Bridging the Performance Gap Between Target-Based and Target-Free Reinforcement Learning With Iterated Q-Learning

Théo Vincent[1,2,†] **Yogesh Tripathi**[1, 2] **Tim Faust**[1, 2]
Yaniv Oren[3] **Jan Peters**[1,2,4] **Carlo D'Eramo**[2,4,5]

[1]DFKI GmbH, SAIROL [2]Department of Computer Science, TU Darmstadt
[3]Department of Computer Science, Delft University of Technology [4]Hessian.ai
[5]Center for Artificial Intelligence and Data Science, University of Würzburg
[†] correspondence to theo.vincent@dfki.de

Abstract

In value-based reinforcement learning, removing the target network is tempting as the boostrapped target would be built from up-to-date estimates, and the spared memory occupied by the target network could be reallocated to expand the capacity of the online network. However, eliminating the target network introduces instability, leading to a decline in performance. Removing the target network also means we cannot leverage the literature developed around target networks. In this work, we propose to use a copy of the last linear layer of the online network as a target network, while sharing the remaining parameters with the up-to-date online network, hence stepping out of the binary choice between target-based and target-free methods. It enables us to leverage the concept of iterated Q-learning, which consists of learning consecutive Bellman iterations in parallel, to reduce the performance gap between target-free and target-based approaches. Our findings demonstrate that this novel method, termed *iterated Shared Q-Learning* (iS-QL), improves the sample efficiency of target-free approaches across various settings. Importantly, iS-QL requires a smaller memory footprint and comparable training time to classical target-based algorithms, highlighting its potential to scale reinforcement learning research. Our code is publicly available at https://github.com/theovincent/iS-DQN.

1 Introduction

Originally, Q-learning (Watkins & Dayan, 1992) was introduced as a reinforcement learning (RL) method that performs asynchronous dynamic programming using a single look-up table. By storing only one Q-estimate, Q-learning benefits from an up-to-date estimate and reduces memory footprint. However, replacing look-up tables with non-linear function approximators and allowing off-policy samples to make the method more tractable introduces training instabilities (Sutton & Barto, 2018). To address this, Mnih et al. (2015) introduce Deep Q-Network (DQN), an algorithm that constructs the regression target from an older version of the online network, known as the *target network*, which is periodically updated to match the online network (see "Target Based" in Figure 1). This modification to the temporal-difference objective helps mitigate the negative effects of function approximation and bootstrapping (Zhang et al., 2021), two elements of the deadly triad (van Hasselt et al., 2018). Recently, new methods have demonstrated that increasing the size of the Q-network can enhance the learning speed and final performance of temporal difference methods (Espeholt et al., 2018; Schwarzer et al., 2023; Nauman et al., 2024; Lee et al., 2025). Numerous ablation studies highlight the crucial role of the target network in maintaining performance improvements over smaller networks (Figure 7 in Schwarzer et al. (2023), and Figure 9b in Nauman et al. (2024)). Interestingly, even methods initially introduced without a target network (Bhatt et al. (2024) and Kim et al. (2019)) benefit from its reintegration (Figure 5 in Palenicek et al. (2025) and Gan et al. (2021)).

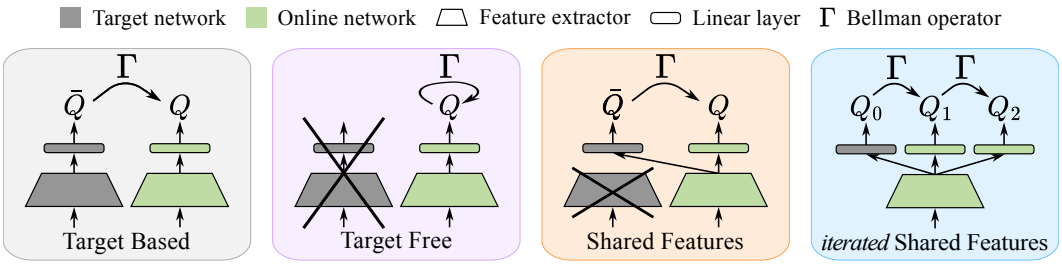

Figure 1: We propose a simple alternative to target-based/target-free approaches, where a linear layer represents the target network, sharing the rest of the parameters with the online network (Shared Features). We apply the concept of iterated Q-learning, which consists of learning multiple Bellman updates in parallel, to reduce the performance gap between target-free and target-based approaches (*iterated* Shared Features).

While temporal difference methods clearly benefit from target networks, their utilization doubles the memory footprint dedicated to Q-networks. This ultimately limits the size of the online network due to the constrained video Random Access Memory (VRAM) of GPUs. This limitation is problematic not only because larger networks can perform better, as previously discussed, but also because some applications inherently require large network sizes. These applications include handling high-dimensional state spaces (Boukas et al., 2021; Pérez-Dattari et al., 2019), processing multi-modal inputs (Schneider et al., 2025), or constructing mixtures of experts (Obando Ceron et al., 2024). This motivates the development of target-free methods.

In this work, we introduce an alternative to the binary choice between target-free and target-based approaches. We propose storing only the smallest possible part of the target network, i.e., the parameters of the last linear layer, while using the parameters of the online network to substitute the other layers of the target network (see "Shared Features" in Figure 1). Although this simple modification alone helps reduce the performance gap between target-free and target-based DQN (see Figure 4, right), we explain in this work how it opens up the possibility of leveraging the target-based literature to reduce this gap further, while maintaining a low memory footprint. Notably, this approach is also orthogonal to regularization techniques that have been shown to be effective for target-free algorithms (Kim et al., 2019; Bhatt et al., 2024; Gallici et al., 2025). Therefore, we will build upon these approaches to benefit from their performance gains.

In the following, we leverage the concept of iterated Q-learning (Vincent et al., 2025) to enhance the learning speed (in terms of number of environment interactions) of target-free algorithms. This concept, initially introduced as a target-based approach, aims at learning multiple Bellman iterations in parallel. This leads to a new algorithm, termed iterated Shared Q-Network (iS-QN), pronounced "ice-QN" to emphasize that it contains a frozen head. iS-QN utilizes a single network with multiple linear heads, where each head is trained to represent the Bellman target of the previous one (see "*iterated* Shared Features" in Figure 1). Our evaluation of iS-QN across various RL settings demonstrates that it improves the learning speed of target-free methods while maintaining a comparable memory footprint and training time.

2 Background

Deep Q-Network (Mnih et al., 2015) The optimal policy of a Markov Decision Process with a discrete action space can be obtained by selecting for each state, the action that maximizes the optimal action-value function Q^*. This function represents the largest achievable expected sum of discounted rewards given a state-action pair. This is why Mnih et al. (2015) estimate the optimal action-value function with a neural network Q_θ, represented by a vector of parameters θ. This neural network is learned to approximate its Bellman iteration ΓQ_θ, leveraging the contraction property of the Bellman operator Γ to guide the optimization process toward the operator's fixed point, i.e., the optimal action-value function Q^*. In practice, a sample estimate of the Bellman iteration is

used, where for a sample (s, a, r, s'), $\Gamma Q_\theta(s, a) = r + \gamma \max_{a'} Q_\theta(s', a')$. However, this learning procedure is unstable because the neural network Q_θ learns from its own values, which change at each optimization step. To address this issue, the authors introduce a target network with parameters $\bar{\theta}$ to stabilize the regression target $\Gamma Q_{\bar{\theta}}$, and periodically update these parameters to the online parameters θ every T steps. This doubles the memory footprint dedicated to Q-networks.

