Supplementary Materials for "Fine-tuning Deep RL with Gradient-Free Optimization"

Tim de Bruin
Delft University of Technology

Jens Kober
Delft University of Technology

Karl Tuyls
Deepmind Paris

Robert Babuška
Delft University of Technology

1 Benchmark details

This section provides additional details for the benchmarks used in the paper.

1.1 CarRacing-v0

We use the de CarRacing-v0 benchmark of the OpenAI gym suite [1], with the following adjustments:

Input pre-processing: To reduce the memory usage of the replay buffer, we convert the original 96x96x3 rgb images to 84x84 gray-scale images. Our initial tests showed that this did not affect the learning performance.

Action discretization: The original task has 3 continuous action dimensions: the steering angle $\in [-1, 1]$, accelerator $\in [0, 1]$ and break $\in [0, 1]$ inputs. We discretize the action space by using the following 7 actions:

Table 1: Discrete actions used during the CarRacing experiments.

action	steering	acceleration	breaking
1	-1	0.2	0
2	-0.5	0.5	0
3	0	0.5	0
4	0	0.8	0
5	0	0	0.8
6	0.5	0.5	0
7	1	0.2	0

1.2 Magman

For the magnetic manipulation problem we use the CoR-control library.[1] The task is to position a ball by controlling the current through four electromagnets positioned under a 1-dimensional track that the ball rolls on. The state of the system is composed of the position and the velocity of the ball. The experiments in the paper are conducted with the absolute reward function, a sampling frequency of 50Hz and an episode length of 2 seconds.

[1] https://github.com/timdebruin/CoR-control-benchmarks

Preprint. Under review.

1.3 Atari

We used the OpenAI gym interface [1], together with the OpenAI baselines `wrap_deepmind` function[2] to the interface with the `EnduroNoFrameskip-v4` and `FreewayNoFrameskip-v4` environments. The `wrap_deepmind` function performs the following modifications to make the environments behave as in the original DQN paper [2]:

- On games with multiple lives, episodes end ($\mathbb{T} = 1$) when a life is lost, but the environment is only reset after all lives are lost.
- Rewards are clipped using the sign function to be one of $\{-1, 0, 1\}$.
- Observations are converted to 84x84 gray-scale images, and four subsequent images are stacked into one new observation.

2 DRL parameters

For the DQN and DDPG algorithms we used the following hyper-parameters and implementation details (in addition to those mentioned in the main paper).

- DQN [2]:
  - optimizer: ADAM[3]
    - learning rate: $\cdot 10^{-4}$
    - $\beta_1 = 0.9, \beta_2 = 0.999$
    - $\epsilon = 10^{-4}$
  - experience buffer size: 10^6 experiences
  - batch size: 32
  - parameter update every 4 environment steps, frozen parameter update every 10^4 environment steps
  - parameter gradient clipping: 10
  - loss functions:
    - TDE: Huber loss with δ = 1
    - $0.01 \cdot ||\theta_{q,a}||_2$
    - cross entropy for policy head
  - $\gamma = 0.99$
- DDPG [4]:
  - optimizer: ADAM[3]
    - learning rate: $\cdot 10^{-3}$ for the critic and $\cdot 10^{-4}$ for the actor
    - $\beta_1 = 0.9, \beta_2 = 0.999$
    - $\epsilon = 10^{-8}$
  - experience buffer size: 10^5 experiences
  - batch size: 64
  - parameter update every steps, frozen parameter update step with $\tau = 0.001$
  - parameter gradient clipping: 10
  - loss function:
    - TDE: Huber loss with δ = 1
    - $0.01 \cdot ||\theta_q||_2$
  - $\gamma = 0.95$
  - $\sigma_a = 0.2$

For DQN, we do not allow the gradients of the policy head to back-propagate into the state encoder.

