Full Network Capacity Framework for Sample-Efficient Deep Reinforcement Learning

Wentao Yang[1] **Xinyue Liu**[1] **Yunlong Gao**[1] **Wenxin Liang**[1]
Linlin Zong[1] **Guanglu Wang**[1] **Xianchao Zhang**[*1]

[1]School of Software, Dalian University of Technology, Dalian, Liaoning, China

Abstract

In deep reinforcement learning (DRL), the presence of dormant neurons leads to a significant reduction in network capacity, which results in sub-optimal performance and limited sample efficiency. Existing training techniques, especially those relying on periodic resetting (PR), exacerbate this issue. We propose the Full Network Capacity (FNC) framework based on PR, which consists of two novel modules: Dormant Neuron Reactivation (DNR) and Stable Policy Update (SPU). DNR continuously reactivates dormant neurons, thereby enhancing network capacity. SPU mitigates perturbation from DNR and PR and stabilizes the Q-values for the actor, ensuring smooth training and reliable policy updates. Our experimental evaluations on the Atari 100K and DMControl 100K benchmarks demonstrate the remarkable sample efficiency of FNC. On Atari 100K, FNC achieves a superhuman IQM HNS of 107.3%, outperforming the previous state-of-the-art method BBF by 13.3%. On DMControl 100K, FNC excels in 5 out of 6 tasks in terms of episodic return and attains the highest median and mean aggregated scores. FNC not only maximizes network capacity but also provides a practical solution for real-world applications where data collection is costly and time-consuming. Our implementation is publicly accessible at `https://github.com/tlyy/FNC`.

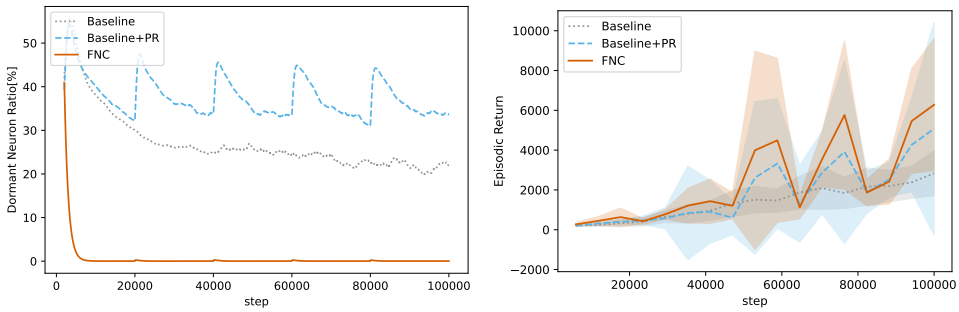
1 INTRODUCTION

Deep reinforcement learning (DRL) has emerged as a pivotal approach in the realm of artificial intelligence, especially when dealing with sequential decision-making problems in environments fraught with uncertainty. Consider the domain of autonomous aerial vehicles (AAVs). The AAVs operate in complex and unpredictable atmospheres, where factors like sudden gusts, rapidly changing weather conditions, and unforeseen obstacles pose significant challenges. Each flight path decision they make is subject to a high degree of uncertainty, and traditional DRL algorithms often struggle to handle these uncertainties efficiently with limited data. Similarly, in financial trading, market conditions are constantly fluctuating due to a myriad of factors as geopolitical events, economic policies, and investor sentiment. Traders using DRL-based strategies need to make decisions in this highly uncertain environment. However, the large number of samples required by traditional DRL methods for effective learning can be an obstacle, as market data is often expensive to obtain and rapidly changing.

To enhance the practicality of DRL, numerous studies have concentrated on improving the sample efficiency of DRL agents[Laskin et al., 2020b, Yarats et al., 2021, Schwarzer et al., 2021, D'Oro et al., 2023, Yu et al., 2021, Schwarzer et al., 2023]. Recent research [Nikishin et al., 2022] has revealed a significant problem: network over-fitting on early interaction samples. This over-fitting problem is particularly prominent in uncertain environments, as the limited initial data may not accurately represent all possible scenarios. Consequently, the learned policies become less reliable, increasing the uncertainty in decision-making. To counter this over-fitting problem, periodic resetting (PR) of network parameters has been proposed [Nikishin et al., 2022]. The BBF method [Schwarzer et al., 2023], which incorporates PR, has achieved state-of-the-art performance on the Atari 100K [Kaiser et al., 2020] benchmark. Nevertheless, as the size of the network continues to increase, further improvements become challenging, accompanied by a substantial increase in computing and storage costs.

Moreover, the neuron dormancy phenomenon in DRL has been discovered in recent studies [Sokar et al., 2023]. Vast neurons remain inactive during training, especially when PR is applied. As depicted in Figure 1, the dormant neuron ratio spikes after each PR operation and remains high through-

[*]Corresponding author



(a) The dormant neurons ratio

(b) The episodic return

Figure 1: The dormant neurons ratio (a) and episodic return (b) during training for a baseline, baseline with periodic resetting (PR), and FNC. A high dormant neuron ratio indicates low network capacity. Although PR improves final performance, it causes a higher dormant neuron ratio than the baseline. FNC fixes the unsteady capacity of PR, rapidly reducing the dormant neuron ratio close to 0 and achieving full network capacity for better performance.

out the training process. This under-utilization of network capacity leads to sub-optimal performance. When neurons are dormant, the network fails to fully utilize its potential, reducing its ability to capture complex environment patterns, which is especially crucial in uncertain scenarios.

To overcome these challenges in DRL under limited samples, we propose the Full Network Capacity (FNC) framework based on PR. FNC consists of two novel modules: Dormant Neuron Reactivation (DNR) and Stable Policy Update (SPU). DNR continuously locates and reactivates dormant neurons. It ensures that the network operates at its full capacity, i.e., it rapidly reduces the dormant neuron ratio close to 0, as shown in Figure 1. However, the parameter perturbations introduced by DNR and from the original PR can cause Q-network instability. Therefore, SPU adopts the momentum Q-network to smooth the perturbations from DNR and PR and evaluate the value of the actor policy.

