

mandalalala

OpenAI教程

英文版：https://gym.openai.com/docs

2016年 5 月 4日，OpenAI发布了人工智能研究工具集 OpenAI Gym。OpenAI Gym是一款用于研发和比较学习算法的工具包。它与很多数值计算库兼容，比如tensorflow和theano。现在支持的语言主要是python。

openai gym 是一个增强学习（reinforcement learning,RL）算法的测试床（testbed）。增强学习和有监督学习的评测不一样。有监督学习的评测工具是数据。只要提供一批有标注的数据18:34:13就能进行有监督学习的评测。增强学习的评测工具是环境。需要提供一个环境给 Agent 运行，才能评测 Agent 的策略的优劣。OpenAI Gym 是提供各种环境的开源工具包。

增强学习有几个基本概念：

- (1) agent：智能体，也就是机器人，你的代码本身。
- (2) environment：环境，也就是游戏本身，openai gym提供了多款游戏，也就是提供了多个环境。
- (3) action：行动，比如玩超级玛丽，向上向下等动作。
- (4) state：状态，每次智能体做出行动，环境会相应地做出反应，返回一个状态和奖励。
- (5) reward：奖励：根据游戏规则的分。智能体不知道怎样才能得分，它通过不断地尝试来理解游戏规则，比如它在这个状态做出向上的动作，得分，那么下一次它处于这个环境状态，就倾向于做出向上的动作。

导航

[博客园](#) [首页](#) [联系](#) [订阅](#) [管理](#)

≤	2017年10月						≥
日	一	二	三	四	五	六	
24	25	26	27	28	29	30	
1	2	3	4	5	6	7	
8	9	10	11	12	13	14	
15	16	17	18	19	20	21	
22	23	24	25	26	27	28	
29	30	31	1	2	3	4	

公告

昵称：[mandalalala](#)

园龄：[9个月](#)

粉丝：[1](#)

关注：[0](#)

[+加关注](#)

统计

随笔 - 24 文章 - 0 评论 - 0

搜索

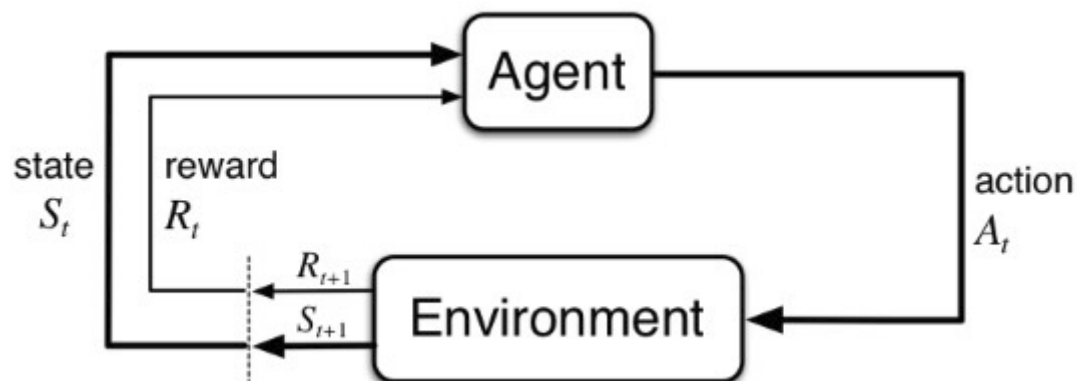
常用链接

[我的随笔](#)

[我的评论](#)

[我的参与](#)

[最新评论](#)



OpenAI Gym由两部分组成：

1. gym开源库：测试问题的集合。当你测试增强学习的时候，测试问题就是环境，比如机器人玩游戏，环境的集合就是游戏的画面。这些环境有一个公共的接口，允许用户设计通用的算法。
2. OpenAI Gym服务。提供一个站点（比如对于游戏cartpole-v0：
<https://gym.openai.com/envs/CartPole-v0>）和api，允许用户对他们的测试结果进行比较。

gym的代码在这上面：<https://github.com/openai/gym>

gym的核心接口是Env，作为统一的环境接口。Env包含下面几个核心方法：

1. reset(self):重置环境的状态，返回观察。
2. step(self,action):推进一个时间步长，返回observation，reward，done，info
3. render(self,mode='human',close=False):重绘环境的一帧。默认模式一般比较友好，如弹出一个窗口。

