincent, Robert Dadashi, Sertan
361 Girgin, Matthieu Geist, Olivier Bachem, Olivier Pietquin, and Marcin Andrychowicz. What
362 matters for adversarial imitation learning? In *Advances in Neural Information Processing*
363 *Systems (NeurIPS)*, 2021.
- 364 [24] Daniel Palenicek, Florian Vogt, Joe Watson, and Jan Peters. Scaling off-policy reinforcement
365 learning with batch and weight normalization. *Advances in Neural Information Processing*
366 *Systems (NeurIPS)*, 2025.
- 367 [25] Daniel Palenicek, Florian Vogt, Joe Watson, Ingmar Posner, and Jan Peters. XQC: Well-
368 conditioned optimization accelerates deep reinforcement learning. *International Conference on*
369 *Learning Representations (ICLR)*, 2026.
- 370 [26] Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *IEEE/RSJ International*
371 *Conference on Intelligent Robots and Systems (IROS)*, 2006.
- 372 [27] Yury Polyanskiy. Information theoretic methods in statistics and computer science: Lecture 1 —
373 f -divergences, 2020.
- 374 [28] D. A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation.
375 *Neural Computation*, 1991.
- 376 [29] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances*
377 *in Neural Information Processing Systems (NeurIPS)*, 2007.
- 378 [30] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel
379 Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement
380 learning and demonstrations. *Robotics: Science and Systems (RSS)*, 2018.
- 381 [31] Carl E. Rasmussen and Chris Williams. *Gaussian Processes for Machine Learning*. MIT Press,
382 2006.
- 383 [32] Konrad Rawlik, Marc Toussaint, and Sethu Vijayakumar. On stochastic optimal control and
384 reinforcement learning by approximate inference. In *Robotics: Science and Systems (RSS)*,
385 2013.
- 386 [33] Tim GJ Rudner, Cong Lu, Michael A Osborne, Yarin Gal, and Yee Teh. On pathologies
387 in KL-regularized reinforcement learning from expert demonstrations. *Advances in Neural*
388 *Information Processing Systems (NeurIPS)*, 2021.
- 389 [34] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch nor-
390 malization help optimization? *Advances in Neural Information Processing Systems (NeurIPS)*,
391 2018.

- 392 [35] Nicol Schraudolph, Peter Dayan, and Terrence J Sejnowski. Temporal difference learning of
393 position evaluation in the game of Go. *Advances in Neural Information Processing Systems*
394 (*NeurIPS*), 1993.
- 395 [36] Noah Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael
396 Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing
397 what worked: Behavior modelling priors for offline reinforcement learning. In *International*
398 *Conference on Learning Representations (ICLR)*, 2020.
- 399 [37] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driess-
400 che, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al.
401 Mastering the game of Go with deep neural networks and tree search. *Nature*, 2016.
- 402 [38] Andrew Stirn, Hans-Hermann Wessels, Megan Schertzer, Laura Pereira, Neville E. Sanjana, and
403 David A. Knowles. Faithful heteroscedastic regression with neural networks. In *International*
404 *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2023.
- 405 [39] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT
406 Press, Cambridge, MA, 1998.
- 407 [40] Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint*
408 *arXiv:1706.05350*, 2017.
- 409 [41] Mel Vecerik, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas
410 Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. Leveraging demonstrations
411 for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint*
412 *arXiv:1707.08817*, 2017.
- 413 [42] Joe Watson, Sandy H. Huang, and Nicolas Heess. Coherent soft imitation learning. In *Advances*
414 *in Neural Information Processing Systems (NeurIPS)*, 2023.
- 415 [43] Yi Zhao, Rinu Boney, Alexander Ilin, Juho Kannala, and Joni Pajarinen. Adaptive behavior
416 cloning regularization for stable offline-to-online reinforcement learning. *Offline Reinforcement*
417 *Learning Workshop at Neural Information Processing Systems*, 2021.

