1. Experimental details

Unless otherwise specified, in all experiments below we report the interquantile mean after 40 million environment steps, aggregated over 15 games with 5 seeds each; error bars indicate 95% stratified bootstrap confidence intervals (Agarwal et al., 2021).

2. Experiments with PPO on MuJoCo

We used the PPO implementation from (Graesser et al., 2022) and ran some initial experiments with MuJoCo, increasing the width by 5x. As with our SAC experiments, we see no real change in performance, with perhaps some mild gains in Humanoid-v2. We see a few reasons why we may not see performance improvements in neither SAC nor PPO:

1. For ALE experiments, all agents use Convolutional layers, whereas for the MuJoCo experiments (where we ran SAC and PPO) the networks only use dense layers. We believe that this is a major contributing factor to why pruning is not providing a greater advantage.
2. The suite of environments in MuJoCo are perhaps less complex than the set of experiments in the ALE, so performance with agents like SAC and PPO is somewhat saturated.

Nonetheless, it is worth noting that Graesser et al. (2022) saw degradation with pruning at width=1 (Figure 16 in their paper), with almost a total collapse at 99% sparsity. In contrast, our results with 5x width shows strong performance even at 99% sparsity.

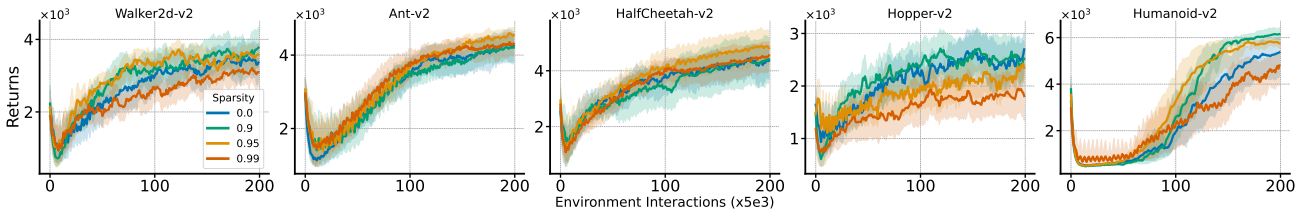

Figure 1. **Proximal Policy Optimization (PPO)** (Schulman et al., 2017) on MuJoCo environments when increasing width x5. We report returns over 10 runs for each experiment.

3. Experiments with IQN and M-IQN

While Rainbow is still a competitive agent in the ALE and both DQN and Rainbow are still regularly used as baselines in recent works, exploring newer agents is a reasonable request. To address this, we ran experiments with Implicit Quantile Networks (IQN) (Dabney et al., 2018) and Munchausen-IQN (Vieillard et al., 2020) with widths of 1 and 3; consistent with our submission's findings, we observe significant gains when using pruning.

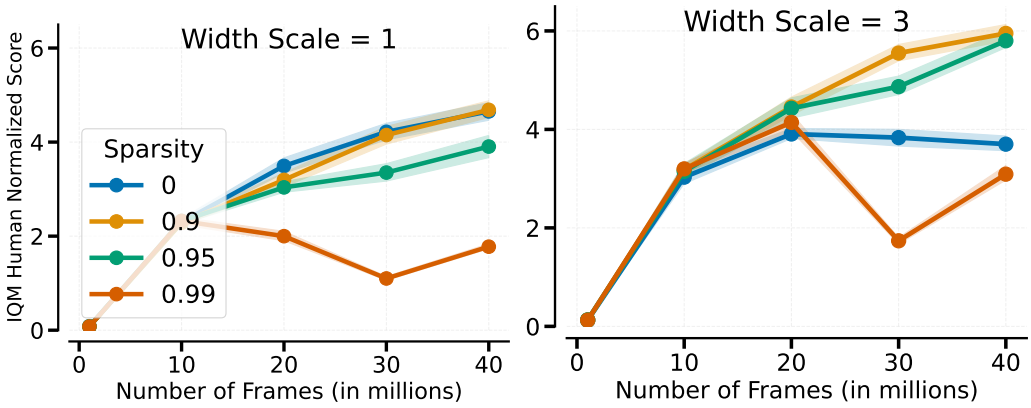

Figure 2. **IQN (*Implicit Quantile Networks*)** (Dabney et al., 2018) with ResNet architecture (with a width factor of 1 and 3).

