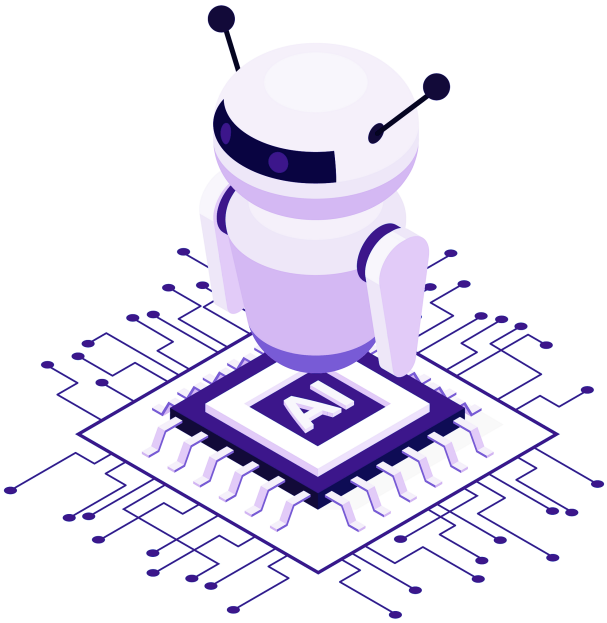


Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Homework 1:

Introduction to RL

Designed By:

MohammadHasan Abbasi

mohasabbasi@gmail.com

RIML

Spring 2025

Preface

Welcome to your first homework!

Reinforcement Learning (RL) is a fundamental branch of artificial intelligence that focuses on training agents to make sequential decisions through trial and error. Unlike supervised learning, RL does not rely on labeled data but instead learns optimal policies by interacting with an environment, receiving feedback in the form of rewards.

This assignment is designed to provide hands-on experience with RL modeling, algorithm implementation, and performance evaluation. Students will explore RL concepts through predefined environments and custom-designed settings. The homework is structured into the following sections:

- **Predefined Environments:** You will select and solve two RL environments from a given set, implementing stable-baselines3 algorithms such as PPO and DQN.
- **Custom Environments:** You will design and modify RL environments, experimenting with changes in rewards, states, and environment dynamics.
- **Algorithm Comparison:** Analyzing the performance of different RL algorithms in terms of sample efficiency, reward accumulation, and hyperparameter sensitivity.
- **Pygame Tutorial:** A short tutorial on using Pygame to create custom RL environments, helping students understand environment modeling from scratch.

The goal of this assignment is not only to implement RL models but also to develop an intuition for how different RL methods perform under varying conditions. By the end of this homework, students should be able to:

- Design and implement RL environments using OpenAI Gym/Gymnasium and Pygame.
- Train RL agents using stable-baselines3 algorithms.
- Compare RL algorithms based on efficiency and performance metrics.
- Understand the impact of environment design on learning outcomes.

We hope this assignment enhances your understanding of RL and encourages further exploration into deep reinforcement learning.

Grading

The grading will be based on the following criteria, with a total of 100 points:

Task	Points
Task 1: Solving Predefined Environments	45
Task 2: Creating Custom Environments	45
Clarity and Quality of Code	5
Clarity and Quality of Report	5
Bonus 1: Writing a wrapper for a known env	10
Bonus 2: Implementing pygame env	20
Bonus 3: Writing your report in Latex	10

Notes:

- Include well-commented code and relevant plots in your notebook.
- Clearly present all comparisons and analyses in your report.
- Ensure reproducibility by specifying all dependencies and configurations.

Acknowledgement

We would like to thank Negin Hashemi and Alireza Nobakht from the QA team for their valuable feedback on this homework.

Submission

The deadline for this homework is 1403/11/28 (February 16th 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 `HW1_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_HW1_[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 10-point bonus for writing your solution in LaTeX using this template and including your LaTeX source code in your submission, named `HW1_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!

Contents

1 Setup Instructions

Before starting this assignment, ensure that your environment is correctly set up with the required libraries and dependencies. The practical component of this homework will be completed in the provided Jupyter Notebook (`HW1_Notebook.ipynb`), which is attached along with this document.

1.1 Environment Setup

You may use `Colab` if you don't want to set up a local environment. If you used a local setup, upload the notebook and run it directly in Google Colab also. To run the provided notebook and complete the assignments, follow these steps:

1. **Install Python and Required Libraries:** Ensure that you have Python 3.8+ installed. We recommend using a virtual environment for package management.

   ```
   python -m venv rl_homework_env
   source rl_homework_env/bin/activate  # On Linux/macOS
   rl_homework_env\Scripts\activate  # On Windows
   ```

   Next, install the required dependencies:

   ```
   pip install -r requirements.txt
   ```

   The provided `requirements.txt` file contains all necessary packages, including:

   - `gymnasium` (for RL environments)
   - `stable-baselines3` (for RL algorithms)
   - `pygame` (for environment customization)
   - `matplotlib, seaborn` (for visualization)
   - `numpy, pandas` (for data processing)
   - `jupyterlab` or `notebook` (for jupyter set up)

2. **Download and Open the Notebook:** Navigate to the directory where you extracted the homework files and launch the Jupyter Notebook:

   ```
   jupyter notebook HW1_Notebook.ipynb
   ```

3. **Verify Installations:** Run the first few cells in the notebook to ensure that all dependencies are correctly installed.

