GenReL-World: Utilizing various RL frameworks for robot manipulation

Márk Czimber

May 2024

Budapest

Aquincum Institute of Technology

Deep Learning

Contents

1 Abstract

GenReL-World is a general Reinforcement Learning framework to utilize various world models as environments for robot manipulation.

The goal is to contrust the framework which utilizes general reinforcement learning algorithms to teach and control a robotic in it's given environment for it's given task. A big problem in robotics is, that agents adapt hardly to new environments or tasks, which fall out of the trained task distribution. The adaptation of a general framework can be worthwhile if they can be trained on different world models (laten representations) and environments, tested on hyperparameters and initial conditions and using different reward and action space constructions.

With that only the world model and the encoding of the lower latent representation have to be repleaced. Implementing different algorithms and finding connections between them is an ongoing research area, which can play a crutial part in robotics. Also data structures, exploration function and the definition of state, reward and action spaces can impact the learning of an agent. The project focuses on different intitial values, action and rewards spaces and the change of agents behaviour in different environments.

The framework involves reinforcement learning concepts from [1] and meta-reinforcement learning and multi-task learning using the MetaWorld open-source simulated benchmark [2].

The project utilizes Google DeepMinds's MuJoCo (Multi-Joint dynamics with Contact) as a general purpose physics engine that aims to facilitate research and development in robotics. [3].

The project also includes a built 7 degree of freedom robotic arm which is simulated with MuJoCo Menagerie's xArm7 (or some other model) as a part of the collection of high-quality models for the MuJoCo physics engine, curated by Google DeepMind [4].

Different algorithm objectives have been consider as the open community provides better and better solutions and implementations, such as torchrl and cleanrl and also Basic PPO algorithms.

I also share the dream of Yann LeCun about how to construct autonomous intelligent agents [5].

2 General Introduction

2.0.1 Reinforcement Learning

The idea of Reinforcement Learning (RL) is actually following up on the approach how humans learn, which is learning through trial and error. When a person begins to learn the guitar (let's say they are already familiar with the music sheet and chords), they can only master it by playing it numerous time over and over again. Learning from the sound it generates, meanwhile the fingers getting used to playing and the feedback of others.

2.0.2 Trial and Error

This concept is the foundation of reinforcement learning, mastering skill by interacting with the environment and learning from series of attempts. Anything that can be learned by humans is possible to be taught to a machine, considering various existing and yet to be explored RL techniques.

2.0.3 Challanges

Reinforcement Learning is one of the three major areas of machine learning, which includes supervised learning and unsupervised learning too. Unlike supervised learning, it is difficult to provide a supervisor in the problem of RL, because we can not determine the right decision (action) in a given state or the number of states and actions are not even finite. This is where deep reinforcement learning techniques, Model Predictive Control, Parametrized Action Spaces come into the picture, cleansing the way for brand new approaches.

2.0.4 Applications

More and more fascinating approaches are constructed in game theory, fluid dynamics and robotics. In addition RL is widely used in in engineering, neuroscience, psychology, mathematics and economics. This document explores and provides insight on some of the general techniques used and discusses prominent and possible solutions for RL problems.

3 Material

The material part covers the different mathematical and reinforcement learning bases and concepts.

3.1 Model-based Reinforcement Learning (MBRL)

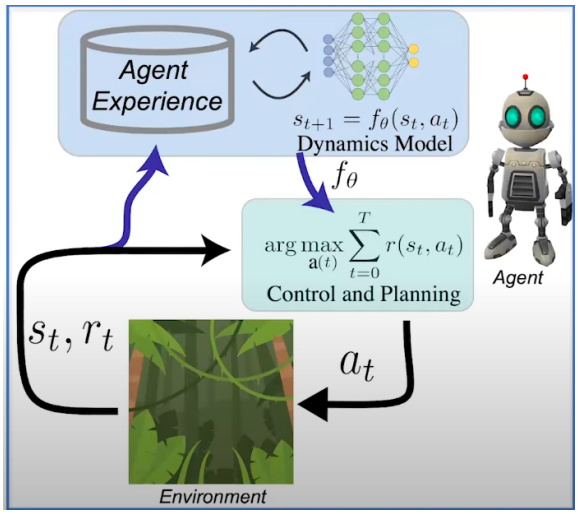

Figure 3.1: Agent and environment interactions in model-based reinforcement learning

