Google Research

The State of Sparse Training in Deep Reinforcement Learning

Laura Graesser, Utku Evci*, Erich Elsen, Pablo Samuel Castro*

Sparsity in Deep RL: Motivation

- **Better scaling** curves compared to dense architectures.
- **Real-time Deep RL**: needed for controlling real-time complex dynamical systems such as Plasma*.
- Deep RL has **challenging training dynamics**.
- Sparse neural network research has the risk of **overfitting to image classification**.

Thus: Sparsity in Deep RL

Our goal:
(1) an extensive study and
(2) strong baselines

Google Research

* Magnetic control of tokamak plasmas through deep reinforcement learning, Nature 2022

Background: Sparse Training

- Dense-to-Sparse Methods
  - **Pruning:** Start dense and prune during training.
- Sparse Training Methods
  - **Static:** Random sparse NN.
  - **SET:** Random sparse NN + dynamic connectivity with random growth.
  - **RigL:** Random sparse NN + dynamic connectivity with gradient guided growth.
- Baseline
  - **Dense:** Dense models with width scaling.

Google Research

Background: Deep RL

- Value function: expected discounted returns under a policy.

$$V^{\pi}(x) := \mathbb{E}_{a\sim\pi(x)}\left[\mathcal{R}(x,a) + \gamma\mathbb{E}_{x'\sim\mathcal{P}(x,a)}V^{\pi}(x')\right]$$

- Goal: find a policy that maximizes the value function

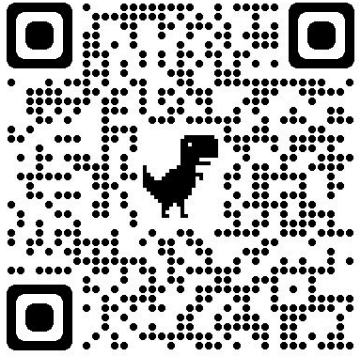
$$\pi^* := \max_{\pi} V^{\pi}$$

- Value-based methods (DQN)
  - Learns state-action value function (critic)
- Actor-Critic methods: (PPO, SAC)
  - Learns the policy (actor) and state-action value function (critic)

Google Research

Experimental Setup

- 23 environments studied across 3 environment classes
  - Discrete (classic control, Atari) and continuous (MuJoCo).
  - High (Atari) and low dimensional (classic control, MuJoCo) state spaces.
- Experiments
  - 10 seeds for each combination + IQM.
- Code :



Google Research

https://github.com/google-research/rigl/tree/master/rigl/rl

Results summary: IQM Plots

- For equivalent param count, Sparse > Dense, RigL ~ SET > Static
- Dynamic sparse training (RigL, SET) maintains performance at 90% sparsity for SAC & DQN
- Results using ResNets are even more promising, especially with pruning