Iterated Q-Network (Vincent et al., 2025) By using a target network, DQN slows down the training process as multiple gradient steps are dedicated to each Bellman iteration. To increase the learning speed, Vincent et al. (2025) proposed to learn consecutive Bellman iterations in parallel. This approach uses a sequence of online parameters $(\theta_i)_{i=1}^K$ and a sequence of target parameters $(\bar{\theta}_i)_{i=0}^{K-1}$. Each online network $Q_{\theta_{i+1}}$ is trained to regress $\Gamma Q_{\bar{\theta}_i}$. Similarly to DQN, each target parameter $\bar{\theta}_i$ is updated to the online parameter θ_{i+1} every T steps. Importantly, the structure of a chain is enforced by setting each $\bar{\theta}_i$ to θ_i every $D \ll T$ steps so that each $Q_{\theta_{i+1}}$, which is learned to regress $\Gamma Q_{\bar{\theta}_i}$, are forced to approximate ΓQ_{θ_i}. This results in $Q_{\theta_K} \approx \Gamma Q_{\theta_{K-1}} \approx \ldots \approx \Gamma^K Q_{\theta_0}$, thus learning K consecutive Bellman iterations in parallel. As a drawback, iterated Q-Network (i-QN) requires storing $2 \times K$ networks, significantly increasing the memory footprint. In the following, we will leverage this concept to reduce the performance gap between target-free and target-based approaches by merging the $2 \times K$ networks into a single network with linear heads.

3 Related Work

Other works have considered removing the target network in different RL scenarios. Vasan et al. (2024) introduce Action Value Gradient, an algorithm designed to work well in a streaming scenario where no replay buffer, no batch updates, and no target networks are available. Gallici et al. (2025) also develop a method for a streaming scenario, in which they rely on parallel environments to cope with the non-stationarity of the sample distribution. Gradient Temporal Difference learning is another line of work that is not using target networks (Sutton et al., 2009; Maei et al., 2009; Yang et al., 2021; Patterson et al., 2022; Elelimy et al., 2025). Instead, they compute the gradient w.r.t. the regression target as well as the gradient w.r.t. the predictions, which doubles the compute requirement. Additionally, to address the double sampling problem, another network is trained to approximate the temporal difference value. This also increases the memory footprint.

Alternatively, some works construct the regression target from the online network instead of the target network, but still use a target network in some other way. For example, Ohnishi et al. (2019) compute the TD(0) loss from the online network and add a term in the loss to constrain the predictions of the online network for the next state-action pair (s', a') to remain close to the one predicted by the target network. Piché et al. (2021; 2023) develop a similar approach, enforcing similar values for the state-action pair (s, a). Lindström et al. (2025) show that the target network can be removed after a pretraining phase in which they rely on expert demonstrations.

Many regularization techniques have been developed, attempting to combat the performance drop that occurs when removing the target network. We stress that our approach is orthogonal to these regularization techniques and we show in Section 5 that our method improves the performance of target-free methods equipped with these advancements. Li & Pathak (2021) encode the input of the Q-network with learned Fourier features. While this approach seems promising, the authors acknowledge that the performance degrades for high-dimensional problems. Shao et al. (2022) remove the target-network and search for an action that maximizes the Q-network more than the action proposed by the policy. Searching for a better action requires additional resources and is only relevant for actor-critic algorithms. Kim et al. (2019) leverage the MellowMax operator to get rid of the target network. However, the temperature parameter needs to be tuned (Kim, 2020), which increases the compute budget, and a follow-up work demonstrates that the reintegration of the target network is beneficial (Gan et al., 2021). Finally, Bhatt et al. (2024) point out the importance of using batch normalization (Ioffe & Szegedy, 2015) to address the distribution shift of the input given to the critic. Our investigation reveals that it degrades the performance in a discrete action setting (see Figure 12, right).

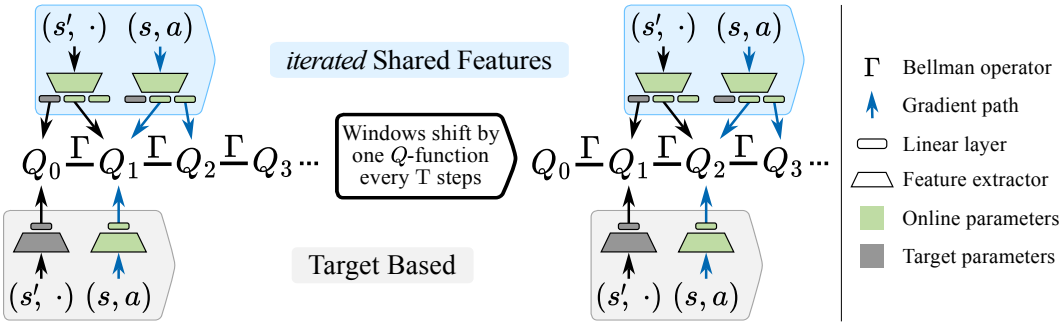

Figure 2: Comparison of the training path defined by the target networks obtained after each target update during training between the target-based approach and our approach (*iterated* Shared Features). While both approaches wait T training steps before shifting by one Q-function, our approach already considers the following Bellman iterations using multiple heads, where each head represents the Bellman iteration of the previous head (right).

The idea of learning multiple Bellman iterations in parallel has been introduced by Schmitt et al. (2022). The authors demonstrate convergence guarantees in the case of linear function approximation. Then, Vincent et al. (2024) used this approach to learn a recurrent hypernetwork generating a sequence of Q-functions where each Q-function approximates the Bellman iteration of the previous Q-function. Finally, Vincent et al. (2025) introduced iterated Q-Network as a far-sighted version of DQN that learns the K following Bellman iterations in parallel instead of only learning the following one. In this work, we propose to leverage the potential of i-QN to boost the learning speed of target-free algorithms.

4 Method

Our goal is to design a new algorithm that improves the learning speed of target-free value-based RL methods without significantly increasing the number of parameters used by the Q-networks. For that, we consider a *single* Q-network parameterized with $K+1$ heads. We note ω_k the parameters of the k^{th} head, ω the shared parameters, and define $\theta = (\omega, \omega_0, .., \omega_K)$ and $\theta_k = (\omega, \omega_k)$. Following Vincent et al. (2025), for a sample $d = (s, a, r, s')$, the training loss is

$$\mathcal{L}_d^{\text{iS-QN}}(\theta) = \sum_{k=1}^{K} \mathcal{L}_d^{\text{QN}}(\theta_k, \theta_{k-1}), \tag{1}$$

where $\mathcal{L}_d^{\text{QN}}$ can be chosen from any temporal-difference learning algorithm. For instance, DQN uses $\mathcal{L}_d^{\text{QN}}(\theta_k, \theta_{k-1}) = (\lceil r + \gamma \max_{a'} Q_{\theta_{k-1}}(s', a') \rceil - Q_{\theta_k}(s, a))^2$, where $\lceil \cdot \rceil$ indicates a stop gradient operation. We stress that ω_0 is not learned. However, every T steps, each ω_k is updated to ω_{k+1}, similarly to the target update step in DQN. That way, iS-QN allows to learn K Bellman iterations in parallel while only requiring a small amount of additional parameters on top of a target-free approach. Indeed, in the general case the size of each head ω_k is negligible compared to the size of shared parameters ω. Algorithm 1 summarizes the changes brought to the pseudo-code of DQN to implement our approach.

