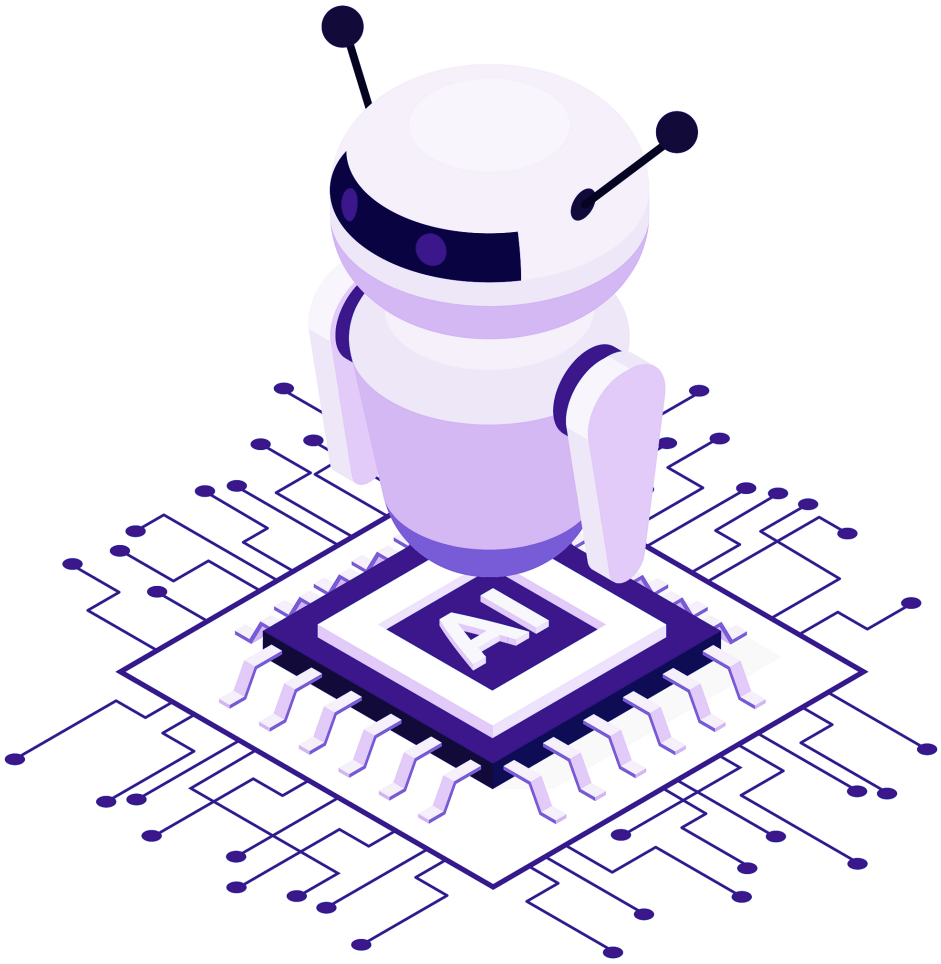


Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Homework 14:

Meta Reinforcement Learning

Designed By:

Armin Saghafian

armin.saghafian@gmail.com

RIML

Deep RL Course

Spring 2025

Contents

1 Preface

1.1 Introduction

Meta-Reinforcement Learning (Meta-RL) considers a family of machine learning (ML) methods that learn to reinforcement learn. That is, meta-RL methods use sample-inefficient ML to learn sample efficient RL algorithms, or components thereof. As such, meta-RL is a special case of meta-learning, with the property that the learned algorithm is an RL algorithm. Meta-RL has the potential to overcome some limitations of existing human-designed RL algorithms. RL remains highly sample inefficient, which limits its real-world applications. Meta-RL can produce (components of) RL algorithms that are much more sample efficient than existing RL methods, or even provide solutions to previously intractable problems.

1.2 Example Application

Consider, as a conceptual example, the task of automated cooking with a robot chef. When such a robot is deployed in somebody's kitchen, it must learn a kitchen-specific policy, since each kitchen has a different layout and appliances. This challenge is compounded by the fact that not all items needed for cooking are in plain sight; pots might be tucked away in cabinets. Therefore, the robot needs not only to understand the general layout but also remember where specific items are once discovered.

Training the robot directly in a new kitchen from scratch is too time consuming and potentially dangerous due to random behavior early in training. One alternative is to pre-train the robot in a single training kitchen and then fine-tune it in the new kitchen. However, this approach does not take into account the subsequent fine-tuning procedure. In contrast, meta-RL would train the robot on a distribution of training kitchens such that it can adapt to any new kitchen in that distribution. This may entail learning some parameters to enable better fine-tuning, or learning the entire RL algorithm that will be deployed in the new kitchen.

A robot trained this way can both make better use of the data collected and also collect better data, e.g., by focusing on the unusual or challenging features of the new kitchen. This meta-learning procedure requires more samples than the simple fine-tuning approach, but it only needs to occur once, and the resulting adaptation procedure can be significantly more sample efficient when deployed in the new test kitchen.

