Transferability of Sparse Subnetworks in Deep Reinforcement Learning

Sebastian Pusch[1] (s5488079), Mihai Damian[1] (s5508983), Yitong Lin Yang[1] (s5460611), Riandika Marshequila Dinu[1] (s5603838)
{s.pusch,m.damian,y.lin.39,riandika.marshequila.dinu}@student.rug.nl

[1]**University of Groningen - Reinforcement Learning Practical - Group 9**

Abstract

Sparse subnetworks such as lottery tickets can match or outperform dense performance in supervised learning, but it remains unclear whether such sparsified representations transfer across reinforcement learning (RL) tasks. We study the transferability of pruned subnetworks in value-based deep reinforcement learning by training Double Deep Q-Network (DDQN) agents on Atari environments and transferring sparse encoders between tasks. We compare subnetworks obtained via Lottery Ticket Hypothesis (LTH) pruning and Gradual Magnitude Pruning (GMP), and evaluate two target task adaptation regimes: Static transfer (no further pruning) and continued magnitude pruning during target training to restore plasticity. Across *Pong*, *Breakout*, and *Space Invaders*, we find that while winning tickets can exist within individual tasks, transferring sparse subnetworks across games yields limited or negative transfer relative to training from scratch, and reintroducing pruning during target task training further degrades performance compared to Static transfer. These results suggest that magnitude-based sparsity alone does not mitigate the representation-value misalignment that arises during cross-task transfer in DDQN, and that effective RL transfer requires mechanisms that preserve or re-establish encoder-head co-adaptation rather than solely relying on sparsity.

1 Introduction

Modern neural networks in supervised learning are typically heavily overparameterized, a phenomenon already noted in classic work such as Optimal Brain Damage (LeCun et al., 1990), which showed that many parameters can be removed with little loss in performance. The Lottery Ticket Hypothesis (LTH) (Frankle & Carbin, 2018) uses iterative magnitude pruning to identify sparse subnetworks—"winning tickets"—inside dense, randomly initialized networks, then resets their surviving weights to their original values and retrains them in isolation. These subnetworks can match or exceed the accuracy of the full model, often training faster and more reliably, and follow-up work shows that they can sometimes transfer across related datasets (**?**), suggesting that they encode meaningful inductive biases rather than merely overfitting to a single task.

In deep reinforcement learning, transfer has been far less successful. Sabatelli and Geurts (2021) examine transfer learning in Atari by pre-training a Dense Deep Q-Network on a source game and transferring the convolutional encoder to a target game while reinitializing the final value layer. They report limited or negative transfer and conjecture that the tight co-evolution of the feature extractor and value function approximator during training causes negative transfer: reinitializing the head breaks this coupling, leaving the pretrained features detached from the new value function. This highlights the importance of preserving representational alignment during transfer.

At the same time, recent work shows that sparsity itself can be effective in reinforcement learning (RL) when handled appropriately. Obando-Ceron et al. (2024) demonstrate that pruned value-based agents can match or surpass dense performance even at high sparsity levels, indicating that significant parameter reductions do not necessarily degrade policy quality. Todorov et al. (2025) further show that dynamically sparse methods such as Gradual Magnitude Pruning (GMP) improve robustness, plasticity, and generalization in multi-task RL settings, outperforming both dense agents and static sparse models. Together, these results suggest that sparse RL networks can perform well when sparsity is applied in a manner that preserves learning dynamics.

Despite these encouraging results, it remains unclear how sparsity interacts with transfer. Prior work has either examined transfer in dense agents (Sabatelli & Geurts, 2021) or studied sparsity in isolation (Obando-Ceron et al., 2024; Todorov et al., 2025), without considering whether sparse subnetworks identified on one RL task—such as winning tickets—can be effectively reused on another. In particular, the role of dynamic sparsification during adaptation to a new task has not been explored. Understanding this interaction is key to determining whether transferable sparse representations can be constructed in RL.