In Figure 2, we compare the training paths defined by the Q-functions obtained after each target update of our approach (top) and the target-based approach (bottom). For each given sample, the target-based approach learns only 1 Bellman iteration at a time and proceeds to the following one after T training steps. In contrast, our approach (iterated Shared Features) learns several consecutive Bellman iterations for each given sample. The considered window also moves forward every T training steps. Similarly to the target-based and target-free approaches, the online parameters are updated with the gradient computed through the forward pass of the state-action pair (s, a), as indicated with a blue arrow. In Figure 2, we depict our approach with $K = 2$. However, the number of heads can be increased at minimal cost. We note that the first Q-function is considered fixed in this

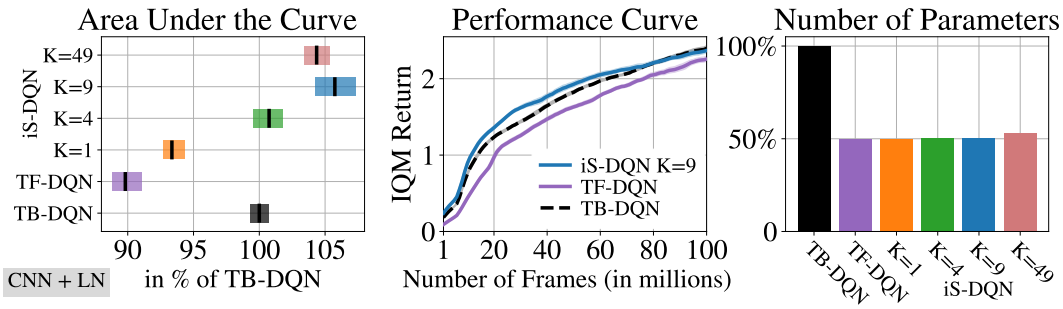

Figure 3: Reducing the performance gap in Online RL on 15 **Atari** games with the CNN architecture and LayerNorm (LN). While removing the target network leads to a 10% drop in AUC (left), our approach, iS-DQN $K = 9$, not only closes the gap but also improves over the target-based approach by 6%. Importantly, iS-DQN uses a comparable number of parameters to the target-free approach.

representation, even if the head is the only frozen element and the previous layers are shared with the other learned Q-estimates. Interestingly, the target-free approach can also be depicted in this figure. Indeed, not using a target network is equivalent to updating the target network to the online network after each training step. Consequently, the target-free approach can be understood as the target-based representation with a window shifting at every step. Therefore, the target-free approach passes through the Bellman iterations faster, creating instabilities as the optimization landscape may direct the training path toward undesirable Q-functions.

In the following, we apply iterated Shared Features to several target-based approaches on multiple RL settings, demonstrating that our approach reduces the gap between target-free and target-based methods. For each algorithm A, we note TB-A as its target-based version, TF-A as its target-free version, and iS-A as our approach, where "iS" stands for iterated Shared. Importantly, we incorporate the insights provided by Gallici et al. (2025) to use LayerNorm (Ba et al., 2016) in the architecture of the Q-networks as we found it beneficial, even for the target-based approach.

5 Experiments

We evaluate our approach in both online and offline RL scenarios to demonstrate that iS-QN can enhance the learning speed of target-free methods. We focus on the learning speed because, in this work, we are interested in the sample efficiency of target-free methods. We use the Area Under the Curve (AUC) to measure the learning speed. The AUC has the benefit of depending less on the training length compared to the end performance, as it accounts for the performance during the entire training. It also favors algorithms that constantly improve during training over those that only emerge at the end of training, thus discounting algorithms that require many samples to perform well. In each experiment, we report the AUC of each algorithm, normalized by the AUC of the target-based approach, to facilitate comparison. By normalizing the AUCs, the resulting metric can also be interpreted as the average performance gap observed during training between the considered approach and the target-based approach. We use the Inter-Quantile Mean (IQM) and 95% stratified bootstrapped confidence intervals to allow for more robust statistics as advocated by Agarwal et al. (2021). The IQMs are computed over 5 seeds per Atari game. 15 games are used for the experiments on the CNN architecture, and 10 games for the experiments on the IMPALA architecture to reduce the computational budget. Importantly, all hyperparameters are kept untouched with respect to the standard values (Castro et al., 2018), only the architecture is modified as described in Section 4. Extensive details about the selection process of the Atari games, the metrics computation, the hyperparameters, and the individual learning curves are reported in Appendix.

5.1 Online Discrete Control

First, we evaluate iS-DQN on 15 Atari games (Bellemare et al., 2013) with the vanilla CNN architecture (Mnih et al., 2015) equipped with LayerNorm. As expected, the target-free approach yields an AUC 10% smaller than the target-based approach, as shown in Figure 3 (left). This performance

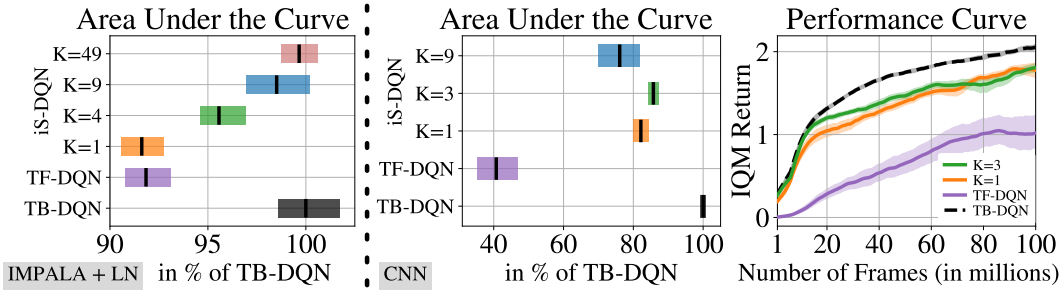

Figure 4: **Left**: Reducing the performance gap in Online RL on 10 **Atari** games with the IMPALA architecture and LayerNorm (LN). Similarly to the results with the CNN architecture, iS-DQN bridges the gap between the target-free and target-based approaches. **Right**: Reducing the performance gap in Online RL on 15 **Atari** games with the CNN architecture. Removing the target network of the vanilla DQN algorithm results in a 60% performance drop ($100\% - 40\%$). By using iS-DQN with $K = 3$, the performance drop is divided by 4 ($100\% - 85\% = 15\% = 60\%/4$), thereby confirming the benefit of sharing features and learning multiple Bellman iterations.

drop is constant across the training, see Figure 3 (middle). Interestingly, iS-DQN $K = 1$ improves over TF-DQN by simply storing an old copy of the last linear head. As more Bellman iterations are learned in parallel, the performance gap between iS-DQN and TB-DQN shrinks. Remarkably, iS-DQN $K = 9$ even outperforms the target-based approach by 6% in AUC. Figure 3 (right) testifies that this performance boost is achieved with approximately half of the parameters used by the target-based approach. We note a slight decline in performance for iS-DQN $K = 49$. We conjecture that this is due to the shared feature representation not being rich enough to enable the network to learn 49 Bellman iterations in parallel.

Our evaluation with the IMPALA architecture (Espeholt et al., 2018) with LayerNorm confirms the ability of iS-DQN to reduce the performance gap between target-free and target-based approaches. Indeed, Figure 4 (left) indicates that removing the target network leads to an 8% performance drop while iS-DQN annuls the performance gap as more Bellman iterations are learned in parallel, i.e., as K increases. Interestingly, as opposed to the CNN architecture, increasing the number of heads to learn 49 Bellman iterations in parallel is beneficial. We believe this is due to IMPALA architecture's ability to produce a richer representation than the CNN architecture, thereby allowing more Bellman iterations to be approximated with a linear mapping. The plots of the performance curve and the number of parameters are similar to the ones shown for the CNN architecture. We report them in Figure 10.

Finally, we confirm the benefit of our approach by removing the normalization layers for all algorithms with the CNN architecture in Figure 4 (right). We observe a major drop in performance for TF-DQN, leading to 60% performance gap ($100\% - 40\%$). Notably, iS-DQN $K = 1$ reduces this performance gap to 18% ($100\% - 82\%$). This highlights the potential of simply storing the last linear layer and using the features of the online network to build a lightweight regression target. While increasing the number of learned Bellman iterations to 3 brings a benefit, the performances are slightly decreasing for higher values of K, indicating that LayerNorm is beneficial to provide useful representations when considering a higher number of linear heads.