418 **A Ablation: Sensitivity to the Number of Expert Demonstrations**

419 Figures 8 to 10 show that XQCfD remains effective across a range of demonstration dataset sizes on
 420 Adroit, Robomimic, and MimicGen. As expected from Hypothesis 2, gains scale with initial BC
 421 quality, indicating that the KL temperature can be tuned to extract further performance when fewer
 422 demonstrations are available.

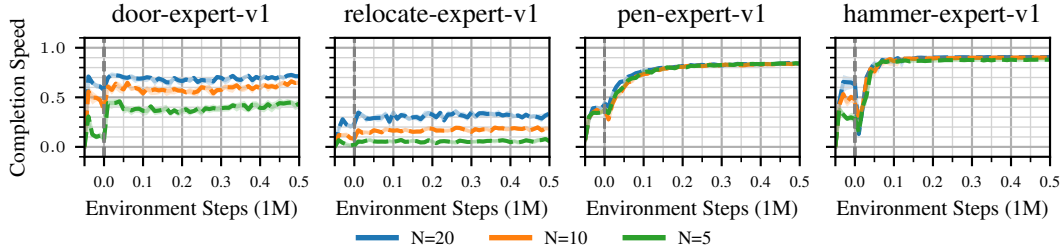


Figure 8: Sensitivity study on XQCfD, replacing the number of expert demonstrations for the Adroit environments over 10 seeds, showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals.

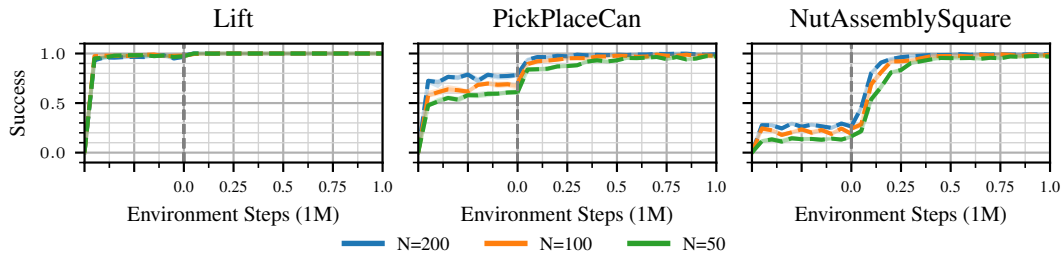


Figure 9: Sensitivity study on XQCfD, replacing the number of expert demonstrations for the Robomimic environments over 10 seeds, showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals.

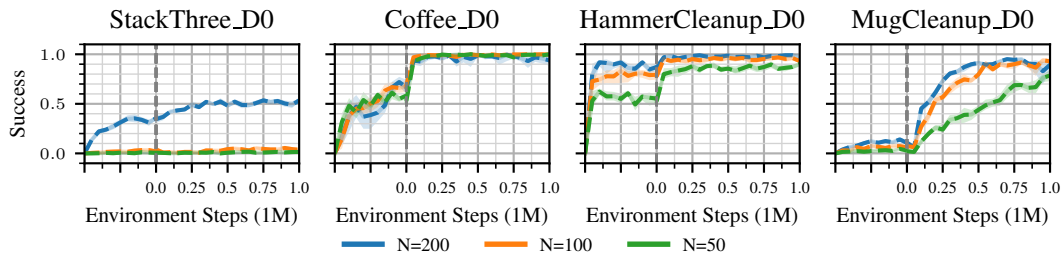


Figure 10: Sensitivity study on XQCfD, replacing the number of expert demonstrations for the MimicGen environments over 10 seeds, showing the IQM and 10th and 90th percentile stratified bootstrap confidence intervals.