We evaluate FNC on two standard benchmarks for DRL under limited samples: the Atari 100K benchmark [Kaiser et al., 2020] and the DMControl 100K benchmark [Hafner et al., 2019]. These benchmarks, although artificial, mimic the uncertainty present in real-world scenarios. In the Atari 100K benchmark, FNC achieves superhuman sample efficiency, outperforming the previous state-of-the-art method BBF. In the DMControl 100K benchmark, FNC also shows remarkable results, leading in most tasks.

Our contributions can be summarized as follows:

- We identify the problem of neuron dormancy reducing network capacity and performance, especially in the training framework with PR under limited samples, which are critical problems in dealing with uncertainty in DRL.
- We propose the FNC framework with two novel modules to maximize network capacity and stabilize training, reducing uncertainty in the learned policies.
- We demonstrate that FNC achieves state-of-the-art performance on the Atari 100K and DMControl 100K benchmarks with limited computational resources, providing a practical solution for DRL in uncertain, sample-constrained settings.

2 RELATED WORK

2.1 SAMPLE EFFICIENCY IN DRL

Sample efficiency is a critical aspect of DRL, as it determines the ability of agents to learn effectively with a limited number of interaction samples. High sample complexity has long been a significant hurdle in the practical application of DRL agents, especially in real-world scenarios where interactions can be costly, time-consuming, or even dangerous.

Numerous techniques have been proposed to enhance sample efficiency:

- **Experience Replay**: Experience replay [Mnih et al., 2013] stores past experiences (s, a, r, s') in a replay buffer. During training, these experiences are randomly sampled and reused. The replay ratio (RR) [D'Oro et al., 2023], defined as the ratio of learning updates to new experiences, plays a crucial role in optimizing performance on limited data. For instance, the original DQN algorithm uses an RR of 0.25. However, more recent and efficient agents often utilize higher ratios, allowing them to learn more from the available data.
- **Data Augmentation**: Techniques like DrQ [Yarats et al., 2021] and RAD [Laskin et al., 2020a] introduce data augmentation methods to DQN [Mnih et al., 2013, 2015] and SAC [Haarnoja et al., 2018a,b]. These methods, such as random shift and intensity adjustments, increase the diversity of the training data. By artificially creating more varied input data, the agent can

learn more generalizable patterns, leading to improved performance.

- **Self-Supervised Learning**: SPR [Schwarzer et al., 2021] builds on the Rainbow [Hessel et al., 2018] algorithm and incorporates a self-supervised temporal consistency loss based on BYOL [Grill et al., 2020], along with data augmentation. The self-supervised loss helps the agent learn about the underlying structure of the environment without relying solely on the rewards provided.
- **Model-Based Methods**: Algorithms such as SimPLe [Kaiser et al., 2020] and EfficientZero [Ye et al., 2021] focus on learning the environment dynamics. By building a model of how the environment behaves, these methods can make more informed decisions and require fewer real-world interactions.

2.2 NETWORK CAPACITY IN DRL

Network capacity, which is related to the number of active neurons in a network, is another key factor in DRL. A higher network capacity improves the agent's ability to model complex situations. Two main ways to enhance network capacity are enlarging the network size and increasing the active neuron ratio in a fixed-size network.

- **Scaling Up Networks**: Ota et al. [2021] showed how increasing the number of layers and neurons in a network can improve its representational power. They also highlighted the challenges, such as increased computational requirements and over-fitting. Schwarzer et al. [2023] achieved sample-efficient performance by scaling the neural networks used for value estimation, combined with other design choices that enabled this scaling. However, increasing the network size brings additional computational and storage costs.
- **Neuron Dormancy**: Recent studies [Sokar et al., 2023, Abbas et al., 2023, Dohare et al., 2024] have uncovered the neuron dormancy phenomenon in neuron network, which means a significant number of neurons remain inactive during training and do not contribute to the network's output. This under-utilization of network capacity wastes computational resources and limits the agent's learning ability.

Most previous methods have focused on enhancing network capacity by enlarging the network size, which inevitably brings additional computational and storage burdens. In contrast, increasing the active neuron ratio in a fixed-size network has been relatively under-explored. Our approach belongs to the latter, enhancing network capacity without introducing additional burdens.

3 PROBLEM FORMULATION

Deep Reinforcement Learning (DRL) trains an agent to make optimal sequential decisions within an environment. The goal is to maximize cumulative rewards. The overall procedure is formalized through the concept of a Markov Decision Process (MDP) [Puterman, 1994], represented by the tuple $M = (S, A, P, R, \rho_0, \gamma)$. The set S encompasses all possible states s that the agent can occupy within the environment. The set A consists of all the actions a that the agent can execute. Given a state s and an action a, the transition probability function $P(s'|s, a)$ quantifies the likelihood of the environment transitioning from the current state s to a new state s'. The reward function $R(s, a)$ assigns a scalar value r to the agent when it takes action a in state s. $\rho_0(s)$ represents the probability distribution over the initial states. The discount factor $\gamma \in [0, 1]$ determines the relative importance of future rewards compared to immediate rewards.

The goal of DRL is to discover a policy $\pi_\phi(a|s)$ that maximizes the expected cumulative reward, expressed as:

$$J(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)|\pi]. \tag{1}$$

Two crucial functions in DRL algorithms are the state-value function $V^\pi(s)$ and the action-value function $Q^\pi(s, a)$:

$$V^\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)|s_0 = s, \pi]. \tag{2}$$

$$Q^\pi(s, a) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)|s_0 = s, a_0 = a, \pi]. \tag{3}$$

Common DRL algorithms include Q-learning [Watkins and Dayan, 1992] and Actor-Critic [Konda, 2002]. Q-learning updates the action-value function $Q(s, a)$ as follows:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(R(s, a) + Q(s', a') - Q(s, a)), \tag{4}$$

where α is the learning rate.