5.2 Offline Discrete Control

We consider an offline RL setting in which the agent has access to 10% of the dataset collected by a vanilla DQN agent trained with a budget of 200 million frames (Agarwal et al., 2020), sampled uniformly. We adapt the loss for learning each Bellman iteration to the one proposed by Kumar et al. (2020b). This leads to an iterated version of Conservative Q-Learning (CQL). This time, the performance is reported as a function of the number of gradient steps as the experiment is performed offline. In Figure 5, iS-CQL $K = 9$ reduces the performance gap by 20 percentage points, ending up

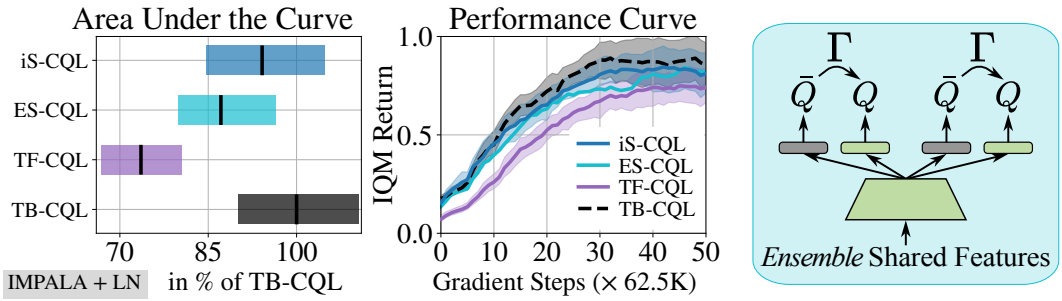

Figure 5: Reducing the performance gap in Offline RL on 10 **Atari** games with the IMPALA architecture and LayerNorm (LN). Our approach, iS-CQL, shrinks the performance gap from 26% to 6%. Interestingly, applying the idea of sharing parameters to Ensemble DQN (*Ensemble* Shared Features, ES-CQL) also reduces the performance gap, demonstrating that this idea is not limited to iterated Q-learning and can be applied to other target-based approaches.

with a performance gap of 6% compared to 26% for TF-CQL. Additionally, we evaluate another way of sharing features to show that this idea is not limited to iterated Q-learning. Instead of building a chain of Q-functions represented by linear heads, we define an ensemble of pairs of linear heads. Each pair contains a frozen head representing a target network $\bar{Q}$ that is used to train the learned head representing the associated online network Q, as depicted in Figure 5 (right). We evaluate this variant that we call Ensemble Shared Features (ES-CQL), with 5 pairs of heads, i.e. 10 heads, to match the number of heads used by iS-CQL $K = 9$, as the number of heads of iS-CQL is always equal to $K + 1$. Importantly, ES-CQL also outperforms TF-CQL, reinforcing the idea that sharing parameters and using linear heads is a fruitful direction.

5.3 Why is iS-QN improving over target-free approaches?

We now provide some insights to understand why iS-QN reduces the performance gap between target-free and target-based approaches. For that, we examine how the state representation is affected by the different losses. We report the effective rank (srank) of the features in the penultimate layer as a proxy for the representation expressivity (Kumar et al., 2020a) in Figure 6. Interestingly, the srank obtained by iS-DQN $K = 1$ is closer to the srank of TB-DQN than the srank of TF-DQN, which further demonstrates the benefit of using the last linear layer to construct the target. Notably, learning $K = 9$ Bellman iterations in parallel increases the representation capacity of the network by a large margin. This behavior is also visible in the offline setting, where iS-CQL reaches a similar srank as the target-based approach at the end of the training (see Figure 6, right). This confirms the benefit of iS-QN to foster a richer representation capacity.

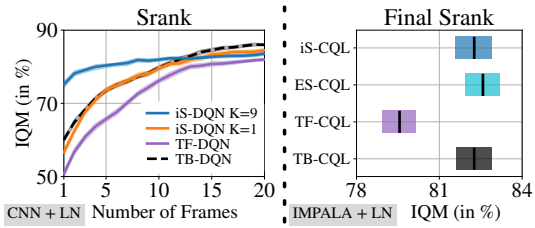

Figure 6: The effective rank (srank) of the features in the penultimate layer is higher for iS-QN, resulting in a higher expressivity of the Q-network.

6 Conclusion

We introduced a simple yet efficient method for mitigating the performance drop that occurs when removing the target network in deep value-based reinforcement learning, while maintaining a low memory footprint. This is made possible by storing a copy of the last linear layer of the online network and using the features of the online network as input to this frozen linear head to construct the regression target. From there, more heads can be added to learn multiple Bellman iterations in parallel. We demonstrated that this new algorithm, iterated Shared Q-Networks, improves over the target-free approach and yields higher returns when the number of heads increases. We believe that generalizing iS-QN to environments with continuous action spaces is a promising direction for future work.

Acknowledgments

This work was funded by the German Federal Ministry of Education and Research (BMBF) (Project: 01IS22078). This work was also funded by Hessian.ai through the project 'The Third Wave of Artificial Intelligence – 3AI' by the Ministry for Science and Arts of the state of Hessen, by the grant "Einrichtung eines Labors des Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI) an der Technischen Universität Darmstadt", and by the Hessian Ministry of Higher Education, Research, Science and the Arts (HMWK). The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project b187cb. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683.

References

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

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, 2021.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *Stat*, 2016.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 2013.

Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. Crossq: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. *International Conference on Learning Representations*, 2024.

Ioannis Boukas, Damien Ernst, Thibaut Théate, Adrien Bolland, Alexandre Huynen, Martin Buchwald, Christelle Wynants, and Bertrand Cornélusse. A deep reinforcement learning framework for continuous intraday market bidding. *Machine Learning*, 2021.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. *JAX: composable transformations of Python+NumPy programs*, 2018.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Pablo Samuel Castro, Subhodeep Moitra, Carles Gelada, Saurabh Kumar, and Marc G Bellemare. Dopamine: A research framework for deep reinforcement learning. *arXiv preprint arXiv:1812.06110*, 2018.

Esraa Elelimy, Brett Daley, Andrew Patterson, Marlos C Machado, Adam White, and Martha White. Deep reinforcement learning with gradient eligibility traces. In *Reinforcement Learning Conference*, 2025.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, 2018.

Matteo Gallici, Mattie Fellows, Benjamin Ellis, Bartomeu Pou, Ivan Masmitja, Jakob Nicolaus Foerster, and Mario Martin. Simplifying deep temporal difference learning. In *International Conference on Learning Representations*, 2025.

Yaozhong Gan, Zhe Zhang, and Xiaoyang Tan. Stabilizing q learning via soft mellowmax operator. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Laura Graesser, Utku Evci, Erich Elsen, and Pablo Samuel Castro. The state of sparse training in deep reinforcement learning. In *International Conference on Machine Learning*, 2022.

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Thomas Paine, Sergio Gómez, Konrad Zolna, Rishabh Agarwal, Josh S Merel, Daniel J Mankowitz, Cosmin Paduraru, et al. Rl unplugged: A suite of benchmarks for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2020.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 2015.

Seungchan Kim. Adaptive tuning of temperature in mellowmax using meta-gradients. *Master Thesis, Brown University*, 2020.

Seungchan Kim, Kavosh Asadi, Michael Littman, and George Konidaris. Deepmellow: removing the need for a target network in deep q-learning. In *International Joint Conference on Artificial Intelligence*, 2019.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2020a.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2020b.

Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical normalization for scalable deep reinforcement learning. *International Conference on Machine Learning*, 2025.

Alexander Li and Deepak Pathak. Functional regularization for reinforcement learning via learned fourier features. In *Advances in Neural Information Processing Systems*, 2021.

Alexander Lindström, Arunselvan Ramaswamy, and Karl-Johan Grinnemo. Pre-training deep q-networks eliminates the need for target networks: An empirical study. In *The 14th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, 2025.

Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 2018.

