Scaling Off-Policy Reinforcement Learning with Batch and Weight Normalization

Daniel Palenicek[1,2] **Florian Vogt**[3] **Joe Watson**[4] **Jan Peters**[1,2,5,6]
[1]Technical University of Darmstadt [2]hessian.AI [3]University of Freiburg [4]University of Oxford
[5]German Research Center for AI (DFKI) [6]Robotics Institute Germany (RIG)
daniel.palenicek@tu-darmstadt.de

Abstract

Reinforcement learning has achieved significant milestones, but sample efficiency remains a bottleneck for real-world applications. Recently, CrossQ has demonstrated state-of-the-art sample efficiency with a low update-to-data (UTD) ratio of 1. In this work, we explore CrossQ's scaling behavior with higher UTD ratios. We identify challenges in the training dynamics, which are emphasized by higher UTD ratios. To address these, we integrate weight normalization into the CrossQ framework, a solution that stabilizes training, has been shown to prevent potential loss of plasticity and keeps the effective learning rate constant. Our proposed approach reliably scales with increasing UTD ratios, achieving competitive performance across 25 challenging continuous control tasks on the DeepMind Control Suite and MyoSuite benchmarks, notably the complex `dog` and `humanoid` environments. This work eliminates the need for drastic interventions, such as network resets, and offers a simple yet robust pathway for improving sample efficiency and scalability in model-free reinforcement learning.

1 Introduction

Reinforcement Learning (RL) has shown great successes in recent years, achieving breakthroughs in diverse areas. Despite these advancements, a fundamental challenge that remains in RL is enhancing the sample efficiency of algorithms. Indeed, in real-world applications, such as robotics, collecting large amounts of data can be time-consuming, costly, and sometimes impractical due to physical constraints or safety concerns. Thus, addressing this is crucial to make RL methods more accessible and scalable.

Different approaches have been explored to address the problem of low sample efficiency in RL. Model-based RL, on the one hand, attempts to increase sample efficiency by learning dynamic models that reduce the need for collecting real data, a process often expensive and time-consuming [38, 21, 11, 16]. Model-free RL approaches, on the other hand, have explored increasing the number of gradient updates on the available data, referred to as the update-to-data (UTD) ratio [32, 9], modifying network architectures [4], or both [8, 17, 19, 31]. A central tension in these research directions is balancing the sample efficiency of the agent against the computational complexity, i.e., wall-clock time, of the underlying algorithm. Algorithmic adjustments such as model-based rollouts, computing auxiliary exploration rewards and higher UTDs can all significantly increase the wall-clock time of the algorithm. Likewise, architectural changes such as larger models [31] and more ensemble members [8] also increase the wallclock time of the method. Ideally, we desire architectural and algorithmic changes that balance sample efficiency and *simplicity*, such as CrossQ [4], which showed that careful use of batch normalization unlocks significantly greater sampler efficiency of deep actor-critic methods without significantly impacting the wallclock time.

Preprint. Under review.

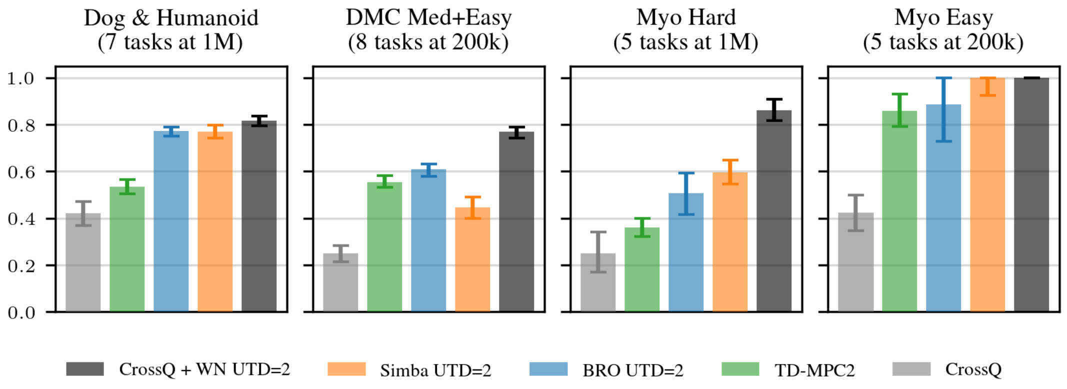

Figure 1: *CrossQ* + WN UTD=2 *outperforms* SIMBA UTD=2 *and* BRO UTD=2. In comparison, our proposed CrossQ + WN is a simple algorithm that, unlike BRO, does not require extra exploration policies or full parameter resets. We present results for 25 complex continuous control tasks from the DMC and MyoSuite benchmarking suites. 1.0 marks the maximum score achievable on the respective benchmarks (DMC *return* up to 1000 / MyoSuite up to 100% *success rate*). We present IQM and 90% stratified bootstrap confidence intervals aggregated over multiple environments and 10 seeds each.

In this work, we build upon CrossQ [4], the model-free RL algorithm that showed state-of-the-art sample efficiency on the `MuJoCo` [41] continuous control benchmarking tasks, and also enabled learning omni-directional locomotion policies in 8 minutes of real-world experience [6]. Notably, the authors achieved this by carefully utilizing batch normalization (BN, Ioffe [20]) within the actor-critic architecture. A technique previously thought not to work in RL, as reported by Hiraoka et al. [17] and others. The insight that Bhatt et al. [4] offered is that one needs to carefully consider the different state-action distributions within the Bellman equation and handle them correctly to succeed. This novelty allowed CrossQ at a low UTD of 1 to outperform the then state-of-the-art algorithms that scaled their UTD ratios up to 20. Even though higher UTD ratios are more computationally expensive, they allow for larger policy improvements using the same amount of data. This naturally raises the question: *How can we extend the sample efficiency benefits of CrossQ and* BN *to the high* UTD *training regime?* Which we address in this manuscript.

Contributions. In this work, we show that the vanilla CrossQ algorithm is brittle to tune on DeepMind Control (DMC) and MyoSuite environments and can fail to scale reliably with increased compute. To address these limitations, we propose the addition of weight normalization (WN),