In the Actor-Critic (AC) approach, the actor is responsible for updating the policy, while the critic evaluates the state or action values. The critic learns the state-value function $V(s)$ or the action-value function $Q(s, a)$ and updates it using the temporal difference (TD) error:

$$\theta \leftarrow \theta + \alpha_C \delta_t \nabla_\theta V(s_t), \tag{5}$$

where

$$\delta_t = R(s, a) + \gamma V(s_{t+1}) - V(s_t), \tag{6}$$

and α_C is the critic's learning rate. The actor then updates the policy using an advantage-based policy gradient:

$$\phi \leftarrow \phi + \alpha_A \delta_t \nabla_\phi \log \pi(a_t|s_t; \phi), \tag{7}$$

where α_A is the actor's learning rate.

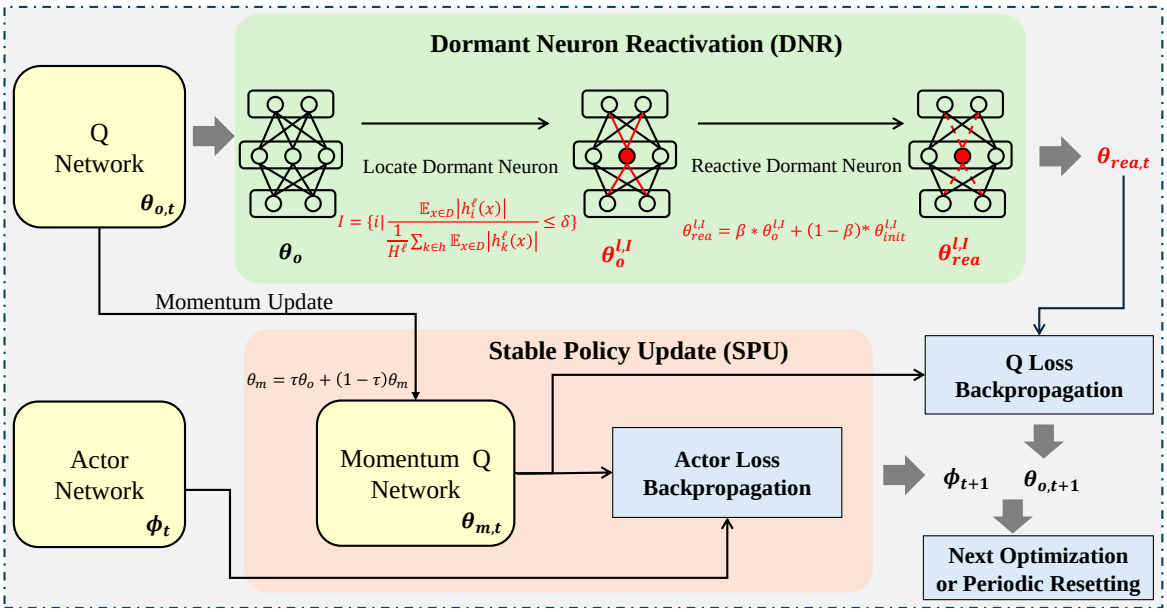


Figure 2: Details of Full Network Capacity (FNC) training framework. FNC introduces two new mechanisms into each training update: Dormant Neuron Reactivation (DNR) to activate the dormant neurons and Stable Policy Update (SPU) to smooth the perturbation from DNR and periodic resetting.

4 FULL NETWORK CAPACITY TRAINING FRAMEWORK

4.1 OVERVIEW

The Full Network Capacity (FNC) framework fully exploits network capacity when applying periodic resetting (PR) under limited samples. FNC incorporates two novel modules into the PR framework: Dormant Neuron Reactivation (DNR) and Stable Policy Update (SPU), as shown in Figure 2. These modules work in tandem to enhance network capacity and ensure stable training, thereby improving the sample efficiency of DRL agents.

4.2 DORMANT NEURON REACTIVATION

The Dormant Neuron Reactivation (DNR) module addresses dormant neurons by detecting and reactivating these neurons.

We first identify dormant neurons. The dormant ratio $d^{l,i}$ of neuron i in layer l is calculated as:

$$a^{l,i} = \frac{\mathbb{E}_{x\in D}|h_i^l(x)|}{\frac{1}{H^l}\sum_{k\in H^l}\mathbb{E}_{x\in D}|h_k^l(x)|} \tag{8}$$

where D is the input distribution, $h_i^l(x)$ is the neuron's activation for input $x \in D$, and H^l is the number of neurons in layer l. Neurons with $d^{l,i} \le \delta$ form the dormant neuron index set I:

$$I = \{i | a^{l,i} \le \delta\} \tag{9}$$

Algorithm 1 Dormant Neuron Reactivation (DNR)