[2] https://github.com/openai/baselines

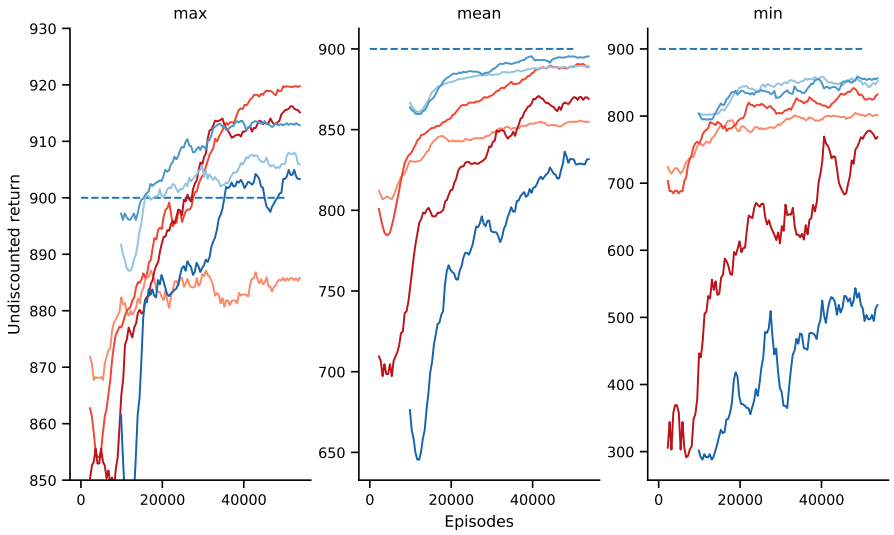


Figure 1: Best, mean and worst performance per iteration on the CarRacing benchmark (accompanies Figure 2c of the main paper).

3 Additional Experimental details and results

3.1 CarRacing-v0

On this benchmark we use the standard DQN architecture apart from added layer normalization and a policy head, as represented in Table 2:

Table 2: Used DQN architecture for the CarRacing benchmark

Input: 84x84 8-bit gray scale images cast to 32 bit float and divided by 255	
Conv1: 8x8x32 (stride 4, ReLU)	
Conv2: 4x4x64 (stride 2, ReLU)	
Conv3: 3x3x64 (stride 1, ReLU)	
Fully connected + layer norm[5]: 512 ReLU ($\bar{s}$)	
policy head: Stop Gradient	
policy head: Fully connected 7, softmax	$\hat{Q}$-head: Fully connected 7, linear

During the gradient based phase, the parameter noise magnitude σ is updated as described in the main paper to choose actions that differ from the policy with a probability $\epsilon = 0.1$. The first 200 episodes are performed with a random policy (actions are sampled uniformly at random in the action space) to fill the experience buffer.

For Figure 2c of the main paper, we show in Figure 1 how the mean and worst performance per iteration compared to the best performance per iteration. This figure also shows what happened when CMA-ES was used to train the most promissing best performer (ep=2500, $\epsilon_0 = 0.5$) for an additional 50,000 episodes. Although the best performance per iteration (over 16 rollouts) steadily improved to 935, testing this policy again over 100 episodes resulted in a mean score of 915.

3.1.1 SRL experiment

For the experiment with state representation losses mentioned in the main paper, we added the following subnetworks and losses:

- Auto encoding:
  - Loss: $10 * \|o - \hat{o}\|^2$

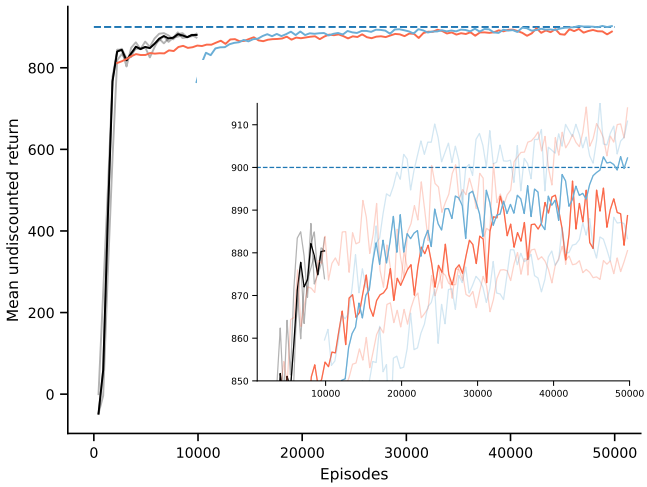


Figure 2: Learning curves for the CarRacing experiment with added state representation learning losses.