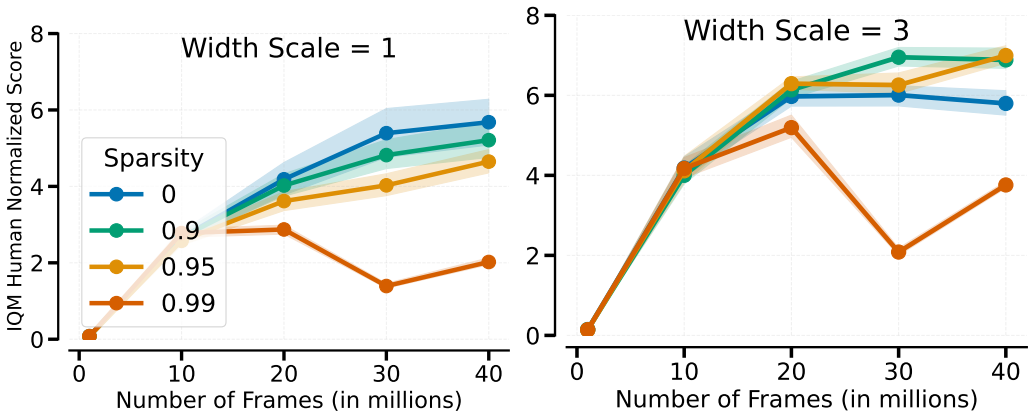

Figure 3. **M-IQN (*Munchausen-Implicit Quantile Networks*)** (Vieillard et al., 2020) with ResNet architecture (with a width factor of 1 and 3).

4. Comparison with RigL

based on other reviewers' comments, we have run a comparison with RigL (Evci et al., 2020). While RigL can be somewhat effective, it is unable to match the performance of pruning.

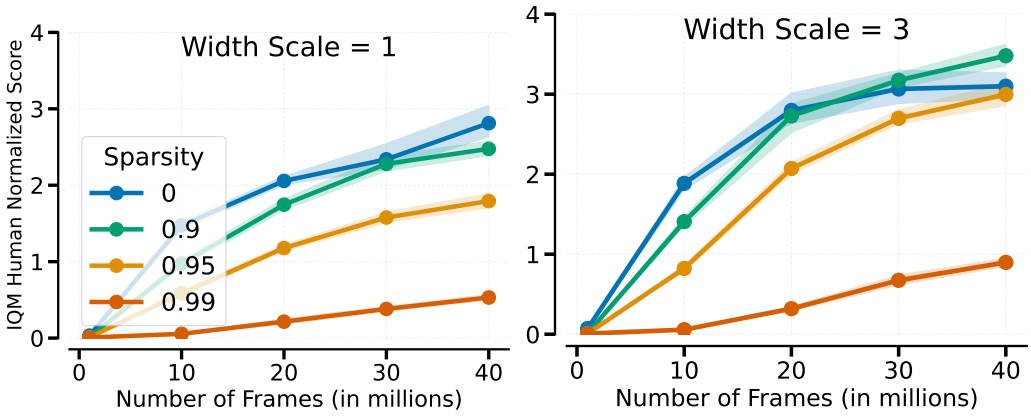

Figure 4. **The Rigged Lottery (RigL)** (Evci et al., 2020) for DQN with ResNet architecture (with a width factor of 1 and 3).

5. Sweeping over ReDo threshold τ

This parameter (introduced in Definition 3.1 of (Sokar et al., 2023)) defines the threshold for determining neuron dormancy. Sokar et al. (2023) suggested using 0.1 with the CNN network. Since we are using the Impala network architecture, we tested three additional values: (0, 0.025, 0.3). We found that 0.1, as used in Figure 12 of our submission, yields the best performance.

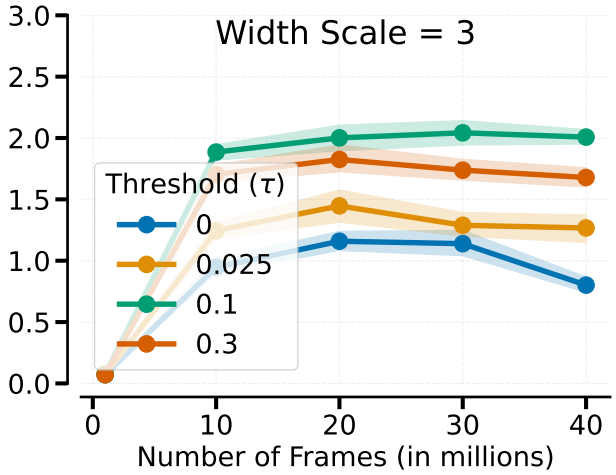

Figure 5. **ReDo's threshold** (Sokar et al., 2023) for DQN with ResNet architecture and a width multiplier of 3. The threshold (τ) allows to detect neurons with low activations.