Then, our research question is: *To what extent do sparse networks discovered on a source reinforcement learning task transfer to related target tasks, and how does introducing sparsification during target task tuning affect transfer performance when compared against dense and sparse-on-target-only baselines?*

We believe that sparse subnetworks (winning tickets) identified on a source RL task encode task-relevant inductive bias that can partially transfer to a related target task. We hypothesize that reintroducing dynamic sparsification during target task training, via GMP, will lead to positive transfer relative to static sparse transfer and dense baselines, by restoring network plasticity and, at the same time, enabling re-alignment between transferred representations and the target value function.

To test this hypothesis, we adopt the following experimental approach. We use DDQN agents with a fixed convolutional architecture, and compare sparse masks obtained via Lottery Ticket Hypothesis pruning and Gradual Magnitude Pruning. Our main intervention is the reintroduction of magnitude-based sparsification during target task training, allowing us to compare static sparse transfer against continued pruning as a mechanism for restoring plasticity. Transfer performance is evaluated using the interquartile mean (IQM) of episodic return with bootstrapped confidence intervals, and quantifies transfer benefit via area ratio, for which we also report 95% bootstrap confidence intervals (CIs), relative to training from scratch and sparse-only baselines.

2 Experimental setup

2.1 Benchmarks

We evaluate our approach on three Atari environments from the Arcade Learning Environment (ALE) (Bellemare et al., 2013): *Pong*, *Breakout*, and *Space Invaders*. These games are single-task, fully observable environments and are widely used in reinforcement learning research (Obando-Ceron et al., 2024). *Pong* and *Breakout* share similar paddle–ball dynamics but differ in objectives and opponent structure, while *Space Invaders* introduces more complex enemy movement and firing patterns. Atari environments have also been commonly used in prior transfer studies (Sabatelli & Geurts, 2021).

2.2 Algorithms and Architecture

All experiments use Double Deep Q-Networks (DDQN) (Van Hasselt et al., 2016), which mitigates Q-value overestimation by decoupling action selection and evaluation. We employ a fixed convolutional architecture with three convolutional layers followed by a fully connected layer, and we do not vary the architecture to isolate the effects of sparsity and transfer.

Importantly, all pruning and transfer operations are applied exclusively to the convolutional encoder. The value head is always randomly reinitialized when transferring to a target task.

2.3 Pruning Methods

We investigate two magnitude-based pruning schedules, denoted using a concise method–sparsity notation.

Lottery Ticket Hypothesis (LTH) pruning (Frankle & Carbin, 2018) prunes the network in discrete rounds. At each round, low-magnitude weights are removed to reach a target sparsity level $X\%$, after which the remaining weights are reset to their initialization values θ_0 from a fixed rewind step $k = 8 \times 10^4$. This procedure is repeated until the final sparsity is reached. To reduce redundant computation, later pruning rounds are early-stopped when the Interquartile Mean (IQM) of episodic returns over the last w steps matches or exceeds the IQM over the first w steps of the initial (unpruned) training iteration. We refer to $w = 5 \times 10^3$ as the *rewind evaluation window*. Models obtained this way are denoted as `LTH{X}` (e.g., `LTH60`).

Gradual Magnitude Pruning (GMP) (Zhu & Gupta, 2017) increases sparsity continuously during training by progressively removing low-magnitude weights without resetting. Sparsity is annealed from 0% to a target level $X\%$, and resulting models are denoted as `GMP{X}` (e.g., `GMP75`).

Both methods use unstructured magnitude-based pruning and differ only in pruning schedule (discrete vs. continuous) and weight treatment (rewind vs. retain). We evaluate sparsity levels of 60% and 75%. To assess whether performance gains arise from structured subnetworks rather than sparsity alone, we include a `Random LTH` baseline with random pruning and random reinitialization.

2.4 Training, Transfer, and Evaluation

All experiments use five random seeds. Source-task agents are trained for 10 million environment steps, which is sufficient for DDQN convergence on Atari (Mnih et al., 2015). Dense agents trained without pruning (`Dense`) serve as the primary baseline.