安装

1.Linux(没试过)：

```
apt-get install -y python-numpy python-dev cmake zlib1g-dev libjpeg-dev xvfb libav-tools xorg-dev python-opengl libboost-all-dev libsdl2-dev swig
```

2.Windows（有两种方法）：

[我的标签](#)

我的标签

[paper\(1\)](#)

随笔档案

[2017年5月 \(1\)](#)

[2017年4月 \(2\)](#)

[2017年3月 \(5\)](#)

[2017年2月 \(6\)](#)

[2016年12月 \(3\)](#)

[2016年10月 \(1\)](#)

[2016年9月 \(2\)](#)

[2016年7月 \(1\)](#)

[2016年5月 \(1\)](#)

[2016年4月 \(1\)](#)

[2016年1月 \(1\)](#)

文章分类

[paper](#)

阅读排行榜

[1. OpenAI教程\(777\)](#)

[2. DCGAN及其tensorflow版源码解读\(137\)](#)

[3. Regions with CNN features \(R-CNN\)\(76\)](#)

[4. 从GBDT到Xgboost\(48\)](#)

[5. pip更新\(26\)](#)

推荐排行榜

[1. OpenAI教程\(1\)](#)

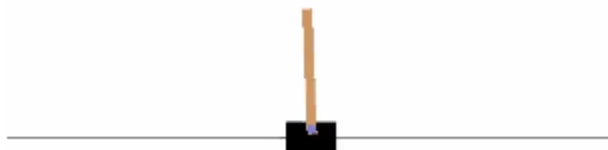
(1) 使用pip :

```
pip install gym
```

(2) 使用git :

```
git clone https://github.com/openai/gym
cd gym
pip install -e . # minimal install
pip install -e .[all] # full install (this requires cmake and a recent pip version)
```

接下来以cartpole-v0 (<https://gym.openai.com/envs/CartPole-v0>) 举例。



这个游戏的规则是让杆不倒。Openai gym提供了行动的集合，环境的集合等等。Cartpole-v0来说，动作空间包括向左拉和向右拉两个动作。其实你并不需要关心它的动作空间是什么，当你的学习算法越好，你就不需要解释这些动作。

运行环境

运行CartPole-v0环境1000个时间步(timestep)。

```
import gym
env = gym.make('CartPole-v0')
env.reset()
for _ in range(1000):
    env.render()
env.step(env.action_space.sample()) # take a random action
```

可以看到随机控制算法发散，游戏很快结束。

观察

如果我们想做得好一点，观察周围的环境是必须的。环境的step函数返回四个值：

- Observation(object):返回一个特定环境的对象，描述对环境的观察。比如，来自相机的像素数据，机器人的关节角度和关节速度，或棋盘游戏中的棋盘状态。
- Reward(float)：返回之前动作收获的总的奖励值。不同的环境计算方式不一样，但总体的目标是增加总奖励。
- Done(boolean)：返回是否应该重新设置（reset）环境。大多数游戏任务分为多个环节（episode），当done=true的时候，表示这个环节结束了。
- Info(dict)：用于调试的诊断信息（一般没用）。

这是一个典型的“智能体-环境循环”的实现。每个时间步长（timestep），智能体选择一个行动，环境返回一个观察和奖励值。

过程一开始调用reset，返回一个初始的观察。并根据done判断是否再次reset。



```
import gym
env = gym.make('CartPole-v0')
for i_episode in range(20):
    observation = env.reset()
    for t in range(100):
        env.render()
        print(observation)
        action = env.action_space.sample()
        observation, reward, done, info = env.step(action)
        if done:
            print("Episode finished after {} timesteps".format(t+1))
            break
```



以上代码有20个episode，打印每次的环境观察值，随机采取行动，返回环境的观察值、奖励、done和调试信息。当done为true时，该episode结束，开始下一个episode。可以看到，观察了环境，每次坚持的时间好

像稍微长了一点。

运行结果如下：



```
[-0.061586   -0.75893141  0.05793238  1.15547541]
[-0.07676463 -0.95475889  0.08104189  1.46574644]
[-0.0958598  -1.15077434  0.11035682  1.78260485]
[-0.11887529 -0.95705275  0.14600892  1.5261692 ]
[-0.13801635 -0.7639636   0.1765323   1.28239155]
[-0.15329562 -0.57147373  0.20218013  1.04977545]
Episode finished after 14 timesteps
[-0.02786724  0.00361763 -0.03938967 -0.01611184]
[-0.02779488 -0.19091794 -0.03971191  0.26388759]
[-0.03161324  0.00474768 -0.03443415 -0.04105167]
```



