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
  - Resources

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.

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

## 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.


## Resources

Read or watch:

    An introduction to Reinforcement Learning
    Simple Reinforcement Learning: Q-learning
    Markov Decision Processes (MDPs) - Structuring a Reinforcement Learning
Problem
    Expected Return - What Drives a Reinforcement Learning Agent in an MDP
    Policies and Value Functions - Good Actions for a Reinforcement Learning
Agent
    What do Reinforcement Learning Algorithms Learn - Optimal Policies
    Q-Learning Explained - A Reinforcement Learning Technique
    Exploration vs. Exploitation - Learning the Optimal Reinforcement Learning
Policy
    OpenAI Gym and Python for Q-learning - Reinforcement Learning Code Project
    Train Q-learning Agent with Python - Reinforcement Learning Code Project
    Markov Decision Processes

Definitions to skim:

    Reinforcement Learning
    Markov Decision Process
    Q-learning

References:

    OpenAI Gym
```

```
    OpenAI Gym: Frozen Lake env

Learning Objectives

    What is a Markov Decision Process?
    What is an environment?
    What is an agent?
    What is a state?
    What is a policy function?
    What is a value function? a state-value function? an action-value function?
    What is a discount factor?
    What is the Bellman equation?
    What is epsilon greedy?
    What is Q-learning?

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), and gym (version
0.7)
    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 OpenAI's Gym

pip install --user gym
```