For transfer, sparse subnetworks discovered on a source task initialize training on a target task. In the `Static` setting (e.g., `LTH60 Static`), transferred masks and weights are kept fixed during target-task training. In the `GMP90` setting (e.g., `LTH60 GMP90`), Gradual Magnitude Pruning is applied during target-task training to further increase sparsity to 90%. We also train target-task baselines from scratch, including `Dense` and `GMP{X}`.

Performance is measured using the Interquartile Mean (IQM) of episodic returns (Agarwal et al., 2021), reported with 95% bootstrap confidence intervals. As a complementary measure of transfer benefit, we compute the area ratio between transfer and from-scratch learning curves. To handle negative returns, learning curves are shifted by a constant offset before integration:

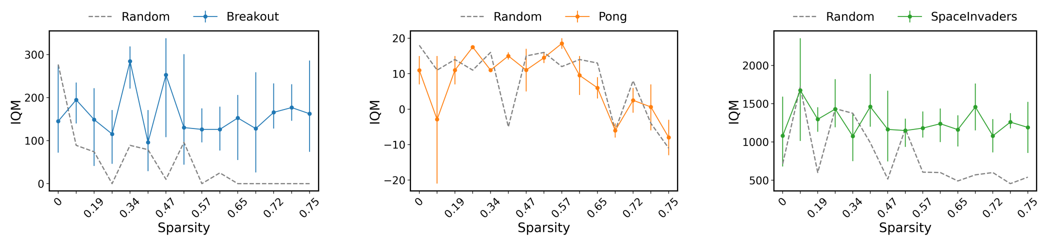
$$\tilde{R}(t) = R(t) + c, \qquad c = -\min_t R(t), \tag{1}$$

and the area ratio is defined as

$$r = \frac{A_{\text{transfer}} - A_{\text{scratch}}}{A_{\text{scratch}}}. \tag{2}$$

3 Results

We organize the results around our main research question: to what extent sparse subnetworks discovered on a source reinforcement learning task transfer to related target tasks, and whether allowing further sparsification during target task training improves transfer performance. We first establish the existence and stability of winning tickets on individual tasks, then analyze transfer performance relative to dense and sparse baselines, and finally examine ablations that isolate the role of pruning schedules and sparsity levels.



(a) Breakout (b) Pong (c) Space Invaders

Figure 1: Existence and stability of winning tickets across Atari environments. Curves show the interquartile mean (IQM) of episodic return across seeds evaluated at the end of each pruning iteration, with vertical bars indicating 95% bootstrap confidence intervals. Performance is reported for increasing sparsity levels and compared against randomly pruned baselines.

