Bigger, Better, Faster: Human-level Atari with human-level efficiency

Max Schwarzer [*123] **Johan Obando-Ceron** [*123] **Aaron Courville** [23] **Marc G. Bellemare** [123]
Rishabh Agarwal [†123] **Pablo Samuel Castro** [†1]

Abstract

We introduce a value-based RL agent, which we call BBF, that achieves super-human performance in the Atari 100K benchmark. BBF relies on scaling the neural networks used for value estimation, as well as a number of other design choices that enable this scaling in a sample-efficient manner. We conduct extensive analyses of these design choices and provide insights for future work. We end with a discussion about updating the goal-posts for sample-efficient RL research on the ALE. **We make our code and data publicly available.**

1. Introduction

Deep reinforcement learning (RL) has been central to a number of successes including playing complex games at a human or super-human level, such as OpenAI Five (Berner et al., 2019), AlphaGo (Silver et al., 2016), and AlphaStar (Vinyals et al., 2019), controlling nuclear fusion plasma in a tokomak (Degrave et al., 2022), and integrating human feedback for conversational agents (Ouyang et al., 2022). The success of these RL methods has relied on large neural networks and an enormous number of environment samples to learn from – a human player would require tens of thousands of years of game play to gather the same amount of experience as OpenAI Five or AlphaGo. It is plausible that such large networks are necessary for the agent's value estimation and/or policy to be expressive enough for the environment's complexity, while large number of samples might be needed to gather enough experience so as to determine the long-term effect of different action choices as well as train such large networks effectively. As such, obtaining human-level sample efficiency with deep RL remains an outstanding goal.

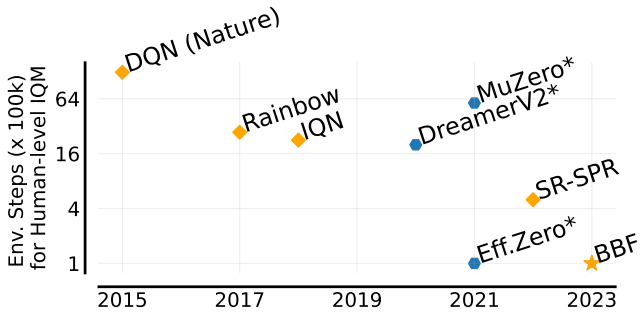

Figure 1: **Environment samples to reach human-level performance**, in terms of IQM (Agarwal et al., 2021b) over 26 games. Our proposed model-free agent, BBF, results in 5× improvement over SR-SPR (D'Oro et al., 2023) and at least 16× improvement over representative model-free RL methods, including DQN (Mnih et al., 2015b), Rainbow (Hessel et al., 2017) and IQN (Dabney et al., 2018). To contrast with the sample-efficiency progress in model-based RL, we also include DreamerV2 (Hafner et al., 2020), MuZero Reanalyse (Schrittwieser et al., 2021) and EfficientZero (Ye et al., 2021).

Although advances in modern hardware enable using large networks, in many environments it may be challenging to scale up the number of environment samples, especially for real-world domains such as healthcare or robotics. While approaches such as offline RL leverage existing datasets to reduce the need for environment samples (Agarwal et al., 2020), the learned policies may be unable to handle distribution shifts when interacting with the real environment (Levine et al., 2020) or may simply be limited in performance without online interactions (Ostrovski et al., 2021).

Thus, as RL continues to be used in increasingly challenging and sample-scarce scenarios, the need for scalable yet sample-efficient online RL methods becomes more pressing. Despite the variability in problem characteristics making a one-size-fits-all solution unrealistic, there are many insights that may transfer *across* problem domains. As such, methods that achieve "state-of-the-art" performance on established benchmarks can provide guidance and insights for others wishing to integrate their techniques.

In this vein, we focus on the Atari 100K benchmark (Kaiser

[*]Equal contribution [†]Equal advising [1]Google DeepMind [2]Mila [3]Université de Montréal. Correspondence to: Max Schwarzer <MaxA.Schwarzer@gmail.com>, Johan Obando-Ceron <jobando0730@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

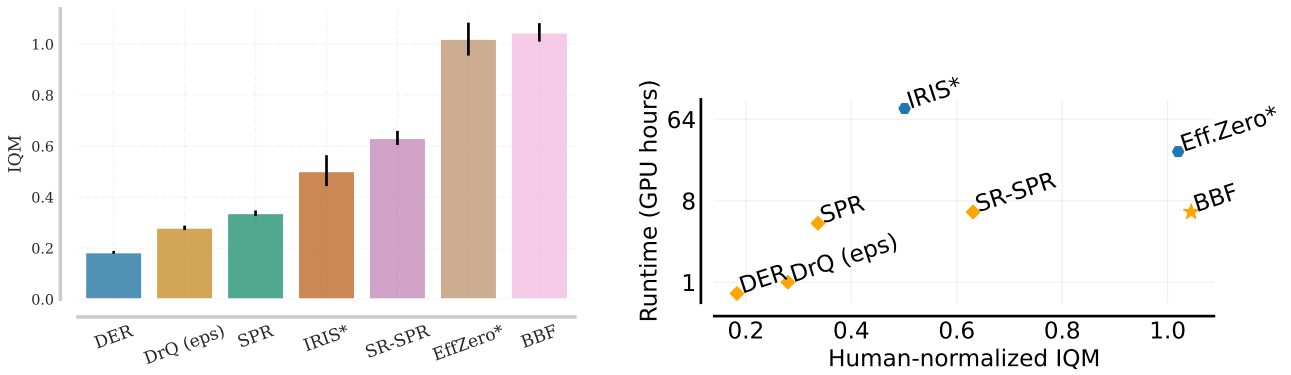

Figure 2: **Comparing Atari 100K performance and computational cost** of our model-free BBF agent to model-free SR-SPR (D'Oro et al., 2023), SPR (Schwarzer et al., 2021), DrQ (eps) (Kostrikov et al., 2020) and DER (Van Hasselt et al., 2019) as well as model-based* EfficientZero (Ye et al., 2021) and IRIS (Micheli et al., 2023). **(Left)** BBF achieves higher performance than all competitors as measured by interquartile mean human-normalized over 26 games. Error bars show 95% bootstrap CIs. **(Right)** Computational cost *vs.* Performance, in terms of human-normalized IQM over 26 games. BBF results in $2\times$ improvement in performance over SR-SPR with nearly the same computational-cost, while results in similar performance to model-based EfficientZero with at least $4\times$ reduction in runtime. For measuring runtime, we use the total number of A100 GPU hours spent per environment.

et al., 2020), a well-known benchmark where agents are constrained to roughly 2 hours of game play, which is the amount of practice time the professional tester was given before human score evaluation. While human-level efficiency has been obtained by the model-based EfficientZero agent (Ye et al., 2021), it has remained elusive for model-free RL agents. To this end, we introduce BBF, a model-free RL agent that achieves super-human performance – interquartile mean (Agarwal et al., 2021b) human-normalized score above 1.0 – while being much more computationally efficient than EfficientZero (Figure 2). Achieving this level of performance required a larger network than the decade-old 3-layer CNN architecture (Mnih et al., 2013), but as we will discuss below, scaling network size is not sufficient on its own. We discuss and analyze the various techniques and components that are necessary to train BBF successfully and provide guidance for future work to build on our findings. Finally, we propose moving the goalpost for sample-efficient RL research on the ALE.

2. Background

The RL problem is generally described as a Markov Decision Proces (MDP) (Puterman, 2014), defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of available actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$[1] is the transition function, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. Agent behavior in RL can be formalized by a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, which maps states to a distribution of actions. The *value* of π when starting from $s \in \mathcal{S}$ is defined as the discounted sum of expected rewards: $V^\pi(s) := \mathbb{E}_{\pi,\mathcal{P}}\left[\sum_{t=0}^{\infty} \gamma^t r\left(s_t, a_t\right)\right]$, where $\gamma \in [0, 1)$ is a discount factor that encourages the agent to accumulate rewards sooner rather than later. The goal of an RL agent is to find a policy π^* that maximizes this sum: $V^{\pi^*} \geq V^\pi$ for all π.