Input: Online Q network parameters θ_o, DNR weight β, dormant threshold δ.
Output: Reactivated parameters θ_{rea}
for each layer l **do**
  for each unit i **do**
    Compute activation value:

$$a^{l,i} = \frac{\mathbb{E}_{x\in D}|h_i^l(x)|}{\frac{1}{H^l}\sum_{k\in H^l}\mathbb{E}_{x\in D}|h_k^l(x)|}$$

    if $a^{l,i} \le \delta$ **then**
      Reactive the neuron parameters:

$$\theta_{\text{rea}}^{l,i} = \beta \cdot \theta_o^{l,i} + (1-\beta) \cdot \theta_{\text{init}}^{l,i}$$

    end if
  end for
end for
return θ_{rea}

After location, we reactivate dormant neurons using shrink and perturb operations. To prevent the network from converging to sharp minima, we shrink the parameters of dormant neurons. Given online parameters θ_o, the shrunk parameters are:

$$\theta_{\text{shrink}}^{l,i} = \beta \cdot \theta_o^{l,i}$$

where $\beta \in (0, 1)$ is the shrink weight. To encourage exploration in the parameter space, we add a fraction of the initial

parameters to the shrunk parameters:

$$\theta^{l,i}_{\text{perturb}} = (1-\beta)\cdot\theta^{l,i}_{\text{init}}$$

The reactivated parameters combine the two operations:

$$\theta^{l,i}_{\text{rea}} = \theta^{l,i}_{\text{shrink}} + \theta^{l,i}_{\text{perturb}}$$

The DNR process is summarized in Algorithm 1. We focus on the dormancy of the critic or the Q-network as empirical study Ma et al. [2023] shows that critic dormancy has a more pronounced impact on sample efficiency. Moreover, directly recovering the actor's dormancy by DNR introduces too much instability into the policy.

4.3 STABLE POLICY UPDATE

The perturbation introduced by the DNR and periodic resetting can disrupt the stability of policy updates. The Stable Policy Update(SPU) mitigates this issue using a momentum Q-network for policy updates. The process is described in Algorithm 2. The reason why delayed copy update is not used is that the policy is derived from the momentum Q-network. The policy does not change within the copy interval time and is not suitable for collecting new data.

Algorithm 2 Stable Policy Update (SPU)

1: **Input:** Online Q-network parameters θ_o, Momentum Q-network parameters θ_m, momentum parameter τ
2: **Output:** Actor policy parameters ϕ
3: Initialize $\theta_m = \theta_o$
4: **for** each training step **do**
5: Update θ_m using the momentum update rule:

$$\theta_m = \tau\cdot\theta_o + (1-\tau)\cdot\theta_m$$

6: Use θ_m to compute Q-values for actor policy parameters ϕ updates
7: **end for**
8: **return** ϕ

In the discrete control setting, although the deep Q-network (DQN) does not have an explicit actor in the traditional sense, the Q-network can be regarded as the actor for action generation. In this case, the parameters of the actor and the Q-network are identical. SPU treats the momentum Q-network in DQN as the actor, and the actual policy is:

$$a = \text{argmax}_a Q_{\theta_m}(s,a), \tag{10}$$

which contrasts with previous approaches that typically employ the online Q-network.

In the continuous control setting, the actor policy is explicitly defined with parameters ϕ. SPU optimizes the actor using the momentum critic as:

$$J_\pi(\phi) = Q_{\theta_m}(s,\pi_\phi(s)) - \alpha * \log\pi_\phi(s). \tag{11}$$

Since DNR-induced perturbations only occur in the online Q-network during training, and periodic resetting (PR) also affects the network periodically, SPU plays a crucial role in preventing drastic changes caused by both PR and DNR. It generates a stable Q-value for policy updates, which is also beneficial in preventing the emergence of dormant neurons, as previously noted in related research [Sokar et al., 2023]. Furthermore, the momentum counterpart is a laggard of the online Q-network, it tends to have the same dormant neurons as the online Q-network has. SPU prevents these neurons of the momentum Q-network from being completely inactive once they are detected in the online Q-network.

5 EXPERIMENT

We conduct a comprehensive evaluation of the proposed Full Network Capacity (FNC) framework on two standard benchmarks for DRL under limited samples. The primary objectives are to thoroughly assess the performance, network capacity, and distinct advantages of FNC over existing methods.

5.1 EXPERIMENTAL SETUP

5.1.1 Benchmark Selection Rationale

Two benchmarks employed in our study are the Atari 100K benchmark [Kaiser et al., 2020] and the DMControl 100K benchmark [Hafner et al., 2019]. The Atari 100K benchmark is a rich source of diverse vision-based control tasks, consisting of 26 games with low-dimensional discrete actions. They test an agent's ability to perceive visual cues, make quick and accurate action selections, and adapt to different game mechanics. The DMControl 100K benchmark, focuses on six control tasks with high-dimensional continuous actions. These tasks evaluate an agent's proficiency in handling continuous control problems, understanding complex dynamics, and making fine-grained decisions in dynamic environments.

5.1.2 Implementation Details

In the discrete action settings, our implementation is grounded in the SPR algorithm framework [Schwarzer et al., 2021]. We inherit a similar architecture and incorporate random shifts and intensity data augmentation techniques [Yarats et al., 2021]. To ensure a fair and reliable comparison, we meticulously follow the same architecture parameters and hyperparameters as those used in the BBF method [Schwarzer et al., 2023]. This includes the utilization of the Impala residual network as the encoder and expanding the network width by 4 times to enhance its representational capability.

Table 1: Final scores and aggregate metrics for FNC and competing methods [Schwarzer et al., 2021, Agarwal et al., 2021, D'Oro et al., 2023, Schwarzer et al., 2023] across the 26 Atari 100K games. Scores are averaged across five runs per game for FNC. We report the standard error for game scores and the 95% bootstrap confidence interval for the aggregate metrics of our method FNC.