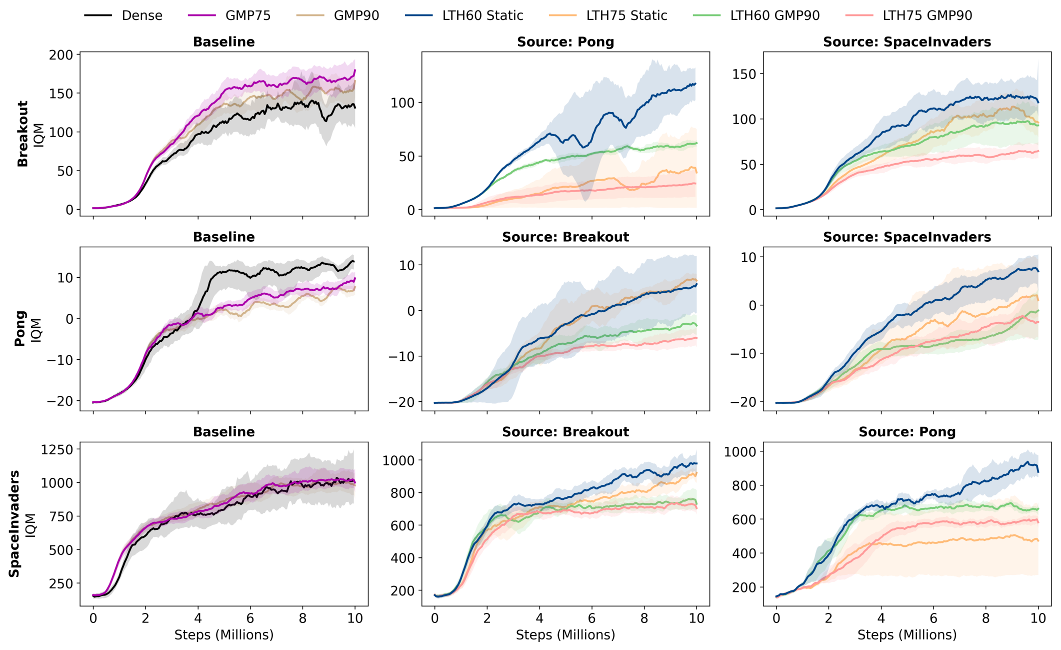


Figure 2: Target task learning curves under transfer for three Atari environments. Curves show the interquartile mean (IQM) of episodic return across seeds, with shaded regions indicating 95% bootstrap confidence intervals. Each row corresponds to a target task; the first column shows the baseline, and each subsequent column corresponds to a source task.

3.1 Existence of Winning Tickets

Figure 1 summarizes the existence of winning tickets across environments. For *Breakout* and *Space Invaders*, performance at the final pruning iteration exceeds that of the initial dense model up to 75% sparsity. Moreover, the performance gap relative to the random baseline widens with increasing sparsity, indicating that the identified subnetworks encode task-relevant structure rather than behaving as random sparse initializations.

In contrast, *Pong* exhibits a marked degradation in performance beyond 60% sparsity. At higher sparsity levels, the agent underperforms relative to the initial dense model, suggesting a failure to identify winning tickets at these sparsities. Furthermore, across all sparsity levels, the LTH pruning trajectory for *Pong* closely follows the random baseline, indicating that subnetworks identified at moderate sparsity do not retain meaningful inductive bias compared to randomly pruned counterparts.

3.2 Transfer

3.2.1 LTH

Figure 2 presents training curves for all baseline models and LTH-based transfer conditions across source–target environment pairs. Across all tasks, winning tickets at 60% sparsity transferred without further pruning on the target task (`LTH60 Static`) achieve the strongest overall transfer performance, both in terms of final IQM and learning stability. Contrary to our hypothesis, reintroducing gradual magnitude pruning during target task training does not improve adaptation and, in most cases, degrades performance relative to `Static` transfer.

The `LTH75` transfer conditions with *Pong* as the source consistently exhibit lower final IQM and a delayed learning onset across all target environments. As shown in Figure 1b, winning tickets could not be identified for *Pong* at 75% sparsity. This observation suggests that effective winning tickets encode inductive biases that are, at least in part, relevant to the target task.

Baseline performance varies across environments. For *Breakout*, the `GMP75` baseline achieves the highest final performance, while for *Pong*, the `Dense` baseline remains strongest. In *Space Invaders*, no statistically significant differences are observed among baseline configurations. These observations motivate comparing the best-performing transfer configuration (`LTH60 Static`) against the strongest baseline for each task.

Table 1: Area ratios comparing `LTH60 Static` transfer against (left) `Dense` and (right) `GMP75` baselines. Rows are source and columns are target environments. † indicates statistically significant negative transfer (95% bootstrap confidence interval does not overlap zero).