6. Frequency of network resets

We varied the frequency of network resets (Nikishin et al., 2022) to evaluate whether this could help mitigate the performance loss when increasing the network width. While more infrequent resets (every 250,000 steps compared to the default value of 100,000) improves performance slightly, it still drastically under-performs with respect to the baseline and the pruning approach.

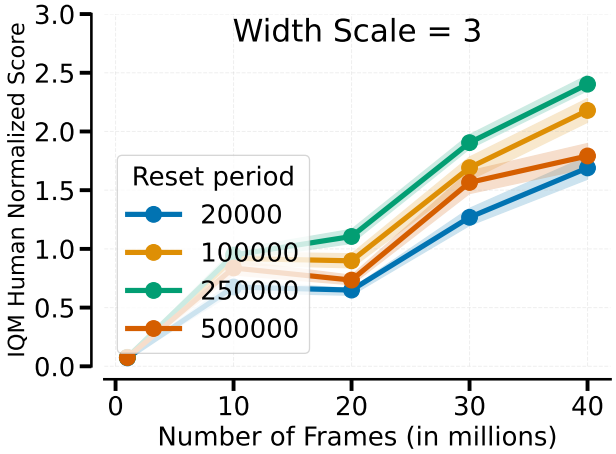

Figure 6. **Reset period (Nikishin et al., 2022)** for DQN with ResNet architecture and a width multiplier of 3.

7. Layer to reset

In our paper we followed the approach of Nikishin et al. (2022) and Sokar et al. (2023) of resetting only the last layer. We explored resetting different layers, but found it resulted in no significant performance difference.

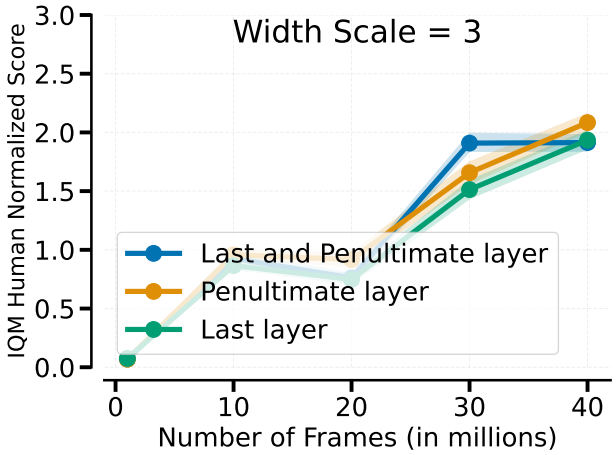

Figure 7. **Resetting different layers** (Nikishin et al., 2022), DQN with ResNet architecture and a width multiplier of 3 .

8. Varying weight decay

We ran a sweep over the following values $10e^{-6}$, $10e^{-5}$, $10e^{-4}$, $10e^{-3}$, and $10e^{-2}$, using the Impala architecture with a width factor of 3. The best performance is obtained with $10e^{-5}$, which is the value suggested by Sokar et al. (2023), and the value used in Figure 12 of our submission.

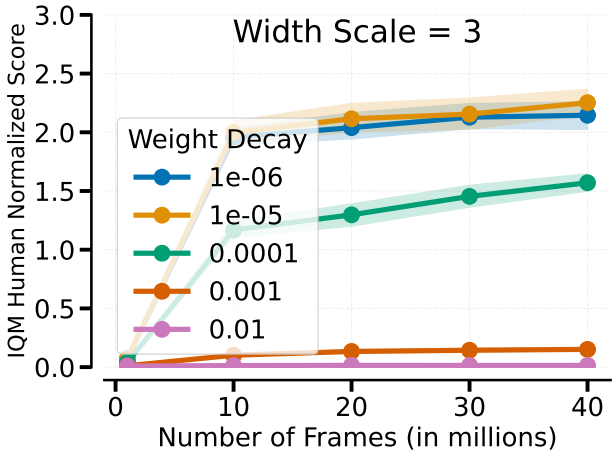

Figure 8. **Weight Decay (WD)** for DQN with ResNet architecture with a width of 3.