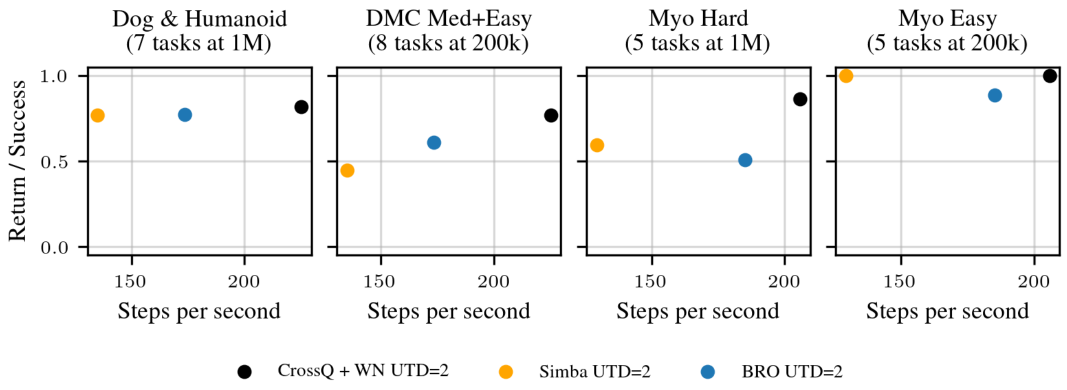

Figure 2: Comparing performance against wall clock time, measured in environment steps per second (so larger is better) on a single `RTX 4090` workstation, we observe the CrossQ + WN outperforms SIMBAand BRO across all environments. We present results for 25 complex continuous control tasks from the DMC and MyoSuite benchmarking suites. 1.0 marks the maximum score achievable on the respective benchmarks (DMC *return* up to 1000 / MyoSuite up to 100% *success rate*).

which we show to be a simple yet effective enhancement that stabilizes CrossQ. We motivate the combined use of WN and BN on insights from the continual learning and loss of plasticity literature and connections to the effective learning rate. Our experiments show that incorporating WN not only improves the stability of CrossQ but also allows us to scale its UTD, thereby significantly enhancing sample efficiency.

2 Preliminaries

This section briefly outlines the required background knowledge for this paper.

Reinforcement learning. A Markov decision process (MDP) [34] is a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \mu_0, \gamma \rangle$, with state space $\mathcal{S} \subseteq \mathbb{R}^n$, action space $\mathcal{A} \subseteq \mathbb{R}^m$, transition probability $\mathcal{P} : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$, the reward function $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, initial state distribution μ_0 and discount factor γ. We define the RL problem according to Sutton and Barto [39]. A policy $\pi : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ is a behavior plan, which maps a state $\boldsymbol{s}$ to a distribution over actions $\boldsymbol{a}$. The discounted cumulative return is defined as $\mathcal{R}^\pi(\boldsymbol{s}, \boldsymbol{a}) = \sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)$, where $\boldsymbol{s}_0 = \boldsymbol{s}$, $\boldsymbol{a}_0 = a$ and $\boldsymbol{s}_{t+1} \sim \mathcal{P}(\,\cdot\,|\boldsymbol{s}_t, \boldsymbol{a}_t)$ and $\boldsymbol{a}_t \sim \pi(\,\cdot\,|\boldsymbol{s}_t)$ otherwise. The Q-function of a policy π is the expected discounted return $Q^\pi(\boldsymbol{s}, \boldsymbol{a}) = \mathbb{E}[\mathcal{R}^\pi(\boldsymbol{s}, \boldsymbol{a})]$. The goal of an RL agent is to find an optimal policy π^* that maximizes the expected return from the initial state distribution $\pi^* = \arg\max_\pi \mathbb{E}_{\boldsymbol{s} \sim \mu_0} \left[Q^\pi(\boldsymbol{s}, \boldsymbol{a}) \right]$.

Soft actor-critic (SAC, Haarnoja et al. [13]) addresses this optimization problem by jointly learning neural network representations for the Q-function and the policy. The policy network is optimized to maximize the Q-values, while the Q-function is optimized to minimize the squared Bellmann error, where the value target is computed by taking an expectation over the learned Q function

$$V(\boldsymbol{s}_{t+1}) = \mathbb{E}_{\boldsymbol{a}_{t+1} \sim \pi_\theta(\cdot|\boldsymbol{s}_{t+1})} \left[Q_{\bar{\theta}}(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) \right]. \tag{1}$$

To stabilize the Q-function learning, Haarnoja et al. [13] found it necessary to use a target Q-network in the computation of the value function instead of the regular Q-network. The target Q-network is structurally equal to the regular Q-network, and its parameters $\bar{\theta}$ are obtained via Polyak Averaging over the learned parameters θ. While this scheme ensures stability during training by explicitly delaying value function updates, it also arguably slows down online learning [33, 22, 30].

Instead of relying on target networks, CrossQ [4] addresses training stability issues by introducing batch normalization (BN, Ioffe [20]) in its Q-function and achieves substantial improvements in sample and computational efficiency over SAC. A central challenge when using BN in Q networks is distribution mismatch: during training, the Q-function is optimized with samples $\boldsymbol{s}_t, \boldsymbol{a}_t$ from the replay buffer. However, when the Q-function is evaluated to compute the target values (eq. (1)), it receives actions sampled from the current policy $\boldsymbol{a}_{t+1} \sim \pi_\theta(\,\cdot\,|\boldsymbol{s}_{t+1})$. Those samples have no guarantee of lying within the training distribution of the Q-function. BN is known to struggle with out-of-distribution samples, as such, training can become unstable if the distribution mismatch is not correctly accounted for [4]. To deal with this issue, CrossQ removes the separate target Q-function and evaluates both Q values during the critic update in a single forward pass, which causes the BN layers to compute shared statistics over the samples from the replay buffer and the current policy. This scheme effectively tackles distribution mismatch problems, ensuring that all inputs and intermediate activations are effectively forced to lie within the training distribution.

Normalization techniques in RL. Normalization techniques are widely recognized for improving the training of neural networks, as they generally accelerate training and improve generalization [18]. There are many ways of introducing different types of normalizations into the RL framework. Most commonly, authors have used layer normalization (LN) within the network architectures to stabilize training [17, 29, 26]. Recently, CrossQ has been the first algorithm to successfully use BN layers in RL [4]. The addition of BN leads to substantial gains in sample efficiency. In contrast to LN, however, one needs to carefully consider the different state-action distributions within the critic loss when integrating BN. In a different line of work, Hussing et al. [19] proposed the integration of unit ball normalization and projected the output features of the penultimate layer onto the unit ball in order to reduce Q-function overestimation.

Increasing update-to-data ratios. Although scaling up the UTD ratio is an intuitive approach to increase the sample efficiency, in practice, it comes with several challenges. Nikishin et al. [32]