Game	Human	Random	DrQ(ϵ)	SPR	SR-SPR	BBF	FNC
Alien	7127.7	227.8	865.2	841.9	1107.8	1121.7	**1250.3** $\pm$ **76.0**
Amidar	1719.5	5.8	137.8	179.7	203.4	**236.6**	173.7 $\pm$ 25.2
Assault	742.0	222.4	579.6	565.6	1088.9	2004.5	**2521.4** $\pm$ **305.7**
Asterix	8503.3	210.0	763.6	962.5	903.1	3169.8	**4410.7** $\pm$ **562.9**
Bank Heist	753.1	14.2	232.9	345.4	531.7	768.8	**781.3** $\pm$ **169.5**
Battle Zone	37187.5	2360.0	10165.3	14834.1	17671.0	23681.4	**23338.0** $\pm$ **2408.1**
Boxing	12.1	0.1	9.0	35.7	45.8	77.4	**79.8** $\pm$ **6.3**
Breakout	30.5	1.7	19.8	19.6	25.5	331.1	**374.2** $\pm$ **8.2**
ChopperCommand	7387.8	811.0	844.6	946.3	2362.1	**4251.6**	2802.2 $\pm$ 995.8
CrazyClimber	35829.4	10780.5	21539.0	36700.5	45544.1	60864.5	**63323.2** $\pm$ **9785.8**
DemonAttack	1971.0	152.1	1321.5	517.6	2814.4	18298.4	**20798.0** $\pm$ **4223.8**
Freeway	29.6	0.0	20.3	19.3	25.4	23.1	**27.1** $\pm$ **1.4**
Frostbite	4334.7	65.2	1014.2	1170.7	**2584.8**	2023.1	1377.4 $\pm$ 705.2
Gopher	2412.5	257.6	621.6	660.6	712.4	1209.4	**1629.7** $\pm$ **285.9**
Hero	30826.4	1027.0	4167.9	5858.6	**8524.0**	5741.8	5604.6 $\pm$ 624.1
Jamesbond	302.8	29.0	349.1	366.5	389.1	**1124.6**	1058.7 $\pm$ 172.8
Kangaroo	3035.0	52.0	1088.4	3617.4	3631.7	5032.1	**8202.0** $\pm$ **1830.8**
Krull	2665.5	1598.0	4402.1	3681.6	5911.8	8069.8	**8075.1** $\pm$ **59.0**
KungFuMaster	22736.3	258.5	11467.4	14783.2	18649.4	16616.9	**21508.6** $\pm$ **4703.6**
Ms Pacman	6951.6	307.3	1218.1	1318.4	1574.1	**2217.8**	1994.9 $\pm$ 206.8
Pong	14.6	-20.7	-9.1	-5.4	2.9	**13.7**	10.2 $\pm$ 6.1
PrivateEye	69571.3	24.9	3.5	86.0	**97.9**	39.1	54.0 $\pm$ 46.0
Qbert	13455.0	163.9	1810.7	866.3	**4044.1**	3245.3	2897.7 $\pm$ 761.7
RoadRunner	7845.0	11.5	11211.4	12213.1	13463.4	26419.0	**30723.0** $\pm$ **2142.3**
Seaquest	42054.7	68.4	352.3	558.1	819.0	**988.6**	835.6 $\pm$ 182.2
UpNDown	11693.2	533.4	4324.5	10859.2	**112450.3**	15122.7	17093.7 $\pm$ 3847.0
IQM HNS (↑)	100.0%	0.0%	28.0%	33.7%	63.1%	94.0%	**107.3%**[96.2%,120.0%]
OG HNS (↓)	0.0%	100.0%	63.1%	57.7%	43.3%	37.7%	**36.7%**[34.2%,39.4%]
Median HNS (↑)	100.0%	0.0%	31.3%	39.6%	68.5%	75.5%	**89.4%**[68.3%,98.1%]
Mean HNS (↑)	100.0%	0.0%	46.5%	61.6%	127.2%	217.5%	**240.1%**[221.2%,257.8%]

In continuous action settings, we build on the DrQ framework [Yarats et al., 2021], a modified version of SAC [Haarnoja et al., 2018a] with integrated data augmentation capabilities. It allows the framework to handle pixel-based input. All hyperparameters, network architectures, and implementation choices are detailed in the Appendix C.

5.2 FNC IMPROVES AGENT PERFORMANCE

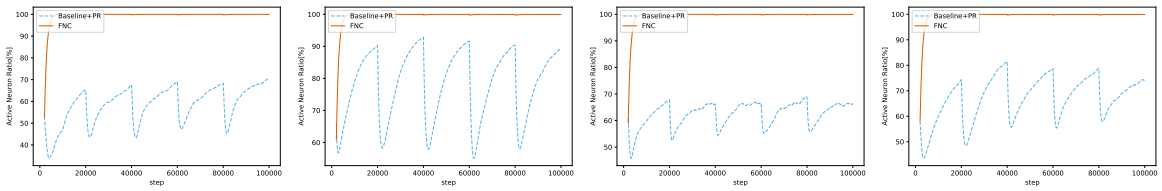
We gauge the agent's performance by measuring the final score after training with limited interaction samples. In the context of a DRL with a fixed sample budget, performance is intrinsically linked to sample efficiency. A high-performance agent is a strong indicator of high sample efficiency, as the agent can learn effectively with fewer samples.

On the Atari 100K benchmark, we collect the final scores of the agents across all 26 tasks. To standardize the comparison, we calculate the human-normalized score (HNS) for each game using the following formula:

$$HNS = \frac{S_A - S_R}{S_H - S_R}, \tag{12}$$

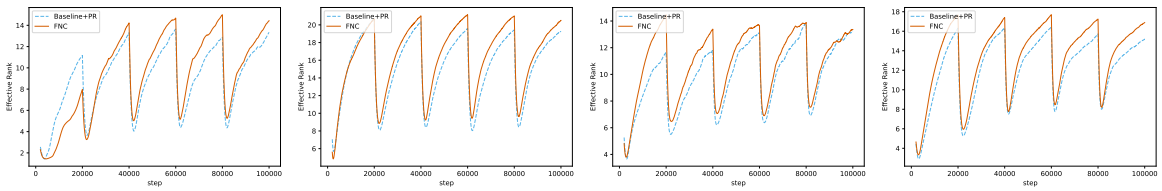
where S_A represents the score achieved by the agent, S_R is the score obtained by random play, and S_H is the score achieved by an expert human player. This normalization allows for a direct comparison of the agent's performance relative to human performance.