Hamid Maei, Csaba Szepesvári, Shalabh Bhatnagar, Doina Precup, David Silver, and Richard S Sutton. Convergent temporal-difference learning with arbitrary smooth function approximation. In *Advances in Neural Information Processing Systems*, 2009.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample-efficient continuous control. *Advances in Neural Information Processing Systems*, 2024.

Johan Samir Obando Ceron, Ghada Sokar, Timon Willi, Clare Lyle, Jesse Farebrother, Jakob Nicolaus Foerster, Gintare Karolina Dziugaite, Doina Precup, and Pablo Samuel Castro. Mixtures of experts unlock parameter scaling for deep RL. In *International Conference on Machine Learning*, 2024.

Shota Ohnishi, Eiji Uchibe, Yotaro Yamaguchi, Kosuke Nakanishi, Yuji Yasui, and Shin Ishii. Constrained deep q-learning gradually approaching ordinary q-learning. *Frontiers in Neurorobotics*, 2019.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *Advances in Neural Information Processing Systems*, 2016.

Georg Ostrovski, Pablo Samuel Castro, and Will Dabney. The difficulty of passive learning in deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2021.

Daniel Palenicek, Florian Vogt, and Jan Peters. Scaling off-policy reinforcement learning with batch and weight normalization, 2025.

Andrew Patterson, Adam White, and Martha White. A generalized projected bellman error for off-policy value estimation in reinforcement learning. *Journal of Machine Learning Research*, 2022.

Rodrigo Pérez-Dattari, Carlos Celemin, Javier Ruiz-del Solar, and Jens Kober. Continuous control for high-dimensional state spaces: An interactive learning approach. In *International Conference on Robotics and Automation*, 2019.

Alexandre Piché, Joseph Marino, Gian Maria Marconi, Valentin Thomas, Christopher Pal, and Mohammad Emtiyaz Khan. Beyond target networks: Improving deep q-learning with functional regularization. In *NeurIPS Workshop on Deep Reinforcement Learning*, 2021.

Alexandre Piché, Valentin Thomas, Joseph Marino, Rafael Pardinas, Gian Maria Marconi, Christopher Pal, and Mohammad Emtiyaz Khan. Bridging the gap between target networks and functional regularization. *Transactions on Machine Learning Research*, 2023.

Simon Schmitt, John Shawe-Taylor, and Hado Van Hasselt. Chaining value functions for off-policy learning. In *AAAI Conference on Artificial Intelligence*, 2022.

Tim Schneider, Cristiana de Farias, Roberto Calandra, Liming Chen, and Jan Peters. Active perception for tactile sensing: A task-agnostic attention-based approach, 2025.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 2020.

Max Schwarzer, Johan Samir Obando Ceron, Aaron Courville, Marc G Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: Human-level atari with human-level efficiency. In *International Conference on Machine Learning*, 2023.

Lin Shao, Yifan You, Mengyuan Yan, Shenli Yuan, Qingyun Sun, and Jeannette Bohg. Grac: Self-guided and self-regularized actor-critic. In *Conference on Robot Learning*, 2022.

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

Richard S. Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *International Conference on Machine Learning*, 2009.

Hado van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *CoRR*, abs/1812.02648, 2018.

Gautham Vasan, Mohamed Elsayed, Alireza Azimi, Jiamin He, Fahim Shariar, Colin Bellinger, Martha White, and A. Rupam Mahmood. Deep policy gradient methods without batch updates, target networks, or replay buffers. In *Advances in Neural Information Processing Systems*, 2024.

Théo Vincent, Alberto Maria Metelli, Boris Belousov, Jan Peters, Marcello Restelli, and Carlo D'Eramo. Parameterized projected bellman operator. In *AAAI Conference on Artificial Intelligence*, 2024.

Théo Vincent, Daniel Palenicek, Boris Belousov, Jan Peters, and Carlo D'Eramo. Iterated q-network: Beyond one-step bellman updates in deep reinforcement learning. *Transactions on Machine Learning Research*, 2025.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 1992.

Guang Yang, Yang Li, Di'an Fei, Tian Huang, Qingyun Li, and Xingguo Chen. Dhqn: a stable approach to remove target network from deep q-learning network. In *International Conference on Tools with Artificial Intelligence*, 2021.

Shangtong Zhang, Hengshuai Yao, and Shimon Whiteson. Breaking the deadly triad with a target network. In *International Conference on Machine Learning*, 2021.

Table of Contents

A Experiment Setup

We build our codebase following Machado et al. (2018) standards and taking inspiration from Castro et al. (2018) codebase. Namely, we use the *game over* signal to terminate an episode instead of the life signal. The input given to the neural network is a concatenation of 4 frames in grayscale of dimension 84 by 84. To get a new frame, we sample 4 frames from the Gym environment (Brockman et al., 2016) configured with no frame-skip, and apply a max pooling operation on the 2 last grayscale frames. We use sticky actions to make the environment stochastic (with $p = 0.25$).

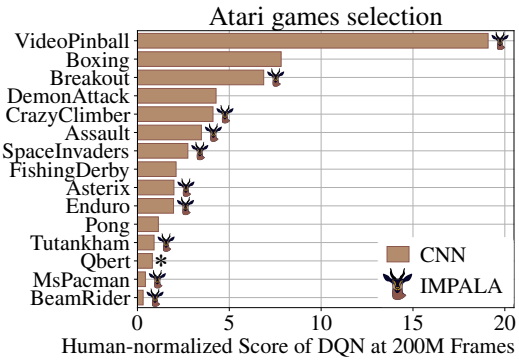

Figure 7: The selected Atari games for different settings cover a variety of normalized returns of DQN at 200M frames, showcasing their diversity. To lower the computational budget of the experiments with the IMPALA architecture, we reduced the set of games to 10 by removing 5 games while maintaining diversity.

Atari games selection Our evaluations on the CNN architecture were performed on the 15 games recommended by Graesser et al. (2022). They were chosen for their diversity of Human-normalized score that DQN reaches after being trained on 200 million frames, as shown in Figure 7. As the IMPALA architecture increases the training length, we removed 5 games while maintaining diversity in the final scores to reduce the computational budget. For the offline experiment, we used the datasets provided by Gulcehre et al. (2020). As the game *Tutankham* is not available in the released dataset, we replaced it with *Qbert*, indicated with an asterisk in Figure 7.

Computing the Area Under the Curve For each experiment, we report the normalized IQM AUC. For that, we first compute the undiscounted return obtained for each epoch, averaged over the episodes, as advocated by Machado et al. (2018). Then, we sum the human-normalized returns over the epochs and compute the IQM and 95% stratified bootstrap confidence intervals over the seeds and games. Finally, we divide the obtained values by the IQM of the target-based approach to facilitate the comparisons. The human-normalized scores are computed from human and random scores that were reported in Schrittwieser et al. (2020). As discussed in Section 5, the normalized AUCs can also be interpreted as the average performance gap between the considered algorithm and the target-based approach. Indeed, dividing the two sums of performances across the training is equivalent to dividing the two averages of performances across the training because the normalizing factors cancel out.

Sampling actions Following Vincent et al. (2025), at each environment interaction, we sample an action from a single head chosen uniformly as shown in Line 3 in Algorithm 1. The authors motivate this choice by arguing that it allows each Q-function to interact with the environment, thereby avoiding passive learning, identified by Ostrovski et al. (2021). This choice is further justified by an ablation study (see Figure 19 in Vincent et al. (2025)) demonstrating a stronger performance against another sampling strategy consisting of sampling one head for each episode, as proposed in Osband et al. (2016). Therefore, we keep this design choice.

Aggregating individual losses In Equation 1, we define the loss of iS-QN as the sum of losses over each Bellman iteration. Other ways of aggregating the losses are possible. Nonetheless, we decided to stick to the version proposed by Vincent et al. (2025) and leave this investigation for future work. While it is true that taking the sum of temporal differences increases the magnitude of the loss, it has a different impact on the updates than simply multiplying the learning rate by the number of terms in the loss. Indeed, the Adam optimizer (Kingma & Ba, 2015) first normalizes the gradient with a running statistic before applying the learning rate. Therefore, changing the aggregation mechanism has a greater impact on the direction of the update than on its magnitude. This is why we do not compare iS-QN against baselines instantiated with different learning rates.