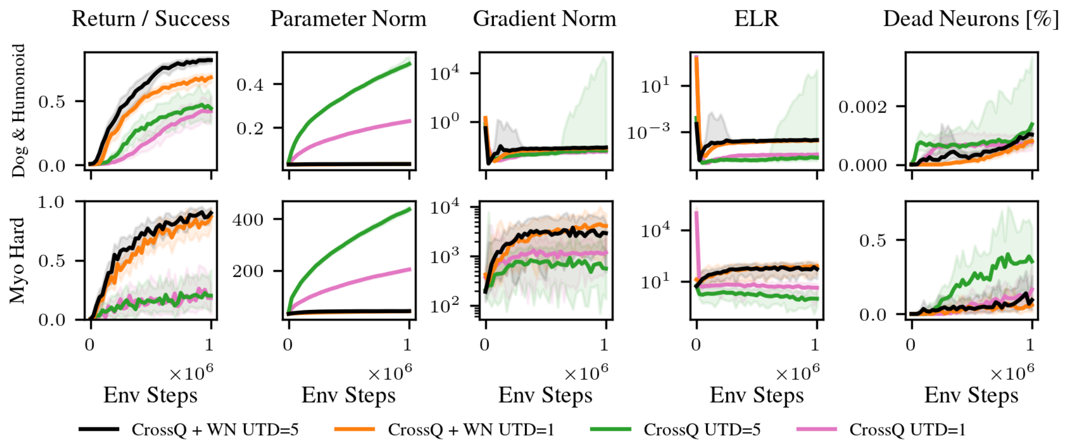

Figure 3: *Growing parameter norms hinder learning.* The performance benefits of CrossQ fail to scale to more complex, higher dimensional tasks such as humanoid locomotion and muscular manipulation. Investigating this, we find that the critic parameter norms increase significantly with increasing UTD ratios. As a result, the effective learning rate (ELR) drops and the number of dead neurons increases. Regularizing the critic parameters with weight norm (WN), we successfully mitigate the parameter norm growth and therefore maintain a more consistent ELR, leading to better performance on these more complex tasks. Uncertainty quantification depicts the 90% stratified bootstrap confidence intervals.

demonstrated that overfitting on early training data can inhibit the agent from learning anything later in the training. The authors dub this phenomenon the primacy bias. To address the primacy bias, they suggest to periodically reset the network parameters while retraining the replay buffer. Many works that followed have adapted this intervention [9, 31]. While often effective, regularly resetting is a very drastic intervention and by design induces regular drops in performance. Since the agent has to start learning from scratch repeatedly, it is also not very computing efficient. Finally, the exact reasons why parameter resets work well in practice are not yet well understood [27]. Instead of resetting there have also been other types of regularization that allowed practitioners to train stably with high UTD ratios. Janner et al. [21] generate additional modeled data, by virtually increasing the UTD. In REDQ, Chen et al. [8] leverage ensembles of Q-functions, while Hiraoka et al. [17] use dropout and LN to effectively scale to higher UTD ratios.

3 Batch normalization alone fails to scale with respect to task complexity

Bhatt et al. [4] demonstrated CrossQ's state-of-the-art sample efficiency on the `MuJoCo` task suite [41], while at the same time also being very computationally efficient. However, on the more extensive DMC and MyoSuite task suites, we find that CrossQ requires tuning. We further find that it works on some, but not all, environments stably and reliably.

Figure 3 shows CrossQ training performance on the DMC `dog` and `humanoid` and the Myo Hard tasks aggregated by environment suite. We plot the IQM and 90% confidence intervals for each metric across 10 seeds. The figure compares a CrossQ with UTDs 1 and 5, where the hyperparameters were identified through a grid search over learning rates and network sizes, as detailed in Table 1. The first shows the agent's training performance, while the other columns show network parameter norms, gradient norms, the effective learning rate [43], and the fraction of dead neurons. Here, we identify three different training behaviors. We notice that CrossQ does not benefit from higher UTD ratios, but performance remains similar on the provided tasks. Overall, for all CrossQ runs we notice large confidence intervals.

Growing network parameter norms. The second column of Figure 3 displays the sum over the L2 norms of the dense layers in the critic network. This includes three dense layers, each with a hidden dimension of 512. On both task suites, CrossQ exhibits growing network weights over the course of training. We find that the effect is particularly pronounced for CrossQ with increasing UTD ratios.

Growing network weights have been linked to a loss of plasticity, a phenomenon where networks become increasingly resistant to parameter update, which can lead to premature convergence [10]. Additionally, the growing magnitudes pose a challenge for optimization, connected to the issue of growing activations, which has recently been analyzed by Hussing et al. [19]. Further, growing network weights decrease the effective learning rate when the networks contain normalization layers [42, 29].

In summary, the scaling results for vanilla CrossQ are mixed. While increasing UTD ratios is known to yield increased sample efficiency, if careful regularization is used [21, 8, 32], CrossQ alone with BN cannot benefit from it. We notice that with increasing UTD ratios, CrossQ's weight layer norms grow significantly faster and overall larger. This observation motivates us to further study the weight norms in CrossQ to increase UTD ratios.

4 Combining batch normalization and weight normalization enables scaling

Inspired by the combined insights of Van Hasselt et al. [42] and Lyle et al. [29], we propose to integrate CrossQ with weight normalization (Salimans and Kingma [35], WN) as a means of counteracting the rapid growth of weight norms we observe with increasing update-to-data (UTD) ratios. A weight normalized parameter $\tilde{\boldsymbol{w}}$ is constrained to have an L2 norm of c, an additional hyperparameter,

$$\tilde{\boldsymbol{w}} = c\,\boldsymbol{w} \,/\, ||\boldsymbol{w}||_2. \tag{2}$$

Our approach is based on the following reasoning: Due to the use of BN in CrossQ, the critic network exhibits scale invariance, as previously noted by Van Laarhoven [43].

Theorem 1 (Van Laarhoven [43]) *Let $f(\boldsymbol{X}; \boldsymbol{w}, b, \gamma, \beta)$ be a function, with inputs $\boldsymbol{X}$ and parameters $\boldsymbol{w}$ and $\boldsymbol{b}$ and γ and β batch normalization parameters. When f is normalized with batch normalization, f becomes scale-invariant with respect to its parameters, i.e.,*

$$f(\boldsymbol{X}; c\boldsymbol{w}, cb, \gamma, \beta) = f(\boldsymbol{X}; \boldsymbol{w}, b, \gamma, \beta), \tag{3}$$

with scaling factor $c > 0$.

The proof is provided in Appendix A.

This property allows us to introduce WN as a mechanism to regulate the growth of weight norms in CrossQ without affecting the critic's outputs. Further, it can be shown, that for such a scale invariant function, the gradient scales inversely proportionally to the scaling factor $c > 0$.

Theorem 2 (Van Laarhoven [43]) *Let $f(\boldsymbol{X}; \boldsymbol{w}, b, \gamma, \beta)$ be a scale-invariant function. Its gradient*

$$\nabla f(\boldsymbol{X}; c\boldsymbol{w}, cb, \gamma, \beta) = \nabla f(\boldsymbol{X}; \boldsymbol{w}, b, \gamma, \beta)/c, \tag{4}$$

scales inversely proportional to the scaling factor $c \in \mathbb{R}$ of its parameters $\boldsymbol{w}$.

The proof is provided in Appendix B.

Recently, Lyle et al. [29] demonstrated that the combination of LN and WN can help mitigate loss of plasticity. Since the gradient scale is inversely proportional to c, keeping norms constant helps to maintain a stable effective learning rate (ELR,Van Hasselt et al. [42]), further enhancing training stability. We conjecture that maintaining a stable ELR could also be beneficial when increasing the UTD ratios in continuous control RL. As the UTD ratio increases, the networks are updated more frequently with each environment interaction. Empirically, we find that the network norms tend to grow quicker with increased UTD ratios (Figure 3), which in turn decreases the ELR even quicker and could be the case for instabilities and low training performance. From this observation, we hypothesize that the training phenomena that affect plasticity also appear when attempting sample-efficient learning with higher UTDs. This hypothesis suggests that regularization techniques for plasticity could also be used to achieve more sample-efficient RL. As a result, we empirically investigate the effectiveness of combining CrossQ with WN with increasing UTD ratios.