空间

以上，我们都是从环境的动作空间中随机选择动作。每个环境都有一个space对象，用来描述有效的

动作和观察：

```
import gym
env = gym.make('CartPole-v0')
print(env.action_space)
#> Discrete(2)
print(env.observation_space)
#> Box(4,)
```

在上面这个例子中，action取非负整数0或1。Box表示一个n维的盒子，因此observation是一个4维的数组。我们可以试试box的上下限。

```
print(env.observation_space.high)
#> array([ 2.4          ,          inf,  0.20943951,          inf])
print(env.observation_space.low)
#> array([-2.4          ,          -inf, -0.20943951,          -inf])
```

Box和discrete是最常见的space。你可以从space中取样或者验证是否属于它。

```
from gym import spaces
space = spaces.Discrete(8) # Set with 8 elements {0, 1, 2, ..., 7}
x = space.sample()
assert space.contains(x)
assert space.n == 8
```

环境

Gym的主要目的是提供一个大环境集合，具有一个公共接口，并且允许比较算法。你可以列出这些环境。

```
from gym import envs
print(envs.registry.all())
#> [EnvSpec(DoubleDunk-v0), EnvSpec(InvertedDoublePendulum-v0), EnvSpec(BeamRider-
v0), EnvSpec(Phoenix-ram-v0), EnvSpec(Asterix-v0), EnvSpec(TimePilot-v0),
EnvSpec(Alien-v0), EnvSpec(Robotank-ram-v0), EnvSpec(CartPole-v0), EnvSpec(Berzerk-
v0), EnvSpec(Berzerk-ram-v0), EnvSpec(Gopher-ram-v0), ...]
```

这列出了一系列的EnvSpec。它们为特定任务定义特定参数，包括运行的实验数目和最多的步数。比如，[EnvSpec\(Hopper-v1\)](#)定义了一个环境，环境的目标是让一个2D的模拟机器跳跃。[EnvSpec\(Go9x9-v0\)](#)定义了9*9棋盘上的围棋游戏。

这些环境ID被视为不透明字符串。为了确保与未来的有效比较，环境永远不会以影响性能的方式更改，只能由较新的版本替代。我们目前使用v0为每个环境添加后缀，以便将来的替换可以自然地称为v1，v2等。

记录和加载结果

Gym使得记录算法在环境中的性能变得简单，同时能记录学习过程的视频。只要使用monitor如下：



```
import gym
env = gym.make('CartPole-v0')
env.monitor.start('/tmp/cartpole-experiment-1')
for i_episode in range(20):
    observation = env.reset()
    for t in range(100):
        env.render()
        print(observation)
        action = env.action_space.sample()
        observation, reward, done, info = env.step(action)
        if done:
            print("Episode finished after {} timesteps".format(t+1))
            break
env.monitor.close()
```



跑了一下发现有错：

env.monitor is deprecated. Wrap your env with gym.wrappers.Monitor to record data.

改为如下：

```
from gym.wrappers import Monitor
env = Monitor(directory='/tmp/cartpole-experiment-0201', video_callable=False,
write_upon_reset=True)(env)
env.close()
```

(mark一下找bug思路，gym\monitoring\tests里面有测试的案例，参考test_monitor.py写代码。)

产生的结果放到'/tmp/cartpole-experiment-1'这个文件夹中，你可以加载到评分板。

```
Finished writing results. You can upload them to the scoreboard via gym.upload('E:\\tmp\\cartpole-experiment-1')
```

monitor支持将一个环境的多个案例写入一个单独的目录。

然后你可以把你的结果加载到OpenAI Gym：

```
import gym
gym.upload('/tmp/cartpole-experiment-1', api_key=' sk_FYp0Gc1dQU69epifs7ZE6w')
```

输出应该是这样：



```
[2016-04-22 23:16:03,123] Uploading 20 episodes of training data
[2016-04-22 23:16:04,194] Uploading videos of 2 training episodes (6306 bytes)
[2016-04-22 23:16:04,437] Creating evaluation object on the server with learning
curve and training video
[2016-04-22 23:16:04,677]
```

You successfully uploaded your agent evaluation to
OpenAI Gym! You can find it at:

https://gym.openai.com/evaluations/eval_tmX7tssiRVtYzZk0PlWhKA



估值

每次加载会产生一个估值对象。官方文件说，应该创建一个Gist（被墙了）显示怎么复制你的结果。估值页会有一个如下方框：

...