Subsequently, we adopt the inter-quartile mean (IQM), optimality gap (OG), median, and mean metrics from the rliable [Agarwal et al., 2021] framework to aggregate the HNS values across the 26 games. The IQM metric represents



(a) Freeway (b) Gopher (c) Kangaroo (d) KungFuMaster

Figure 3: The active neuron ratio during training for baseline with periodic resetting (PR) and FNC. The active neuron ratio of baseline with PR is relatively low, especially at the time after each PR, FNC quickly recovers the full network capacity by increasing the active neuron ratio close to 100% and maintains it even with the PR executed.



(a) Freeway (b) Gopher (c) Kangaroo (d) KungFuMaster

Figure 4: The effective rank [Roy and Vetterli, 2007] during training for baseline with periodic resetting (PR) and FNC. FNC enhances the effective rank, enabling a better representation and expressivity on all selected games.

the average score of the middle 50% of the runs combined across all games and seeds. It reduces the impact of extreme values, providing a more stable and representative assessment. A higher IQM, mean, and median score indicates a better overall performance, while a lower OG, which quantifies the performance gap between the agent and the human, is more favorable.

As presented in Table 1, FNC showcases remarkable performance. It outperforms human performance in 11 games and surpasses previous model-free methods in 15 out of 26 games in terms of the final score. FNC achieves the highest IQM HNS of 107.3%, Median HNS of 89.4%, Mean HNS of 240.1%, and the lowest OG HNS of 36.7%. Notably, FNC outperforms the previous state-of-the-art method BBF by 13.3% in the IQM score under the same replay ratio of 2.

On the DMControl 100K benchmark, we calculate the final scores of the six tasks and aggregate them using the median and mean metrics. As shown in Table 2 in Appendix A, FNC achieves the best final scores on 5 tasks and attains the best median and mean scores. These results, on both the Atari 100K and DMControl 100K benchmarks, demonstrate the effectiveness of FNC in improving sample efficiency.

5.3 FNC IMPROVES NETWORK CAPACITY

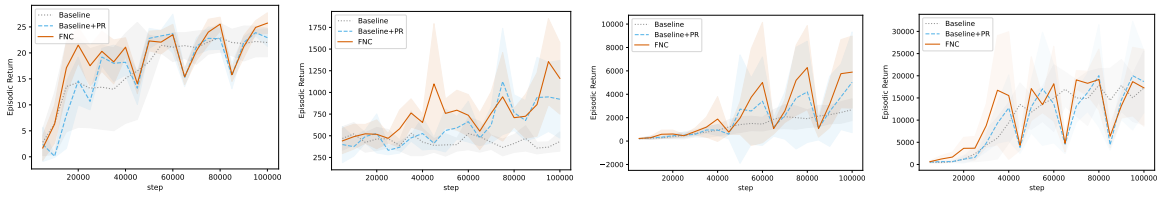
To evaluate the network capacity, we monitor the active neuron ratio. Since the same network architecture is applied across different variants, the active neuron ratio is reliable to assess how effective the network is.

We conduct a comparative analysis of the active neuron ratio of the baseline with periodic resetting and FNC during the training process on 4 selected Atari games. As illustrated in Figure 3, the active neuron ratio of the baseline with periodic resetting remains relatively low, especially immediately after each resetting event. This low ratio indicates that a significant portion of the network's potential is left untapped, leading to under-utilization of the network capacity. In contrast, FNC exhibits a remarkable ability to rapidly reduce the dormant neuron ratio to close to 0. Even when periodic resetting is executed, FNC manages to maintain a high active neuron ratio, suggesting that it can effectively utilize nearly the entire capacity of the network.

The full utilization of the network capacity endows the network with a higher effective rank [Roy and Vetterli, 2007], which contributes to better representation and expressivity, as shown in Figure 4. Consequently, the performance of FNC is higher than that of the baseline with PR during the training procedure, as shown in Figure 5.

5.4 FNC REDUCES THE TRAINING COMPUTATION COST

In the online setting, the agent interacts with the environment and trains the policy simultaneously. Each interaction is contingent upon the completion of the update procedure. Therefore, reducing the cost of updating is vital to minimize the expenditures and risks associated with interactions in real-world scenarios.



(a) Freeway (b) Gopher (c) Kangaroo (d) KungFuMaster

Figure 5: The episodic return during training for baseline with period resetting (PR) and FNC. FNC outperforms the baseline with periodic resetting on all selected games.

To evaluate the computation cost, we record the training runtime. The IQM HNS and runtime comparison are depicted in Figure 6, along with relevant model-free [van Hasselt et al., 2019, Yarats et al., 2021, Schwarzer et al., 2021, D'Oro et al., 2023, Schwarzer et al., 2023] and model-based [Micheli et al., 2023, Ye et al., 2021] algorithms. FNC only consumes approximately the same amount of time as BBF with a replay ratio of 2 to complete training on a task. However, its performance rivals BBF with a replay ratio of 8, which requires four times the computational resources. FNC achieves superhuman sample efficiency with a low replay ratio of 2 for the first time, demonstrating its cost-effectiveness in achieving high-performance results.

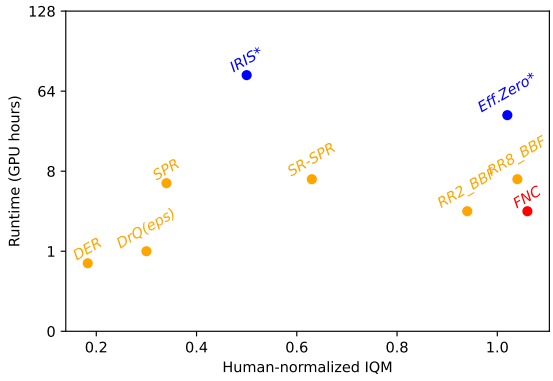


Figure 6: Computational cost versus performance, measured by IQM HNS across 26 games and the total number of GPU hours spent per environment for Runtime. FNC improves performance even with lower costs.