423 **B The Categorical Critic Objective**

424 In C51 [3], the critic models the return distribution as a categorical distribution over N fixed
 425 atoms $\{z_i\}_{i=1}^N$ with probabilities $p_\phi(s, a) = (p_{\phi,1}(s, a), \dots, p_{\phi,N}(s, a))$. The target distribution
 426 is obtained by projecting the Bellman update $r + \gamma Z$ onto the atom support, yielding projected

427 probabilities $m(\mathbf{s}, \mathbf{a}, \mathbf{s}')$. The critic is then trained by minimizing the cross-entropy loss,

$$\mathcal{L}_Q(\theta) = -\mathbb{E}_{\mathbf{s}, \mathbf{a}, r, \mathbf{s}' \sim \mathcal{B}} \left[\sum_{i=1}^N m_i(\mathbf{s}, \mathbf{a}, \mathbf{s}') \log p_{\theta, i}(\mathbf{s}, \mathbf{a}) \right], \quad (6)$$

428 where the target projection coefficients are computed as

$$m_i(\mathbf{s}, \mathbf{a}, \mathbf{s}') = \sum_{j=1}^N \left[1 - \frac{[r + \gamma z_j]_{z_1^N} - z_i}{z_{i+1} - z_i} \right]_0^1 p_{\bar{\theta}, j}(\mathbf{s}', \mathbf{a}'). \quad (7)$$

429 with $\mathbf{a}' \sim \pi_{\phi}(\cdot | \mathbf{s}')$. Function $[\cdot]_a^b$ denoting clipping to $[a, b]$, and $\bar{\theta}$ denoting the parameters of a
430 target network.

431 C Experimental Details

432 **Evaluation protocol.** All experiments were run using 10 random seeds. For Robomimic and
433 MimicGen, evaluations are performed every 50000 environment steps, whereas for Adroit, evaluations
434 are performed every 10000 environment steps. In each evaluation phase, we execute 100 rollouts
435 using a deterministic policy. For Robomimic and MimicGen, we report the average success rate, and
436 for Adroit, we report the completion speed, following the protocol of Ball et al. [2].

437 **Implementation details.** Our implementation is based on the publicly available XQC code-
438 base [25] (<https://github.com/danielpalenicek/xqc>). We use two sets of hyperparameters that differ
439 only in their fixed temperature. For the environments MugCleanup, NutAssemblySquare, Pen,
440 and Hammer, we set the temperature to 0.001, while for all other environments, we use a temperature
441 of 0.01. We observe that stronger BC policies perform better with a higher temperature, whereas
442 weaker BC policies benefit from a lower temperature. Table 1 summarizes our hyperparameters
443 alongside those used in the baseline methods.

Hyperparameter	XQCfD	XQC	SAC	RLPD
Learning rate	3e-4	3e-4	3e-4	3e-4
hidden dim	512	512	512	256
Policy Delay	3	3	3	20
Initial Temperature	0.01 / 0.001	0.01	0.01	1.0
Target entropy	-	$ A /2$	$ A /2$	$ A /2$
Target network momentum	0.005	0.005	0.005	0.005
Update-to-data ratio	2	2	2	20
Critic ensemble size	2	2	2	10
Critic loss	Categorical	Categorical	MSE	MSE
Discount	Heuristic	Heuristic	Heuristic	0.99

Table 1: Overview of hyperparameters, including SAC (same parameters as XQC) and critic loss settings.

444 RLPD uses environment-specific hyperparameters, which we summarize in Table 2. Since RLPD
445 does not provide hyperparameters for Robomimic, we manually adjusted them. We did not perform
446 any additional tuning of these hyperparameters.

Environment	CDQ	Entropy	MLP Size	MLP Layers
Adroit	True	False	256	3
Robomimic	True	False	512	3

Table 2: Overview of RLPD environment-specific hyperparameters.

447 **Compute resources.** The experiments for this paper were conducted on a high-performance
448 computing cluster utilizing NVIDIA A100 and H100 GPUs. A single GPU could run 5 seeds in
449 parallel in 16 hours, leveraging JAX vmap over seeds. We estimate the total computational effort
450 for this paper at approximately 1,000 GPU-hours. This total encompasses the primary experimental
451 suite, ablation studies, hyperparameter tuning, and prior investigations.

452 **Licenses.** We use the following assets: XQC codebase [25] (MIT), Adroit [30] (Apache 2.0),
453 Robomimic [18] (MIT), MimicGen [19] (NVIDIA License), RLPD [2] (MIT). All assets are used in
454 accordance with their respective licenses.

455 **D Discount Heuristic**

456 We set the discount factor based on the environment’s horizon, following Hansen et al. [11]:

$$\gamma = \text{clip} \left(\frac{T/5-1}{T/5}, [0.95, 0.995] \right),$$

457 where T denotes the episode length.