(a) `LTH60 Static` vs. `Dense`

	Breakout	Pong	SpaceInvaders
Breakout	-	$-0.3437^{\dagger}$	$-0.0551^{\dagger}$
Pong	$-0.2979^{\dagger}$	-	$-0.1645^{\dagger}$
SpaceInvaders	-0.0940	$-0.2903^{\dagger}$	-

(b) `LTH60 Static` vs. `GMP75`

	Breakout	Pong	SpaceInvaders
Breakout	-	-0.2394	$-0.0876^{\dagger}$
Pong	$-0.4571^{\dagger}$	-	$-0.1957^{\dagger}$
SpaceInvaders	$-0.2960^{\dagger}$	$-0.1785^{\dagger}$	-

Tables 1a and 1b report the corresponding area ratios. Across all source-target pairs, transfer results in negative area ratios, with most comparisons being statistically significant, indicating consistent negative transfer relative to training from scratch. Overall, these results provide a negative answer to our research question and contradict our hypothesis: the inductive bias encoded by sparse winning tickets identified on a source Atari task does not transfer effectively across environments under the considered settings.

3.2.2 GMP

	Breakout	Pong	SpaceInvaders
Breakout	-	$-0.2186^{\dagger}$	$-0.1826^{\dagger}$
Pong	$-0.2485^{\dagger}$	-	$-0.1871^{\dagger}$
SpaceInvaders	$-0.3821^{\dagger}$	$-0.1766^{\dagger}$	-

Table 2: Area ratios comparing `GMP60 Static` transfer against `Dense` baseline.

Table 2 reports the area ratios for sparse networks identified using GMP on the source task and transferred without further pruning to the target task, compared against the dense baseline. All source-target combinations yield negative area ratios, indicating that GMP-based sparse networks also fail to transfer effectively across environments.

3.2.3 LTH vs. GMP

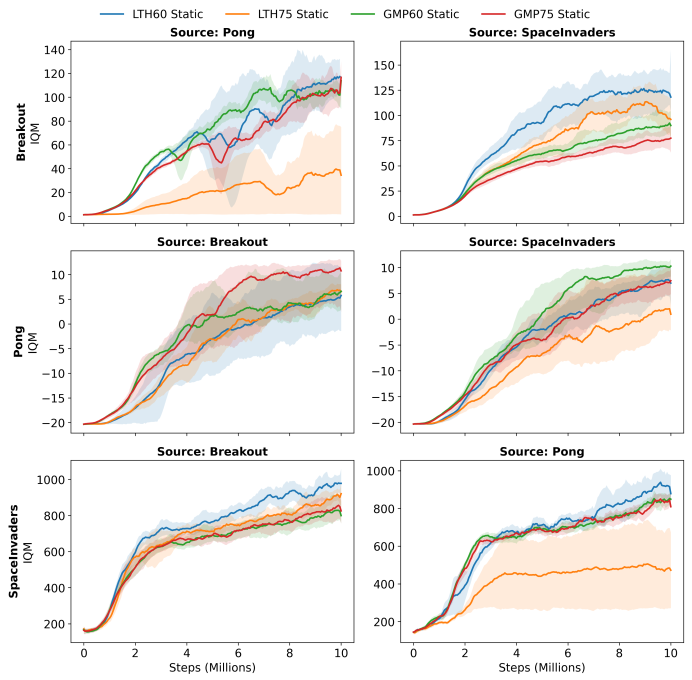


Figure 3: Comparison of `LTH Static` and `GMP Static` transfer conditions across source–target pairs. While individual source–target combinations exhibit differences in early learning dynamics or stability, neither pruning strategy consistently outperforms the other in terms of final performance across all environments.

Figure 3 compares the training curves of the `LTH Static` and `GMP Static` transfer conditions. The results are mixed: neither pruning approach consistently outperforms the other across all source-target combinations. This suggests that neither pruning schedule induces a universally transferable representation: subnetworks discovered via LTH or GMP appear to capture task-specific structure that does not consistently generalize across environments.

4 Discussion, Limitations, and Future Work

This work set out to investigate whether sparse subnetworks discovered on a source reinforcement learning task can transfer to related target tasks, and whether reintroducing sparsification during target-task training can improve such transfer by restoring network plasticity. Across all ablations, our results provide a clear and consistent answer to both parts of this research question. Sparse subnetworks identified on a source task do not transfer effectively across Atari environments, and dynamic sparsification during target-task training further degrades performance relative to Static transfer. Together, these findings provide strong evidence against our hypothesis that sparsity-driven plasticity can recover positive transfer in value-based deep reinforcement learning.

