Supplementary Material: Locally Constrained Representations in Reinforcement Learning

Anonymous submission

Environments

In this section, we describe the environments used in the main paper.

MiniGrid

There are two MiniGrid environments used: RandomGoal and FourRooms. Both the environments are based on Chevalier-Boisvert, Willems, and Pal (2018). For our experiments we used the fully observable state input.

RandomGoal: This environment is similar to the MiniGrid Empty Room environment, where the goal of the agent is to navigate a gridworld and reach a goal. The agent receives a reward of $+1$ on reaching the goal and also a negative reward of -0.01 per step it takes to reach that goal. The maximum steps per episode is $4 \times$ grid size $\times$ grid size. The agent can take 3 actions: turn left, turn right and go forward. The input state of the agent is provided in the form of a 2D matrix. After the end of every episode, the agent can spawn anywhere in the grid with the exception of the starting position of the agent.

FourRooms: This environment is an extension of the RandomGoal environment, except there are four rooms separated by walls. The reward structure, state dimension and the actions are exactly the same. This is a little bit more challenging from a representation learning perspective, as the agent has to incorporate the collision with the walls.

MuJoCo

Multi Joint dynamics with Contact (MuJoCo) (Todorov, Erez, and Tassa 2012) is a physics engine supported by OpenAI Gym (Brockman et al. 2016). We used 8 environments from the MuJoCo suite.

1. **Half-Cheetah-v2** is an environment where the RL agent is a two-legged robot. It needs to learn to sprint fast.
2. **Ant-v2** is a similar environment, except that the agent controls a four-legged robot. The goal of the agent is to learn to sprint.
3. **Humanoid-v2** is an environment in which the agent learns to control a bipedal robot to walk on the ground without falling over.
4. **HumanoidStandUp-v2** is an environment, where the agent has to control a humanoid bipedal robot and get it to stand up.
5. **Pusher-v2** is an environment where the agent operates a robot arm and the goal is to push a cylindrical object to a designated position.
6. **Reacher-v2** is an environment where the agent is a two-jointed arm and the goal is to reach a randomly spawning point with the tip.
7. **Striker-v2** is an environment where the agent is a robot arm that needs to strike an object so that it reaches a goal.
8. **Thrower-v2** is an environment in which the RL agent controls a robot arm that throws a ball to a target using a scoop.

Atari

We use the Arcade Learning Environment (Bellemare et al. 2013) to run our experiments on Atari 2600. The environments we tested our algorithm are:

1. **Asterix:** In this game, the agent needs to eat hamburgers while avoiding dynamites.
2. **Breakout:** Breakout is a popular game, where the agent controls a paddle and the goal is to break all the bricks in the game using a ball.
3. **Gopher:** In gopher, the agent is a person who needs to protect three carrots from a gopher.
4. **Ms Pacman:** Here, the agent has to navigate a maze while eating pellets and avoiding ghosts.

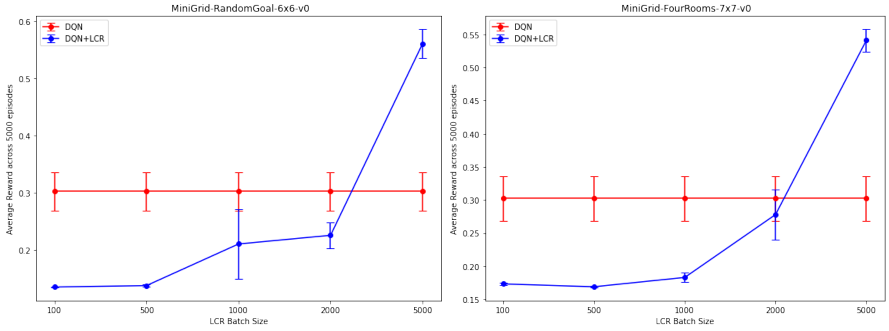
Gym Control

These are the classic control environments in Open AI Gym (Brockman et al. 2016). We used the popular CartPole-v1 and Acrobot-v1 environments.

1. **Acrobot-v1:** In this environment, the agent controls a linked chain and the goal is to push the low hanging chain up to a certain height while applying torque to the joint.
2. **CartPole-v1:** The goal of this environment is to balance a paddle with a pole on it by applying forces to move the paddle to the left or right.