458 **NeurIPS Paper Checklist**

459 **1. Claims**

460 Question: Do the main claims made in the abstract and introduction accurately reflect the
461 paper’s contributions and scope?

462 Answer: [Yes]

463 Justification: Yes, the abstract and introduction list the contributions of XQCfD. We have
464 additionally added a contributions paragraph to the end of the introduction.

465 Guidelines:

- 466 • The answer [N/A] means that the abstract and introduction do not include the claims
467 made in the paper.
- 468 • The abstract and/or introduction should clearly state the claims made, including the
469 contributions made in the paper and important assumptions and limitations. A [No] or
470 [N/A] answer to this question will not be perceived well by the reviewers.
- 471 • The claims made should match theoretical and experimental results, and reflect how
472 much the results can be expected to generalize to other settings.
- 473 • It is fine to include aspirational goals as motivation as long as it is clear that these goals
474 are not attained by the paper.

475 **2. Limitations**

476 Question: Does the paper discuss the limitations of the work performed by the authors?

477 Answer: [Yes]

478 Justification: We name limitations and assumptions in the paper and provide a dedicated
479 limitations paragraph at the end of the conclusion.

480 Guidelines:

- 481 • The answer [N/A] means that the paper has no limitation while the answer [No] means
482 that the paper has limitations, but those are not discussed in the paper.
- 483 • The authors are encouraged to create a separate “Limitations” section in their paper.
- 484 • The paper should point out any strong assumptions and how robust the results are to
485 violations of these assumptions (e.g., independence assumptions, noiseless settings,
486 model well-specification, asymptotic approximations only holding locally). The authors
487 should reflect on how these assumptions might be violated in practice and what the
488 implications would be.
- 489 • The authors should reflect on the scope of the claims made, e.g., if the approach was
490 only tested on a few datasets or with a few runs. In general, empirical results often
491 depend on implicit assumptions, which should be articulated.
- 492 • The authors should reflect on the factors that influence the performance of the approach.
493 For example, a facial recognition algorithm may perform poorly when image resolution
494 is low or images are taken in low lighting. Or a speech-to-text system might not be
495 used reliably to provide closed captions for online lectures because it fails to handle
496 technical jargon.
- 497 • The authors should discuss the computational efficiency of the proposed algorithms
498 and how they scale with dataset size.
- 499 • If applicable, the authors should discuss possible limitations of their approach to
500 address problems of privacy and fairness.
- 501 • While the authors might fear that complete honesty about limitations might be used by
502 reviewers as grounds for rejection, a worse outcome might be that reviewers discover
503 limitations that aren’t acknowledged in the paper. The authors should use their best
504 judgment and recognize that individual actions in favor of transparency play an impor-
505 tant role in developing norms that preserve the integrity of the community. Reviewers
506 will be specifically instructed to not penalize honesty concerning limitations.

507 **3. Theory assumptions and proofs**

508 Question: For each theoretical result, does the paper provide the full set of assumptions and
509 a complete (and correct) proof?

510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563

Answer: [Yes]

Justification: Our Lemma 1 is an established result we use to motivate our algorithm design.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have taken great care to aid reproducibility. First, we describe the proposed algorithm in detail throughout the main paper. Second, Section C in the Appendix details implementation details, evaluation protocols and lists hyperparameters.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

564 Question: Does the paper provide open access to the data and code, with sufficient instruc-
565 tions to faithfully reproduce the main experimental results, as described in supplemental
566 material?

567 Answer: [No]

568 Justification: We do not provide the code at the current time, however, we plan to release it
569 upon acceptance of the paper.