Predicting Trajectories: (Optimization over finite horizon)
Choose action sequence with the highest cumulative reward and execute 1^{th} *action.*

$$s_T = f_\theta(f_\theta(\ldots(f_\theta(s_i, a_i)\ldots)) \tag{1}$$

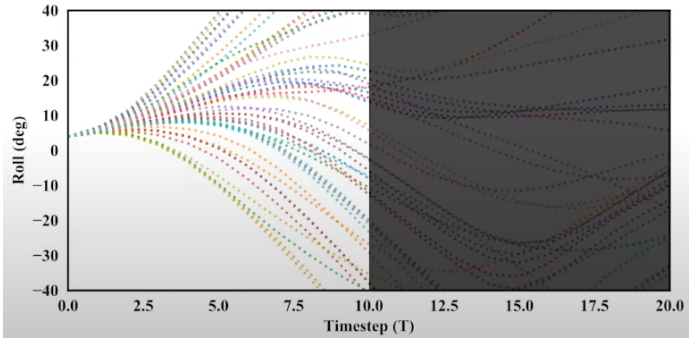

Figure 3.2: The trajectory with the highest comulative reward is selected by the arg max function

3.2 Proximity Policy Optimalization

- PPO is a computer agent, whcih computes a decision function to accomplish difficult tasks, complete MDP's.
- PPO is classified as a policy gradient method for training an agent's policy network. The policy network is the function that the agent uses to make decisions also called the value function.
- Gradient descent is sensitive to policy updates (step size), too-big step may direct policy in the false direction, too-small step lowers overall efficiency. A clip function is usually implemented that constrains the policy update of an agent from being too large or too small.
- Policy search methods may converge slowly given noisy data. For example, this happens in episodic problems when the trajectories are long and the variance of the returns is large. Value-function based methods that rely on temporal differences provide some solution to this problem. In recent years, actor–critic methods have been proposed and performed well on various MDP's.

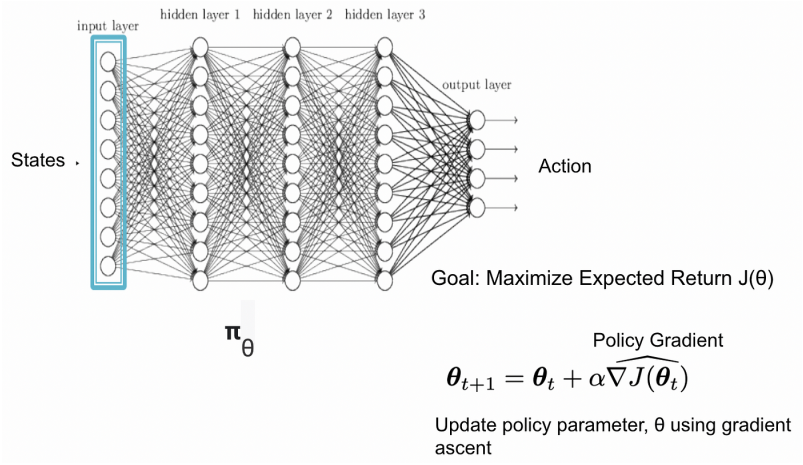

Figure 3.3: Actor-critic optimizing policy update to find optimal action sequence

3.3 General Advantage Estimation

- GAE introduces a parameter γ that allows us to reduce variance by downweighting rewards corresponding to delayed effects
- GAE resides on the idea of Temporal Difference such as we sample k steps from the future and use γ as a biased parameter to estimate gradients of future steps.
- It allows is to reduce variance by downweighting rewards at a much smaller cost of bias.
- It is an important estimator and enjoys widespread use in many advanced algorithms including VPG, TRPO, and PPO [6].