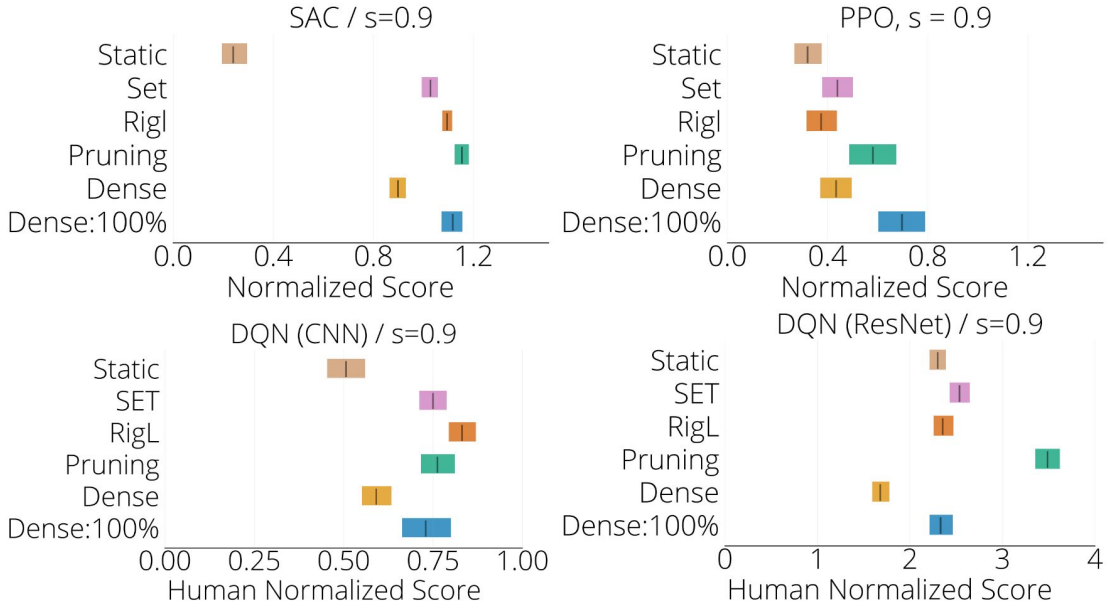


Figure 1. IQM plots for networks at 90% sparsity for various architecture and algorithm combinations. SAC and PPO are averaged over 5 MuJoCo environments, whereas DQN is averaged over 15 Atari environments. Results at different sparsities can be found at Appendix E. "Dense: 100%" corresponds to the standard dense model. Atari scores were normalized using human performance per game. MuJoCo scores were normalized using the average returns obtained by the Dense: 100% SAC agent per game.

Google Research

Results: Scaling Curves

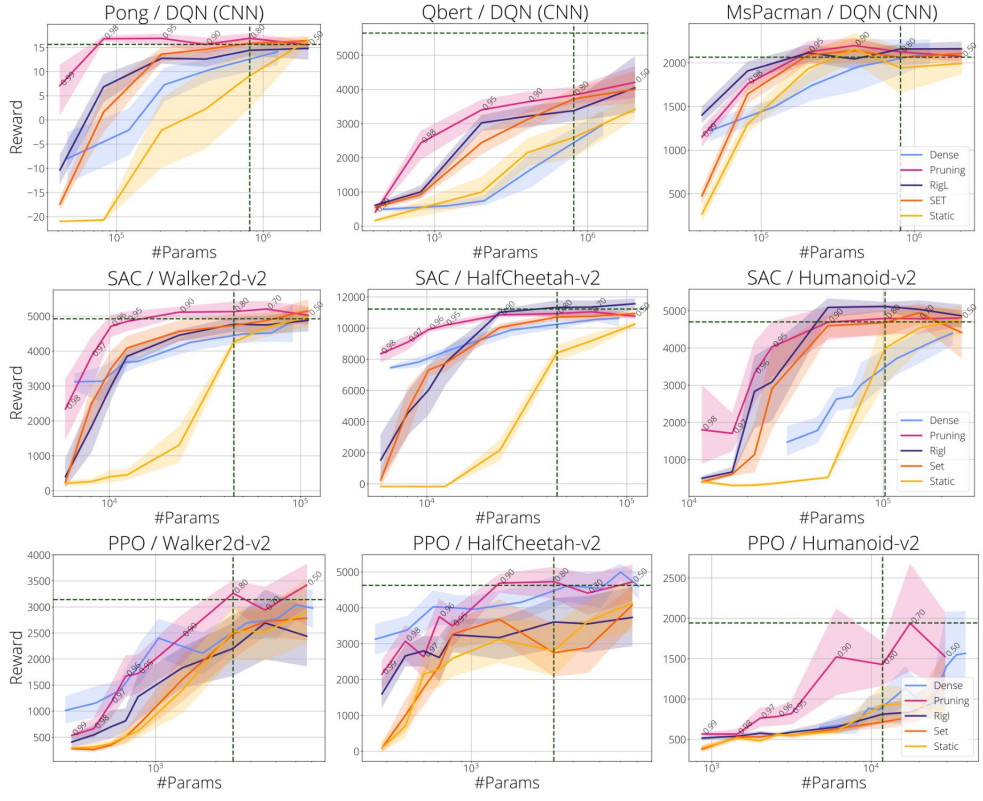


Figure 2. Comparison of the final reward relative to parameter count for the various methods considered: (row-1) DQN on Atari (CNN) (row-2) SAC on MuJoCo, (row-3) PPO on MuJoCo. We consider sparsities from 50% to 95% (annotated on the pruning curve) for sparse training methods and pruning. Parameter count for networks with 80% sparsity and the reward obtained by the dense baseline are highlighted with vertical and horizontal lines. Shaded areas represent 95% confidence intervals. See Appendix B for results on additional environments.