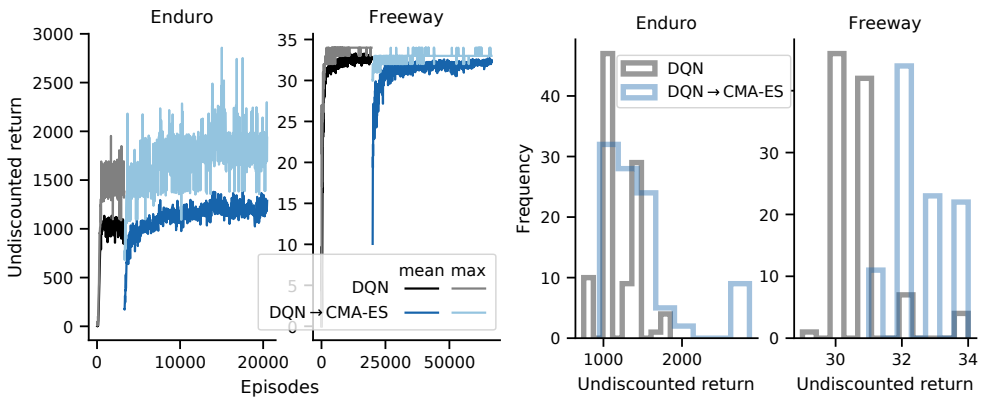
  - Subnetwork: $\bar{s} \rightarrow$ fully connected ELU layer with 441 units $\rightarrow$ reshaped to (21,21,1) $\rightarrow$ (3,3,32) transpose convolution with stride 2, ELU activation and batch normalization $\rightarrow$ (3,3,1) transpose convolution with stride 2, ELU activation and batch normalization $\rightarrow \hat{o}$.

- Reward prediction:
  - Loss: $0.5(r - \hat{r})^2$
  - Subnetwork: $\bar{s}$ concatenated with one hot representation of $a \rightarrow$ fully connected ELU layer with 32 units $\rightarrow$ fully connected linear layer with 1 unit $\rightarrow \hat{r}$.
- Forward dynamics:
  - Loss: $10 \left\| \bar{s}' - \hat{\bar{s}}' \right\|^2$
  - Subnetwork: $\bar{s}$ concatenated with one hot representation of $a \rightarrow$ fully connected ELU layer with 64 units $\rightarrow$ fully connected linear layer with 512 units $\rightarrow \hat{\bar{s}}'$.
- Inverse dynamics:
  - Loss: $1 \cdot$ cross entropy based on actually taken action a and the assigned probability.
  - Subnetwork: $\bar{s}$ concatenated with $\bar{s}' \rightarrow$ fully connected ELU layer with 64 units $\rightarrow$ softmax layer with 7 units.

For this experiment the learning curves are displayed in Figure 2. It can be seen that while switching after 10k episodes (blue line) now resulted in better performance than switching after 2.5k episodes (red line), there is a distinct drop in performance after the switch. This drop in performance was not observed during the experiment without SRL losses (Figure 2 of the main paper).

3.2 Atari experiments

For the Atari experiments, the same exploration decay as in [6] was used, with ϵ decaying linearly from 1 to 0.1 during the first one million episodes and remaining constant afterwards. The learning curves of the experiments, together with the performance when testing the best resulting controllers, are shown in Figure 3.



(a) Learning curves

(b) Test performance

Figure 3: Train and test performance on two Atari games.

References

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016.

[2] 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*, 518(7540):529–533, 2015.

[3] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference for Learning Representations (ICLR)*, 2015.

[4] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2016.

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016.

[6] Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y. Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. In *International Conference on Learning Representations (ICLR)*, 2018.