4 Environment

4.1 Metaworld and Gymnasium

- Metaworld is an open-source simulated benchmark for meta-reinforcement learning and multi-task learning.
- It contains 50 distinct robotic manipulation tasks and contains task distributions that are sufficiently broad to evaluate meta-RL algorithms to explore new behaviors.
- GenReL-World uses the pick-place-v2-goal-observable environment (Figure 4.4) from the benchmark to analyze a simple PPO agent's behaviour in the environment.

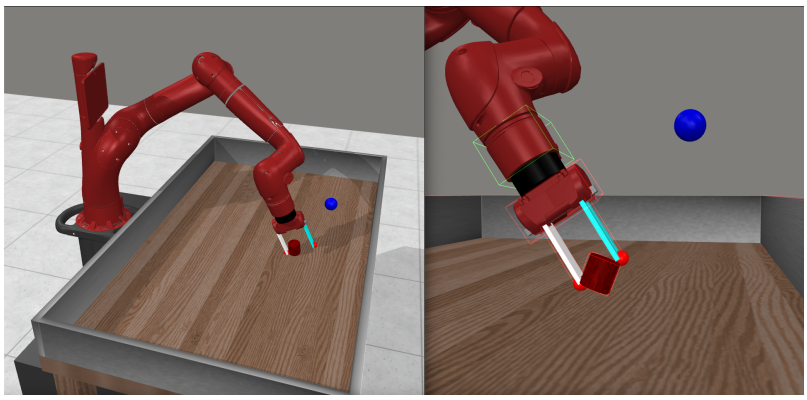

Figure 4.4: Two agents learning on pick-place-v2 task in Metaworld

- Gymnasium is an open source Python library deriving from OpenAI Gym for developing and comparing reinforcement learning algorithms by providing a standard API to communicate between learning algorithms and environments.
- GenReL-World experiments with Lunar Lander environment, a classic rocket trajectory optimization problem. Basic PPO was tested on MoonlanderContinuous-v2 environment.
- Gymnasium provides various environments (Figure 4.5) and tasks to test different agents, which is a powerful tool to conduct research on reinforcement learning.

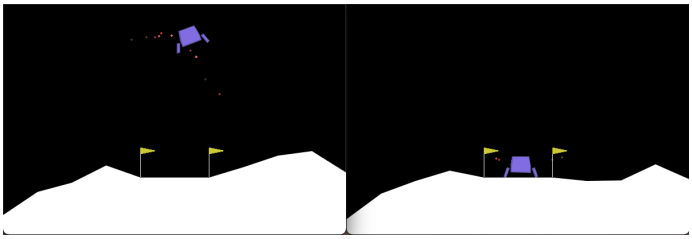

Figure 4.5: Two agents learning on MoonlanderContinuous-v2 in gym environment

5 Methodology

Due to training on a Mac M2 chip I couldn't obtain as good result of a long term train, although could derive multiple conclusions on early phases of different trains.

5.1 Scratch PPO

- My PPO model implements GAE and uses a Linear or an LSTM actor-critic module to complete Lunar Lander or Pick-place task. Currently the algorithm struggles with learning due to falsh implementation or incorrect hyperparameters, the issue is being tracked.