While there are a number of valid approaches (Sutton & Barto, 1998), in this paper we focus on model-free *value-based* methods. Common value-based algorithms approximate the Q^*-values, defined via the Bellman recurrence: $Q^*(s, a) := R(s, a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(s,a)}[\max_{a' \in \mathcal{A}} Q^*(s', a')]$. The optimal policy π^* can then be obtained from the optimal state-action value function Q^* as $\pi^*(x) := \max_{a \in \mathcal{A}} Q^*(s, a)$. A common approach for learning Q^* is the method of temporal differences, optimizing the Bellman temporal difference:

$$\left(r\left(s_t, a_t\right) + \gamma \max_{a_{t+1}} Q\left(s_{t+1}, a_{t+1}\right)\right) - Q\left(s_t, a_t\right).$$

We often refer to $\left(r\left(s_t, a_t\right) + \gamma \max_{a_{t+1}} Q\left(s_{t+1}, a_{t+1}\right)\right)$ as the *Bellman target*.

Mnih et al. (2015a) introduced the agent DQN by combining temporal-difference learning with deep networks, and demonstrated its capabilities in achieving human-level performance on the Arcade Learning Environment (ALE) (Bellemare et al., 2013). They used a network consisting of 3 convolutional layers and 2 fully connected layers, parameterized by θ, to approximate Q (denoted as Q_θ). We will refer to this architecture as the CNN architecture. Most of the work in value-based agents is built on the original DQN

[1] $\Delta(S)$ denotes a distribution over the set S.

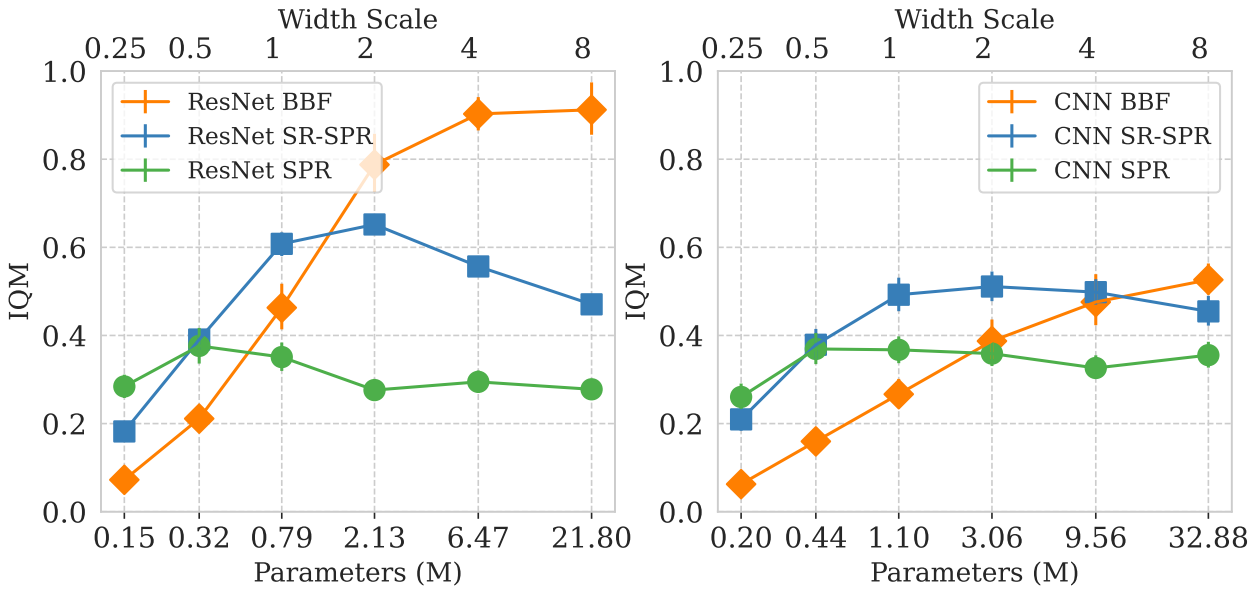

Figure 3: **Scaling network widths for both ResNet and CNN architectures**, for BBF, SR-SPR and SPR at replay ratio 2, with an Impala-based ResNet **(left)** and the standard 3-layer CNN (Mnih et al., 2015b) **(right)**. We report interquantile mean performance with error bars indicating 95% confidence intervals. On the x-axis we report the approximate parameter count of each configuration as well as its width relative to the default (width scale = 1).

agent, and we discuss a few of these advances below which are relevant to our work.

Hessel et al. (2018) combined six components into a single agent they called *Rainbow*: prioritized experience (Schaul et al., 2016), n-step learning (Sutton, 1988), distributional RL (Bellemare et al., 2017), double Q-learning (van Hasselt et al., 2016), dueling architecture (Wang et al., 2016) and NoisyNets (Fortunato et al., 2018b). Hessel et al. (2018) and Ceron & Castro (2021) both showed that Multi-step learning is one of the most crucial components of Rainbow, in that removing it caused a large drop in performance.

In n-step learning, instead of computing the temporal difference error using a single-step transition, one can use n-step targets instead (Sutton, 1988), where for a trajectory $(s_0, a_0, r_0, s_1, a_1, \cdots)$ and update horizon n: $R_t^{(n)} := \sum_{k=0}^{n-1} \gamma^k r_{t+k+1}$, yielding the multi-step temporal difference: $R_t^{(n)} + \gamma^n \max_{a'} Q_\theta(s_{t+n}, a') - Q_\theta(s_t, a_t)$.

Most modern RL algorithms store past experiences in a *replay buffer* that increases sample efficiency by allowing the agent to use samples multiple times during learning, and to leverage modern hardware such as GPUs and TPUs by training on sampled mini-batches. An important design parameter is the **replay ratio**, the ratio of learning updates to online experience collected (Fedus et al., 2020a). It is worth noting that DQN uses a replay ratio of 0.25 (4 environment interactions for every learning update), while some sample-efficient agents based on Rainbow use a value of 1.

Nikishin et al. (2022) showed that the networks used by deep RL agents have a tendency to overfit to early experience, which can result in sub-optimal performance. They proposed a simple strategy consisting of periodically resetting the parameters of the final layers of DQN-based agents to counteract this. Building on this promising work, D'Oro et al. (2023) added a shrink-and-perturb technique for the parameters of the convolutional layers, and showed that this allowed them to scale the replay ratio to values as high as 16, with no performance degradation.

3. Related Work

Sample-Efficient RL on ALE: Sample efficiency has always been an import aspect of evaluation in RL, as it can often be expensive to interact with an environment. Kaiser et al. (2020) introduced the Atari 100K benchmark, which has proven to be useful for evaluating sample-efficiency, and has led to a number of recent advances.

Kostrikov et al. (2020) use data augmentation to design a sample-efficient RL method, DrQ, which outperformed prior methods on Atari 100K. Data-Efficient Rainbow (DER) (Van Hasselt et al., 2019) and DrQ(ϵ) (Agarwal et al., 2021b) simply modified the hyperparameters of existing model-free algorithms to exceed the performance of existing methods without any algorithmic innovation.

Schwarzer et al. (2021) introduced SPR, which builds on Rainbow (Hessel et al., 2017) and uses a self-supervised tem-

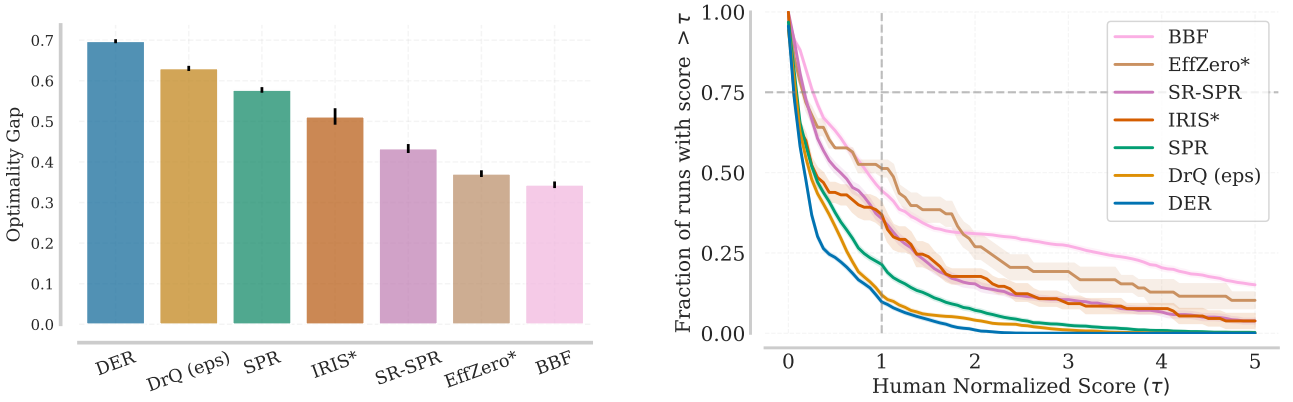

Figure 4: (**Left**). Optimality Gap (lower is better) for BBF at replay ratio 8 and competing methods on Atari 100K. Error bars show 95% CIs. BBF, has a lower optimality gap than any competing algorithm, indicating that it comes closer on average to achieving human-level performance across all tasks. **(Right)** Performance profiles showing the distribution of scores across all runs and 26 games at the end of training (higher is better). Area under an algorithm's profile is its mean performance while τ value where it intersects y = 0.75 shows its 25th percentile performance. BBF has better performance on challenging tasks that may not otherwise contribute to IQM or median performance.

