Instruction:

In this final project, you will work in groups of 2-3 to develop a reinforcement learning solution for an OpenAI Gymnasium environment of your choosing. The goal of this project is to gain hands-on experience with state-of-the-art AI techniques and the OpenAI Gymnasium library, and to showcase your ability to apply these techniques to a real-world problem.

Through this project, you will have the opportunity to explore the latest advances in reinforcement learning, including Q-learning, SARSA, actor-critic methods, and other algorithms used in industry and academia. You will also gain experience in working with OpenAI Gymnasium, a powerful and widely used library for developing and testing AI solutions.

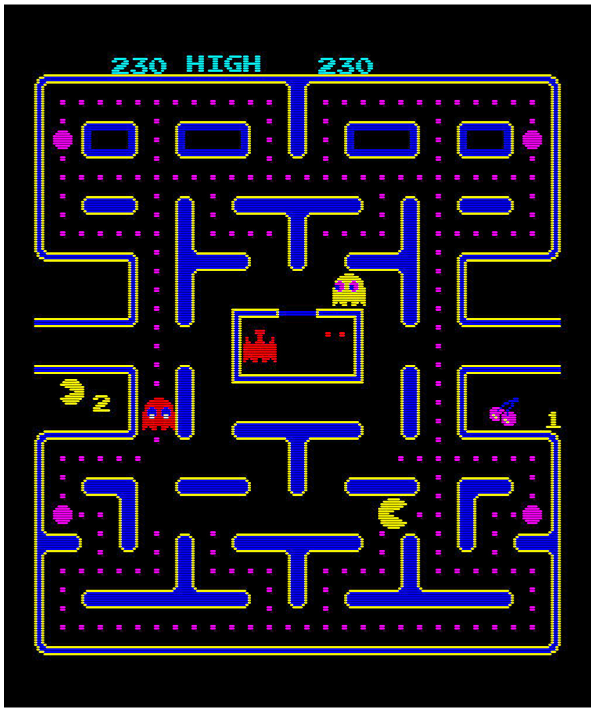
At the end of this project, you will have developed a working reinforcement learning solution for your chosen environment, and will have gained valuable experience in problem-solving, algorithm development, and project management. You will also have the opportunity to present your work to your peers and instructors, and to receive feedback on your approach and results.

We encourage you to be creative and innovative in your approach to this project, and to collaborate closely with your team members to achieve the best possible results. We are excited to see the solutions you develop and to support you in your exploration of the fascinating world of AI and reinforcement learning.

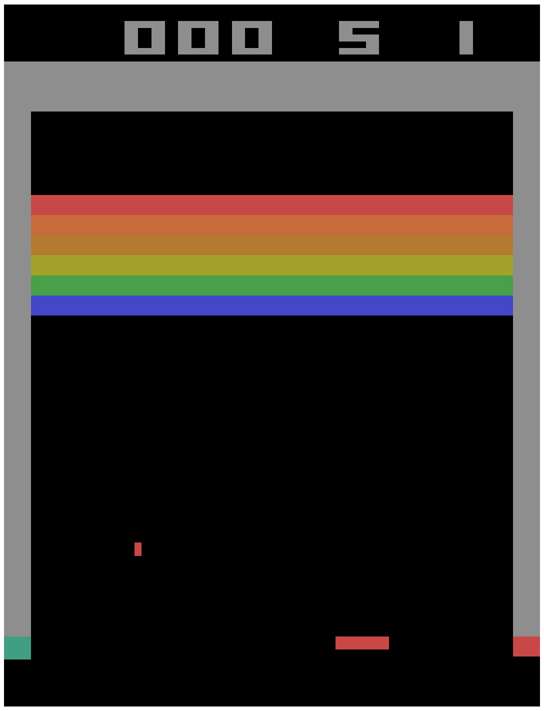
Topics

There are a wide variety of environments available in the OpenAI Gymnasium library, each with their own unique challenges and rewards. For example, you might choose to work on an environment like Atari Breakout, where the agent must learn to navigate a ball and paddle game using reinforcement learning. Alternatively, you might tackle an environment like CartPole, where the agent must learn to balance a pole on a cart by moving left or right. Other popular environments include classic games like Pac-Man, robotic simulations like Roboschool, and complex physics simulations like MuJoCo. Ultimately, the choice of environment will depend on your interests and the skills and experience of your team.

