Getting Started with Gym

Gym adalah toolkit untuk mengembangkan dan membandingkan algoritma pembelajaran penguatan. Itu tidak membuat asumsi tentang struktur agen Anda, dan kompatibel dengan perpustakaan komputasi numerik apa pun, seperti TensorFlow atau Theano.

The gym library is a collection of test problems — environments — that you can use to work out your reinforcement learning algorithms. These environments have a shared interface, allowing you to write general algorithms.

pip install gym

Kelompok 13

Nomor 1 : Dikri Nur Abdillah (persevere)

Nomor 2 : Nur Apita Lia Andini (Atlas)

Nomor 3 : Mikhael (Alan Turing)

Nomor 4 : kharir Makhfudz (visioner)

```
pip install gym
```

```
Requirement already satisfied: gym in /usr/local/lib/python3.7/dist-packages (0.17.3
Requirement already satisfied: cloudpickle<1.7.0,>=1.2.0 in /usr/local/lib/python3.7
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: numpy>=1.10.4 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: pyglet<=1.5.0,>=1.4.0 in /usr/local/lib/python3.7/dis
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-packages (fro
```

```
pip install pyglet==1.5.11
```

```
Collecting pyglet==1.5.11
  Using cached pyglet-1.5.11-py3-none-any.whl (1.1 MB)
Installing collected packages: pyglet
  Attempting uninstall: pyglet
    Found existing installation: pyglet 1.5.0
    Uninstalling pyglet-1.5.0:
      Successfully uninstalled pyglet-1.5.0
ERROR: pip's dependency resolver does not currently take into account all the packag
gym 0.17.3 requires pyglet<=1.5.0,>=1.4.0, but you have pyglet 1.5.11 which is incom
Successfully installed pyglet-1.5.11
```

```
%%bash
```

```
# install required system dependencies
apt-get install -y xvfb x11-utils

# install required python dependencies (might need to install additional gym extras depenc
pip install gym[box2d]==0.17.* pyvirtualdisplay==0.2.* PyOpenGL==3.1.* PyOpenGL-accelerate

#import
import pyvirtualdisplay
_display = pyvirtualdisplay.Display(visible=0,  # remember to use visible=0 and not False
                                    size=(1400, 900))
_ = _display.start()

#check
!echo $DISPLAY
```

```
Reading package lists...
Building dependency tree...
Reading state information...
x11-utils is already the newest version (7.7+3build1).
xvfb is already the newest version (2:1.19.6-1ubuntu4.10).
0 upgraded, 0 newly installed, 0 to remove and 39 not upgraded.
Requirement already satisfied: gym[box2d]==0.17.* in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pyvirtualdisplay==0.2.* in /usr/local/lib/python3.7/d
Requirement already satisfied: PyOpenGL==3.1.* in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: PyOpenGL-accelerate==3.1.* in /usr/local/lib/python3.
Requirement already satisfied: EasyProcess in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: numpy>=1.10.4 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: cloudpickle<1.7.0,>=1.2.0 in /usr/local/lib/python3.7
Collecting pyglet<=1.5.0,>=1.4.0
  Using cached pyglet-1.5.0-py2.py3-none-any.whl (1.0 MB)
Requirement already satisfied: box2d-py~=2.3.5 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-packages (fro
Installing collected packages: pyglet
  Attempting uninstall: pyglet
    Found existing installation: pyglet 1.5.11
    Uninstalling pyglet-1.5.11:
      Successfully uninstalled pyglet-1.5.11
Successfully installed pyglet-1.5.0
bash: line 9: import: command not found
bash: line 10: syntax error near unexpected token `('
bash: line 10: `_display = pyvirtualdisplay.Display(visible=0,  # remember to use vi
```

Environments Here's a bare minimum example of getting something running. This will run an instance of the CartPole-v0 environment for 1000 timesteps, rendering the environment at each step. You should see a window pop up rendering the classic cart-pole problem:

```
# install required system dependencies
!apt-get install -y xvfb x11-utils > /dev/null
!pip install gym[box2d] pyvirtualdisplay  PyOpenGL PyOpenGL-accelerate > /dev/null

import pyvirtualdisplay
_display = pyvirtualdisplay.Display(visible=False,size=(1400, 900))  # use False with Xvfb
```

```
_ = _display.start()
!echo $DISPLAY

import gym
import matplotlib.pyplot as plt
import numpy as np
from IPython import display

env = gym.make("CartPole-v0").env
#env = gym.make("CartPole-v0").env
#env = gym.make('LunarLander-v2')

env.reset()
fig, ax = plt.subplots(figsize=(20, 10))
ax.axis('off')
img = ax.imshow(env.render(mode='rgb_array'))
#img.set_data(env.render(mode='rgb_array'))
display.display(plt.gcf())
display.clear_output(wait=True)
```

```
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
env.close()
```

```
/usr/local/lib/python3.7/dist-packages/gym/logger.py:30: UserWarning: WARN: You are
  warnings.warn(colorize('%s: %s'%('WARN', msg % args), 'yellow'))
```