9. Lottery ticket hypothesis experiment

After training a network with pruning, we train a new network with the final mask *fixed* (i.e. not adjusted during training) and with the parameters initialized as in the original dense network. We found that the proposal under-performs both the pruning approach and the unpruned baseline. It is interesting to observe that both the pruning approach and the lottery ticket experiment seem to still be progressing at 40M, whereas the baseline seems to start deteriorating.

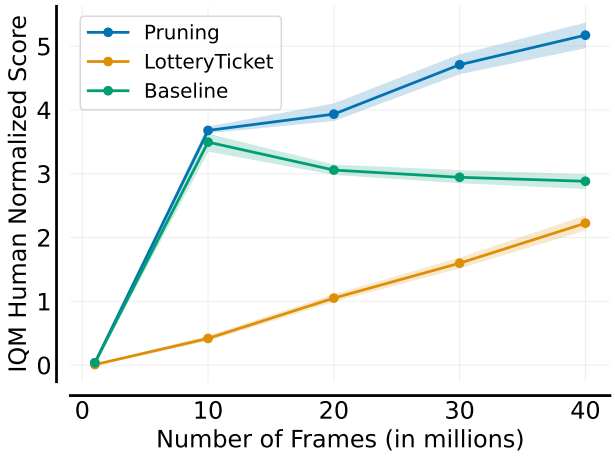

Figure 9. **Lottery ticket hypothesis experiment**. Taking the final pruned network (with a width factor of 3) and retraining with the original initialization results in worse performance.

10. Varying learning rates

The default learning for DQN is 6.25e-5. As suggested by the reviewer, we have run experiments with a learning rate divided by the width scale factor (so 2.08e-5 for a width factor of 3, and 1.25e-5 for a width factor of 5). These learning rates do improve the performance of the baseline, but it is still surpassed by pruning. All in all, these results are consistent with the thesis of the paper: pruning can serve as a drop-in mechanism for increasing agent performance.

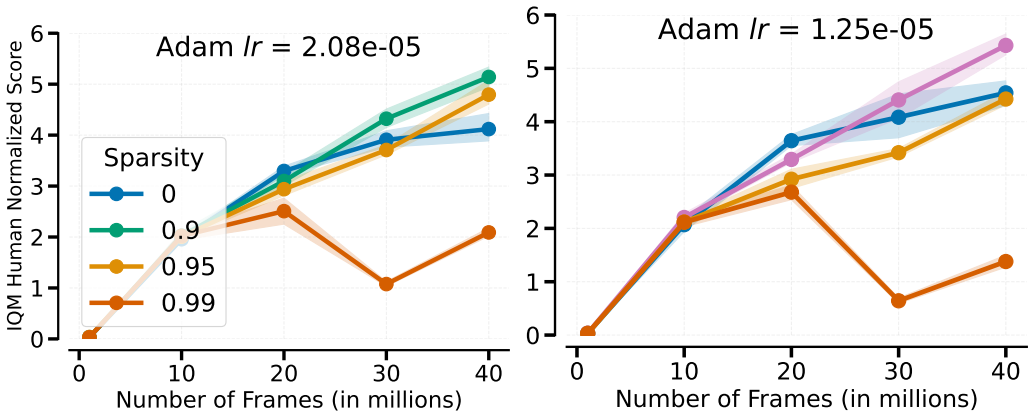

Figure 10. **Learning rate evaluation** for DQN agent with ResNet architecture (with a width factor of 1 and 3). These learning rate values correspond to dividing the default learning rate by the factor we used to amplify the size of the neural network.

11. Varying Adam's ϵ

The default value for Adam's ϵ is 1.5e-5; we ran experiments by dividing/multiplying this value by 3 (5e-5 and 4.5e-4, respectively). In all these cases, pruning maintains a significant advantage over the dense baseline.

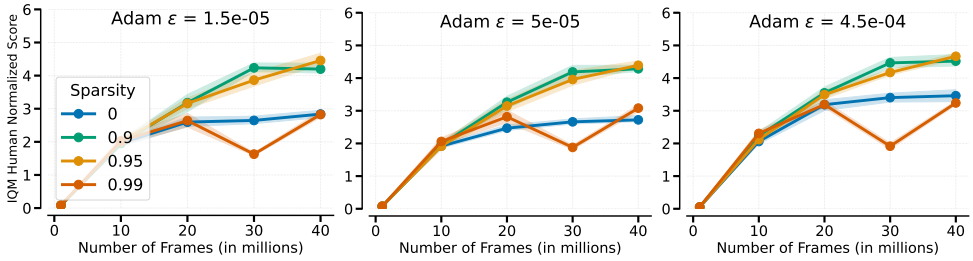

Figure 11. **Adam's epsilon** (ϵ) (Kingma & Ba, 2014) for DQN with ResNet architecture and a width multiplier of 3.