5.5 ABLATION STUDY

To gain a deeper understanding of the contribution of each component of FNC, we conduct a systematic ablation study. In this study, we remove one component at a time and observe the impact on the performance of the framework. The results are presented in Figure 7.

When we remove Dormant Neuron Reactivation (FNC-DNR), we observe a notable decline in the IQM score. This decline clearly indicates that the activation of dormant neurons in each update step is crucial to maintaining high network expressivity. Without DNR, the network fails to fully exploit its capacity, leading to sub-optimal performance.

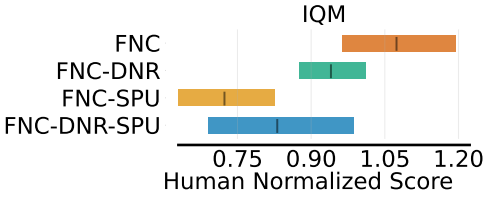


Figure 7: Ablation study results show the impact of removing different components of FNC on the Atari 100K benchmark.

Removing the Stable Policy Update (FNC-SPU) also results in a significant decrease in the IQM score, even worse than the baseline that only applies PR (FNC-DNR-SPU). It indicates that directly introducing DNR into the PR framework without SPU causes excess parameter perturbation and unsteady training. This finding highlights the importance of SPU in stabilizing the training process.

Hyperparameters β and δ were tuned by grid search in 4 Atari games (Freeway, Gopher, Kangaroo, KungFuMaster). The results are depicted in Appendix B. We set $\beta = 0.5$ and $\delta = 0.0$ to maximize reactivation while avoiding over-perturbation, achieve the best mean HNS.

6 CONCLUSION

In this study, we have introduced the Full Network Capacity (FNC) framework to address dormant neurons and to exploit network capacity in deep reinforcement learning (DRL) under limited samples. The FNC framework, with its two novel modules, Dormant Neuron Reactivation (DNR) and Stable Policy Update (SPU), has achieved state-of-the-art performance on the Atari 100K and DMControl 100K benchmarks with limited computational resources.

Our work has not only provided a practical solution to the sample-efficiency problem in DRL but also opened up new research directions. We believe that the insights gained from

this study will inspire further research in the area of sample-efficient reinforcement learning, leading to the development of more advanced and efficient DRL algorithms. These advancements could have far-reaching implications in various real-world applications with uncertainty, such as robotics, autonomous vehicles, and financial trading, where data collection is often costly and time-consuming.

7 ACKNOWLEDGEMENT

This work was supported by National Natural Science Foundation of China (No. 62476040).

References

Zaheer Abbas, Rosie Zhao, Joseph Modayil, Adam White, and Marlos C. Machado. Loss of plasticity in continual deep reinforcement learning. In Sarath Chandar, Razvan Pascanu, Hanie Sedghi, and Doina Precup, editors, *Conference on Lifelong Learning Agents, 22-25 August 2023, McGill University, Montréal, Québec, Canada*, volume 232 of *Proceedings of Machine Learning Research*, pages 620–636. PMLR, 2023.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C. Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 29304–29320, 2021.

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nat.*, 632(8026):768–774, 2024.

Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G. Bellemare, and Aaron C. Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent - A new approach to self-supervised learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018a.

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

Danijar Hafner, Timothy P. Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 3215–3222. AAAI Press, 2018.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H. Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model based reinforcement learning for atari. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Vijaymohan Konda. *Actor-critic algorithms*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2002.

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020a.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5639–5650. PMLR, 2020b.

Guozheng Ma, Lu Li, Sen Zhang, Zixuan Liu, Zhen Wang, Yixin Chen, Li Shen, Xueqian Wang, and Dacheng Tao. Revisiting plasticity in visual reinforcement learning: Data, modules and training stages. *CoRR*, abs/2310.07418, 2023.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nat.*, 518(7540):529–533, 2015.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron C. Courville. The primacy bias in deep reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 16828–16847. PMLR, 2022.

Kei Ota, Devesh K. Jha, and Asako Kanezaki. Training larger networks for deep reinforcement learning. *CoRR*, abs/2102.07920, 2021.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *15th European Signal Processing Conference, EUSIPCO 2007, Poznan, Poland, September 3-7, 2007*, pages 606–610. IEEE, 2007.

Max Schwarzer, Ankesh Anand, Rishab Goel, R. Devon Hjelm, Aaron C. Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Max Schwarzer, Johan Samir Obando-Ceron, Aaron C. Courville, Marc G. Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: Human-level atari with human-level efficiency. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 30365–30380. PMLR, 2023.

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 32145–32168. PMLR, 2023.

Hado van Hasselt, Matteo Hessel, and John Aslanides. When to use parametric models in reinforcement learning? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 14322–14333, 2019.

Christopher J. C. H. Watkins and Peter Dayan. Technical note q-learning. *Mach. Learn.*, 8:279–292, 1992.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering atari games with limited data. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 25476–25488, 2021.

Tao Yu, Cuiling Lan, Wenjun Zeng, Mingxiao Feng, Zhizheng Zhang, and Zhibo Chen. Playvirtual: Augmenting cycle-consistent virtual trajectories for reinforcement learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman

Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 5276–5289, 2021.