570 Guidelines:

- 571 • The answer [N/A] means that paper does not include experiments requiring code.
- 572 • Please see the NeurIPS code and data submission guidelines ([https://neurips.](https://neurips.cc/public/guides/CodeSubmissionPolicy)
573 [cc/public/guides/CodeSubmissionPolicy](https://neurips.cc/public/guides/CodeSubmissionPolicy)) for more details.
- 574 • While we encourage the release of code and data, we understand that this might not
575 be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not
576 including code, unless this is central to the contribution (e.g., for a new open-source
577 benchmark).
- 578 • The instructions should contain the exact command and environment needed to run to
579 reproduce the results. See the NeurIPS code and data submission guidelines ([https:](https://neurips.cc/public/guides/CodeSubmissionPolicy)
580 [//neurips.cc/public/guides/CodeSubmissionPolicy](https://neurips.cc/public/guides/CodeSubmissionPolicy)) for more de-
581 tails.
- 582 • The authors should provide instructions on data access and preparation, including how
583 to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- 584 • The authors should provide scripts to reproduce all experimental results for the new
585 proposed method and baselines. If only a subset of experiments are reproducible, they
586 should state which ones are omitted from the script and why.
- 587 • At submission time, to preserve anonymity, the authors should release anonymized
588 versions (if applicable).
- 589 • Providing as much information as possible in supplemental material (appended to the
590 paper) is recommended, but including URLs to data and code is permitted.

591 6. Experimental setting/details

592 Question: Does the paper specify all the training and test details (e.g., data splits, hyperpa-
593 rameters, how they were chosen, type of optimizer) necessary to understand the results?

594 Answer: [Yes]

595 Justification: We provide details in the main paper, as well as a dedicated section in the
596 appendix listing experimental details, evaluation protocols and hyperparameters.

597 Guidelines:

- 598 • The answer [N/A] means that the paper does not include experiments.
- 599 • The experimental setting should be presented in the core of the paper to a level of detail
600 that is necessary to appreciate the results and make sense of them.
- 601 • The full details can be provided either with the code, in appendix, or as supplemental
602 material.

603 7. Experiment statistical significance

604 Question: Does the paper report error bars suitably and correctly defined or other appropriate
605 information about the statistical significance of the experiments?

606 Answer: [Yes]

607 Justification: We report IQM and 10th and 90th percentile stratified bootstrap confidence
608 intervals across 10 random seeds for all training curves, following best practices in deep RL
609 research (Agarwal et. al. 2021).

610 Guidelines:

- 611 • The answer [N/A] means that the paper does not include experiments.
- 612 • The authors should answer [Yes] if the results are accompanied by error bars, confidence
613 intervals, or statistical significance tests, at least for the experiments that support the
614 main claims of the paper.

- 615 • The factors of variability that the error bars are capturing should be clearly stated (for
616 example, train/test split, initialization, random drawing of some parameter, or overall
617 run with given experimental conditions).
- 618 • The method for calculating the error bars should be explained (closed form formula,
619 call to a library function, bootstrap, etc.)
- 620 • The assumptions made should be given (e.g., Normally distributed errors).
- 621 • It should be clear whether the error bar is the standard deviation or the standard error
622 of the mean.
- 623 • It is OK to report 1-sigma error bars, but one should state it. The authors should
624 preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis
625 of Normality of errors is not verified.
- 626 • For asymmetric distributions, the authors should be careful not to show in tables or
627 figures symmetric error bars that would yield results that are out of range (e.g., negative
628 error rates).
- 629 • If error bars are reported in tables or plots, the authors should explain in the text how
630 they were calculated and reference the corresponding figures or tables in the text.

631 8. Experiments compute resources

632 Question: For each experiment, does the paper provide sufficient information on the com-
633 puter resources (type of compute workers, memory, time of execution) needed to reproduce
634 the experiments?

635 Answer: [Yes]

636 Justification: We include a paragraph in the Appendix, which details compute resources.

637 Guidelines:

- 638 • The answer [N/A] means that the paper does not include experiments.
- 639 • The paper should indicate the type of compute workers CPU or GPU, internal cluster,
640 or cloud provider, including relevant memory and storage.
- 641 • The paper should provide the amount of compute required for each of the individual
642 experimental runs as well as estimate the total compute.
- 643 • The paper should disclose whether the full research project required more compute
644 than the experiments reported in the paper (e.g., preliminary or failed experiments that
645 didn't make it into the paper).