poral consistency loss based on BYOL (Grill et al., 2020) combined with data augmentation. SR-SPR (Schwarzer et al., 2021) combines SPR with periodic network resets to achieve state-of-the-art performance on the 100K benchmark. Ye et al. (2021) used a self-supervised consistency loss similar to SPR (Chen & He, 2021).

EfficientZero (Ye et al., 2021), an efficient variant of MuZero (Schrittwieser et al., 2020), learns a discrete-action latent dynamics model from environment interactions, and selects actions via lookahead MCTS in the latent space of the model. Micheli et al. (2023) introduce IRIS, a data-efficient agent that learns in a world model composed of an autoencoder and an auto-regressive Transformer.

Scaling in Deep RL: Deep neural networks are useful for extracting features from data relevant for various downstream tasks. Recently, there has been interest in the scaling properties of neural network architectures, as scaling model size has led to commensurate performance gains in applications ranging from language modelling to computer vision.

Based on those promising gains, the deep RL community has begun to investigate the effect of increasing the model size of the function approximator. Sinha et al. (2020) and Ota et al. (2021) explore the interplay between the size, structure, and performance of deep RL agents to provide intuition and guidelines for using larger networks. Kumar et al. (2022) find that with ResNets (up to 80 million parameter networks) combined with distributional RL and feature normalization, offline RL can exhibit strong performance that scales with model capacity. Taiga et al. (2023) show that generalization capabilities on the ALE benefit from higher capacity networks, such as ResNets. Cobbe et al. (2020) and Farebrother et al. (2023) demonstrate benefits when scaling the number of features in each layer of the ResNet architecture used by Impala (Espeholt et al., 2018), which motivated the choice of feature width scaling in this work. Different from these works, our work focus on improving sample-efficiency in RL as opposed to offline RL or improving generalization in RL.

In the context of online RL, Hafner et al. (2023) demonstrate that increased dynamics model size, trained via supervised learning objectives, leads to monotonic improvements in the agent's final performance. Recently, AdA (Team et al., 2023) scales transformer encoder for a Muesli agent up to 265M parameters. Interestingly, AdA required distillation from smaller models to bigger models to achieve this scaling, in the spirit of reincarnating RL (Agarwal et al., 2022). However, it is unclear whether findings from above papers generalize to scaling typical value-based deep RL methods in sample-constraint settings, which we study in this work.

4. Method

The question driving this work is: *How does one scale networks for deep RL when samples are scarce?* To investigate this, we focus on the well-known Atari 100K benchmark (Kaiser et al., 2020), which includes 26 Atari 2600 games of diverse characteristics, where the agent may perform only 100K environment steps, roughly equivalent to two hours of human gameplay[2]. As we will see, näively scaling networks can rarely maintain performance, let alone improve it.

The culmination of our investigation is the Bigger, Better,

[2] 100k steps (400k frames) at 60 FPS is 111 minutes.

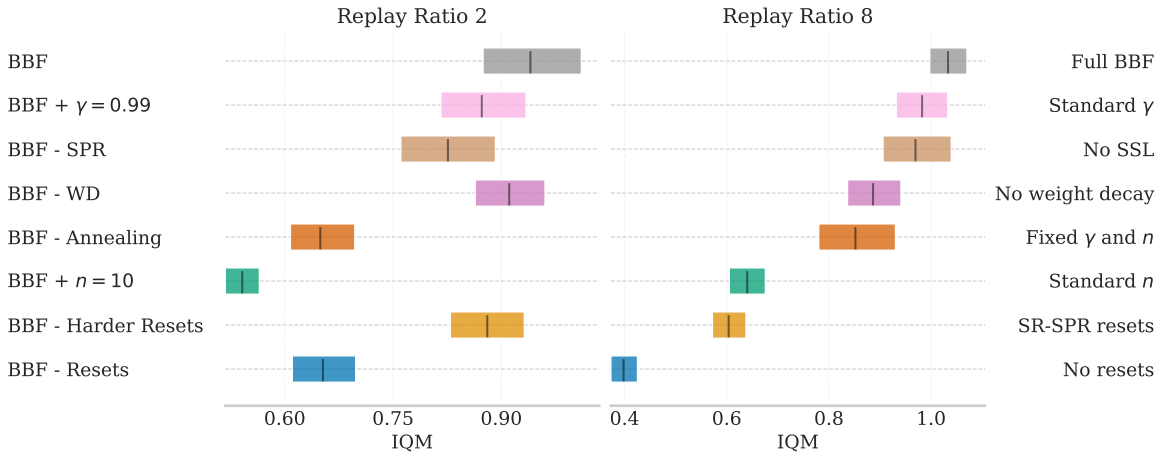

Figure 5: **Evaluating the impact of removing the various components that make up BBF with RR=2 and RR=8**. Reporting interquantile mean averaged over the 26 Atari 100k games, with 95% CIs over 15 independent runs.

Faster agent, or BBF in short, which achieves super-human performance on Atari 100K with about 6 hours on single GPU. Figure 2 demonstrates the strong performance of BBF relative to some of the best-performing Atari 100K agents: EfficientZero (Ye et al., 2021), SR-SPR (D'Oro et al., 2023), and IRIS (Micheli et al., 2023). BBF consists of a number of components, which we discuss in detail below.

Our implementation is based on the Dopamine framework (Castro et al., 2018) and uses mostly already previously-released components. For evaluation, we use rliable (Agarwal et al., 2021b) and in particular, the interquartile mean (IQM) metric, which is the average score of the middle 50% runs combined across all games and seeds.

Base agent. BBF uses a modified version of the recently introduced SR-SPR agent (D'Oro et al., 2023). Through the use of periodic network resets, SR-SPR is able to scale up its replay ratio (RR) to values as high as 16, yielding better sample efficiency. For BBF, we use RR=8 in order to balance the increased computation arising from our large network. Note that this is still very high relative to existing Atari agents – Rainbow and its data-efficient variant DER (Van Hasselt et al., 2019) use RR=0.25 and 1, respectively.

As we expect that many users will not wish to pay the computational costs of running at replay ratio 8, we also present results for BBF and ablations at replay ratio 2 (matching SPR). For all experiments we state which replay ratio is being used in the captions.

Harder resets. The original SR-SPR agent (D'Oro et al., 2023) used a shrink-and-perturb method for the convolutional layers where parameters were only perturbed 20% of the way towards a random target, while later layers were fully reset to a random initialization. An interesting result of our investigation is that using harder resets of the convolutional layers yields better performance. In our work, we move them 50% towards the random target, resulting in a stronger perturbation and improving results (see Figure 5). This may be because larger networks need more regularization, as we find that they reduce loss faster (Figure A.1).

Larger network. Scaling network capacity is one of the motivating factors for our work. As such, we adopt the Impala-CNN (Espeholt et al., 2018) network, a 15-layer ResNet, which has previously led to substantial performance gains over the standard 3-layer CNN architecture in Atari tasks where large amounts of data are available (Agarwal et al., 2022; Schmidt & Schmied, 2021). Additionally, BBF scales the width of each layer in Impala-CNN by 4×. In Figure 3, we examine how the performance of SPR, SR-SPR and BBF varies with different choices of scaling width, for both the ResNet and original CNN architectures. Interestingly, although the CNN has roughly 50% more parameters than the ResNet at each scale level, the ResNet yields better performance at all scaling levels for both SR-SPR and BBF.

