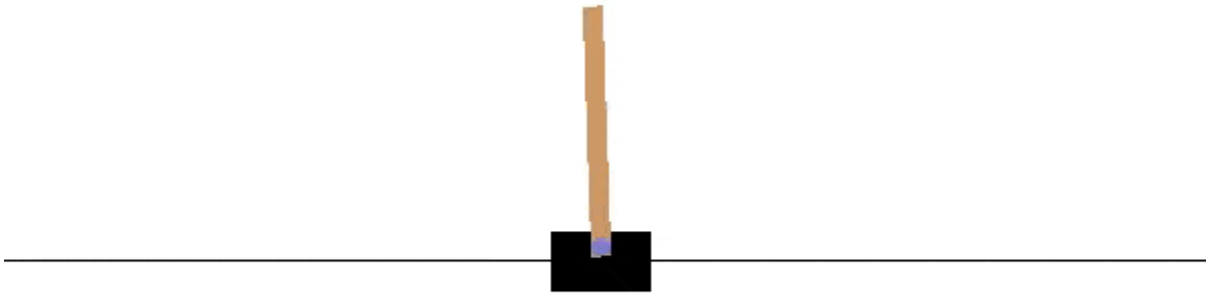
Making a custom environment in gym

Ashish Poddar Follow
Jul 21, 2018 · 3 min read

Gym is a toolkit for developing and comparing Reinforcement Learning algorithms. It is implemented in Python and R(though the former is primarily used) and can be used to make your code for training RL models short, simple and easy-to-read.



Gym has a lot of built-in environments like the cartpole environment shown above and when starting with Reinforcement Learning, solving them can be a great help. However, what we are interested in is not these built-in environments. When we want to use Reinforcement learning to solve our problems, we need to be able to create our own environment that behaves as we want it to. And while this is documented in a lot of places, I think there is no one place where I could find all the different parts together. So, I am writing this to put all the things you would need in one place.

So, let's first go through what a gym environment consists of. A gym environment will basically be a class with 4 functions. The first function is the initialization function of the class, which will take no additional parameters and initialize a class. It also sets the initial state of our RL problem. The second function is the step function, that will take an

action variable and will return the a list of four things — the next state, the reward for the current state, a boolean representing whether the current episode of our model is done and some additional info on our problem. The other functions are reset, which resets the state and other variables of the environment to the start state and render, which gives out relevant information about the behavior of our environment so far.

So, when we create a custom environment, we need these four functions in the environment. Let's now get down to actually creating and using the environment. For creating the gym, environment, we will need to create the following file structure.

```
gym-foo/
  README.md
  setup.py
  gym_foo/
    __init__.py
    envs/
      __init__.py
      foo_env.py
```

README.md will basically only be a description of what the environment is meant to do. The gym-foo/setup.py should contain the following lines.

```
from setuptools import setup

setup(name='gym_foo',
      version='0.0.1',
      install_requires=['gym']#And any other dependencies required
)
```

What you enter as the name variable of the setup is what you will use to import your environment(for eg. here, import gym_foo).

The `gym-foo/gym_foo/__init__.py` should have -

```
from gym.envs.registration import register

register(
    id='foo-v0',
```

```
    entry_point='gym_foo.envs:FooEnv',
)
```

The id variable we enter here is what we will pass into gym.make() to call our environment.

The file `gym-foo/gym_foo/envs/__init__.py` should include -

```
from gym_foo.envs.foo_env import FooEnv
```

The final file which will contain the “custom” part of your environment is `gym-foo/gym_foo/envs/foo_env.py` You will fill in this file as the following -

```
import gym
from gym import error, spaces, utils
from gym.utils import seeding

class FooEnv(gym.Env):
  metadata = {'render.modes': ['human']}

  def __init__(self):
    ...
  def step(self, action):
    ...
  def reset(self):
    ...
  def render(self, mode='human', close=False):
    ...
```

The four functions defined here will define what the gym environment will do. Once you are done writing this file, you just have to install the environment to gym and we are done.

To install the environment, just go to the gym-foo folder and run the command -

```
pip install -e .
```

This will install the gym environment. Now, we can use our gym environment with the following -

```
import gym
import gym_foo
env = gym.make('foo-v0')
```

We can now use this environment to train our RL models efficiently.

As suggested by one of the readers, I implemented an environment for the tic-tac-toe in the gym environment. The code for the same is included here.

Feel free to ask questions if you do not understand any part of the above post. I will try to clear them out as and when I can. And yeah, thanks for reading. :)

Python Reinforcement Learning Open Gym