1 Benchmarking RL on Real World Robots

There is lack of benchmarking tasks on physical robots for the case of studying Reinforcement Learning algorithms on them. The work introduces several RL tasks with commercially available robots. The work introduces 6 RL tasks on based on 3 commercially available robots. They test 4 policy gradient algorithms i.e. PPO, TRPO, DDPG and Soft Q learning (not PG i suppose). The main contributions being

- Introducing benchmark tasks for physical robots to share across the community
- setting up task conducive to learning
- empirical study of multiple policy learning algos on multiple physical robots

Robots used : The robots used are 1. **UR5** , a six joint robotic arm , 2. **Dynamixel MX-64AT** : A servo motor , 3. **Create 2** , A differential drive robot base.

1.1 Tasks

- UR-Reacher 2 : Actuate the second and third joint from the base by sending angular speeds between [-0.3, +0.3] rad/s. Observation vector is composed of joint angles, joint velocities, previous actions and vector difference between target and fingertip coordinates. Reward function is $R_t = -d_t + exp(-100d_t^2)$. Episodes are 4 seconds long , the starting boundary is $0.7m \times 0.5m$.
- UR-Reacher 6 : Extended version of Reacher 2 , with addition of another dimension for target position. A box of size : $0.7m \times 0.5m \times 0.4m$.
- DXL Reacher : based on current control, action space is 1D, with control signal of current between [-100, 100]mA. Reward $R_t = -d_t$.
- DXL Tracker : Current control to allow velocity matching and position tracking of a moving motor.
- Create Mover : Task is to move a robot forward in an area as fast as possible. Areas is 3ft times 2.5 ft, action space is -150mm/s, 150mm/s for both the wheels. Observation vector contains 6 wall sensors values, and previous actions.
- Create Docker : objective is to dock to a charging station, which has a huge positive reward, penalty for bumping.

In all the above cases the reward is scaled by the action cycle time in all cases. the space is normalized between +1 and -1.

1.2 Experimental Protocol

TO measure the sensitivity of hyper-parameter within task s and consistency across tasks, random search of 7 hyper parameters is performed. 30 Hyper parameters were drawn uniformly in the log scale. All the hyper-parameters were first tested for UR Reacher 2 task and then taken forward for other tasks. Each run is 150,000 steps (3 hours) for UR Rreacher 2 , 200,000 steps (4 hours) for Reacher 6, 50,000 steps (45 minutes) for DXL reacher, 150,000 steps (2 hours 15 minutes) for DXL Tracker, 40,000 steps (2 hours) Create Mover, 300,000 steps for Create Docker.

1.3 Findings

TRPO seemed to be least sensitive to its hyperparameters giving good performance on the robots, whereas DDPG didn't perform too well. Soft Q Learning often resulted in overheating of DXL (probably due to the random noise) and jerky movements. Create 2 Docker was sometimes achieved by the use of TRPO, which is very difficult to achieve in general from a scripted agent as well.

2 Survey on Reproducibility

Taxonomy of reproducible research

- **Repeatability** : same team, running the same experimental setup. Needed for reporting statistically sound results.
- **Reproducibility** : Different team , same setup , achieving results within marginals of experimental error.
- **Replicability** : differente team, different experimental setup. Trying to replicate the results , without access to the code, data and environment.

2.1 Methods

The paper proposes to uniformly configure the RL experiments by collecting all the parameters be it agent or the environment into a YAML configuration file. Separate the algorithms from the environments and metric collection routines.

2.2 Experimental Protocol

The task that they have used is the UR Reacher 2D. They have reused the OpenAI baselines, and used PPO and TRPO for their evaluations. They task is used partly to investigate their proposed methodology and to also check reproducibility. The paper approaches by performing ten runs for each hyperparameter , to determine the statistical significance. Then it approximates the Empirical distribution function, using boostrapping on the collected data. Bootstrapping is performed on 10k resamples on the ten average returns values to find the EDF.

2.3 Results

The paper reports the repeatability of the code base to hold alongside the issues that are there with the physical hardware.

The report that they are able to replicate the results that were proposed by the former paper.

2.4 Recommendations

Reproducing RL results of a reral world robot is **DIFFICULT**.
Managing Software dependicies should be reduced to **minimum** possible.
Presetting and reporting random seeds is **essential**
Distinguishing btw experimental and library code
Logging of return values Reporting Hyper Parameters is **Essential**.
Extensive Documentation
Open Source you research