What stands out from Figure 3 is that BBF's performance continues to grow as width is increased, whereas SR-SPR seems to peak at 1-2× (for both architectures). Given that ResNet BBF performs comparably at 4× and 8×, we chose 4× to reduce the computational burden. While reducing widths beyond this could further reduce computational costs, this comes at the cost of increasingly sharp reductions in performance for all methods tested.

Receding update horizon. One of the surprising components of BBF is the use of an update horizon (n-step) that decreases exponentially from 10 to 3 over the first 10K gradient steps following each network reset. Given that we follow the schedule of D'Oro et al. (2023) and reset every

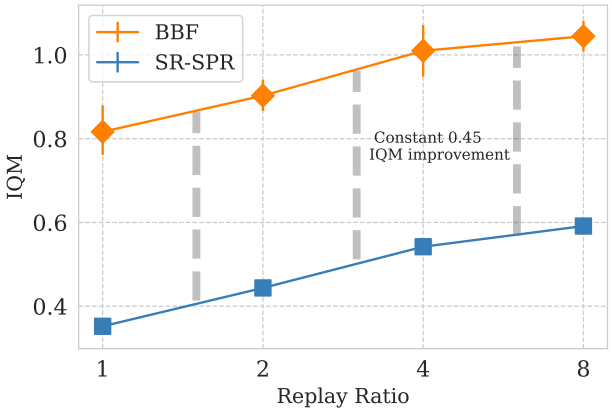

Figure 6: **Comparison of BBF and SR-SPR across different replay ratios**. We report IQM with 95% CIs for each point. BBF achieves an almost-constant 0.45 IQM improvement over SR-SPR at each replay ratio.

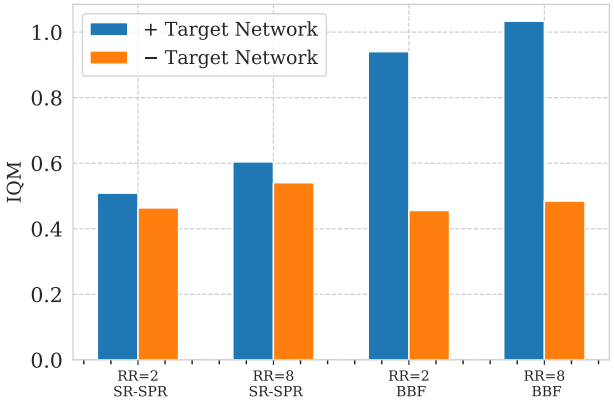

Figure 7: **Comparison of BBF and SR-SPR at replay ratios 2 and 8 with and without EMA target networks**. Human-normalized IQM on the 26 Atari 100k games.

40k gradient steps, the annealing phase is always 25% of training, regardless of the replay ratio. As can be seen in Figure 5, this yields a much stronger agent than using a fixed value of $n = 3$, which is default for Rainbow, or $n = 10$, which is typically used by Atari 100K agents like SR-SPR.

Our n-step schedule is motivated by the theoretical results of Kearns & Singh (2000) – larger values of n-step leads to faster convergence but to higher asymptotic errors with respect to the optimal value function. Thus, selecting a fixed value of n corresponds to a choice between having either rapid convergence to a worse asymptote, or slower convergence to a better asymptote. As such, our exponential annealing schedule closely resembles the optimal decreasing schedule for n-step derived by Kearns & Singh (2000).

Increasing discount factor. Motivated by findings that increasing the discount factor γ during learning improves performance (François-Lavet et al., 2015), we increase γ from γ_1 to γ_2, following the same exponential schedule as for the update horizon. Note that increasing γ has the effect of progressively giving more weights to delayed rewards. We choose $\gamma_1 = 0.97$, slightly lower than the typical discount used for Atari, and $\gamma_2 = 0.997$ as it is used by MuZero (Schrittwieser et al., 2021) and EfficientZero (Ye et al., 2021). As with the update horizon, Figure 5 demonstrates that this strategy outperforms using a fixed value.

Weight decay. We incorporate weight decay in our agent to curb statistical overfitting, as BBF is likely to overfit with its high replay ratio. To do so, we use the AdamW optimizer (Loshchilov & Hutter, 2019) with a weight decay value of 0.1. Figure 5 suggests the gains from adding weight decay are significant and increase with replay ratio, indicating that the regularizing effects of weight decay enhance replay ratio scaling with large networks.

Removing noisy nets. Finally, we found that NoisyNets (Fortunato et al., 2018a), used in the original SPR (Schwarzer et al., 2021) and SR-SPR, did not improve performance. This could be due to NoisyNets causing overexploration due to increased policy churn (Schaul et al., 2022) from added noise during training, or due to added variance in optimization, and we leave investigation to future work. Removing NoisyNets results in large computational and memory savings, as NoisyNets creates duplicate copies of the weight matrices for the final two linear layers in the network, which contain the vast majority of all parameters: turning on NoisyNets increases the FLOPs per forward pass and the memory footprint by a factor of 2.5× and 1.6×, respectively, which both increases runtime and reduces the number of training runs that can be run in parallel on a single GPU. Removing NoisyNets is thus critical to allowing BBF to achieve reasonable compute efficiency despite its larger networks. We found that this decision had no significant impact on task performance (see Figure A.2 in appendix).

5. Analysis

In light of the importance of BBF's components, we discuss possible consequences of our findings for other algorithms.

The importance of self-supervision. One unifying aspect of the methods compared in Figure 2 is that they all use some form of self-supervised objective. In sample-constrained scenarios, like the one considered here, relying on more than the temporal-difference backups is likely to improve learning speed, provided the self-supervised losses are consistent with the task at hand. We test this by removing the SPR objective (inherited from SR-SPR) from BBF, and observe a substantial performance degredation (see Figure 5). It is worth noting that EfficientZero uses a self-supervised objective that is extremely similar to SPR, a striking commonality between BBF and EfficientZero.

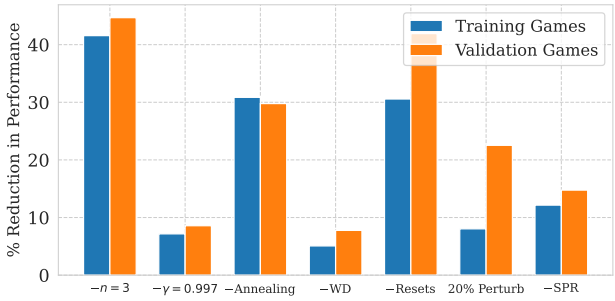

Figure 8: **Validating BBF design choices at RR=2 on 29 unseen games**. While Atari 100K training set consists of 26 games, we evaluate the performance of various components in BBF on 29 validation games in ALE that are not in Atari 100K. Interestingly, all BBF components lead to a large performance improvement on unseen games. Specifically, we measure the % decrease in human-normalized IQM performance relative to the full BBF agent at RR=2.

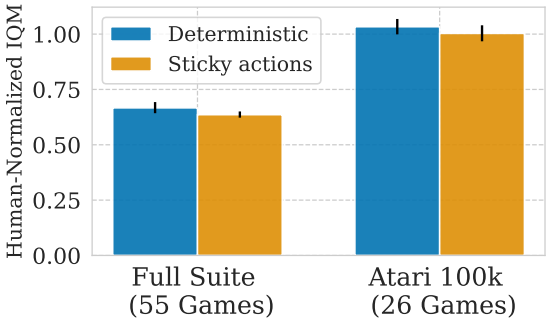

Figure 9: **Evaluating BBF on ALE with and w/o sticky actions**. We report IQM human-normalized performance at replay ratio 8 on 26 games in Atari 100K as well the full set of 55 games in ALE. While performance on the full set of 55 games is lower, neither setting has its performance significantly affected by sticky actions.