- Scaling curves vary from environment to environment
- Strong performance by dynamic sparse training algorithms for MsPacman and SAC Humanoid
- In Qbert none of the sparse training methods are able to match the dense baseline

Google Research

Results: Parameter distribution matters

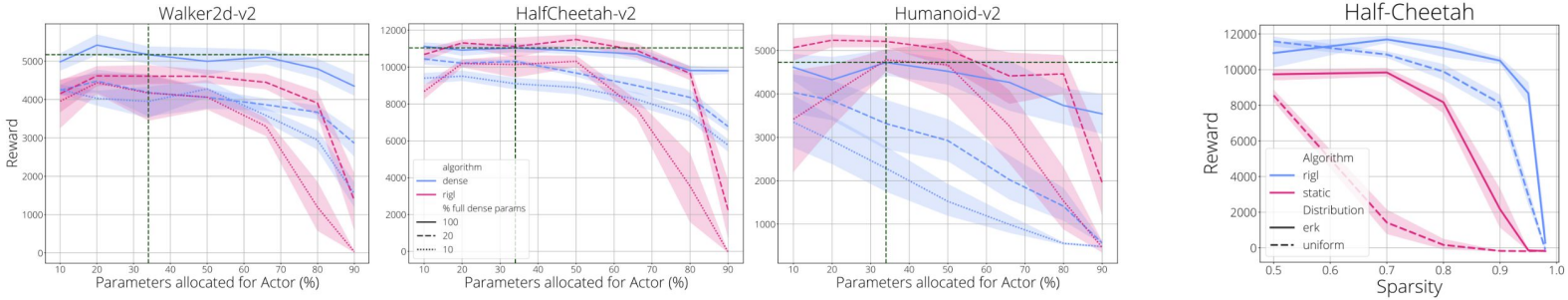


Figure 3. Evaluating how varying the actor-critic parameter ratio affects performance for a given parameter budget on different environment with policies trained using SAC. % of parameters allocated to the actor network is reported on the x axis. In SAC the parameter count of both critic networks is summed to give the overall critic parameter count. The vertical line corresponds the the standard parameter split and the horizontal line to the full dense training reward.

- The distribution of parameters across and within networks matters.
- Best performance is obtained by allocating the majority of parameters to the critic network ....
- ... and by using Erdos Renyi Kernel (ERK) sparsity distributions.

Google Research

Results: Robustness to Noise

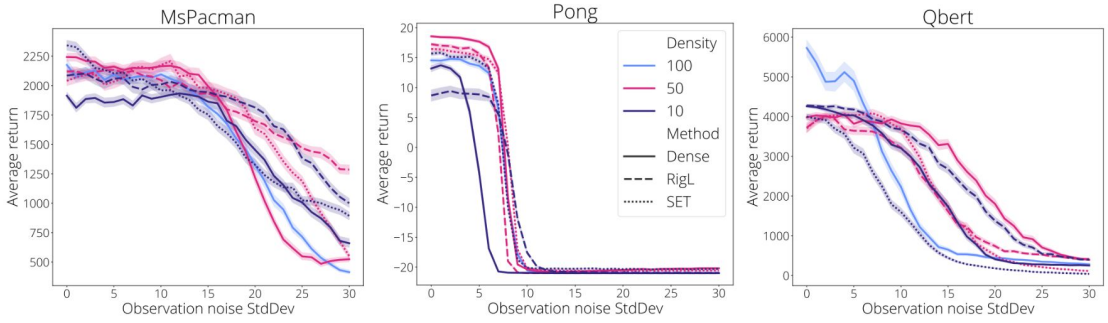
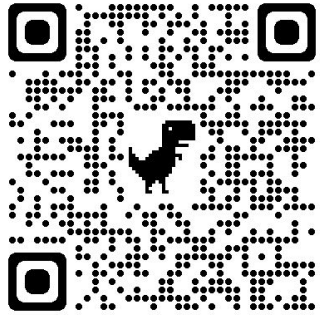


Figure 6. Robustness to observation noise. We test the robustness of networks trained using sparse and dense methods (denoted by line style) by adding Gaussian noise to the observations. We examine three parameter regimes, 100% (blue), 50% (pink) and 10% (purple) of the standard dense model parameter count. All policies were trained using DQN.

- Smaller models are generally more robust to high noise than larger models.
- Sparse models are more robust to high noise than dense models on average.
- In most cases there are minimal differences when the noise is low.

Google Research

Summary



- Sparse neural networks perform better than their dense counterparts for a given parameter count in DRL.
- It is possible to train up to 80 - 90% sparse networks with minimal loss in performance compared to the standard dense networks.
- Pruning often obtains the best results, and dynamic sparse training improves over static sparse training significantly.
- Gradient based growth seems to have a limited effect on performance. We argue this is due to low signal-to-noise ratio in gradients.
- The distribution of parameters among the actor and critic networks, as well as among different layers, impact training greatly.

Thank you for watching!

Google Research