

# Atari Breakout with Reinforcement Learning

---

## Table of Contents

- [Atari Breakout with Reinforcement Learning](#)
  - [Table of Contents](#)
  - [Overview](#)
  - [Requirements](#)
  - [Setting Up a Conda Environment](#)
  - [Installing Dependencies](#)
  - [Running the Code](#)
  - [Troubleshooting](#)
  - [Contributing](#)
- [Project badge](#)
  - [Deep Q-learning](#)
- [Resources](#)
  - [Read or watch:](#)
  - [References:](#)
  - [Learning Objectives](#)
- [Requirements](#)
  - [General](#)
- [Tasks](#)
  - [0. Breakout](#)

## Overview

This project aims to train a reinforcement learning agent to play Atari's Breakout game. We use Python 3.5 and various libraries like Gym, Keras, and Keras-RL to accomplish this. The project contains two main scripts:

- [train.py](#): Trains the agent using DQN (Deep Q-Network).
- [play.py](#): Allows the trained agent to play the game.

## Requirements

- Python 3.5
- NumPy 1.15
- Gym 0.17.2
- Keras 2.2.5
- Keras-RL 0.4.2

## Setting Up a Conda Environment

Conda is a package and environment management system that allows you to install software packages and manage different environments for various projects. Follow these steps to set up a Conda environment:

1. **Install Anaconda or Miniconda:** Download from [Anaconda](#) or [Miniconda](#).
2. **Open Terminal:** Open your terminal (Command Prompt on Windows, Terminal on macOS or Linux).
3. **Create a New Environment:** Run `conda create --name atari_breakout python=3.5`.
4. **Activate the Environment:** Run `conda activate atari_breakout` on Windows or `source activate atari_breakout` on macOS and Linux.

#### Environment Creation/Configuration

Create these files under a folder of your choice.

Environment.yml

```
name: deep
channels:
  - anaconda
  - defaults
dependencies:
  - python=3.6
  - pip
  - pip:
    - pip
```

(notice that I wrote pip three times; the first one is to install pip, but, that installs pip version 20.2.4  
the two other lines are to upgrade pip to the latest version to avoid annoying errors/warnings later)

Requirements.txt

```
h5py==2.10.0
keras==2.2.4
keras-rl==0.4.2
numpy==1.18.5
opencv-python==4.4.0.42
pyyaml==5.3.1
six==1.15.0
gym
Pillow
tensorflow==1.14.0
```

Store both these files in a folder and open command prompt (powershell for windows and terminal in ubuntu :) ) and cd to the directory where you put those files.

You need to create the anaconda environment, so enter this command:

```
conda env create -f environment.yml
```

Second, you should activate the environment created

```
conda activate deep
```

## Initialize Conda for Your Shell

**Open Terminal:** Open your terminal (Command Prompt on Windows, Terminal on macOS or Linux).

**Run Conda Init:** Run the `conda init` command for your specific shell. The shell name is usually the name of the terminal you are using. For example, if you are using `bash`, you would run:

```
bash
```

```
conda init bash
```

If you're using a different shell, replace `bash` with the name of your shell (e.g., `zsh`, `fish`, `cmd.exe`, etc.).

**Restart Shell:** Close and reopen your terminal window or start a new shell session to apply the changes.

## Verify Configuration

After restarting your shell, try activating your environment again:

```
bash
```

```
conda activate deepQ
```

If everything is set up correctly, this should activate your `deepQ` environment without any issues.

(notice that the name of the environment `deep` is mentioned in the first line in the `environment.yml` file)

So now, that you are inside the `anaconda` environment. You need to complete the installation of the remaining requirements.

```
pip install -r requirements.txt # installs all packages in the file
```

Lastly, is to install `atari_py`  
Windows:

```
pip install --no-index -f https://github.com/Kojoley/atari-py/releases atari_py
```

For more details, check the [Conda documentation](#).

## Installing Dependencies

After activating your Conda environment, install the required packages:

```
conda install numpy=1.15 gym=0.17.2
pip install keras==2.2.5 keras-rl==0.4.2
```

**Playing Atari Games:** The `atari_py` library is generally used for training machine learning models rather than human gameplay. However, you can use the `Gym` library to create an environment where you can play Atari games using Python code. It won't be a traditional gaming experience but more of a programmatic one.