Sample efficiency via more gradient steps. The original DQN agent (Mnih et al., 2015b) has a replay ratio of 0.25, which means a gradient update is performed only after every 4 environment steps. In low-data regimes, it is more beneficial to perform more gradient steps, although many algorithms cannot benefit from this without additional regularization (D'Oro et al., 2023). As Figure 6 confirms, performance of BBF grows with increasing replay ratio in the same manner as its base algorithm, SR-SPR. More strikingly, we observe a *linear* relationship between the performance of BBF and SR-SPR across all replay ratios, with BBF performing roughly 0.45 IQM above SR-SPR. While the direction of this relationship is intuitive given the network scaling introduced by BBF, its linearity is unexpected, and further investigation is needed to understand the nature of the interaction between replay ratio and network scaling.

One interesting comparison to note is that, although EfficientZero uses a replay ratio of 1.2, they train with a batch size that is 8 times larger than ours. Thus, their *effective* replay ratio is comparable to ours.

The surprising importance of target networks Many prior works on Atari 100k, such as DrQ and SPR (Kostrikov et al., 2020; Schwarzer et al., 2021) chose not to use target networks, seeing them unnecessary or an impediment to sample efficiency. Later, D'Oro et al. (2023) re-introduced an exponential moving average target network, used both for training and action selection, and found that it improved performance somewhat, especially at high replay ratios. With network scaling, however, using a target network becomes a critical, but easy-to-overlook, component of the algorithm at all replay ratios (see Figure 7).

Reset Strength Increasing the replay ratio is in general challenging, as explored by Fedus et al. (2020b) and Kumar et al. (2020). Periodic resetting, as suggested by Nikishin et al. (2022) and D'Oro et al. (2023), has proven effective to enable scaling to larger replay ratios, quite possibly a result of reduced overfitting. This is confirmed in Figure 5, where the importance of resets is clear. Further, Figure 5 and Figure 8 demonstrate the added benefit of more aggressive perturbations, relative to SR-SPR.

Scale is not enough on its own. The naïve approach of simply scaling the capacity of the CNN used by SR-SPR turns out to be insufficient to improve performance. Instead, as Figure 3 shows, the performance of SR-SPR collapses as network size increases. As discussed in section 4, it is interesting to observe that the smaller Impala-CNN ResNet (as measured by number of parameters and FLOPs) yields stronger performance at all width scales.

Computational efficiency. As machine learning methods become more sophisticated, an often overlooked metric is their computational efficiency. Although EfficientZero trains in around 8.5 hours, it requires about 512 CPU cores and 4 distributed GPUs. IRIS uses half of an A100 GPU for a week per run. SR-SPR, at its highest replay ratio of 16, uses 25% of an A100 GPU and a single CPU for roughly 24 hours. Our BBF agent at replay ratio 8 takes only 10 hours with a single CPU and half of an A100 GPU. Thus, measured by GPU-hours, BBF provides the best trade-off between performance and compute (see Figure 2).

6. Revisiting the Atari 100k benchmark

A natural question is whether there is any value in continuing to use the Atari 100K benchmark, given that both EfficientZero and BBF are able to achieve human-level perfor-

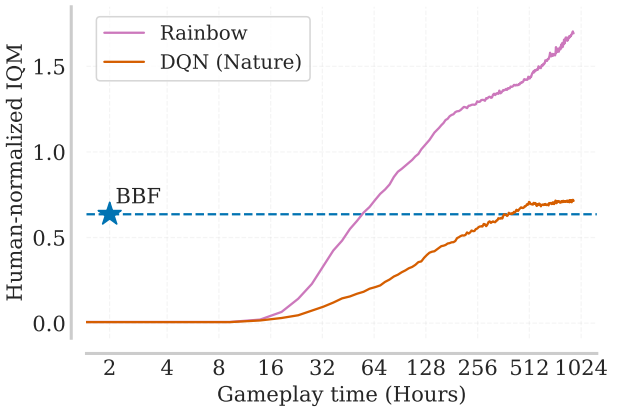

Figure 10: **Sample efficiency progress on ALE**, measured via human-normalized IQM over 55 Atari games with sticky actions, as a function of amount of human game play hours, with BBF at RR=8. Shaded regions show 95% CIs.

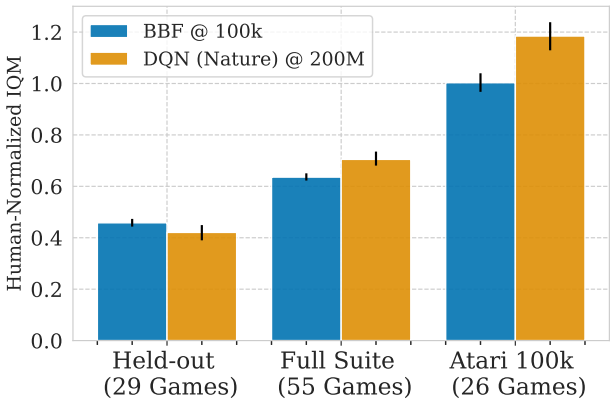

Figure 11: **Comparing performance on the 29 unseen games to the 26 Atari 100k games**. BBF trained with sticky actions at RR=8 for 100k steps approximately matches DQN (Nature) with 500 times more training data on each set. While we find that the 29 games not included in the Atari 100k setting are significantly harder than the 26 Atari 100k games, we see no evidence that BBF has overfitted to Atari 100k compared to DQN.

mance (IQM ≥ 1.0) in just 100K steps. When considering this, it is important to remember that IQM is an *aggregate* measure. Indeed, in the left panel of Figure 4 we can see there is still room for improvement with regards to the *optimality gap*, which measures the amount by which each algorithm fails to meet a minimum score of 1.0 (Agarwal et al., 2021b). Specifically, despite monotonic progress over the years, no agent is yet able to achieve human-level performance on all 26 games, which would yield an optimality gap of zero, without using dramatically more than two hours of data (Kapturowski et al., 2022).

Overfitting on Atari 100K. Another important consideration is that the Atari 100K benchmark uses only 26 of the 55 games from the full ALE suite, and it does not include sticky actions[3] (Machado et al., 2018), which may make tasks significantly harder. Since we extensively benchmark BBF on Atari 100K, this raises the question of whether BBF works well on unseen Atari games and with sticky actions.

Fortunately, it does. In Figure 9, we compare the performance of BBF on all 55 games with sticky actions, and show that sticky action do not significantly harm performance. We do observe that the held-out games not included in the Atari 100k set are significantly more challenging than the 26 Atari 100k games (see Figure 11) – but this is even more true for baselines such as DQN (Nature) that did not use Atari 100k. Furthermore, as shown in Figure 8, we find that BBF's design choices generally provide even more benefit on these held-out games, possibly due to their increased difficulty.

New Frontiers In fact, BBF works so well on the standard Atari setting that it is able to roughly match DQN's performance at 256 hours with only two hours of gameplay time (Figure 10). This suggests a clear new milestone for the community: can we match Rainbow's final performance with just two hours of gameplay? To facilitate future research toward this, we release scores on the set of 55 games with sticky actions, at various scales and replay ratios.

Data Scaling Prior works have indicated that many sample-efficient RL algorithms plateau in performance when trained for longer than they were originally designed for (e.g., Agarwal et al., 2022). To examine this phenomenon, we train BBF, SPR and SR-SPR at replay ratio 2 out to one million environment steps (Figure 12), keeping all parameters unchanged (including conducting resets as normal past 100k steps). We observe that SPR and SR-SPR experience stagnating performance, with SR-SPR's advantage over SPR fading by 1M steps. BBF, however, remains consistently ahead of both, matching DQN's final performance before 200k environment steps and matching Rainbow's performance at 20M environment steps by 1M steps. We note that this experiment costs only 2.5 times more than training at replay ratio 8 to 100k steps, so we encourage other researchers to run similar experiments.

Additionally, we note in Figure 13 that it is possible to compare algorithms even with extremely small amounts of data, such as 20k or 50k steps, by which point BBF at replay ratio 2 (even with sticky actions enabled) outperforms most recently proposed algorithms (Robine et al., 2023; Micheli et al., 2023; Hafner et al., 2023), which did not use sticky actions. We thus suggest that compute-constrained groups consider this setting, as training BBF at replay ratio 2 for 40k environment steps takes only half of an A100 for 1 hour.

[3]With 25% probability, the environment will execute the previous action again, instead of the agent's executed action.