Beyond individual comparisons, several systematic trends emerge. First, higher initial sparsity on the target task leads to progressively worse transfer performance across environments. Second, Static transfer consistently outperforms configurations that further increase sparsity during adaptation. These patterns suggest that, while sparse subnetworks may encode task-relevant structure for the source environment, this structure is either insufficiently general or only marginally useful for the target task. Increasing sparsity further reduces network capacity, limiting the model's ability to form new task-specific representations during adaptation and exacerbating negative transfer.

Our findings align with prior observations by Sabatelli and Geurts (2021), who report limited or negative transfer when transferring convolutional encoders between Atari tasks. Their conjecture—that transfer breaks the co-adaptation between the feature extractor and the value head—offers a plausible explanation for our results. In our setting, this issue is likely amplified by sparsity. Pruning-induced constraints restrict representational flexibility in the encoder, while the reinitialized value head must adapt to a new reward structure. The combination of disrupted co-adaptation and reduced capacity appears to hinder the realignment between representations and value estimation required for successful transfer. From this perspective, sparsity does not merely fail to improve transfer, but can actively worsen the underlying mismatch.

While our results are consistent across ablations, several limitations must be acknowledged. Our evaluation is restricted to three Atari environments and value-based DDQN agents, and our conclusions therefore apply primarily to this experimental setting. Although similar trends may hold across additional Atari games, it would be inappropriate to generalize these findings to other algorithms, architectures, or continuous-control domains without further empirical evidence. Moreover, our study focuses exclusively on magnitude-based pruning methods. Alternative sparsification techniques—such as structured pruning, gradient-based pruning, or learned sparsity mechanisms—may interact differently with transfer and remain an open area for investigation.

An additional limitation concerns the rewind evaluation window used during LTH pruning. If this window is too short, early stopping may occur before a subnetwork has fully stabilized, resulting in lower-quality tickets that do not faithfully capture robust training dynamics. While the chosen window ensured consistency across runs, larger rewind evaluation windows may yield higher-quality subnetworks and could potentially alter transfer behavior.

While our results do not directly isolate the root cause of transfer failure, they are consistent with the hypothesis that limitations arise from how representations are transferred and adapted across tasks. This motivates future work that shifts focus away from sparsity-driven transfer alone and toward mechanisms that explicitly preserve or restore encoder–head co-adaptation. One potential direction is to investigate partial or structured transfer of the value head, rather than full reinitialization, potentially through action-space alignment methods. Such approaches may help retain useful representational couplings while still accommodating differing action spaces.

Another concrete direction is to more directly inspect what sparse networks learn and how those learned features change after transfer. Simple analyses—such as visualizing activations, comparing feature statistics, or examining how sparsity patterns differ between source and target tasks—could help reveal which aspects of the learned representations fail to carry over. These observations may, in turn, guide more intuitive and targeted transfer strategies.

Finally, extending this investigation to multi-task or continual learning settings—where encoder–head co-adaptation is preserved across tasks—may help clarify why dynamic sparsity improves robustness in those regimes but fails in cross-task transfer. Understanding this distinction could help bridge the gap between successful sparsity-driven plasticity in multi-task learning and its failure in cross-task transfer.

In summary, our results demonstrate that magnitude-based sparsity-driven transfer is not a promising approach for improving transfer performance in value-based deep reinforcement learning. Rather than enabling generalization, sparsity exacerbates representational misalignment and capacity limitations. Effective transfer in reinforcement learning likely requires interventions that directly address encoder–head co-adaptation, rather than relying on sparsity as a proxy for plasticity.

5 Supplementary Materials / Appendix

This appendix reports the full set of hyperparameters and experimental settings used in our experiments.

5.1 DDQN Training Hyperparameters

Table 3: DDQN training hyperparameters used across all Atari experiments.