Implementation details. We apply WN to the first two linear layers, ensuring that their weights remain unit norm after each gradient step by projecting them onto the unit ball, similar to Lyle et al. [29]. While we could employ a learning rate schedule [29] we did not investigate this here as this would add additional complexity. Additionally, we impose weight decay on all parameters that remain unbounded—specifically, the final dense output layer. In practice, we use AdamW [28] with a decay of 0 (which falls back to vanilla Adam [23]) for the normalized intermediate dense layers and 1e$-$2 otherwise.

Target networks. CrossQ removes the target networks from the actor-critic framework and showed that using BN training remains stable even without them [4]. While we find this to be true in many cases, we find that especially in DMC, the re-integration of target networks can help stabilize training overall (see Section 5.4). However, not surprisingly, we find that the integration of target networks with BN requires careful consideration of the different state-action distributions between the $\boldsymbol{s}, \boldsymbol{a}$ and $\boldsymbol{s}', \boldsymbol{a}' \sim \pi(\boldsymbol{s}')$ exactly as proposed by Bhatt et al. [4]. To satisfy this, we keep the joined forward pass through both the critic network as well as the target critic network. We evaluate both networks in *training mode*, i.e., they calculate the joined state-action batch statistics on the current batches. As is common, we use Polyak-averaging with a $\tau = 0.005$ from the critic network to the target network.

5 Experimental results

To evaluate the effectiveness of our proposed CrossQ + WN method, we conduct a comprehensive set of experiments on the DeepMind Control Suite [40] and MyoSuite [7] benchmarks. For DMC we report individual results for the hard (`dog` and `humanoid`), as well as Medium+Easy (`cheetah-run`, `walker-run`, `hopper-stand`, `finger-turn-hard`, `quadruped-run`, `fish-swim`, `hopper-hop`, `pendulum-swingup`) due to their varying difficulties. Equally, we split MyoSuite hard and easy environments. Our primary goal is to investigate the scalability of CrossQ + WN with increasing UTD ratios and to assess the stabilizing effects of combining CrossQ with WN. We compare our approach to several baselines, including the recent BRO [31], CrossQ [4], TD-MPC2 [15], and SIMBA [26], a concurrent approach utilizing layer norm. Figure 6 in the appendix further provides a SR-SAC [9] baseline, a version of SAC [13] with high UTD ratios and network resets.

5.1 Experimental setup

Our implementation is based on the SAC implementation of `jaxrl` codebase [24]. We implement CrossQ following the author's original codebase and add the architectural modifications introduced by [4], incorporating batch normalization in the actor and critic networks. We extend this approach by introducing WN to regulate the growth of weight norms and prevent loss of plasticity and add target networks. We perform a grid search to focus on learning rate selection and layer width.

We evaluate 25 diverse continuous control tasks, 15 from DMC and 10 from MyoSuite. These tasks vary significantly in complexity, requiring different levels of fine motor control and policy adaptation with high-dimensional state spaces up to $\mathbb{R}^{223}$. Each experiment is run for 1 million environment steps and across 10 random seeds to ensure statistical robustness. We evaluate agents every $25,000$ environment steps for 5 trajectories. For the DMC Medium&Easy and Myo Easy we plot the first $200k$ steps, as all methods learn much faster than the 1 million steps. As proposed by Agarwal et al. [1], we report the interquartile mean (IQM) and 90% stratified bootstrap confidence intervals (CIs) of the return (or success rate, respectively), if not otherwise stated. For the BRO and SIMBA baseline results, for computational reasons, we take the official evaluation data that the authors provide. The official BRO codebase is also based on `jaxrl`, and the authors followed the same evaluation protocol. All experiments were run on a compute cluster with `RTX 3090` and `A5000` GPUs, where all 10 seeds run in parallel on a single GPU via `jax.vmap`.

5.2 Weight normalization allows CrossQ to scale to harder tasks effectively

We provide empirical evidence for our hypothesis that controlling the weight norm and, thereby, the ELR can stabilize training (Figure 3). We show that through the addition of WN, CrossQ + WN shows stable training and can stably scale with increasing UTD ratios.

Figure 7 shows per environment results of our experiments encompassing all 25 tasks evaluated across 10 seeds each. Based on that, Figure 1 shows aggregated performance over all environments from Figure 7 per task suite, with a separate aggregation for the most complex `dog` and `humanoid` environments.

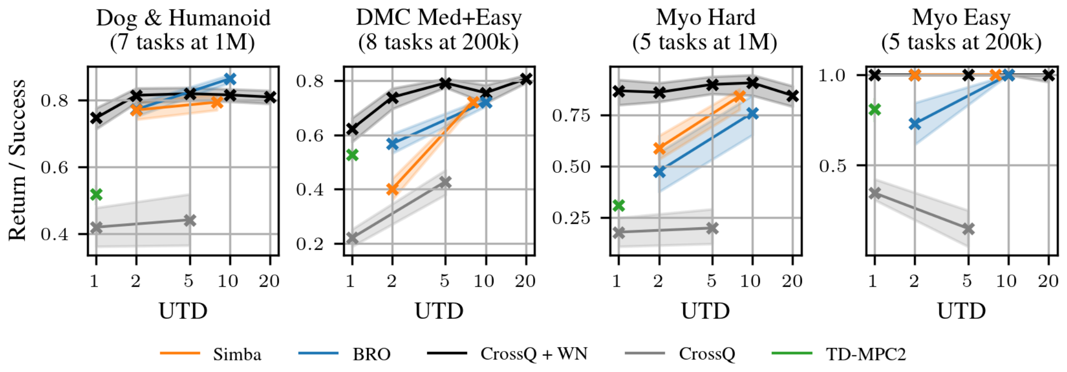

Figure 4: *CrossQ* WN UTD *scaling behavior.* We plot the IQM return and 90% confidence intervals for different UTD ratios. Results are aggregated over 15 DMC environments and 10 random seeds each according to Agarwal et al. [1]. The sample efficiency scales reliably with increasing UTD ratios and remains stable even when there are no more performance gains, which is a crucial property.

Scaling results in Figure 4 show that CrossQ + WN is competitive to the BRO and SIMBA baselines on both DMC and MyoSuite, especially on the more complex `dog` and `humanoid` tasks already on lower UTD ratios. Notably, CrossQ + WN UTD=5 uses only half the UTD of BRO and does not require any parameter resets and no additional exploration policy. Further, it uses $\sim 90\%$ fewer network parameters—BRO reports $\sim 5M$, while our proposed CrossQ + WN uses only $\sim 600k$ (these numbers vary slightly per environment, depending on the state and action dimensionalities).