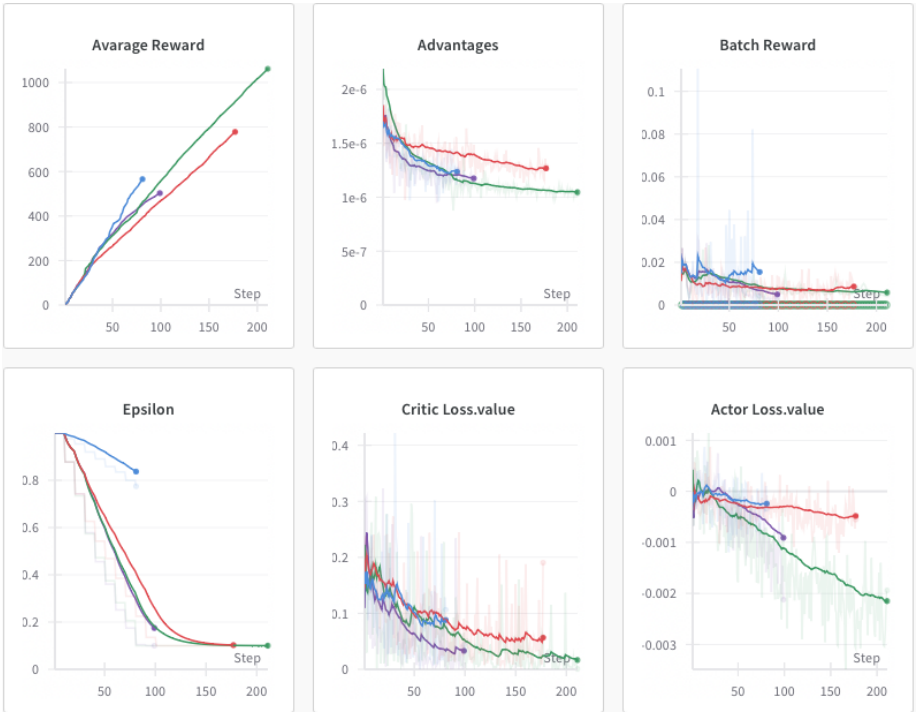

Figure 5.6: Learning of scratch PPO agent on Pick-palce-v2 (reward should converge)

5.2 Basic PPO for Lunar Lander

- Basic PPO model was trained on both environments with slight modifications. Adjustments was different actor-critic agents, simple linear feed-forward and long-short term memory. The LSTM preformed badly on both, but the linear achieved good results.
- Several hyperparameter optimalization was conducted, for Lunar Lander the best train was with 3e-4 learning rate, which had close completion around 800 steps (Figure 5.7).