B List of Hyperparameters and Algorithm

Our codebase is written in Jax (Bradbury et al., 2018). The details of hyperparameters used for the Atari experiments are provided in Table 1.

Table 1: Summary of the shared hyperparameters used for the Atari experiments. We note $\text{Conv}_{a,b}^{d}C$ a $2D$ convolutional layer with C filters of size $a \times b$ and of stride d, and FC E a fully connected layer with E neurons. The CNN architecture is described here. Please refer to Espeholt et al. (2018) for details on the IMPALA architecture.

Environment	
Discount factor γ	0.99
Horizon H	27 000
Full action space	No
Reward clipping	clip$(-1, 1)$
All experiments	
Batch size	32
Torso architecture	$\text{Conv}_{8,8}^{4}32$ $-\text{Conv}_{4,4}^{2}64$ $-\text{Conv}_{3,3}^{1}64$
Head architecture	FC 512 $-$FC $n_{\mathcal{A}}$ (TB-QN, TF-QN) $-$FC $(K+1)\cdot n_{\mathcal{A}}$ (iS-QN)
Activations	ReLU
Offline experiments	
Number of training steps per epochs	62 500
Target update period T	2 000
Dataset size	5 000 000
Learning rate	5×10^{-5}
Adam ϵ	3.125×10^{-4}

Online experiments	
Number of training steps per epochs	250 000
Target update period T	8 000
Type of the replay buffer $\mathcal{D}$	FIFO
Initial number of samples in $\mathcal{D}$	20 000
Maximum number of samples in $\mathcal{D}$	1 000 000
Gradient step period G	4
Starting ϵ	1
Ending ϵ	0.01
ϵ linear decay duration	250 000
Batch size	32
Learning rate	6.25×10^{-5}
Adam ϵ	1.5×10^{-4}

Algorithm 1 *iterated* Shared Deep Q-Network (iS-DQN). Modifications to DQN are in purple.

Initialize a network Q_θ with $K+1$ heads, where each head is defined by the parameters ω_k. We note $\theta_k = (\omega, \omega_k)$, and ω the shared parameters such that $\theta = (\omega, \omega_0, .., \omega_K)$. $\mathcal{D}$ is an empty replay buffer.
Repeat
  Set $u \sim \texttt{Uniform}(\{1, .., K\})$.
  Take action $a \sim \epsilon$-greedy.$(Q_{\theta_u}(s, \cdot))$; Observe reward r, next state s'.
  Update $\mathcal{D} \leftarrow \mathcal{D} \bigcup \{(s, a, r, s')\}$.
  every G **steps**
    Sample a mini-batch $\mathcal{B} = \{(s, a, r, s')\}$ from $\mathcal{D}$.
    Store $[Q_0(s', \cdot), .., Q_K(s', \cdot)] \leftarrow Q_\theta(s', \cdot)$ and $[Q_0(s, a), .., Q_K(s, a)] \leftarrow Q_\theta(s, a)$.
    Compute the loss ▷ $\lceil \cdot \rceil$ indicates a stop gradient operation.
    $\mathcal{L}^{\text{iS-QN}} = \sum_{(s,a,r,s') \in \mathcal{B}} \sum_{k=1}^{K} (\lceil r + \gamma \max_{a'} Q_{k-1}(s', a') \rceil - Q_k(s, a))^2$.
    Update θ from $\nabla_\theta \mathcal{L}^{\text{iS-QN}}$.
  every T **steps**
    Update $\omega_k \leftarrow \omega_{k+1}$, for $k \in \{0, \ldots, K-1\}$.

C Individual Learning Curves

C.1 Deep Q-Network with CNN and LayerNorm

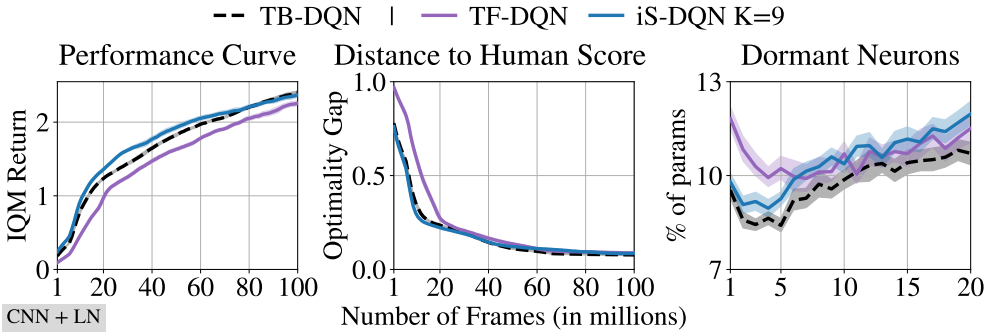

Figure 8: Reducing the performance gap in Online RL on 15 **Atari** games with the CNN architecture and LayerNorm (LN). **Left**: iS-DQN $K = 9$ not only reduces the performance gap but outperforms the target-based approach. **Middle**: iS-DQN annuls the performance gap for the game where the score is below the human level. **Right**: iS-DQN exhibits a lower amount of dormant neurons at the beginning of the training compared to the target-free approach.

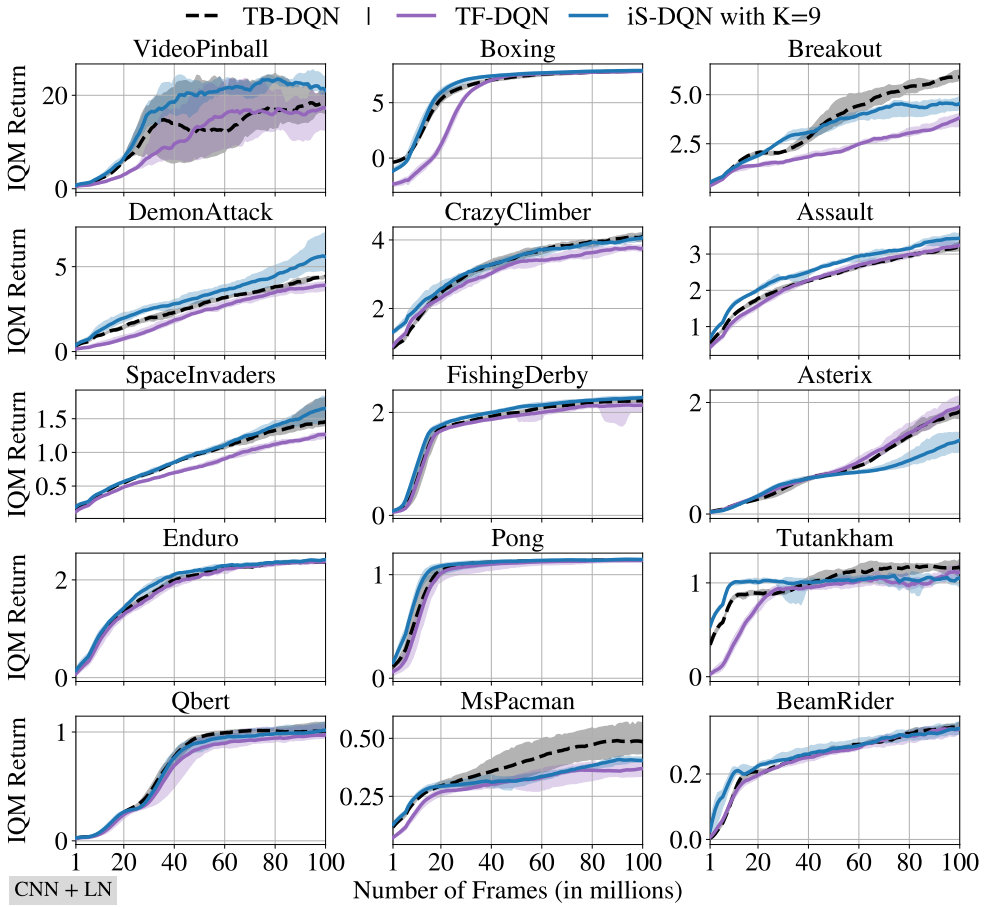

Figure 9: Per game training curves of iS-DQN and the relevant baselines with the CNN architecture and LayerNorm (LN). Except on *Asterix*, our approach outperforms or is on par with the target-free approach (TF-DQN).