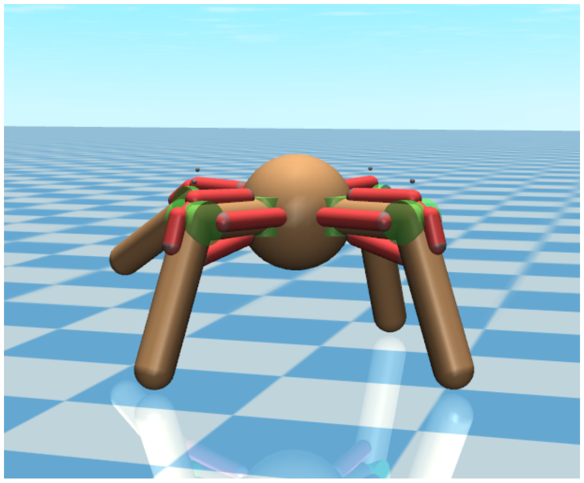
Example of Gymnasium library:



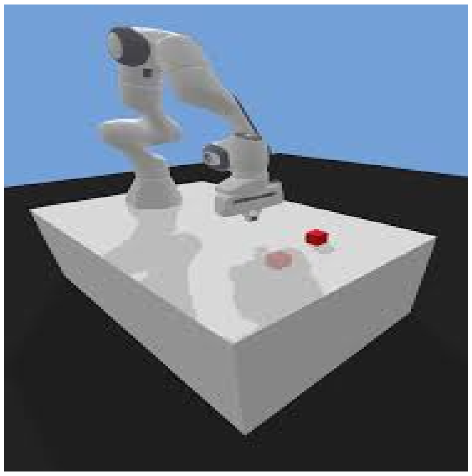
Pac-man



Breakout



MuJoCo



Franka Emika Panda robot

<u>Requirement</u>

- The project must be completed in groups of 2-3 students.
- Each group must choose an environment from the OpenAI Gymnasium library and develop a reinforcement learning solution for it.

- You may use online examples as a starting point for your project, but you must adapt the code to your own settings and ensure that it is fully functional and able to demonstrate the capabilities of your solution during a demo.
- Your solution must be based on a deep understanding of the underlying reinforcement learning algorithms and principles, and you should be able to explain the logic and reasoning behind your approach in detail.
- Each group must submit a written report describing their approach, results, and lessons learned during the project.
- Each group must also give a demonstration of their solution, showcasing its capabilities and strengths. The demo should include a clear explanation of how the solution works and what its benefits are.
- Collaboration within groups is strongly encouraged, and all team members should be actively involved in the development and testing of the solution.

Grading:

You will be evaluated based on the following criteria:

- The completeness and clarity of your solution and report will be a significant portion of the final grade.
- Your report should clearly explain the problem you tackled, your approach to solving it, and the results you achieved.
- If students from the same group share the same report, it must be clearly stated which team member was responsible for each part of the project and the report.
- The report should also include a detailed analysis of the strengths and weaknesses of your solution, along with any lessons learned during the project.
- The report should be well-structured, easy to follow, and free from errors in grammar and spelling.

Appendix:

If you're new to OpenAI Gymnasium library, a simple tutorial is available to help you get started. In addition, there are many online documents and resources that you can refer to for further guidance and support. OpenAI Gym Documentation: https://gymnasium.farama.org/ or https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/

OpenAI Gym is a toolkit for developing and comparing reinforcement learning algorithms. It provides an interface for developing and testing RL algorithms using a wide range of environments, including classic Atari games, board games, robotics simulations, and more.
Here are the basic steps for using OpenAI Gym:

1. Install OpenAI Gym: First, you need to install the OpenAI Gym library. You can install it using pip, the Python package manager, with the command: "pip install gym"
2. Choose an environment: Once you have installed Gym, you can choose an environment to work with. Gym provides a wide range of environments, such as Atari games, classic control problems, and robotics simulations. You can browse the list of available environments on the OpenAI Gym website.
3. Create a Gym environment: Once you have chosen an environment, you can create an instance of the environment in your code using the "gym.make()" function. This creates an instance of the environment that you can use to interact with the environment.
4. Reset the environment: Before you can start interacting with the environment, you need to reset it to its initial state using the "reset()" function.
5. Interact with the environment: Once you have reset the environment, you can interact with it by taking actions and receiving observations and rewards. To take an action in the environment, you use the "step()" function, which takes an action as input and returns a tuple containing the next observation, reward, done flag, and any additional information.
6. Repeat the process: You can continue to interact with the environment by taking actions and receiving feedback until the episode is complete. Once the episode is complete, you can reset the environment and start the process over again.

A simple API explanation:

1. gym.make(): This function creates an instance of the environment you want to work with. It takes the name of the environment as a parameter and returns the environment object.
2. env.reset(): This function resets the environment to its initial state and returns the initial observation.
3. env.render(): This function renders the environment so you can see what's happening. You can use it to visualize the actions your agent takes.
4. env.step(): This function takes an action as a parameter and returns the next observation, reward, done, and info. Done is a boolean indicating if the episode is over, and info contains additional debugging information.
5. env.action_space: This attribute contains information about the action space of the environment, such as the shape of the action space and the number of discrete actions.
6. env.observation_space: This attribute contains information about the observation space of the environment, such as the shape of the observation space and the type of observations.