Parameter	**Value**
Algorithm	Double Deep Q-Network (DDQN)
Discount factor γ	0.99
Optimizer	Adam
Learning rate	1.0×10^{-4}
Adam ϵ	1.0×10^{-8}
Batch size	32
Replay buffer size	1.0×10^{6}
Learning starts	80,000 steps
Training frequency	Every 4 environment steps
Target network update frequency	1,000–5,000 steps
Gradient clipping norm	10.0
Hidden layer dimension	512
Network head	Linear

5.2 Exploration Schedule

Table 4: Epsilon-greedy exploration parameters.

Parameter	**Value**
Initial ϵ	1.0
Final ϵ	0.01 (0.05 for some baselines)
Decay schedule	Linear
Decay fraction	0.1–0.2 of total timesteps

5.3 Atari Environment and Preprocessing

Table 5: Atari environment configuration and observation preprocessing.

Setting	**Value**
Environment suite	Arcade Learning Environment (ALE v5)
Frame stacking	4 frames
Observation resolution	84×84
Grayscale observations	Yes
Reward clipping	Yes
No-op starts	Up to 30 steps
Observation scaling	Disabled (handled by network)
Number of parallel environments	1

5.4 Lottery Ticket Hypothesis (LTH) Pruning Hyperparameters

Table 6: Hyperparameters for Lottery Ticket Hypothesis (LTH) pruning.

Parameter	**Value**
Pruning method	Iterative magnitude pruning
Final sparsity	0.60, 0.75
Pruning rate per iteration	0.10
Rewind step	80,000 steps
Rewind evaluation window	5,000 steps
Pruned layers	Encoder only
Weight reset	Yes (to rewind initialization)
Ticket evaluation window	IQM over last 20 evaluations

5.5 Gradual Magnitude Pruning (GMP) Hyperparameters

Table 7: Hyperparameters for Gradual Magnitude Pruning (GMP).

Parameter	**Value**
Pruning method	Gradual magnitude pruning
Initial sparsity	0.0
Final sparsity	0.60, 0.75, 0.90
Pruning start step	1,000,000
Pruning end step	6,000,000
Pruning update frequency	1,000–10,000 steps
Sparsity scheduler	Cubic
Pruned layers	Encoder only
Weight reset	No

5.6 Training Budget and Evaluation

Table 8: Training budget and evaluation protocol.

Parameter	**Value**
Total training timesteps	10,000,000
Evaluation metric	Interquartile Mean (IQM) of episodic return
Confidence intervals	95% bootstrap
Number of random seeds	5 (GMP), 3–5 (LTH)
Checkpoint interval	100,000 steps
Logging interval	100 steps

References

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 *Advances in Neural Information Processing Systems*, volume 34, pp. 29304–29320, 2021.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, June 2013. ISSN 1076-9757. DOI: 10.1613/jair.3912. URL http://dx.doi.org/10.1613/jair.3912.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in Neural Information Processing Systems*, 2, 1990.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei Rusu, Joel Veness, Marc G. Bellemere, Alex Graves, Martin 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. *Nature*, 518(7540):529–533, feb 2015. ISSN 0028-0836. URL http://dx.doi.org/10.1038/nature14236.

Johan Obando-Ceron, Aaron Courville, and Pablo Samuel Castro. In value-based deep reinforcement learning, a pruned network is a good network, 2024. URL https://arxiv.org/abs/2402.12479.

Matthia Sabatelli and Pierre Geurts. On the transferability of deep-q networks. *arXiv preprint arXiv:2110.02639*, 2021.

Aleksandar Todorov, Juan Cardenas-Cartagena, Rafael F. Cunha, Marco Zullich, and Matthia Sabatelli. Sparsity-driven plasticity in multi-task reinforcement learning. *Transactions on Machine Learning Research*, 2025.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Michael Zhu and Suyog Gupta. To prune, or not to prune: Exploring the efficacy of pruning for model compression. In *International Conference on Learning Representations (Workshop Track)*, 2017.