C.2 Deep Q-Network with IMPALA and LayerNorm

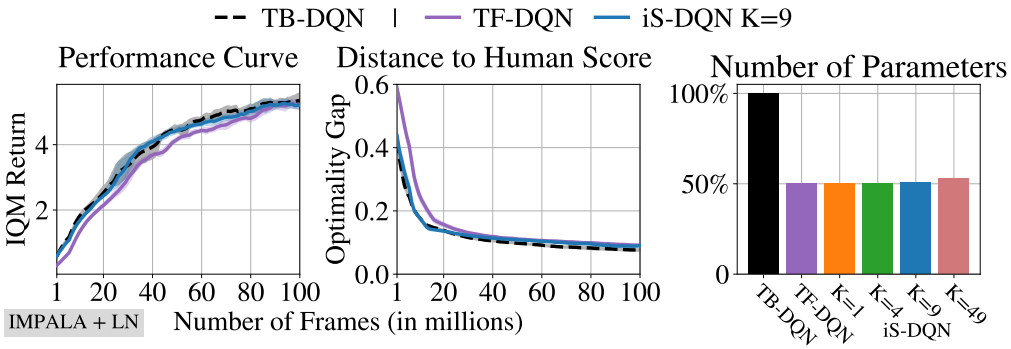

Figure 10: Reducing the performance gap in Online RL on 10 **Atari** games with the IMPALA architecture and LayerNorm (LN). **Left**: iS-DQN $K = 9$ is outperforms the target-free approach. **Middle**: iS-DQN annuls the performance gap for the game where the score is below the human level. **Right**: iS-DQN requires significantly fewer parameters than the target-based approach while reaching similar performances.

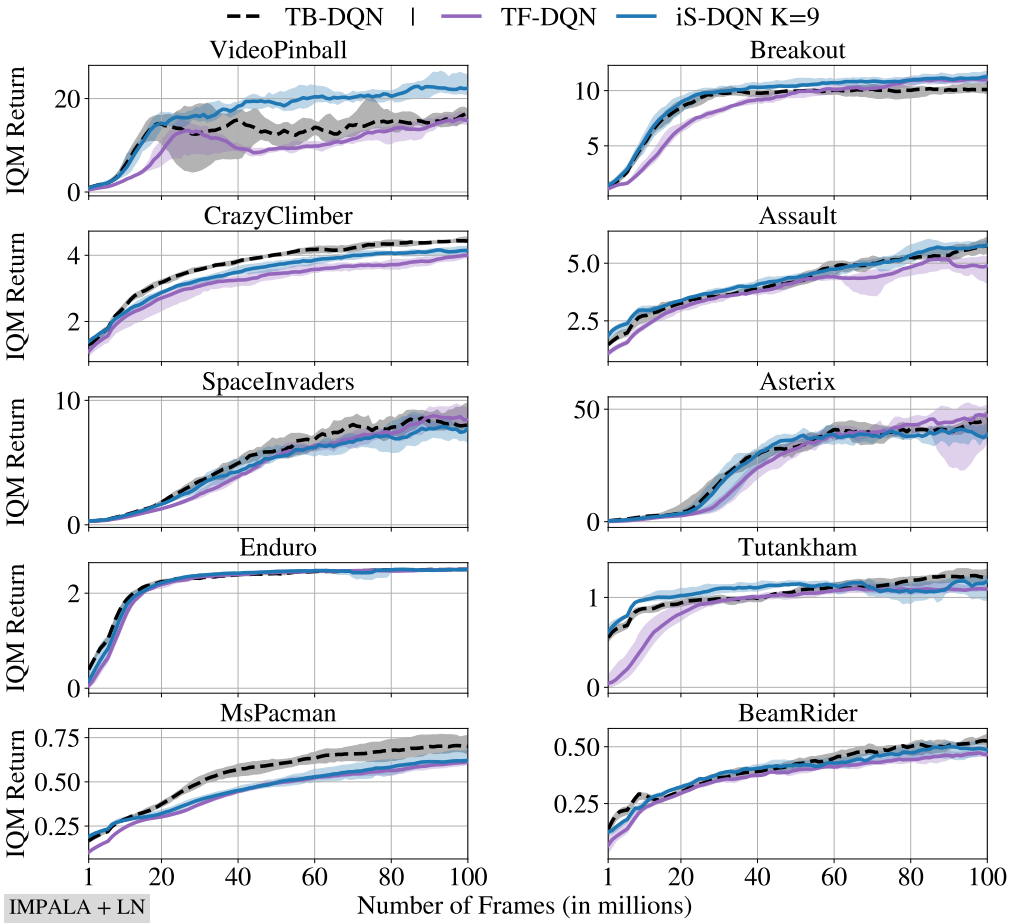

Figure 11: Per game training curves of iS-DQN and the relevant baselines with the IMPALA architecture and LayerNorm (LN). Our approach outperforms or is on par with the target-free approach (TF-DQN) on all games.

C.3 Deep Q-Network with CNN

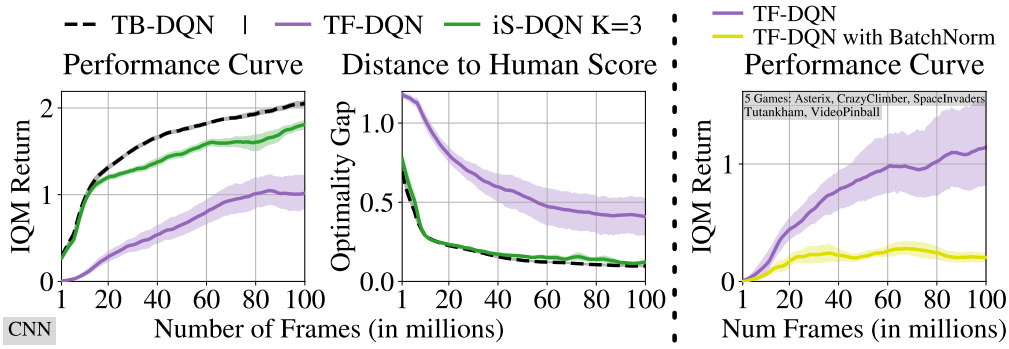

Figure 12: Reducing the performance gap in Online RL on 15 **Atari** games with the CNN architecture. **Left**: iS-DQN $K = 3$ significantly reduces the performance gap between the target-free and target-based approaches. **Middle**: iS-DQN annuls the performance gap for the game where the score is below the human level. **Right**: Including BatchNorm in the architecture damages the performance on the 5 considered games. This is why we do not consider including BatchNorm in our experiments.