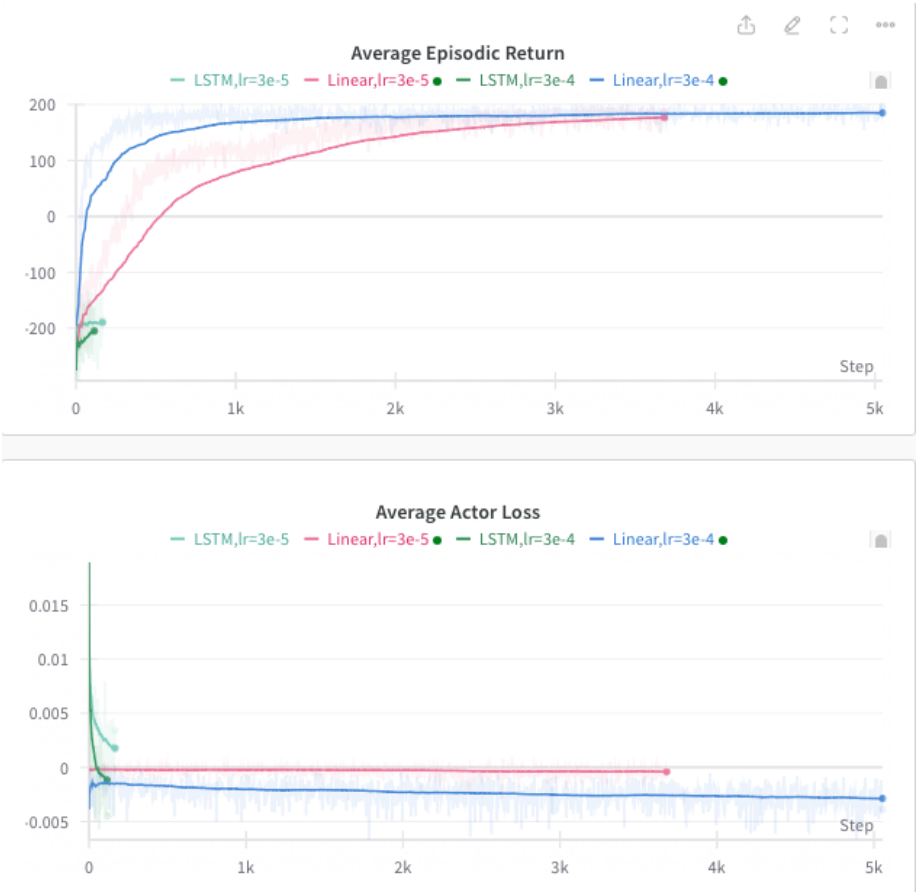

Figure 5.7: Learning curves of PPO on Lunar Lander

5.3 Basic PPO for Pick-place-v2

- Reward function is separated into reach-push-pick-place, 3 separate environments. The arm learns the optimal policy for the given task by maximizing it's expected return.
- The two reward function is based on various metrics, such as the distance between the object and the target, the distance between the TCP (Tool Center Point) and the object, and whether the object is grasped.

- These metrics are used to calculate a reward value, which is higher when the object is close to the target and the gripper is grasping the object.
- For the second reward logic (Figure 5.8) penalty was implemented to push the agent towards the correct policy and had increased rewards such as the arm lifting the object above a certain height and on task completion.
- This quickly taught the arm to receive positive rewards. The expected rewards continue to increase while actor loss decreases, so the learning might converge.
- Although the penalty for staying away from the object helps, the higher reward for favoured action sequences might cause a stuck in local minima.

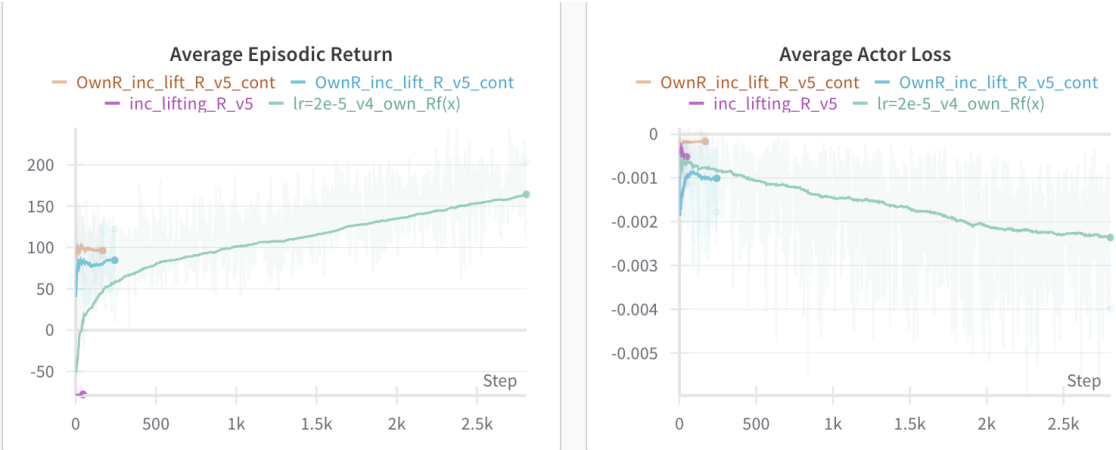

Figure 5.8: Learning curves of PPO on Pick-place-v2 with positive rewards

- The base reward function computes rewards between 0-10 for each step with a maximum of 5000 before truncation.
- Teaching the arm to pick up an object is a very challenging task, yet the actor finds correct solutions to the problem and continues to improve (Figure 5.9), so the place task may be learned in long time training.

