Internship project, master level, 2023

Title: Evolving Reservoirs for Meta Reinforcement Learning

Supervision: Clément Moulin-Frier, Xavier Hinaut, Eleni Nisioti and Gautier Hamon

Team: Flowers team and Mnemosyne team, Inria Bordeaux

Duration: 6 months, around February 2023 (with flexibility). **Level:** Master 2 research internship

Keywords: meta reinforcement learning, reservoir computing, evolutionary computation, neuroevolution, simulation environments, scientific programming with Python.

How to apply: contact clement.moulin-frier@inria.fr ; xavier.hinaut@inria.fr ; gautier.hamon@inria.fr ; eleni.nisioti@inria.fr with a CV and letter of motivation, ideally by the end of November 2023. We also recommend sending documents or reports describing previous projects you have been working on (even if they are not directly related to the topic), as well as your grades and links to some of your code repositories.

Requirements: We are looking for highly motivated MSc students (Master II). Programming skills and prior experience with Python and deep learning frameworks (Pytorch, Tensorflow) are expected.

Project

How can a reinforcement learning (RL) agent autonomously learn a diversity of skills with minimal external supervision? Recent works have shown that a single goal-conditioned policy can be trained to solve multiple tasks in parallel (see e.g. Colas et al., 2021 for a recent review). Other works in Meta Reinforcement Learning (Meta-RL) have shown that artificial agents can "learn how to learn" (hence the term meta-learning) multiple skills without having access to goal-related information (see e.g. Weng, 2019 for an introduction, as well as Figure 1).

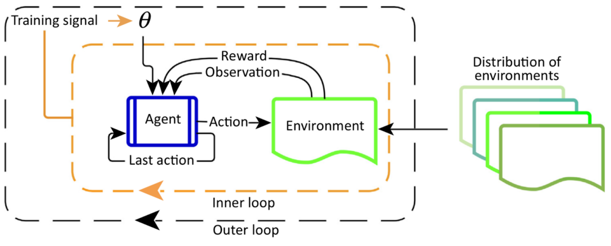

Figure 1. Example of a Meta-RL architecture with inner and *outer loops*, from (Botvinick et al., 2019).

Meta-RL aims at equipping agents with the ability to generalize to tasks or environments that have not been encountered during training. Two nested processes of adaptation are traditionally considered: an *outer adaptation loop* optimizing the hyperparameters of an *inner adaptation loop*. The *inner adaptation loop* can be a standard RL algorithm operating on a given environment. The *outer loop* is tuning the hyperparameters of the *inner loop* such that it performs well on a wide distribution of environments. The end result of this nested training process is an algorithm that *learns how to learn*, i.e. learns how to adapt to tasks that have never been encountered during training. Works differ in the optimization technique and adaptation mechanism they consider in the two loops. The outer loop often optimizes higher-level structures such as the architecture of a neural network (Baker et al., 2017), the morphology (Gupta et al., 2021), a curiosity module providing intrinsic rewards (Alet et al., 2020) or plasticity rules (Najarro & Risi, 2021), employing either evolutionary or gradient-based optimization.

In parallel, Reservoir Computing (RC, (Jaeger, 2007)) is particularly well suited for extracting information from data stream with complex temporal dynamics (e.g. weather forecast or language). It is a machine learning paradigm on sequential data where a recurrent neural network is only partially trained (e.g. only training a linear readout, Figure 2).

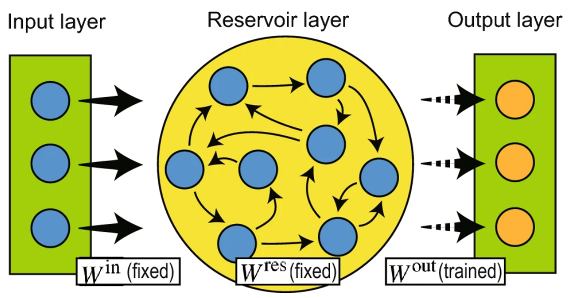
Input layer
Reservoir layer
Output layer
W^{in} (fixed)
W^{res} (fixed)
W^{out} (trained)

Figure 2. A typical RC architecture, from (Sakemi et al., 2020).

One of the major interests of these recurrent neural networks is their reduced computational cost and the possibility to learn both in on-line and off-line fashion. They have been successfully applied on a wide variety of tasks, from prediction/generation of chaotic timeseries to discrimination of audio sequences, such as bird song recognition. They offer de facto a fast, simple yet efficient way to train RNNs. This "reservoir of computations" works thanks to random projections in large dimensions, and is thus similar to temporal Support Vector Machines (SVM).