In contrast, vanilla CrossQ UTD=1 exhibits much slower learning on most tasks and, in some environments, fails to learn performant policies. Moreover, the instability of vanilla CrossQ at UTD=5 is particularly notable, as it does not reliably converge across environments (Figure 7).

These findings highlight the necessity of incorporating additional normalization techniques to sustain effective training at higher UTD ratios. This leads us to conclude that CrossQ benefits from the addition of WN, which results in stable training and scales well with higher UTD ratios. The resulting algorithm can match or outperform state-of-the-art baselines on the continuous control DMC and MyoSuite benchmarks while being much simpler algorithmically.

5.3 Stable scaling of CrossQ + WN with UTD ratios

To visualize the stable scaling behavior of CrossQ + WN we ablate across UTD ratios. Figure 4 shows training performances aggregated over multiple environments for 10 seeds each at 1M steps and 200k steps respectively. We confirm that CrossQ + WN shows reliable scaling behavior. With increasing compute, the performance increases or stays constant which is desirable. We see, that for the same UTD ratio, CrossQ + WN nearly always beats both the BRO and SIMBA baselines.

5.4 Hyperparameter ablation studies

We also ablate the different hyperparameters of CrossQ + WN, by changing each one at a time. Figure 5 shows aggregated results of the final performances of each ablation. We will briefly discuss each ablation individually.

Removing weight normalization. Not performing weight normalization results in the biggest drop in performance across all our ablations. This loss is most drastic on the MyoSuite tasks and often results in no meaningful learning. Showing that, as hypothesized, the inclusion of WN into the CrossQ framework yields great improvements in terms of sample efficiency and training stability, especially for larger UTD ratios. In general, lower UTD ratios are already reasonably competitive in overall performance.

Target networks. Ablating the target networks shows that on MyoSuite, there is a small but significant difference between using a target network and or no target network. Results on DMC show a large drop in performance. There, removing target networks leads to a significant drop in

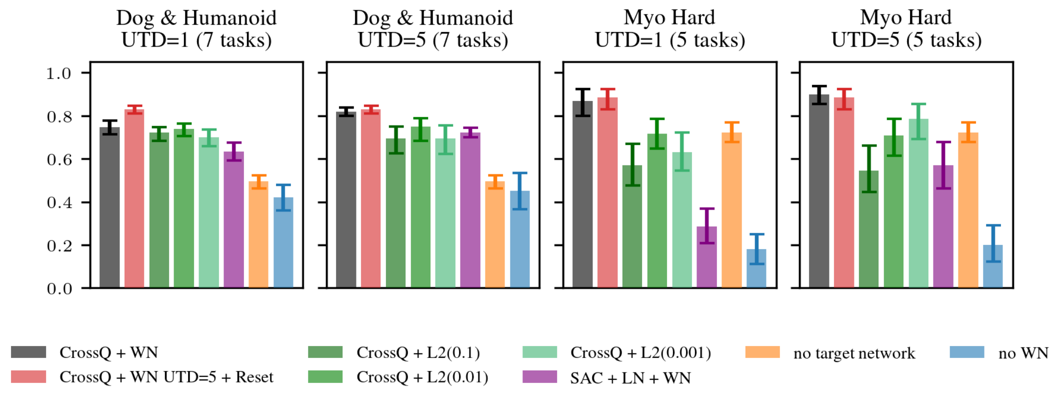

Figure 5: An ablation study comparing CrossQ + WN against a soft L2 penality on the weights, as well as other design decisions such as target networks. The results show that the hard constraint outperforms the soft approach across a range of regularization scales and tasks. Uncertainty quantification depicts the 90% stratified bootstrap confidence intervals.

performance, nearly as large as removing weight normalization. This finding is interesting, as it suggests that CrossQ + WN without target networks is not inherently unstable. But there are situations where the inclusion of target networks is required. Further investigating the role and necessity of target networks in RL is an interesting direction for future research.

L2 regularization. Figure 5 investigates the performance of a soft L2 penalty on the weights compared to the weight normalization proposed in this paper. Across all tasks, and sweeping across soft regularization scalings, weight normalization outperforms the soft L2 penalty. Our hypothesis is that, in principle, soft L2 regularization could work in a similar way to the proposed WN; however, it would require per-task tuning and potentially even scheduling to result in a stable ELR. In comparison, the *hard constraint* via WN guarantees a stable weight norm by design and as such is easier to employ.

Parameter resets. Our experiments show that CrossQ + WN is able to scale without requiring drastic interventions such as parameter resets. However, it is still interesting to investigate whether CrossQ + WN's performance could benefit from parameter resets. CrossQ + WN UTD$= 5$ + Reset in Figure 5 investigates this question. We see, that there is a slight improvement on the 7 `dog` and `humanoid` tasks, on all other 18 tasks, performance remains the same. The main takeaway is, that CrossQ + WN scales stably *without requiring* parameter resets.

6 Related work

RL has demonstrated remarkable success across various domains, yet sample efficiency remains a significant challenge, especially in real-world applications where data collection is expensive or impractical. Various approaches have been explored to address this issue, including model-based RL, UTD ratio scaling, and architectural modifications.

Update-to-data ratio scaling. Model-free RL methods, including those utilizing higher UTD ratios, focus on increasing the number of gradient updates per collected sample to maximize learning from available data. High UTD training introduces several challenges, such as overfitting to early training data, a phenomenon known as primacy bias [32]. This can be counteracted by periodically resetting the network parameters [32, 9]. However, network resets introduce abrupt performance drops. Nauman et al. [31] demonstrated that full parameter resets of the critic can effectively preserve learning capacity using a UTD ratio up to 10. However, such resets inevitably impact the wall-clock time due to relearning function approximation several times during learning. Alternative approaches use an ensemble of Q-functions to reduce overestimation bias that occurs under high UTD ratio training regimes [8]. Due to the decreased computational efficiency of using a large number of Q-functions, Hiraoka et al. [17] propose to replace the ensemble of critics with dropout and layer normalization. Both methods utilize a UTD ratio of 20, which is highly inefficient.

Normalization techniques in RL. Normalization techniques have long been recognized for their impact on neural network training. LN [3] and other architectural modifications have been used to stabilize learning in RL [17, 31]. Yet BN has only recently been successfully applied in this context [4], challenging previous findings, where BN in critics caused training to fail [17]. WN has been shown to keep ELRs stable and prevent loss of plasticity [29], when combined with LN, making it a promising candidate for integration into existing RL frameworks.

Normalization techniques have long been recognized for their impact on neural network training. Bjorck et al. [5] show that using spectral normalization (SN) enables training with large scale neural networks in deep RL. SN divides a layer's weight matrix by its largest singular value, regularizing the Lipschitz continuity of the function approximation and therefore stabilizing the gradients. BN has been used by Bhatt et al. [4], where it achieves impressive results, but requires slight adjustments when the UTD ratio is scaled. Concurrent work [26] injected 'simplicity bias' into their actor and critic architecture, encouraging the model to favor 'simpler' features for its predictions. Simba incorporates a residual feedforward block and layer normalization in both the actor and critic networks and achieved state of the art results. WN has been shown to keep ELRs stable and prevent loss of plasticity [29], when combined with LN, making it a promising candidate for integration into existing RL frameworks.