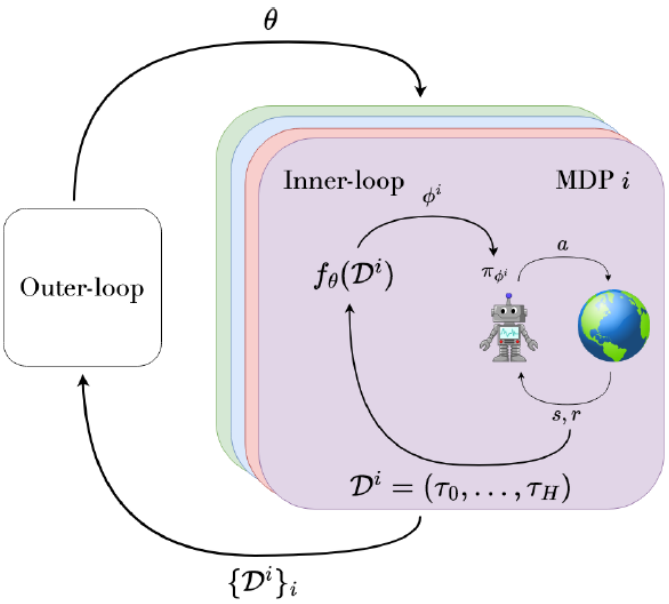
This example illustrates how, in general, meta-RL may be particularly useful when the need for efficient adaptation is frequent, so the cost of meta-training is relatively small.

1.3 Meta-RL definition

The idea of meta-RL is instead to learn (parts of) an algorithm f using machine learning. Where RL learns a policy, meta-RL learns the RL algorithm f that outputs the policy. This does not remove all of the human effort from the process, but shifts it from directly designing and implementing the RL algorithms into developing the training environments and parameterizations required for learning parts of them in a data-driven way.

Due to this bi-level structure, the algorithm for learning f is often referred to as the outer-loop, while the learned f is called the inner-loop. The relationship between the inner-loop and the outer-loop is illustrated in following figure. Since the inner-loop and outer-loop both perform learning, we refer to the inner-loop as performing adaptation and the outer-loop as performing meta-training, for the sake of clarity. The

learned inner-loop, that is, the function f, is assessed during meta-testing. We want to meta-learn an RL algorithm, or an inner-loop, that can adapt quickly to a new MDP. This meta-training requires access to a set of training MDPs. These MDPs, also known as tasks, come from a distribution denoted p(M). In principle, the task distribution can be supported by any set of tasks.



2 Grading and Submission

2.1 Grading

The grading will be based on the following criteria, with a total of 110 points:

- Model-Agnostic Meta-Learning : 100 points
- Clarity and Quality of Code : 5 points
- Clarity and Quality of Report : 5 points

2.2 Submission

The deadline for this homework is 1404/06/20 (Sep 11th 2025) at 11:59 PM. Please submit your work by following the instructions below:

- Place your solution alongside the Jupyter notebook(s).
  - Your written solution must be a single PDF file named `HW14_Solution.pdf`.
  - If there is more than one Jupyter notebook, put them in a folder named `Notebooks`.
- Zip all the files together with the following naming format:

  `DRL_HW14_[StudentNumber]_[FullName].zip`

  - Replace `[FullName]` and `[StudentNumber]` with your full name and student number, respectively. Your `[FullName]` must be in CamelCase with no spaces.
- Submit the zip file through Quera in the appropriate section.
- We provided this LaTeX template for writing your homework solution. There is a 5-point bonus for writing your solution in LaTeX using this template and including your LaTeX source code in your submission, named `HW14_Solution.zip`.
- If you have any questions about this homework, please ask them in the Homework section of our Telegram Group.
- If you are using any references to write your answers, consulting anyone, or using AI, please mention them in the appropriate section. In general, you must adhere to all the rules mentioned here and here by registering for this course.

Keep up the great work and best of luck with your submission!

3 Model-Agnostic Meta-Learning

Model-Agnostic Meta-Learning (MAML) is a prominent meta-learning algorithm introduced in the paper "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks"[1] by Chelsea Finn, Pieter Abbeel, and Sergey Levine. This paper proposes a method that trains a model to learn a good initial set of parameters, enabling it to adapt quickly to new, unseen tasks with only a few gradient steps and limited data.

3.1 Key characteristics of MAML

- **Model-Agnostic:** MAML can be applied to any model architecture that can be trained with gradient descent, making it versatile across various domains like classification, regression, and reinforcement learning.
- **Fast Adaptation:** The core idea is to learn a set of initial parameters such that a small number of gradient updates on a new task will lead to significant performance improvements on that task.
- **Bi-level Optimization:** MAML formulates meta-learning as a bi-level optimization problem. The inner loop optimizes the model parameters for a specific task, while the outer loop optimizes the meta-parameters (the initial parameters) by considering the performance across multiple tasks after their respective inner-loop adaptations.
- **Few-shot Learning:** It is particularly effective in few-shot learning scenarios where only a limited amount of training data is available for new tasks.

3.2 Conclusion

In this assignment notebook, we will implement Model-Agnostic Meta-Learning (MAML) from scratch for a custom HalfCheetahBackward environment. The goal is to learn policy parameters that can quickly adapt to new tasks in this case, running the HalfCheetah agent backward with just a few gradient steps.

References

[1] Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks

[2] A Survey of Meta-Reinforcement Learning