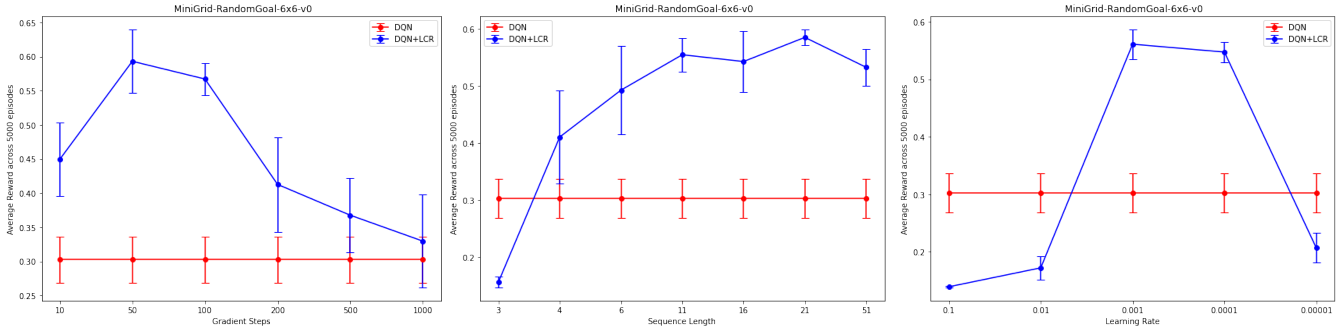
Additional Experiments

In this section, we describe additional ablation studies.



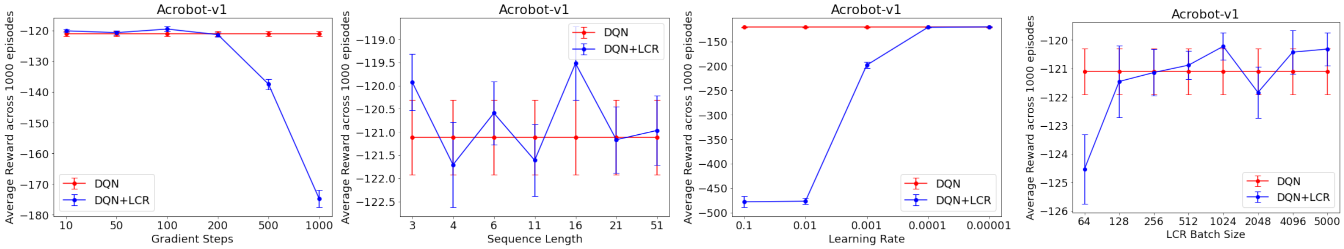
(a) (b)

Figure 1: Sensitivity to the batch size of Locally Constrained Representations (LCR). The constant hyper-parameters are sequence length of 11, 100 steps and learning rate of 0.0001.



(a) (b) (c)

Figure 2: Sensitivity to LCR hyper-parameters for the MiniGrid RandomGoal environment over 10 runs. The constant hyper-parameters are sequence length of 11, 100 steps and learning rate of 0.0001.



(a) (b) (c) (d)

Figure 3: Sensitivity to LCR hyper-parameters for the Acrobot-v1 environment over 10 runs. The constant hyper-parameters are sequence length of 11, 100 steps and learning rate of 0.0001.

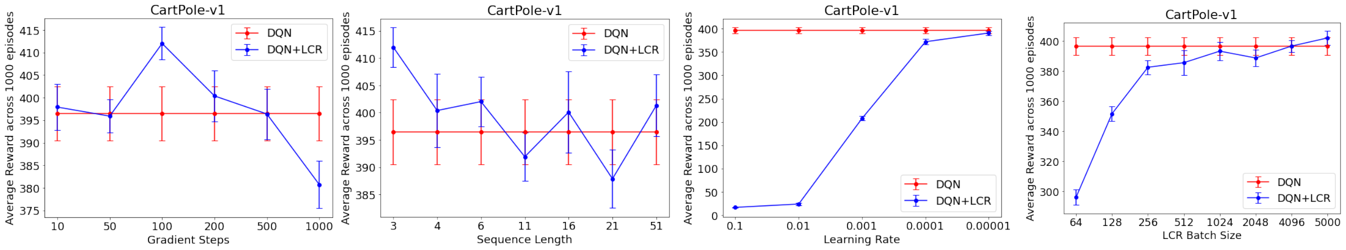
Sensitivity to Batch Size

There is no trade-off for LCR optimization when it comes to batch sizes as higher batch size means more state space coverage and more global gradients per update. As demonstrated in Figure 1 a larger batch size is better in our experiments.

Ablation studies for remaining environments

In this section, we provide ablation studies on the remaining environments. Figure 2 are the sensitivity parameters with respect to the gradient steps (a), sequence length (b) & learning rate (c) respectively. In Figure 2 (b), we see that the performance of LCR does not drop even for higher values of the sequence length. This is probably due to little variation in the states because it is an empty room and as a result LCR is able to find a good solution.