Hafner et al. [14] designed a vision-based MBRL algorithm Dreamer that leverages world models to master a wide range of diverse tasks, and also relies on several normalization techniques. They utilize root mean square layer normalization (RMSNORM) [45] before the activation function and normalize the returns.

To successfully scale deep RL on the Atari 100k benchmark and achieve human-level sample efficiency, Schwarzer et al. [36] rely on regularization techniques. Rather than use normalization methods, they use the shrink-and-perturb method [2] in shallow layers and full parameter resets in deeper layers to preserve network plasticity. To scale to higher UTDs, they introduce AdamW [28] and gradually decrease the number of steps for the computation of the TD error for faster convergence. Lee et al. [25] argue that the loss of plasticity observed in deep RL when the UTD ratio is increased can be mitigated by using LN, sharpness-aware minimization (SAM, Foret et al. [12]), incorporating parameter resets in the last layers and replacing the ReLU activation function with a 'concetenated ReLU' function [37]. Voelcker et al. [44] demonstrated that parameter resets are not strictly necessary when the UTD ratio is increased to improve sample efficiency. They identify the generalization ability of the critic as the main source of training instabilities under high UTD regimes. They demonstrate empirically that architectural regularization can mitigate overestimation and divergence, but it does not guarantee proper generalization. On the other hand, leveraging synthetic data generated by a learned world model can help mitigate the effects of distribution shift, thereby enhancing generalization.

7 Conclusion, limitations & future work

In this work, we have addressed the instability and scalability limitations of CrossQ in RL by integrating weight normalization. Our empirical results demonstrate that WN effectively stabilizes training and allows CrossQ to scale reliably with higher UTD ratios. The proposed CrossQ + WN approach achieves competitive or superior performance compared to state-of-the-art baselines across a diverse set of 25 complex continuous control tasks from the DMC and MyoSuite benchmarks. These tasks include complex and high-dimensional `humanoid` and `dog` environments. This extension preserves simplicity while enhancing robustness and scalability by eliminating the need for drastic interventions such as network resets.

In this work, we only consider continuous state-action benchmarking tasks. While our proposed CrossQ + WN performs competitively on these tasks, its performance on discrete state-action spaces or vision-based tasks remains unexplored. We plan to investigate this in future work. Moreover, the theoretical basis of our work does not directly connect to the convergence rates or sample efficiency of the underlying RL algorithm, but rather to mitigate observed empirical phenomena regarding the function approximation alone.

Acknowledgments and Disclosure of Funding

We would also like to thank Tim Schneider, Cristiana de Farias, João Carvalho and Theo Gruner for proofreading and constructive criticism on the manuscript. This research was funded by the research cluster "Third Wave of AI", funded by the excellence program of the Hessian Ministry of Higher Education, Science, Research and the Arts, hessian.AI. This work was also supported by a UKRI/EPSRC Programme Grant [EP/V000748/1].

References

[1] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 2021.

[2] Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training, 2020. URL https://arxiv.org/abs/1910.08475.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[4] Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. CrossQ: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *International conference on learning representations*, 2024.

[5] Johan Bjorck, Carla P. Gomes, and Kilian Q. Weinberger. Towards deeper deep reinforcement learning with spectral normalization, 2022. URL https://arxiv.org/abs/2106.01151.

[6] Nico Bohlinger, Jonathan Kinzel, Daniel Palenicek, Lukasz Antczak, and Jan Peters. Gait in eight: Efficient on-robot learning for omnidirectional quadruped locomotion. *arXiv preprint arXiv:2503.08375*, 2025.

[7] Vittorio Caggiano, Huawei Wang, Guillaume Durandau, Massimo Sartori, and Vikash Kumar. Myosuite–a contact-rich simulation suite for musculoskeletal motor control. *arXiv preprint arXiv:2205.13600*, 2022.

[8] Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double Q-learning: Learning fast without a model. In *International conference on learning representations*, 2021.

[9] Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G Bellemare, and Aaron Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *International conference on learning representations*, 2022.

[10] Mohamed Elsayed, Qingfeng Lan, Clare Lyle, and A Rupam Mahmood. Weight clipping for deep continual and reinforcement learning. In *Reinforcement Learning Conference*, 2024.

[11] Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I. Jordan, Joseph E. Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. In *International Conference on Machine Learning*, 2018.

[12] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations (ICLR)*, 2021.

[13] 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. *arXiv preprint arXiv:1812.05905*, 2018.

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

[15] Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control, 2024. URL https://arxiv.org/abs/2310.16828.

[16] Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. In *Advances in neural information processing systems*, 2015.

[17] Takuya Hiraoka, Takahisa Imagawa, Taisei Hashimoto, Takashi Onishi, and Yoshimasa Tsuruoka. Dropout q-functions for doubly efficient reinforcement learning. In *International conference on learning representations*, 2021.

[18] Lei Huang, Jie Qin, Yi Zhou, Fan Zhu, Li Liu, and Ling Shao. Normalization techniques in training dnns: Methodology, analysis and application. *IEEE transactions on pattern analysis and machine intelligence*, 2023.

[19] Marcel Hussing, Claas Voelcker, Igor Gilitschenski, Amir-massoud Farahmand, and Eric Eaton. Dissecting deep rl with high update ratios: Combatting value overestimation and divergence. *arXiv preprint arXiv:2403.05996*, 2024.

[20] Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[21] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in neural information processing systems*, 2019.

[22] Seungchan Kim, Kavosh Asadi, Michael Littman, and George Konidaris. Deepmellow: Removing the need for a target network in deep q-learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 2733–2739. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/379. URL https://doi.org/10.24963/ijcai.2019/379.

[23] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Ilya Kostrikov. JAXRL: Implementations of Reinforcement Learning algorithms in JAX, 2021. URL https://github.com/ikostrikov/jaxrl.

[25] Hojoon Lee, Hanseul Cho, Hyunseung Kim, Daehoon Gwak, Joonkee Kim, Jaegul Choo, Se-Young Yun, and Chulhee Yun. Plastic: Improving input and label plasticity for sample efficient reinforcement learning, 2023. URL https://arxiv.org/abs/2306.10711.

[26] Hojoon Lee, Dongyoon Hwang, Donghu Kim, Hyunseung Kim, Jun Jet Tai, Kaushik Subramanian, Peter R Wurman, Jaegul Choo, Peter Stone, and Takuma Seno. Simba: Simplicity bias for scaling up parameters in deep reinforcement learning. *International Conference on Learning Representations*, 2025.

[27] Qiyang Li, Aviral Kumar, Ilya Kostrikov, and Sergey Levine. Efficient deep reinforcement learning requires regulating overfitting. In *International conference on learning representations*, 2023.