Full Network Capacity Framework for Sample-Efficient Deep Reinforcement Learning (Supplementary Material)

Wentao Yang[1] **Xinyue Liu**[1] **Yunlong Gao**[1] **Wenxin Liang**[1]
Linlin Zong[1] **Guanglu Wang**[1] **Xianchao Zhang**[*1]

[1]School of Software, Dalian University of Technology, Dalian, Liaoning, China

A DMCONTROL 100K BENCHMARK PERFORMANCE TABLE

Table 2: Scores(mean and standard deviation) and aggregate metrics for FNC and competing methods [Hafner et al., 2020, Laskin et al., 2020b, Yarats et al., 2021, Yu et al., 2021] across the 6 DMControl 100K games. We run our FNC with five random seeds per game. The scores of other methods refer to the work of Yu et al.[Yu et al., 2021].

Game	Dreamer	CURL	DrQ	PlayVirtual	FNC
Ball In Cup Catch	246 ± 174	769 ± 43	913 ± 53	926 ± 31	$\mathbf{962 \pm 6}$
Cartpole Swingup	326 ± 27	582 ± 146	759 ± 92	816 ± 36	$\mathbf{850 \pm 17}$
Cheetah Run	235 ± 137	299 ± 48	344 ± 67	474 ± 50	$\mathbf{475 \pm 40}$
Finger Spin	341 ± 70	767 ± 56	901 ± 104	$\mathbf{915 \pm 49}$	799 ± 191
Reacher Easy	314 ± 155	538 ± 233	601 ± 213	785 ± 142	$\mathbf{936 \pm 87}$
Walker Walk	277 ± 12	403 ± 24	612 ± 164	460 ± 173	$\mathbf{767 \pm 92}$
Median Score	295.5	560.0	685.5	800.5	**824.5**
Mean Score	289.8	559.7	688.3	729.3	**798.2**

B HYPERPARAMETER ABLATION STUDY

C EXPERIMENT SETTINGS

We use an open-source JAX implementation of BBF from `https://github.com/google-research/google-research/tree/master/bigger_better_faster`, and a JAX implementation of the DrQ algorithm from `https://github.com/evgenii-nikishin/rl_with_resets/tree/main`. All experiments are performed on one RTX 3080 Ti GPU and require GPU memory less than 12G. The runtime is about 3-4 hours for one seed on one game.

The experiments use five random seeds to evaluate performance. We largely reuse the hyperparameters from previous methods [Schwarzer et al., 2023, Yarats et al., 2021], and report the hyperparameter settings used in the DMControl 100k in Table 4 and in the Atari 100k experiments in Table 5.

[*]Corresponding author
[*]Corresponding author

Table 3: Hyperparameter selection for β and δ in DNR with the human-normalized scores. We run each variant with 3 random seeds per game and report the HNS score.

	Freeway	Gopher	Kangaroo	KungFuMaster	Mean HNS
$\beta = 0$	84.7%	55.0%	110.33%	101.33%	87.8%
$\beta = 0.25$	95.7%	74.7%	97.7%	73.0%	85.3%
$\beta = 0.5$	97.3%	83.3%	154.3%	93.0%	107.0%
$\beta = 0.75$	88.3%	44.0%	125.7%	101.3%	89.8%
$\beta = 1$	85.7%	34.3%	125.3%	87.3%	83.2%
$\delta = 0$	97.3%	83.3%	154.3%	93.0%	107.0%
$\delta = 0.1$	87.3%	76.0%	141.7%	107.3%	103.1%
$\delta = 0.2$	76.3%	77.3%	153.7%	87.3%	98.6%

Table 4: Hyperparameters for FNC in DMControl 100K benchmark. The ones introduced by this work are at the bottom of the table.

Parameter	Setting
Grey-scaling	True
Observation down-sampling	64×64
Frames stacked	3
Action repetitions	
Cartpole Swingup	8
Reacher Easy	4
Cheetah Run	4
Finger Spin	2
Ball In Cup Catch	4
Walker Walk	2
Memory size	100000
Seed steps	1000
Discount factor	0.99
Minibatch size	512
Optimizer	Adam
Learning rate	0.0003
First moment decay	0.9
Second moment decay	0.999
ϵ	0.00015
Critic update frequency	2
Critic Q-function soft-update rate	0.005
Actor update frequency	2
Actor log std bounds	[-10, 2]
Init temperature	0.1
Data augmentation	Shifts (± 4 pixels), Intensity
Reset interval	20000
Layers getting hard reset	Final 3
Dormant threshold δ	0.0
DNR weight β	0.8

Table 5: Hyperparameters for FNC in Atari 100K benchmark. The ones introduced by this work are at the bottom of the table.

Parameter	Setting
Grey-scaling	True
Observation down-sampling	84×84
Frames stacked	4
Action repetitions	4
Reward clipping	[-1, 1]
Terminal on loss of life	True
Max frames per episode	108k
Update	Distributional Q
Dueling	True
Support of Q-distribution	51
Discount factor	0.97$\rightarrow$ 0.997
Minibatch size	32
Optimizer	AdamW
Learning rate	0.0001
First moment decay	0.9
Second moment decay	0.999
ϵ	0.00015
Weight decay	0.1
Max gradient norm	10
Priority exponent	0.5
Priority correction	0.4$\rightarrow$ 1
Training steps	100k
Evaluation episodes	100
Memory size	Unbounded
Min replay size for sampling	2000
Replay period every	1 step
Updates per step	2
Multi-step return length	10 $\rightarrow$ 3
Encoder	Impala ResNet
Hidden units	2048
Non-linearity	ReLU
Target network	
Update period	1
EMA coefficient τ	0.005
λ (SPR loss coefficient)	5
K (SPR prediction depth)	5
Data augmentation	Shifts (± 4 pixels), Intensity
Action selection	Target network
Reset interval	20000
Cycle steps	5000
Layers getting hard reset	Final 2
Shrink and Perturb	0.5
Dormant threshold δ	0.0
DNR weight β	0.5