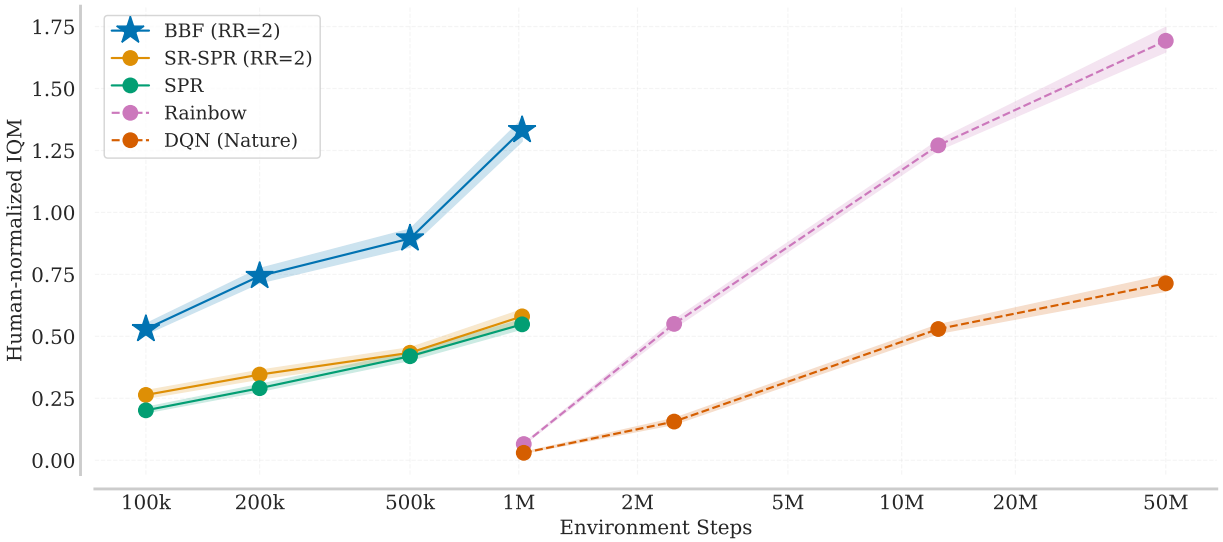

Figure 12: **Learning curves for BBF, SR-SPR and SPR at replay ratio 2**, measured via human-normalized IQM over 55 Atari games with sticky actions, as a function of number of environment interactions. Shaded regions show 95% CIs.

7. Discussion and Future Work

We introduced BBF, an algorithm that is able to achieve super-human level performance on the ALE with only 2-hours of gameplay. Although BBF is not the first to achieve this milestone, it is able to do so in a computationally efficient manner. Furthermore, BBF is able to better handle the scaling of networks and replay ratios, which are crucial for network expressivity and learning efficiency. Indeed, Figure 3 suggests that BFF is better-able to use over-parameterized networks than prior agents.

The techniques necessary to achieve this result invite a number of research questions for future work. Large replay ratios are a key element of BFF's performance, and the ability to scale them is due to the periodic resets incorporated into the algorithm. These resets are likely striking a favourable balance between catastrophic forgetting and network plasticity. An interesting avenue for future research is whether there are other mechanisms for striking this balance that perhaps are more targeted (e.g. not requiring resetting the full network, as was recently explored by Sokar et al. (2023)). We remarked on the fact that all the methods compared in Figure 2 use a form of self-supervision. Would other self-supervised losses (e.g. (Mazoure et al., 2020; Castro et al., 2021; Agarwal et al., 2021a)) produce similar results? Surprisingly, Li et al. (2022) argue that self-supervision from pixels does not improve performance; our results seem to contradict this finding.

Recent attention has shifted towards more realistic benchmarks (Fan et al., 2022) but such benchmarks exclude the majority of researchers outside certain resource-rich labs, and may require an alternative paradigm (Agarwal et al., 2022). One advantage of the Atari 100k benchmark is that, while still a challenging benchmark, it is relatively cheap compared to other benchmarks of similar complexity. However, despite its apparent saturation, scientific progress can still be made on this benchmark if we expand its scope. We hope our work provides a solid starting point for this.

Overall, we hope that our work inspires other researchers to continue pushing the frontier of sample efficiency in deep RL forward, to ultimately reach human-level performance across all tasks with human-level or superhuman efficiency.

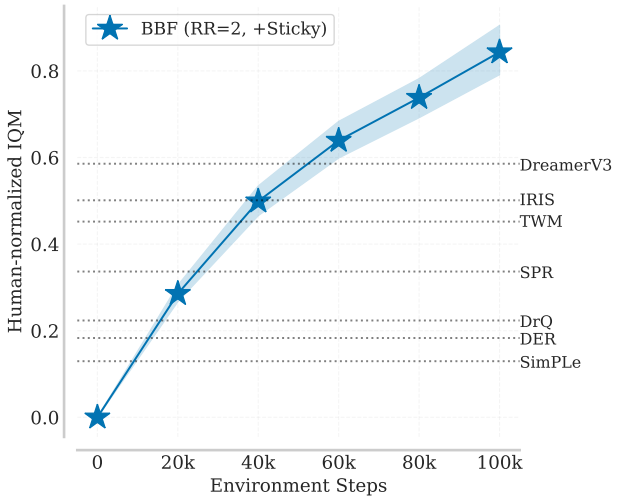

Figure 13: **IQM Human-normalized learning curve for BBF at RR=2 with sticky actions on the 26 Atari 100k games**, with final performances of many recent algorithms after they have trained for 100k steps. Even a weakened BBF outperforms all by 50k steps.

References

Agarwal, R., Schuurmans, D., and Norouzi, M. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pp. 104–114. PMLR, 2020.

Agarwal, R., Machado, M. C., Castro, P. S., and Bellemare, M. G. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. *arXiv preprint arXiv:2101.05265*, 2021a.

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., and Bellemare, M. G. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021b.

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., and Bellemare, M. G. Reincarnating reinforcement learning: Reusing prior computation to accelerate progress. In *Advances in Neural Information Processing Systems*, 2022.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bellemare, M. G., Dabney, W., and Munos, R. A distributional perspective on reinforcement learning. In *ICML*, 2017.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., et al. Jax: composable transformations of python+ numpy programs. 2018.

Castro, P. S., Moitra, S., Gelada, C., Kumar, S., and Bellemare, M. G. Dopamine: A Research Framework for Deep Reinforcement Learning. 2018. URL http://arxiv.org/abs/1812.06110.

Castro, P. S., Kastner, T., Panangaden, P., and Rowland, M. MICo: Improved representations via sampling-based state similarity for markov decision processes. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=wFp6kmQELgu.

Ceron, J. S. O. and Castro, P. S. Revisiting rainbow: Promoting more insightful and inclusive deep reinforcement learning research. In *International Conference on Machine Learning*, pp. 1373–1383. PMLR, 2021.

Chen, X. and He, K. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Cobbe, K., Hesse, C., Hilton, J., and Schulman, J. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pp. 2048–2056. PMLR, 2020.

Dabney, W., Ostrovski, G., Silver, D., and Munos, R. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pp. 1096–1105. PMLR, 2018.

Degrave, J., Felici, F., Buchli, J., Neunert, M., Tracey, B. D., Carpanese, F., Ewalds, T., Hafner, R., Abdolmaleki, A., de Las Casas, D., Donner, C., Fritz, L., Galperti, C., Huber, A., Keeling, J., Tsimpoukelli, M., Kay, J., Merle, A., Moret, J., Noury, S., Pesamosca, F., Pfau, D., Sauter, O., Sommariva, C., Coda, S., Duval, B., Fasoli, A., Kohli, P., Kavukcuoglu, K., Hassabis, D., and Riedmiller, M. A. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022. doi: 10.1038/s41586-021-04301-9. URL https://doi.org/10.1038/s41586-021-04301-9.

D'Oro, P., Schwarzer, M., Nikishin, E., Bacon, P.-L., Bellemare, M. G., and Courville, A. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OpC-9aBBVJe.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Farebrother, J., Greaves, J., Agarwal, R., Lan, C. L., Goroshin, R., Castro, P. S., and Bellemare, M. G. Proto-value networks: Scaling representation learning with auxiliary tasks. In *Submitted to The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=oGDKSt9JrZi. under review.