[28] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[29] Clare Lyle, Zeyu Zheng, Khimya Khetarpal, James Martens, Hado van Hasselt, Razvan Pascanu, and s Will Dabney. Normalization and effective learning rates in reinforcement learning. In *Neural information processing systems*, 2024.

[30] Miguel Morales. Grokking deep reinforcement learning. 2020.

[31] Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample-efficient continuous control. In *Advances in neural information processing systems*, 2024.

[32] Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International conference on machine learning*, 2022.

[33] Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, Vikash Kumar, and Wojciech Zaremba. Multi-goal reinforcement learning: Challenging robotics environments and request for research, 2018. URL https://arxiv.org/abs/1802.09464.

[34] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[35] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[36] Max Schwarzer, Johan Obando-Ceron, Aaron Courville, Marc Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: Human-level atari with human-level efficiency, 2023. URL https://arxiv.org/abs/2305.19452.

[37] Wenling Shang, Kihyuk Sohn, Diogo Almeida, and Honglak Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units, 2016. URL https://arxiv.org/abs/1603.05201.

[38] Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. *Machine learning*, 1990.

[39] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2018.

[40] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[41] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *International conference on intelligent robots and systems*, 2012.

[42] Hado P Van Hasselt, Matteo Hessel, and John Aslanides. When to use parametric models in reinforcement learning? In *Advances in Neural Information Processing Systems*, 2019.

[43] Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

[44] Claas A Voelcker, Marcel Hussing, Eric Eaton, Amir massoud Farahmand, and Igor Gilitschenski. Mad-td: Model-augmented data stabilizes high update ratio rl, 2025. URL https://arxiv.org/abs/2410.08896.

[45] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

A Proof Scale Invariance

Proof of Theorem 1.

$$\begin{aligned}
f(\boldsymbol{X}; c\boldsymbol{w}, cb, \gamma, \beta) &= \frac{g(c\boldsymbol{X}\boldsymbol{w} + cb) - \mu(g(c\boldsymbol{X}\boldsymbol{w} + cb))}{\sigma(g(c\boldsymbol{X}\boldsymbol{w} + cb))}\gamma + \beta && (5)\\
&= \frac{cg(\boldsymbol{X}\boldsymbol{w} + b) - c\mu(g(\boldsymbol{X}\boldsymbol{w} + b))}{|c|\sigma(g(\boldsymbol{X}\boldsymbol{w} + b))}\gamma + \beta && (6)\\
&= \frac{g(\boldsymbol{X}\boldsymbol{w} + b) - \mu(g(\boldsymbol{X}\boldsymbol{w} + b))}{\sigma(g(\boldsymbol{X}\boldsymbol{w} + b))}\gamma + \beta = f(\boldsymbol{X}; \boldsymbol{w}, b, \gamma, \beta) && (7)
\end{aligned}$$

B Proof Inverse Proportional Gradients

To show that the gradients scale inversely proportional to the parameter norm, we can first write

$$\begin{aligned} f(\boldsymbol{X}; c\boldsymbol{w}, cb, \gamma, \beta) &= \frac{g(c\boldsymbol{X}\boldsymbol{w} + cb) - \mu(g(c\boldsymbol{X}\boldsymbol{w} + cb))}{\sigma(g(c\boldsymbol{X}\boldsymbol{w} + cb))}\gamma + \beta \quad (8) \\ &= \frac{g(c\boldsymbol{X}\boldsymbol{w} + cb)}{\sigma(g(c\boldsymbol{X}\boldsymbol{w} + cb))}\gamma - \frac{\mu(g(c\boldsymbol{X}\boldsymbol{w} + cb))}{\sigma(g(c\boldsymbol{X}\boldsymbol{w} + cb))}\gamma + \beta. \quad (9) \\ & \quad (10) \end{aligned}$$

As the gradient of the weights is not backpropagated through the mean and standard deviation, we have

$$\begin{aligned} \nabla_w f(\boldsymbol{X}; c\boldsymbol{w}, cb, \gamma, \beta) &= \frac{g'(c\boldsymbol{X}\boldsymbol{w} + cb)X}{|c|\sigma(g(\boldsymbol{X}\boldsymbol{w} + b))}\gamma. \quad (11) \\ & \quad (12) \end{aligned}$$

The gradient of the bias can be computed analogously

$$\begin{aligned} \nabla_b f(\boldsymbol{X}; c\boldsymbol{w}, cb, \gamma, \beta) &= \frac{g'(c\boldsymbol{X}\boldsymbol{w} + cb)}{|c|\sigma(g(\boldsymbol{X}\boldsymbol{w} + b))}\gamma. \quad (13) \\ & \quad (14) \end{aligned}$$

C Hyperparameters

Table 1 gives an overview of the hyperparameters that were used for each algorithm that was considered in this work.

Table 1: Hyperparameters

Hyperparameter	**CrossQ**	**CrossQ + WN**	**Simba**	**SRSAC**	**BRO**
Critic learning rate	0.0003	0.0003	0.0001	0.0003	0.0003
Critic hidden dim	512	512	512	256	512
Actor learning rate	0.0003	0.0003	0.0001	0.0003	0.0003
Actor hidden dim	256	256	128	256	256
Initial temperature	1.0	1.0	0.01	1.0	1.0
Temperature learning rate	0.0001	0.0001	0.0001	0.0003	0.0003
Target entropy	$\lvert A\rvert /2$	$\lvert A\rvert /2$	$\lvert A\rvert /2$	$\lvert A\rvert$	$\lvert A\rvert$
Target network momentum	0.005	0.005	0.005	0.005	0.005
UTD	1,2,5,10,20	1,5	2,8	32	2,10
Number of critics	2	2	1	2	1
Action repeat	2	2	2	2	1
Discount	0.99 (DMC) 0.95 (Myo)	0.99 (DMC) 0.95 (Myo)	0.99 (DMC) 0.95 (Myo)	0.99 (DMC) 0.95 (Myo)	0.99 (DMC) 0.99 (Myo)
Optimizer	Adam	AdamW	AdamW	Adam	AdamW
Optimizer momentum (β_1, β_2)	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)
Policy delay	3	3	1	1	1
Warmup transitions	5000	5000	5000	10000	10000
AdamW weight decay critic	0.0	0.01	0.01	0.0	0.0001
AdamW weight decay actor	0.0	0.01	0.01	0.0	0.0001
AdamW weight decay temperature	0.0	0.0	0.0	0.0	0.0
Batch Normalization momentum	0.99	0.99	N/A	N/A	N/A
Reset Interval of networks	N/A	N/A	N/A	every 80k steps	*at 15k, 50k, 250k, 500k and 750k steps*
Batch Size	256	256	256	256	128

