Instructions

Declaimer

- Most work in the following projects are modified from open source code and are not written by us;
- Some projects especially Atari need GPU to boost the speed of train. You can book GPU from our college. Detail instructions are in https://gist.github.com/fly-cfchen/073749982e2e71e24f4d21c4a5897b3e, which contains how to book GPU and how to build the environment;
- If you want more details, please refer to the *Readme file* in each project.

Part 1 Maze

Note

This project is based on http://www.cnblogs.com/cjnmy36723/p/7017549.html

Overview

To strengthen the understanding of DQN, we do some research and design a small example of **Maze Problem**, which apply neural network to train the Q value.

Installation Dependencies

- python 2.7 or 3
- tensorflow

How to run

```
git clone https://github.com/qiyexue777/maze.git

cd maze

python q-learning.py

python DQN.py
```

Part 2 FlappyBird

Note

This project is based on https://github.com/yenchenlin/DeepLearningFlappyBird

Overview

This project follows the description of the Deep Q Learning algorithm described in Playing Atari with Deep Reinforcement Learning and shows that this learning algorithm can be further generalized to the notorious Flappy Bird.

Installation Dependencies:

- Python 2.7 or 3
- TensorFlow 0.7
- pygame
- OpenCV-Python

How to Run?

```
git clone https://github.com/qiyexue777/DeepLearningFlappyBird.git
cd DeepLearningFlappyBird
python deep_q_network.py
```

Part 3 Atari

Note

This project is based on https://github.com/kuz/DeepMind-Atari-Deep-Q-Learner

Overview

This project contains the source code of DQN 3.0, a Lua-based deep reinforcement learning architecture, necessary to reproduce the experiments described in the paper "Human-level control through deep reinforcement learning", Nature 518, 529–533 (26 February 2015) doi:10.1038/nature14236.

Installation Dependencies:

The installation requires Linux with apt-get.

Note: In order to run the GPU version of DQN, you should additionally have the NVIDIA® CUDA® (version 5.5 or later) toolkit installed prior to the Torch installation below. This can be downloaded from https://developer.nvidia.com/cuda-toolkit and installation instructions can be found in http://docs.nvidia.com/cuda/cuda-getting-started-guide-for-linux

To train DQN on Atari games, the following components must be installed:

- LuaJIT and Torch 7.0
- nngraph
- Xitari
- AleWrap

To install all of the above in a subdirectory called 'torch', it should be enough to run

```
./install_dependencies.sh
```

from the base directory of the package.

Note: The above install script will install the following packages via apt-get: build-essential, gcc, g++, cmake, curl, libreadline-dev, git-core, libjpeg-dev, libpng-dev, ncurses-dev, imagemagick, unzip

Training DQN on Atari games

Prior to running DQN on a game, you should copy its ROM in the 'roms' subdirectory. It should then be sufficient to run the script

```
./run_cpu <game name>
```

Or, if GPU support is enabled,

```
./run_gpu <game name>
```

Note: On a system with more than one GPU, DQN training can be launched on a specified GPU by setting the environment variable GPU_ID, e.g. by

```
GPU_ID=2 ./run_gpu <game name>
```

If GPU_ID is not specified, the first available GPU (ID 0) will be used by default.

Storing a .gif for a trained network

Once you have a snapshot of a network you can run

```
./test_gpu <game name> <snapshopt filename>
```

to make it play one game and store the .gif under `gifs`. For example

```
./test_gpu breakout DQN3_0_1_breakout_FULL_Y.t7
```