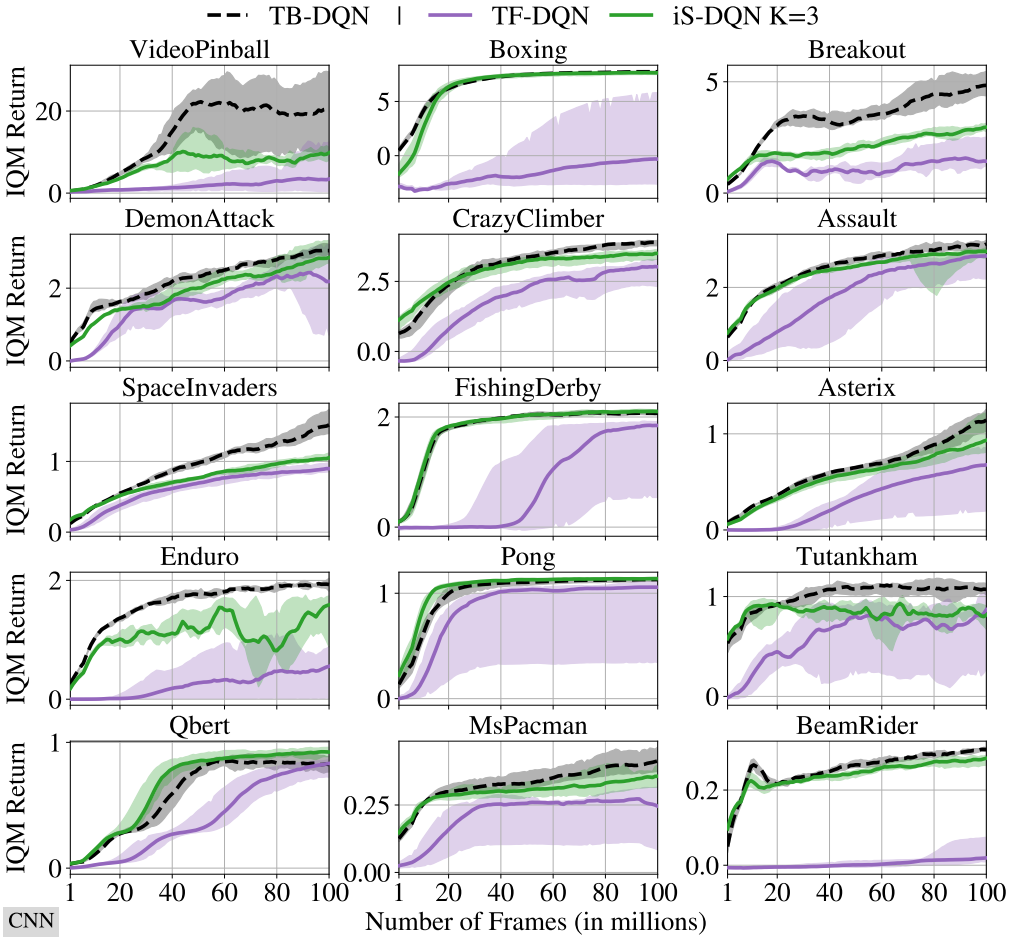

Figure 13: Per game training curves of iS-DQN and the relevant baselines with the CNN architecture. Remarkably, our approach outperforms the target-free approach (TF-DQN) on all games.

C.4 Conservative Q-Learning with IMPALA and LayerNorm

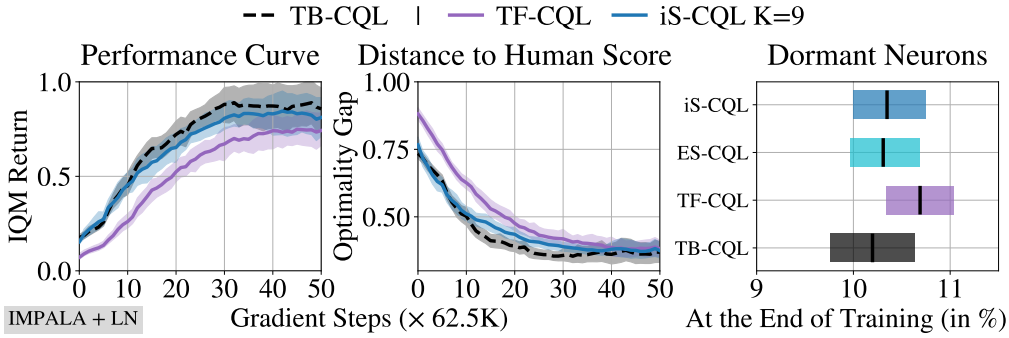

Figure 14: Reducing the performance gap in Offline RL on 10 **Atari** games with the IMPALA architecture and LayerNorm (LN). **Left**: iS-CQL $K = 9$ significantly reduces the performance gap between the target-free and target-based approaches. **Middle**: iS-CQL significantly reduces the performance gap for the game where the score is below the human level. **Right**: Just like ES-CQL, iS-CQL exhibits a lower amount of dormant neurons at the end of the training compared to the target-free approach. However, we note that the confidence intervals are overlapping.

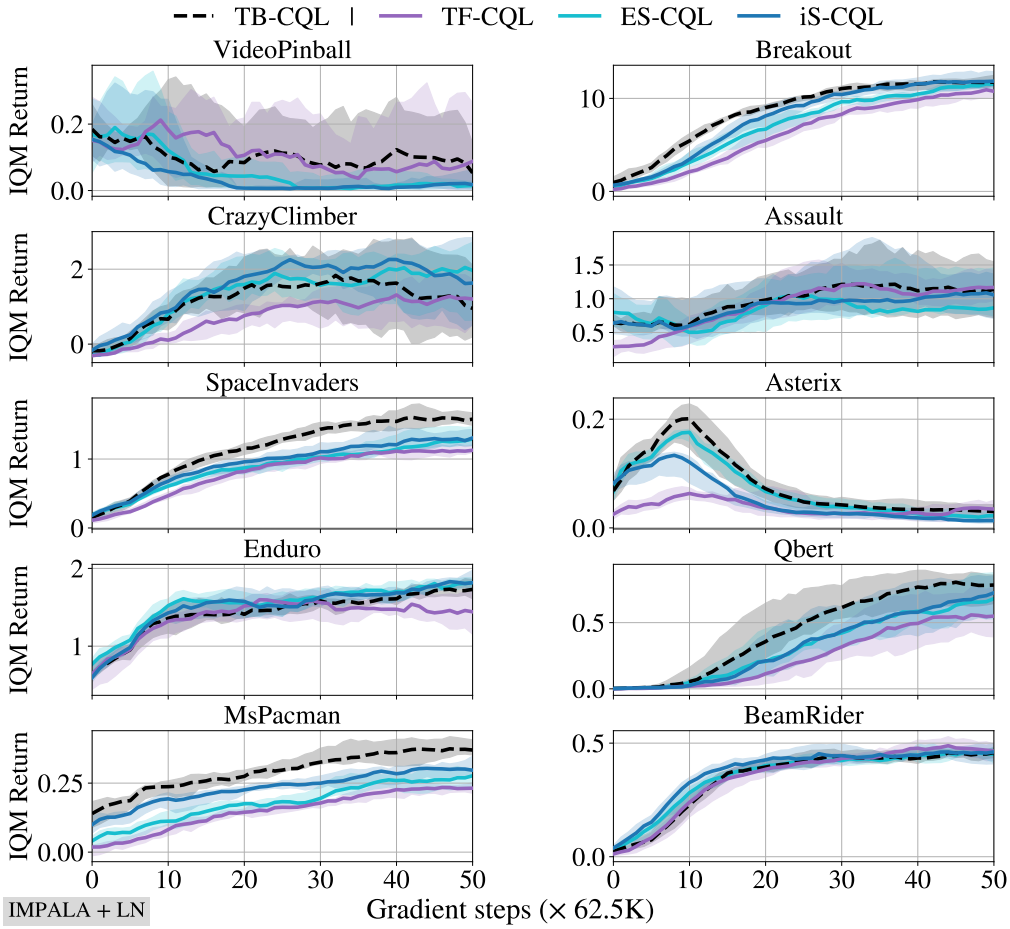

Figure 15: Per game training curves of iS-CQL and the relevant baselines with the IMPALA architecture and LayerNorm (LN). Except on *VideoPinball*, our approach outperforms or is on par with the target-free approach (TF-DQN).