In this internship, we will explore how Reservoir Computing can be leveraged in the context of Meta-RL. In this context, we will consider evolving the architecture of a reservoir for performing well on a wide range of RL tasks (see Seoane, 2019 for theoretical arguments on this evolutionary perspective). For this aim, we will use the ReservoirPy[1] Python library developed in

[1] https://github.com/reservoirpy/reservoirpy

the Mnemosyne team (Trouvain & Hinaut, 2022) that we will interface with evolutionary computation algorithms (e.g. Tang et al., 2022). The project will involve:

- Designing of a wide range of RL tasks in an existing simulation environment (see e.g. Figure 3)
- Learning a single RL task with a reservoir
- Learning multiple tasks with a goal-conditioned reservoir
- Learning how to learn multiple tasks without access to the goal information (i.e. meta-RL with a reservoir).
- Evolving a reservoir for meta-RL

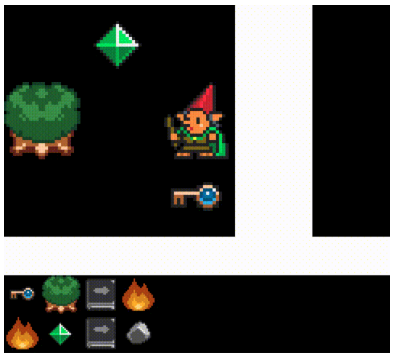

Figure 3. Simulation environment example developed in the Flowers team where an agent has to combine different objects to create new ones, in the spirit of the Minecraft video game.

References

Alet, F., Schneider, M. F., Lozano-Perez, T., & Kaelbling, L. P. (2020). *Meta-learning curiosity algorithms*. International Conference on Learning Representations (ICLR 2020). https://openreview.net/forum?id=BygdyxHFDS

Baker, B., Gupta, O., Naik, N., & Raskar, R. (2017). Designing neural network architectures using reinforcement learning. *ArXiv:1611.02167 [Cs]*. http://arxiv.org/abs/1611.02167

Botvinick, M., Ritter, S., Wang, J. X., Kurth-Nelson, Z., Blundell, C., & Hassabis, D. (2019). Reinforcement Learning, Fast and Slow. *Trends in Cognitive Sciences*, *23*(5), 408–422. https://doi.org/10.1016/j.tics.2019.02.006

Colas, C., Karch, T., Sigaud, O., & Oudeyer, P.-Y. (2021). Intrinsically Motivated Goal-Conditioned Reinforcement Learning: A Short Survey. *ArXiv:2012.09830 [Cs]*.

http://arxiv.org/abs/2012.09830

Gupta, A., Savarese, S., Ganguli, S., & Fei-Fei, L. (2021). Embodied intelligence via learning and evolution. *Nature Communications*, *12*(1), 5721. https://doi.org/10.1038/s41467-021-25874-z

Jaeger, H. (2007). Echo state network. *Scholarpedia*, *2*(9), 2330. https://doi.org/10.4249/scholarpedia.2330

Najarro, E., & Risi, S. (2021). Meta-Learning through Hebbian Plasticity in Random Networks. *ArXiv:2007.02686 [Cs]*. http://arxiv.org/abs/2007.02686

Sakemi, Y., Morino, K., Leleu, T., & Aihara, K. (2020). Model-size reduction for reservoir computing by concatenating internal states through time. *Scientific Reports*, *10*(1), Article 1. https://doi.org/10.1038/s41598-020-78725-0

Seoane, L. F. (2019). Evolutionary aspects of reservoir computing. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *374*(1774), 20180377. https://doi.org/10.1098/rstb.2018.0377

Tang, Y., Tian, Y., & Ha, D. (2022). EvoJAX: Hardware-Accelerated Neuroevolution. *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 308–311. https://doi.org/10.1145/3520304.3528770

Trouvain, N., & Hinaut, X. (2022). *reservoirpy: A Simple and Flexible Reservoir Computing Tool in Python*. https://hal.inria.fr/hal-03699931 https://github.com/reservoirpy/reservoirpy

Weng, L. (2019, June 23). *Meta Reinforcement Learning*. https://lilianweng.github.io/posts/2019-06-23-meta-rl/