**Watching the Model Play:** You can absolutely watch your trained model play an Atari game. When you set up your `Gym` environment, you can enable rendering to visualize the game in a window. This is often done by calling `env.render()` within the game loop in your code.

Here's a simplified example using `Gym` to visualize a random agent playing an Atari game:

```
import gym

# Initialize environment
env = gym.make('Breakout-v0')

# Number of episodes
for i_episode in range(5):
    observation = env.reset()
    for t in range(1000):
        env.render()
        action = env.action_space.sample() # Take a random action
        observation, reward, done, info = env.step(action)
        if done:
            print(f"Episode finished after {t+1} timesteps")
            break

env.close()
```

In this example, `env.render()` opens a window displaying the game. You can replace the random action with actions determined by your trained model to watch it play.

Remember to close the rendering window by calling `env.close()` when you're done.

## Running the Code

- Train the Agent: Run `python train.py` to train the agent. The trained model will be saved as `policy.h5`.
- Play the Game: Run `python play.py` to see the trained agent in action.

## Troubleshooting

- Conda Command Not Found: Make sure Anaconda/Miniconda is installed and added to your system's PATH. See detailed guide.
- Environment Doesn't Exist: Ensure you have the correct Gym version and have installed the Atari dependencies (`pip install gym[atari]`).

## Contributing

Feel free to contribute to this project by opening issues or submitting pull requests.

## Project badge

---

## Deep Q-learning

Master

By: Alexa Orrico, Software Engineer at Holberton School

Weight: 6

Manual QA review must be done (request it when you are done with the project)

## Resources

---

### Read or watch:

Deep Q-Learning - Combining Neural Networks and Reinforcement Learning  
Replay Memory Explained - Experience for Deep Q-Network Training  
Training a Deep Q-Network - Reinforcement Learning  
Training a Deep Q-Network with Fixed Q-targets - Reinforcement Learning

### References:

Setting up anaconda for keras-rl  
keras-rl  
rl.policy

```
rl.memory
rl.agents.dqn
Playing Atari with Deep Reinforcement Learning
```

## Learning Objectives

What is Deep Q-learning?  
What is the policy network?  
What is replay memory?  
What is the target network?  
Why must we utilize two separate networks during training?  
What is keras-rl? How do you use it?

## Requirements

---

### General

Allowed editors: vi, vim, emacs  
All your files will be interpreted/compiled on Ubuntu 16.04 LTS using python3 (version 3.5)  
Your files will be executed with numpy (version 1.15), gym (version 0.17.2), keras (version 2.2.5), and keras-rl (version 0.4.2)  
All your files should end with a new line  
The first line of all your files should be exactly `#!/usr/bin/env python3`  
A README.md file, at the root of the folder of the project, is mandatory  
Your code should use the pycodestyle style (version 2.4)  
All your modules should have documentation (python3 -c 'print(\_\_import\_\_("my\_module").\_\_doc\_\_)')  
All your classes should have documentation (python3 -c 'print(\_\_import\_\_("my\_module").MyClass.\_\_doc\_\_)')  
All your functions (inside and outside a class) should have documentation (python3 -c 'print(\_\_import\_\_("my\_module").my\_function.\_\_doc\_\_)' and python3 -c 'print(\_\_import\_\_("my\_module").MyClass.my\_function.\_\_doc\_\_)')  
All your files must be executable  
Your code should use the minimum number of operations

### Installing Keras-RL

```
pip install --user keras-rl
```

Dependencies (that should already be installed)

```
pip install --user keras==2.2.4
```

```
pip install --user Pillow
```

pip install --user h5py

# Tasks

---

## 0. Breakout

mandatory

Write a python script train.py that utilizes keras, keras-rl, and gym to train an agent that can play Atari's Breakout:

Your script should utilize keras-rl's DQNAgent, SequentialMemory, and EpsGreedyQPolicy  
Your script should save the final policy network as policy.h5

Write a python script play.py that can display a game played by the agent trained by train.py:

Your script should load the policy network saved in policy.h5  
Your agent should use the GreedyQPolicy

Repo:

GitHub repository: holbertonschool-machine\_learning  
Directory: reinforcement\_learning/deep\_q\_learning  
File: train.py, play.py