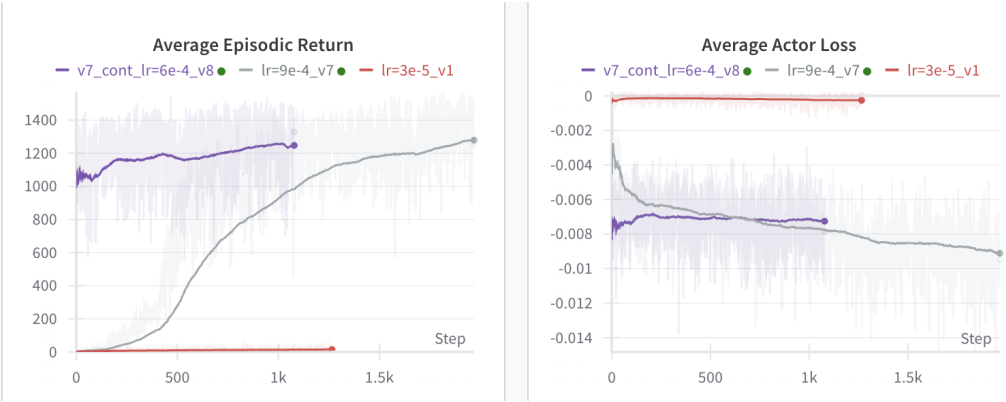

Figure 5.9: Learning curves of PPO on Pick-place-v2 with mixed rewards

6 Discussions

- MDP's and reinforcement learning are sensitive to initial values and the definition of action, reward and state spaces, along with exploring and various hyperparameters.
- One way is to carefully adjust every initial value and optimize for different hyperparameters, which is a costy and complicated method.
- Even simple algorithms such as basic PPO agents can perform well on simple problems, such as Lunar Lander and complex ones such as Pick-place-v2, meaning complex algorithms can achieve even higher performances like the RL^2POO (9).
- While different methodologies such as curriculum learning, MPC, hierarchical RL, imitation learning, and meta-learning offer promising avenues for improving sample efficiency, generalization, and safety in RL systems, has it's limitations.
- The battle of the transformers, leading RL methods and newly arising techniques may determine the future of robotics and Reinforcement Learning.

7 Improvements

- Correction of scratch PPO implementation and usage of different RL methods, for example the promising Dreamer model.
- Implementing world models along with Model Predictive Control to provide a latent representation of the environment, which decreases computation usage and provides more precise understanding of the surroundings.
- Consideration of Parametrized Action Spaces, which is also a promising research area about the definition of initial spaces(8).
- Further understanding MDP's as initial value problems improving existing methods.

8 References

(1) Richard S. Sutton and Andrew G. Barto. (2018). ReinforcementLearning: An Introduction (second edition). The MIT Press.
https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf

(2) @inproceedingsyu2019meta,
title=Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning,
author=Tianhe Yu and Deirdre Quillen and Zhanpeng He and Ryan Julian and Karol Hausman and Chelsea Finn and Sergey Levine,
booktitle=Conference on Robot Learning (CoRL),
year=2019
eprint=1910.10897,
archivePrefix=arXiv,
primaryClass=cs.LG
url=https://arxiv.org/abs/1910.10897

(3) @inproceedingstodorov2012mujoco,
title=MuJoCo: A physics engine for model-based control,
author=Todorov, Emanuel and Erez, Tom and Tassa, Yuval,
booktitle=2012 IEEE/RSJ International Conference on Intelligent Robots and Systems,
pages=5026–5033,
year=2012,
organization=IEEE,
doi=10.1109/IROS.2012.6386109

(4) @softwaremenagerie2022github,
author = Zakka, Kevin and Tassa, Yuval and MuJoCo Menagerie Contributors,
title = MuJoCo Menagerie: A collection of high-quality simulation models for MuJoCo,
$url = http://github.com/google-deepmind/mujoco_menagerie$,
year = 2022

(5) Yann LeCun. (2022). A Path Towards Autonomous Machine Intelligence, Version 0.9.2, 2022-06-27. Courant Institute of Mathematical Sciences, New York University and Meta - Fundamental AI Research.

(6) Siwei Causevic. (2023). Generalized Advantage Estimation in Reinforcement Learning, Towards Data Science Medium article

(7) Danijar Hafner et al. (2020). Mastering Atari with Discrete World Models. arXiv:2010.02193

(8) Renhao Zhang et al. (2024) Model-based Reinforcement Learning for Parameterized Action Spaces. arXiv:2404.03037

(9) Yan Duan et al. (2016) RL2: FAST REINFORCEMENT LEARNING VIA SLOW REINFORCEMENT LEARNING.
url = https://arxiv.org/pdf/1611.02779.pdf