4. **Additional Notes:**

   - If you face issues with Gymnasium environments, ensure that the necessary dependencies (such as `pygame` and `mujoco` for specific environments) are installed.

1.2 Submission Requirements

- Ensure that your Jupyter Notebook runs without errors before submission.
- Include all code, outputs, and plots within the notebook.
- Submit a ZIP file containing:
  - The completed `HW1_Notebook.ipynb`.
  - Any additional scripts used for custom environments.

If you encounter any issues, please reach out on the course forum or contact the teaching assistants.

2 Problem Descriptions

This homework consists of multiple tasks that will guide you through designing, implementing, and analyzing reinforcement learning environments and algorithms. You will work with both predefined environments and custom ones, applying different RL algorithms from stable-baselines3 and comparing their performance.

2.1 Task 1: Solving Predefined Environments

Objective: Solve two predefined reinforcement learning environments using stable-baselines3 algorithms.

2.1.1 Instructions:

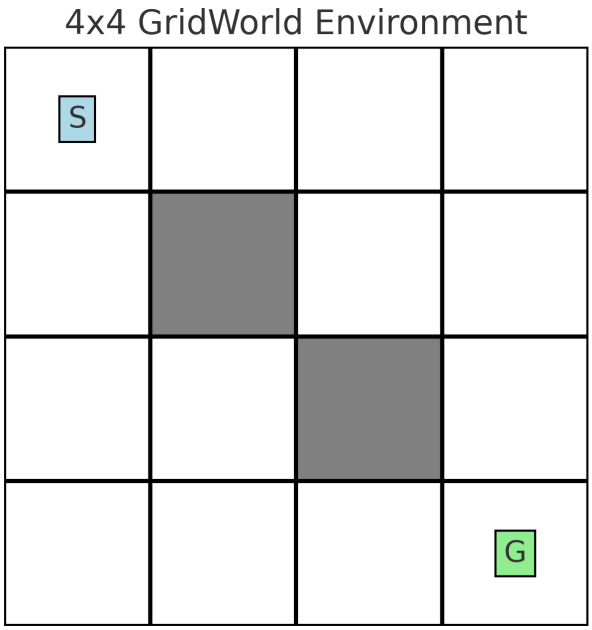
- Choose **two** environments from the following options:
  1. **CartPole** - https://gymnasium.farama.org/environments/classic_control/cart_pole/
  2. **Frozen Lake** https://gymnasium.farama.org/environments/toy_text/frozen_lake/
  3. **Flappy Bird** (Custom Environment) - No official Gymnasium link
  4. **Taxi** - https://gymnasium.farama.org/environments/toy_text/taxi/
- Implement and train agents using at least two RL algorithms (e.g., PPO, DQN).
- Try to write a wrapper for reward function and plot the changes. (Bonus)
- Record and analyze:
  - Learning curves (reward over episodes)
  - Sample efficiency (training time and required episodes)
  - Algorithm hyperparameters and their effect
- Visualize and compare the results.
- Write down your thoughts on the fitness of SL for problems like this. Guess in what problems SL fails or is not practical, explain why (bonus).

Deliverables:

- Code implementation in the attached Jupyter Notebook.
- Graphs comparing the performance of different algorithms.
- A brief discussion of the findings in the report.

2.2 Task 2: Creating Custom Environments

Objective: Design and implement custom reinforcement learning environments with Gymnasium.



2.2.1 Instructions:

- In this question, you are required to model a custom 4*4 grid-world problem as Markov Decision Processes (MDPs). You must define the following components:
  - **State Space (S)**: The set of all possible states the agent can be in.
  - **Action Space (A)**: The set of all possible actions the agent can take.
  - **Reward Function (R)**: The reward the agent receives for taking an action in a given state.
  - **Transition Probability (P)**: The probability of transitioning to a new state given the current state and action. If the environment is deterministic, this can be omitted.
- Train agents using at least one algorithm and evaluate performance.

Deliverables:

- Code for the environment.
- Training results, learning curves, and observations.
- A short discussion on implementation and how modifications affect learning.

2.3 Task 3: Pygame for RL environment (Bonus)

Objective: Learn to use Pygame for RL environment customization.

2.3.1 Instructions:

- Complete a step-by-step Pygame tutorial provided in the notebook.
- Extend the example by adding new:
  - Obstacles
  - Rewards
  - Action mechanics
- Experiment with different agent interactions.

Deliverables:

- Updated Pygame environment.
- Screenshots of modifications and explanations.

References

[1] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, 2nd Edition, 2020. Available online: http://incompleteideas.net/book/the-book-2nd.html

[2] A. Raffin et al., "Stable Baselines3: Reliable Reinforcement Learning Implementations," GitHub Repository, 2020. Available: https://github.com/DLR-RM/stable-baselines3.

[3] Gymnasium Documentation. Available: https://gymnasium.farama.org/.

[4] Pygame Documentation. Available: https://www.pygame.org/docs/.

[5] CS 285: Deep Reinforcement Learning, UC Berkeley, Pieter Abbeel. Course material available: http://rail.eecs.berkeley.edu/deeprlcourse/.

[6] Cover image designed by freepik