12. Varying update horizon

We explored using an update horizon of 3 for DQN (the default is 1) and found that pruning still maintains its advantage.

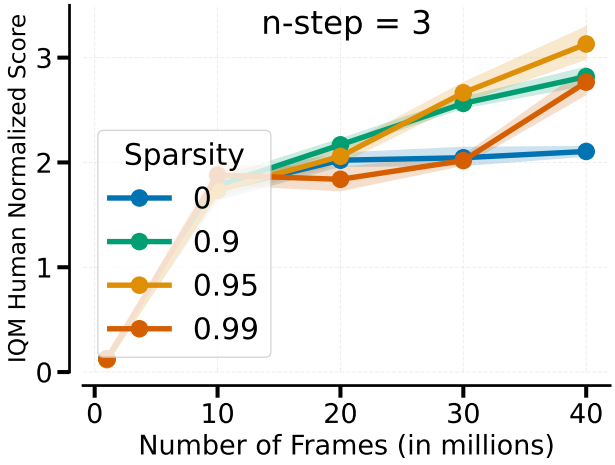

Figure 12. **Multi-step return** (Sutton & Barto, 2018) for DQN with ResNet architecture and a width multiplier of 3.

13. Varying batch size

The default batch size is 32, and ran experiments with batch sizes of 16 and 64. In all cases, pruning maintains its strong advantage.

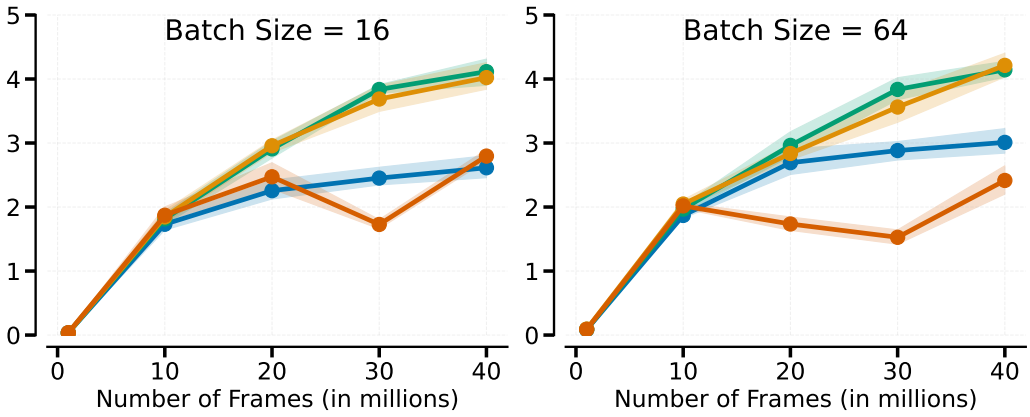

Figure 13. **Batch size** (Obando Ceron et al., 2024) for DQN with ResNet architecture and a width multiplier of 3.

References

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., and Bellemare, M. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

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

Evci, U., Gale, T., Menick, J., Castro, P. S., and Elsen, E. Rigging the lottery: Making all tickets winners. In *International conference on machine learning*, pp. 2943–2952. PMLR, 2020.

Graesser, L., Evci, U., Elsen, E., and Castro, P. S. The state of sparse training in deep reinforcement learning. In *International Conference on Machine Learning*, pp. 7766–7792. PMLR, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Nikishin, E., Schwarzer, M., D'Oro, P., Bacon, P.-L., and Courville, A. The primacy bias in deep reinforcement learning. In *International conference on machine learning*, pp. 16828–16847. PMLR, 2022.

Obando Ceron, J., Bellemare, M., and Castro, P. S. Small batch deep reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Sokar, G., Agarwal, R., Castro, P. S., and Evci, U. The dormant neuron phenomenon in deep reinforcement learning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32145–32168. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/sokar23a.html.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Vieillard, N., Pietquin, O., and Geist, M. Munchausen reinforcement learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 4235–4246. Curran Associates, Inc., 2020.