Fedus, W., Ramachandran, P., Agarwal, R., Bengio, Y., Larochelle, H., Rowland, M., and Dabney, W. Revisiting fundamentals of experience replay. In *International Conference on Machine Learning*, pp. 3061–3071. PMLR, 2020a.

Fedus, W., Ramachandran, P., Agarwal, R., Bengio, Y., Larochelle, H., Rowland, M., and Dabney, W. Revisiting fundamentals of experience replay. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3061–3071. PMLR, 13–18 Jul 2020b.

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Hessel, M., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., Blundell, C., and Legg, S. Noisy networks for exploration. In *International Conference on Learning Representations*, 2018a. URL https://openreview.net/forum?id=rywHCPkAW.

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., Blundell, C., and Legg, S. Noisy networks for exploration. 2018b.

François-Lavet, V., Fonteneau, R., and Ernst, D. How to discount deep reinforcement learning: Towards new dynamic strategies. *arXiv preprint arXiv:1512.02011*, 2015.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models, 2023. URL https://arxiv.org/abs/2301.04104.

Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. *arXiv preprint arXiv:1710.02298*, 2017.

Hessel, M., Modayil, J., Hasselt, H. V., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M. G., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI*, 2018.

Hunter, J. D. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(03):90–95, 2007.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., et al. Model-based reinforcement learning for atari. *International Conference on Learning Representations*, 2020.

Kapturowski, S., Campos, V., Jiang, R., Rakićević, N., van Hasselt, H., Blundell, C., and Badia, A. P. Human-level atari 200x faster. *arXiv preprint arXiv:2209.07550*, 2022.

Kearns, M. J. and Singh, S. Bias-variance error bounds for temporal difference updates. In *COLT*, pp. 142–147, 2000.

Kostrikov, I., Yarats, D., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

Kumar, A., Agarwal, R., Ghosh, D., and Levine, S. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. *arXiv preprint arXiv:2010.14498*, 2020.

Kumar, A., Agarwal, R., Geng, X., Tucker, G., and Levine, S. Offline q-learning on diverse multi-task data both scales and generalizes, 2022. URL https://arxiv.org/abs/2211.15144.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *CoRR*, abs/2005.01643, 2020. URL https://arxiv.org/abs/2005.01643.

Li, X., Shang, J., Das, S., and Ryoo, M. S. Does self-supervised learning really improve reinforcement learning from pixels? In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=fVslVNBfjd8.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., and Bowling, M. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *J. Artif. Int. Res.*, 61(1): 523–562, jan 2018. ISSN 1076-9757.

Mazoure, B., Tachet des Combes, R., Doan, T. L., Bachman, P., and Hjelm, R. D. Deep reinforcement and infomax learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3686–3698. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/26588e932c7ccfa1df309280702fe1b5-Paper.pdf.

Micheli, V., Alonso, E., and Fleuret, F. Transformers are sample-efficient world models. In *To appear in The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=vhFu1Acb0xb.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, February 2015a.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015b.

Nikishin, E., Schwarzer, M., D'Oro, P., Bacon, P.-L., and Courville, A. The primacy bias in deep reinforcement learning. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16828–16847. PMLR, 17–23 Jul 2022.

Oliphant, T. E. Python for scientific computing. *Computing in Science & Engineering*, 9(3):10–20, 2007. doi: 10.1109/MCSE.2007.58.

Ostrovski, G., Castro, P. S., and Dabney, W. The difficulty of passive learning in deep reinforcement learning. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=nPHA8fGicZk.

Ota, K., Jha, D. K., and Kanezaki, A. Training larger networks for deep reinforcement learning. *arXiv preprint arXiv:2102.07920*, 2021.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Gray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=TG8KACxEON.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Robine, J., Höftmann, M., Uelwer, T., and Harmeling, S. Transformer-based world models are happy with 100k interactions. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=TdBaDGCpjly.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *CoRR*, abs/1511.05952, 2016.

Schaul, T., Barreto, A., Quan, J., and Ostrovski, G. The phenomenon of policy churn. *Advances in Neural Information Processing Systems*, 2022.

Schmidt, D. and Schmied, T. Fast and data-efficient training of rainbow: an experimental study on atari. *arXiv preprint arXiv:2111.10247*, 2021.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Schrittwieser, J., Hubert, T., Mandhane, A., Barekatain, M., Antonoglou, I., and Silver, D. Online and offline reinforcement learning by planning with a learned model. *Advances in Neural Information Processing Systems*, 34: 27580–27591, 2021.

Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A., and Bachman, P. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016. URL

http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html.

Sinha, S., Bharadhwaj, H., Srinivas, A., and Garg, A. D2rl: Deep dense architectures in reinforcement learning. *arXiv preprint arXiv:2010.09163*, 2020.

Sokar, G., Agarwal, R., Castro, P. S., and Evci, U. The dormant neuron phenomenon in deep reinforcement learning. In *ICML*, 2023.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, August 1988.

Sutton, R. S. and Barto, A. G. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0262193981.

Taiga, A. A., Agarwal, R., Farebrother, J., Courville, A., and Bellemare, M. G. Investigating multi-task pretraining and generalization in reinforcement learning. In *Submitted to The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sSt9fROSZRO. under review.

Team, A. A., Bauer, J., Baumli, K., Baveja, S., Behbahani, F., Bhoopchand, A., Bradley-Schmieg, N., Chang, M., Clay, N., Collister, A., et al. Human-timescale adaptation in an open-ended task space. *arXiv preprint arXiv:2301.07608*, 2023.

van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirthieth AAAI Conference On Artificial Intelligence (AAAI), 2016*, 2016. cite arxiv:1509.06461Comment: AAAI 2016.

Van Hasselt, H. P., Hessel, M., and Aslanides, J. When to use parametric models in reinforcement learning? *Advances in Neural Information Processing Systems*, 32, 2019.

Van Rossum, G. and Drake Jr, F. L. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pp. 1995–2003, 2016.

Ye, W., Liu, S., Kurutach, T., Abbeel, P., and Gao, Y. Mastering atari games with limited data. *Advances in Neural Information Processing Systems*, 34:25476–25488, 2021.

Acknowledgements. Many thanks to Ross Goroshin, Georg Ostrovski, and Gopeshh Subbaraj for their feedback on an earlier draft of this paper. The authors would like to thank the anonymous reviewers for useful discussions and feedback on this paper. We would also like to thank the Python community (Van Rossum & Drake Jr, 1995; Oliphant, 2007) for developing tools that enabled this work, including NumPy (Harris et al., 2020), Matplotlib (Hunter, 2007) and JAX (Bradbury et al., 2018).

Societal impact. Although the work presented here is mostly academic, it aids in the development of more capable autonomous agents. While our contributions do not directly contribute to any negative societal impacts, we urge the community to consider these when building on our research.