Figures 3 and 4 show the ablation studies for all the hyper-parameters in *Acrobot-v1* and *CartPole-v1* environments, respectively. Our observations in the MiniGrid experiments are also valid for these simple environments, i.e. locally linear representations do not hamper learning the original problem, provided the appropriate hyper-parameters are chosen.



(a) (b) (c) (d)

Figure 4: Sensitivity to LCR hyper-parameters for the CartPole-v1 environment over 10 runs. The constant hyper-parameters are sequence length of 11, 100 steps and the learning rate of 0.0001

The detailed hyper-parameters for all the algorithms are provided in Table 1. The code base is also included along with this supplementary material for the reproduction of these results.

References

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

Chevalier-Boisvert, M.; Willems, L.; and Pal, S. 2018. Minimalistic Gridworld Environment for OpenAI Gym. https://github.com/maximecb/gym-minigrid.

Ding, Z.; Yu, T.; Huang, Y.; Zhang, H.; Mai, L.; and Dong, H. 2020. RLzoo: A Comprehensive and Adaptive Reinforcement Learning Library. *arXiv preprint arXiv:2009.08644*.

Hessel, M.; Modayil, J.; van Hasselt, H.; Schaul, T.; Ostrovski, G.; Dabney, W.; Horgan, D.; Piot, B.; Azar, M. G.; and Silver, D. 2017. Rainbow: Combining Improvements in Deep Reinforcement Learning. *CoRR*, abs/1710.02298.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033.

Table 1: Hyper-Parameters of all experiments

Environments	Algorithm	Base Algorithm Parameters	LCR Parameters	Hardware and Software
MiniGrid	DQN	‘runs’ = 10 ‘episodes’ = 5000 ‘batch_size’: 32, ‘gamma’: 0.99, ‘learning_rate’: 1e-3, ‘start_epsilon’: 1.0, ‘stop_epsilon’: 1e-3, ‘epsilon_decay’: 1e-3**(1/5000) ‘hidden_units’: [64, 64], ‘max_buffer_size’: 10000, ‘min_buffer_size’: 1000, ‘copy_step’: 5, #episodes	‘K’: 10, ‘lcr_batch_size’: 5000, ‘gradient_steps’: 100, ‘lcr_learning_rate’: 1e-4	Hardware- CPU: Intel Gold 6148 Skylake RAM: 6 GB Software- Tensorflow: 2.8.0 Python: 3.8
MuJoCo	SAC	‘runs’ = 5 ‘episodes’ = 5000 ‘max_steps per episode’: ‘Ant-v2’: 150, ‘HalfCheetah-v2’: 150, ‘Humanoid-v2’: 1000, ‘HumanoidStandup-v2’: 1000, ‘Reacher-v2’ : 100, ‘Striker-v2’: 100, ‘Thrower-v2’: 100, ‘Pusher-v2’:100, Remaining hyper-parameters same as Ding et al. (2020)	‘K’: 10, ‘gradient_steps’: 100, ‘lcr_learning_rate’: 3e-4, ‘lcr_batch_size’: 5000	Hardware- CPU: 6 Intel Gold 6148 Skylake GPU: 1 NVidia V100 RAM: 48 GB Software- Tensorflow: 2.8.0 Python: 3.8
Atari	Rainbow	‘runs’ = 5 ‘frames’ = 5 million Remaining hyper-parameters same as Hessel et al. (2017)	‘K’: 10, ‘gradient_steps’: 20, ‘lcr_learning_rate’: 6.25e-5, ‘lcr_batch_size’: 1000	Hardware- CPU: 6 Intel Gold 6148 Skylake GPU: 1 NVidia V100 RAM: 32 GB Software- Pytorch: 1.10.0 Python: 3.8
Gym Control	DQN	‘runs’ = 10 ‘episodes’ = 1000 ‘batch_size’: 64, ‘gamma’: 0.99, ‘learning_rate’: 1e-3, ‘start_epsilon’: 1.0, ‘stop_epsilon’: 1e-3, ‘epsilon_decay’: 1e-3, ‘hidden_units’: [32] ‘max_buffer_size’: 5000, ‘min_buffer_size’: 100, ‘copy_step’: 25, #steps	‘K’: 10, ‘lcr_batch_size’: 5000, ‘gradient_steps’: 100, ‘lcr_learning_rate’: 1e-4	Hardware- CPU: Intel Gold 6148 Skylake RAM: 1.2 GB Software- Tensorflow: 2.8.0 Python: 3.8