646 9. Code of ethics

647 Question: Does the research conducted in the paper conform, in every respect, with the
648 NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines?>

649 Answer: [Yes]

650 Justification: We have read and conform to the code of ethics.

651 Guidelines:

- 652 • The answer [N/A] means that the authors have not reviewed the NeurIPS Code of
653 Ethics.
- 654 • If the authors answer [No], they should explain the special circumstances that require a
655 deviation from the Code of Ethics.
- 656 • The authors should make sure to preserve anonymity (e.g., if there is a special consid-
657 eration due to laws or regulations in their jurisdiction).

658 10. Broader impacts

659 Question: Does the paper discuss both potential positive societal impacts and negative
660 societal impacts of the work performed?

661 Answer: [No]

662 Justification: This work concerns a sample-efficient deep actor-critic implementation and
663 has no broader societal impact implications we believe would require a discussion.

664 Guidelines:

- 665 • The answer [N/A] means that there is no societal impact of the work performed.

- 666
- 667
- 668
- 669
- 670
- 671
- 672
- 673
- 674
- 675
- 676
- 677
- 678
- 679
- 680
- 681
- 682
- 683
- 684
- 685
- 686
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
 - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
 - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
 - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
 - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

687 **11. Safeguards**

688 Question: Does the paper describe safeguards that have been put in place for responsible
689 release of data or models that have a high risk for misuse (e.g., pre-trained language models,
690 image generators, or scraped datasets)?

691 Answer: [N/A]

692 Justification: Our work focuses on fundamental research on small scale deep reinforcement
693 learning algorithms. We neither train, nor provide any large scale model checkpoints that
694 could be misused. Therefore, this question is not applicable.

695 Guidelines:

- 696
- 697
- 698
- 699
- 700
- 701
- 702
- 703
- 704
- 705
- The answer [N/A] means that the paper poses no such risks.
 - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
 - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
 - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

706 **12. Licenses for existing assets**

707 Question: Are the creators or original owners of assets (e.g., code, data, models), used in
708 the paper, properly credited and are the license and terms of use explicitly mentioned and
709 properly respected?

710 Answer: [Yes]

711 Justification: We cite used codebases and benchmarks extensively in the paper and have
712 added a section with references and corresponding licenses to the Appendix.

713 Guidelines:

- 714
- 715
- 716
- 717
- 718
- The answer [N/A] means that the paper does not use existing assets.
 - The authors should cite the original paper that produced the code package or dataset.
 - The authors should state which version of the asset is used and, if possible, include a URL.
 - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- 719
- 720
- 721
- 722
- 723
- 724
- 725
- 726
- 727
- 728
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
 - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
 - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
 - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

729 **13. New assets**

730 Question: Are new assets introduced in the paper well documented and is the documentation
731 provided alongside the assets?

732 Answer: [N/A]

733 Justification: This paper does not release new assets.

734 Guidelines:

- 735
- 736
- 737
- 738
- 739
- 740
- 741
- 742
- The answer [N/A] means that the paper does not release new assets.
 - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
 - The paper should discuss whether and how consent was obtained from people whose asset is used.
 - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

743 **14. Crowdsourcing and research with human subjects**

744 Question: For crowdsourcing experiments and research with human subjects, does the paper
745 include the full text of instructions given to participants and screenshots, if applicable, as
746 well as details about compensation (if any)?

747 Answer: [N/A]

748 Justification: This work does not involve crowdsourcing nor research with human subjects.

749 Guidelines:

- 750
- 751
- 752
- 753
- 754
- 755
- 756
- 757
- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
 - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
 - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

758 **15. Institutional review board (IRB) approvals or equivalent for research with human
759 subjects**

760 Question: Does the paper describe potential risks incurred by study participants, whether
761 such risks were disclosed to the subjects, and whether Institutional Review Board (IRB)
762 approvals (or an equivalent approval/review based on the requirements of your country or
763 institution) were obtained?

764 Answer: [N/A]

765 Justification: This research does not involve crowdsourcing nor research with human
766 subjects.

767 Guidelines:

- 768
- 769
- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [No]

Justification: This work does not use LLMs as part of the core methodology.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.