Gist URL

你还可以在加载的时候提供你的gist，通过一个writeup参数

```
import gym
gym.upload('/tmp/cartpole-experiment-1',
```



```
writeup='https://gist.github.com/gdb/b6365e79be6052e7531e7ba6ea8caf23', api_key='
sk_FYp0Gc1dQU69epifs7ZE6w')
```

这一步是将你的结果提交到在线网站上进行评估，你的结果会被自动评分，并且会产生一个漂亮的界面，如：

https://gym.openai.com/evaluations/eval_lr5NHkdNRGqympBDcdNNNw

在大多数环境中，你的目标是用最少的步数达到性能的要求（有一个阈值）。而在一些特别复杂的环境中，阈值是什么还不清楚，因此，你的目标是最优化性能。

注意，现在writeup被舍弃了，所以不需要writeup。

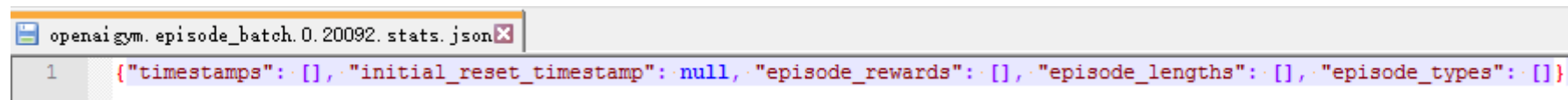
另外，api_key是指

再mark一个错误：

```
raise error.Error("[%s] You didn't have any recorded training data in {}. Once you've used
'env.monitor.start(training_dir)' to start recording, you need to actually run some rollouts. Please join the
community chat on https://gym.openai.com if you have any issues.".format(env_id, training_dir))
```

这是由于之前更改monitor，改成调用wrappers里面的文件时候出的错。将结果放到/tmp/cartpole-experiment-1'这个文件夹的时候出错，也就是在monitor这一步出错，通过查看文件夹里面的文件也可看到文件夹里面的文件大小都很小，内容也不对，如下：

名称	类型	大小	修改日期
 openaigym.episode_batch.0.20092.stats.json	JSON File	1 KB	2016/12/27 16:51
 openaigym.manifest.0.20092.manifest.json	JSON File	1 KB	2016/12/27 16:51



```
1 {"timestamps": [], "initial_reset_timestamp": null, "episode_rewards": [], "episode_lengths": [], "episode_types": []}
```

因此从monitor入手更改。

改完之后产生的文件如下：

opnagym.episode_batch.0.14980.stats.json - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
{ "initial_reset_timestamp": 1482825342.885998, "episode_types": ["t",  
"t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t",  
"t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t",  
"t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t",  
"t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t",  
"t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t",  
"t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t", "t",  
"t"], "timestamps": [1482825343.5884657], "episode_lengths": [200],  
"episode_rewards": [200.0]}
```

好文要顶

关注我

收藏该文

[mandalalala](#)[关注 - 0](#)[粉丝 - 1](#)

1

0

[+加关注](#)« 上一篇: [ValueError: The hardcoded shape for thenumber of rows in the filter \(5\) isn't the run time shape \(6\)](#)» 下一篇: [Regions with CNN features \(R-CNN\)](#)posted on 2016-12-27 18:49 [mandalalala](#) 阅读(776) 评论(0) [编辑](#) [收藏](#)[刷新评论](#) [刷新页面](#) [返回顶部](#)注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。[【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库](#)[【推荐】搭建微信小程序 就选腾讯云](#)[【推荐】报表开发有捷径：快速设计轻松集成，数据可视化和交互](#)



最新IT新闻:

- [Facebook认怂React专利，但问题依旧没有解决？](#)
- [多款重量级产品亮相，Google认真做起硬件来连苹果都害怕](#)
- [未向苹果征收130亿欧元税款 欧盟起诉爱尔兰政府](#)
- [苹果要求三星向韩国监管机构施压严查高通](#)
- [小米被摄影师指责盗用视频素材 未获授权即使用](#)
- » [更多新闻...](#)



最新知识库文章:

- [实用VPC虚拟私有云设计原则](#)
- [如何阅读计算机科学类的书](#)
- [Google 及其云智慧](#)
- [做到这一点，你也可以成为优秀的程序员](#)
- [写给立志做码农的大学生](#)
- » [更多知识库文章...](#)