Solving Complex Sparse Reinforcement Learning Environments

Javier Rando Ramírez   
javier.rando@tum.deShreyash Agarwal  
shreyash.agarwal@tum.de

Keywords – reinforcement learning, sparse rewards, Scheduled Auxiliary Control (SAC-X)

# Objective

In this project, we will try to extend the current state-of-the-art systems to solve complex sparse reinforcement learning tasks. This sparse setting is very useful to avoid reward shaping; i.e. the agent overfitting to the rewards found in the environment. The idea behind these scenarios is that the reward is rare and usually found after a complex series of events. Therefore, discovering this reward by random exploration is unfeasible, especially for complex environments.

We will build on top of SAC-X [1], a technique proposed to solve these tasks by creating auxiliary policies that allow the agent to efficiently explore the environment. We will analyze potential shortcomings of this implementation and explore how it can be used for object manipulation, e.g., with a hand. Furthermore, we will try to implement temporal coherency using recurrent neural networks such as LSTM.

# Related Work

## Reward shaping

This was the first and most intuitive idea introduced to solve the problem [2]. The idea is including additional reward features to reward or punish interactions with the environment before the final state. However, this may need experts to come up with the correct additional rewards that will help the agent solve the task. This is not usually easy.

## Curiosity driven methods

In this approach, the agent must explore the environment in order to discover the final reward state. Therefore, curiosity must be modelled as a reward signal. There have been several contributions following this idea. For instance, curiosity can be seen as an intrinsic reward for the agent [3] or we may create self-supervised exploration [4].

## Curriculum learning

The idea is presenting the agent increasingly complex tasks over time until it is able to solve the initial sparse task. One technique used for curriculum learning is GoalGAN which uses a Generative Advesarial Network to create new goals that are harder but still solvable for the agent [5].

## Auxiliary tasks

In this case, we will show the agent auxiliary policies that lead to an efficient exploration of the environment. Riedmiller et al [1] put forward Scheduled Auxiliary Control (SAC-X). Based on the auxiliary tasks presented to the agent, it will explore the environment until the external reward is found. Jaderberg et al. [6] proposed an extension using unsupervised auxiliary tasks.

# Technical Outline

Since we don’t have previous experience implementing Reinforcement Learning our outline will start from very basic tasks that will enable us to achieve the final result.

* Developing a toy continuous environment according to OpenAI Guidelines which is defined as follows: square world with an agent that is able to move freely. In this world, three different points (blue, red, green) will be found. Reward must be sparse so the agent will only be rewarded after visiting the three dots in a specific order: blue, red, green. Any other combination or intermediate step will result in no reward.
* Once the environment is built, we will implement the benchmark algorithms presented in by Riedmiller et al. [1] to solve the problem on Google Cloud.
* Also, different approaches such as the ones presented in Related Work section can be implemented for comparison. For instance, the work from Jaderberg et al. [6] could be an interesting option since they are both based on auxiliary tasks.
* Then, when everything is set up in a toy environment, we will move towards a more complex and 3D scenario for object manipulation. For example, a hand rotating a cube. In this case, the agent will only be rewarded after achieving a specific position.
* Again, we will see how the algorithms perform and possible shortcomings of their behaviour.
* In a final step, we will advance on one of these implementations to achieve better performance in our object manipulation task. For instance, we could try to implement recurrent neural networks such as LSTM to impose temporal consistency for moves. This might be really important for real-world scenarios in which uncoordinated actions can result on failure. For example, when rotating a cube if a finger is lifted before the previous one was placed in its final position this will result on the cube falling.

1. M. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J. Degrave, T. Van de Wiele, V. Mnih, N. Heess, T. Springenberg, “Learning by Playing – Solving Sparse Reward Tasks from Scratch”. Proceedings of the 35th International Conference on Machine Learning, in PMLR 80:4344-4353
2. M. J. Mataric, “Reward functions for accelerated learning.”
3. D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, “Curiosity-driven exploration by self-supervised prediction. ”
4. R. Sekar, O. Rybkin, K. Daniilidis, P. Abbeel, D. Hafner, and D. Pathak, “Planning to explore via self-supervised world models.”
5. C. Florensa, D. Held, X. Geng, and P. Abbeel, “Automatic goal generation for reinforcement learning agents.”
6. M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Sil-ver, and K. Kavukcuoglu, “Reinforcement learning with unsupervised auxiliary tasks.”