D Aggregated learning curves

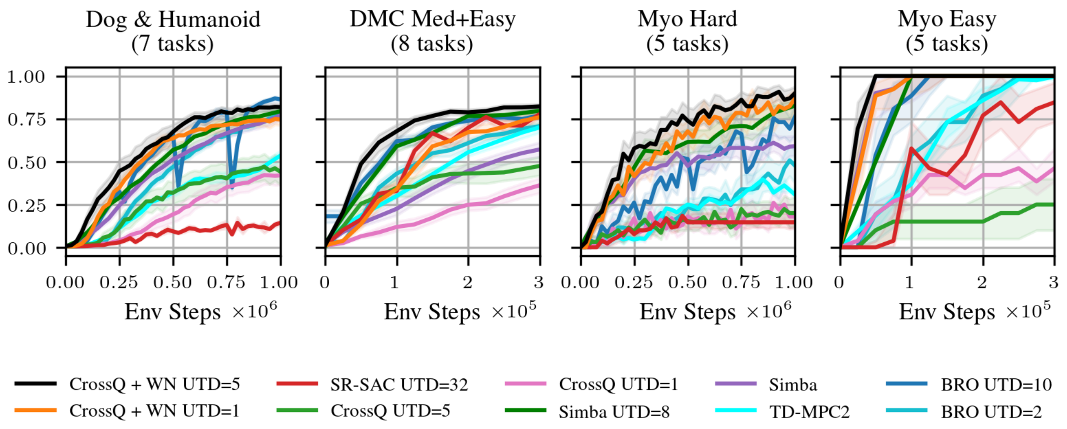

Figure 6: *CrossQ* WN UTD *scaling behavior.* We plot the IQM return and 90% stratified bootstrapped confidence intervals for different UTD ratios. The results are aggregated over 15 DMC environments and 10 random seeds each according to Agarwal et al. [1]. The sample efficiency scales reliably with increasing UTD ratios.

E Per environment learning curves

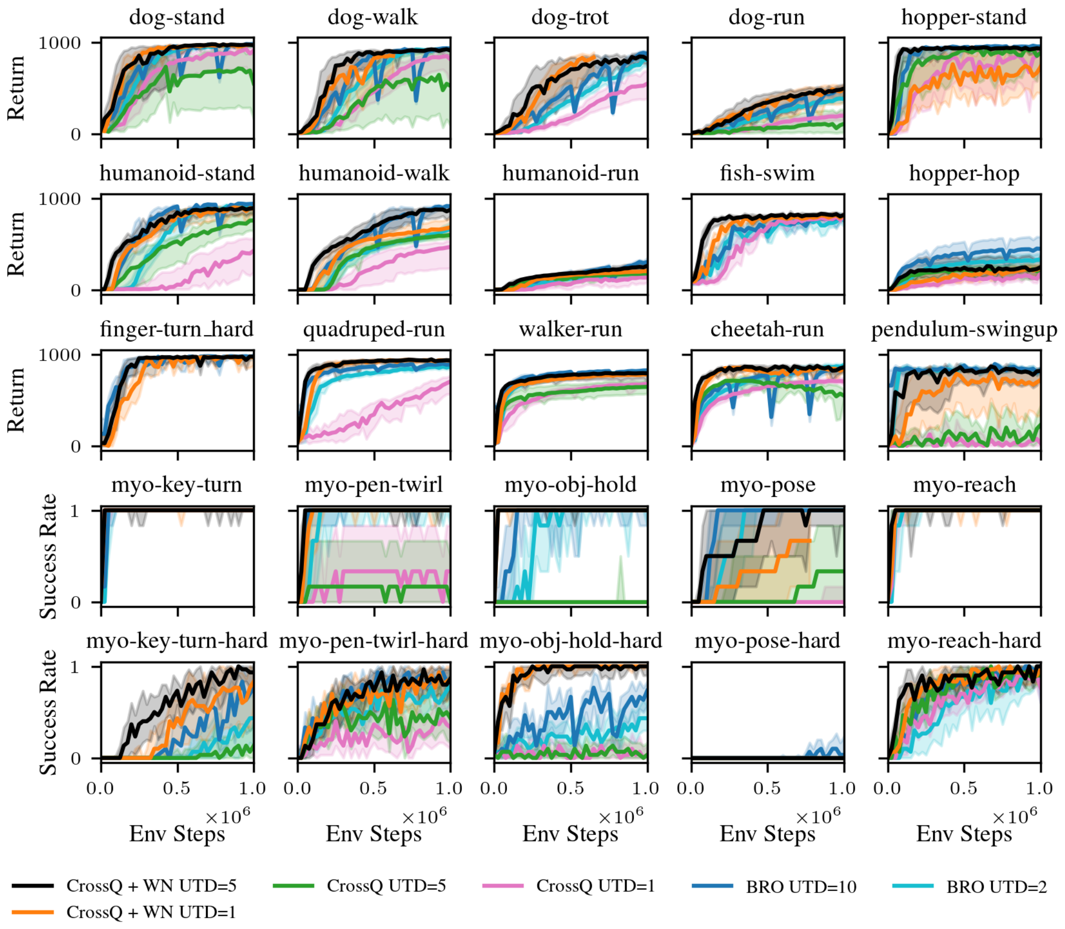

Figure 7: *CrossQ* WN + UTD$=5$ *against baselines.* We compare our proposed CrossQ + WN UTD$=5$ against two baselines, BRO [31] and SR-SAC UTD$=32$. Results are reported on all 15 DMC and 10 MyoSuite tasks. We plot the IQM and 90% stratified bootstrapped confidence intervals over 10 random random seeds. Our proposed approach proves competitive to BRO and outperforms the CrossQ baseline. We want to note that our approach achieves this performance without requiring any parameter resetting or additional exploration policies.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We have made sure that our claims reflect our contributions.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We have a dedicated conclusion and limitations discussion at the end of the paper.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: Our theory is adopted from prior work and the proofs are provided in the appendix.

Guidelines:

- The answer NA 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 included information about implementation details and hyperparameters. To aid reproducibility, we plan to release the code together with the camera-ready version of the paper.

   Guidelines:

   - The answer NA 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**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: At the current time we do not provide the code, however, we already provide all implementation details in the paper. We plan to release the code together with the publication of the paper.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We have parts explaining implementation details and choice of hyperparameters in detail.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer:[Yes]

   Justification: We provide IQM and 90$ stratified bootstrap confidence intervals on all our results (if not stated otherwise). Results are aggregated over multiple environments and 10 seeds each.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We reference the type of GPUs used for the experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: As this paper did not involve human participants or real-world datasets or experiments, there were no ethical concerns with the research.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [NA]

    Justification: The paper examines performance improvements of existing actor-critic methods. As actor-critic methods already enjoy a long history, there is no additional societal impact with this research contribution.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA 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).

11. **Safeguards**

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

    Answer: [NA] ,

    Justification: [NA]

    Guidelines:

    - The answer NA 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.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cite and link to the codebase which our code is based upon.

    Guidelines:

    - The answer NA 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.
    - 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.

13. **New assets**

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

    Answer: [NA]

    Justification: We do not provide any new assets.

    Guidelines:

    - The answer NA 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.

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

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

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA 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.

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

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

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: [NA]

    Guidelines: [NA]

    - The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.