A. Additional Results

	Random	Human	DER	DrQ(ϵ)	SPR	IRIS	SR-SPR	EfficientZero	BBF
Alien	227.8	7127.7	802.3	865.2	841.9	420.0	1107.8	808.5	**1173.2**
Amidar	5.8	1719.5	125.9	137.8	179.7	143.0	203.4	148.6	**244.6**
Assault	222.4	742.0	561.5	579.6	565.6	1524.4	1088.9	1263.1	**2098.5**
Asterix	210.0	8503.3	535.4	763.6	962.5	853.6	903.1	**25557.8**	3946.1
BankHeist	14.2	753.1	185.5	232.9	345.4	53.1	531.7	351.0	**732.9**
BattleZone	2360.0	37187.5	8977.0	10165.3	14834.1	13074.0	17671.0	13871.2	**24459.8**
Boxing	0.1	12.1	-0.3	9.0	35.7	70.1	45.8	52.7	**85.8**
Breakout	1.7	30.5	9.2	19.8	19.6	83.7	25.5	**414.1**	370.6
ChopperCommand	811.0	7387.8	925.9	844.6	946.3	1565.0	2362.1	1117.3	**7549.3**
CrazyClimber	10780.5	35829.4	34508.6	21539.0	36700.5	59324.2	45544.1	**83940.2**	58431.8
DemonAttack	152.1	1971.0	627.6	1321.5	517.6	2034.4	2814.4	13003.9	**13341.4**
Freeway	0.0	29.6	20.9	20.3	19.3	**31.1**	25.4	21.8	25.5
Frostbite	65.2	4334.7	871.0	1014.2	1170.7	259.1	**2584.8**	296.3	2384.8
Gopher	257.6	2412.5	467.0	621.6	660.6	2236.1	712.4	**3260.3**	1331.2
Hero	1027.0	30826.4	6226.0	4167.9	5858.6	7037.4	8524.0	**9315.9**	7818.6
Jamesbond	29.0	302.8	275.7	349.1	366.5	462.7	389.1	517.0	**1129.6**
Kangaroo	52.0	3035.0	581.7	1088.4	3617.4	838.2	3631.7	724.1	**6614.7**
Krull	1598.0	2665.5	3256.9	4402.1	3681.6	6616.4	5911.8	5663.3	**8223.4**
KungFuMaster	258.5	22736.3	6580.1	11467.4	14783.2	21759.8	18649.4	**30944.8**	18991.7
MsPacman	307.3	6951.6	1187.4	1218.1	1318.4	999.1	1574.1	1281.2	**2008.3**
Pong	-20.7	14.6	-9.7	-9.1	-5.4	14.6	2.9	**20.1**	16.7
PrivateEye	24.9	69571.3	72.8	3.5	86.0	**100.0**	97.9	96.7	40.5
Qbert	163.9	13455.0	1773.5	1810.7	866.3	745.7	4044.1	**14448.5**	4447.1
Roadrunner	11.5	7845.0	11843.4	11211.4	12213.1	9614.6	13463.4	17751.3	**33426.8**
Seaquest	68.4	42054.7	304.6	352.3	558.1	661.3	819.0	1100.2	**1232.5**
UpNDown	533.4	11693.2	3075.0	4324.5	10859.2	3546.2	**112450.3**	17264.2	12101.7
Games > Human	0	0	2	3	6	9	9	**14**	12
IQM (↑)	0.000	1.000	0.183	0.280	0.337	0.501	0.631	1.020	**1.045**
Optimality Gap (↓)	1.000	0.000	0.698	0.631	0.577	0.512	0.433	0.371	**0.344**
Median (↑)	0.000	1.000	0.189	0.313	0.396	0.289	0.685	**1.116**	0.917
Mean (↑)	0.000	1.000	0.350	0.465	0.616	1.046	1.272	1.945	**2.247**

Table A.1: **Scores and aggregate metrics for BBF and competing methods across the 26 Atari 100k games**. Scores are averaged across 50 seeds per game for BBF, 30 for SR-SPR, 5 for IRIS, 3 for EfficientZero, and 100 for others.

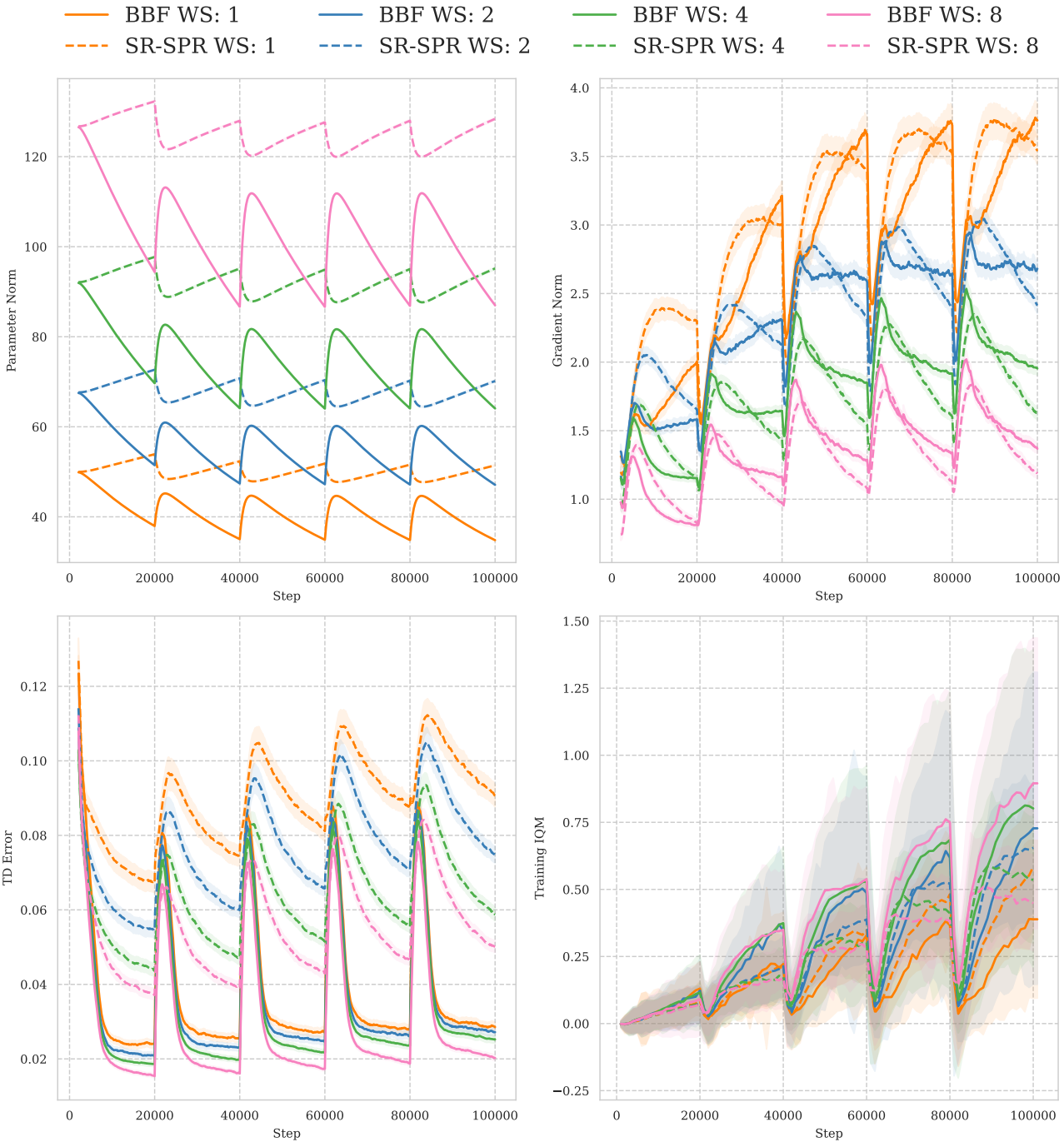

Figure A.1: **Learning curves for BBF and SR-SPR at RR=2 with a ResNet encoder at various width scales**, on the 26 Atari 100k games. Larger networks consistently have lower TD errors and higher gradient norms, and higher parameter norms, but only BBF translates this to higher environment returns. The large, systematic difference in TD error between BBF and SR-SPR is due to BBF's use of a shorter update horizon, which makes each step of the TD backup easier to predict.

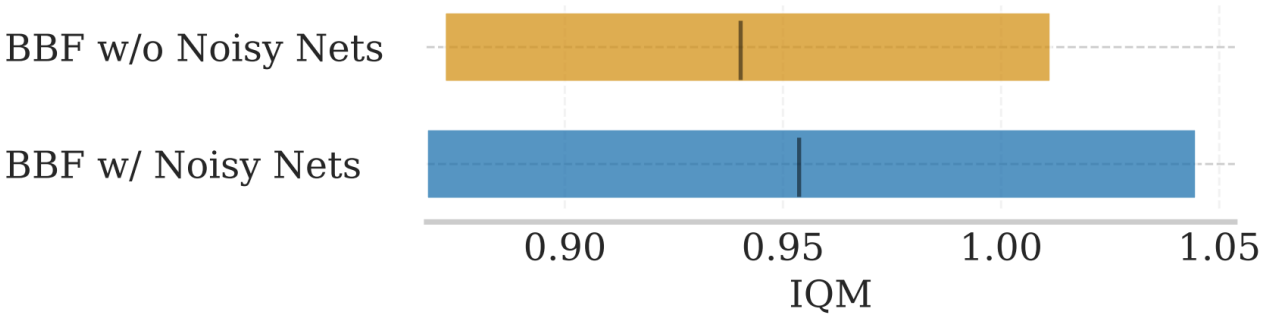

Figure A.2: BBF at RR=2 on the 26 Atari 100k tasks, with and without Noisy Nets.