Observations

Jika kita ingin melakukan lebih baik daripada mengambil tindakan acak di setiap langkah, mungkin ada baiknya untuk benar-benar mengetahui apa yang dilakukan tindakan kita terhadap lingkungan.

Fungsi langkah lingkungan mengembalikan persis apa yang kita butuhkan. Faktanya, langkah mengembalikan empat nilai. Ini adalah:

observasi (objek): objek khusus lingkungan yang mewakili pengamatan Anda terhadap lingkungan. Misalnya, data piksel dari kamera, sudut sambungan dan kecepatan sambungan robot, atau status papan dalam permainan papan. reward (float): jumlah reward yang dicapai oleh tindakan sebelumnya. Skalanya bervariasi antar lingkungan, tetapi tujuannya selalu untuk meningkatkan total hadiah Anda. done (boolean): apakah sudah waktunya untuk mengatur ulang lingkungan lagi. Sebagian besar (tetapi tidak semua) tugas dibagi menjadi beberapa episode yang terdefinisi dengan baik, dan dilakukan dengan True menunjukkan bahwa episode telah berakhir. (Misalnya, mungkin tiangnya miring terlalu jauh, atau Anda kehilangan nyawa terakhir Anda.)

info (dict): informasi diagnostik yang berguna untuk debugging. Kadang-kadang dapat berguna untuk belajar (misalnya, mungkin berisi probabilitas mentah di balik perubahan keadaan terakhir lingkungan). Namun, evaluasi resmi agen Anda tidak diperbolehkan menggunakan ini untuk pembelajaran. Ini hanyalah implementasi dari "loop agen-lingkungan" klasik. Setiap langkah waktu, agen memilih tindakan, dan lingkungan mengembalikan pengamatan dan hadiah.

Selengkapnya tentang teks sumber iniDiperlukan teks sumber untuk mendapatkan informasi terjemahan tambahan Kirim masukan Panel samping Histori Disimpan Beri kontribusi

Proses dimulai dengan memanggil reset(), yang mengembalikan observasi awal. Jadi cara yang lebih tepat untuk menulis kode sebelumnya adalah dengan menghormati flag done:

```
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
env.close()
```

```
[ 0.00312849 -0.00639911 -0.04929967  0.01671158]
[ 0.00300051 -0.20078065 -0.04896544  0.2934414 ]
[-0.0010151  -0.00499602 -0.04309661 -0.01427365]
[-0.00111502  0.19071665 -0.04338208 -0.32023654]
[ 0.00269931  0.38642871 -0.04978681 -0.62627876]
[ 0.01042789  0.19203578 -0.06231239 -0.34968195]
[ 0.0142686   0.38798604 -0.06930603 -0.66134489]
[ 0.02202832  0.19389341 -0.08253293 -0.39126455]
[ 0.02590619  0.39008377 -0.09035822 -0.7087855 ]
[ 0.03370787  0.58633339 -0.10453393 -1.02848822]
[ 0.04543453  0.39274626 -0.12510369 -0.77036964]
[ 0.05328946  0.5893475  -0.14051108 -1.09965139]
[ 0.06507641  0.7860107  -0.16250411 -1.43291334]
[ 0.08079662  0.98272153 -0.19116238 -1.77165837]
Episode finished after 14 timesteps
[ 0.02321311  0.03723523 -0.01765049  0.0499298 ]
[ 0.02395782 -0.15762924 -0.01665189  0.33699201]
[ 0.02080523  0.03772567 -0.00991205  0.03910483]
[ 0.02155974  0.23298835 -0.00912996 -0.25668889]
[ 0.02621951  0.03799793 -0.01426373  0.03310038]
[ 0.02697947 -0.1569166  -0.01360173  0.32124904]
[ 0.02384114  0.03839638 -0.00717674  0.02430788]
[ 0.02460907  0.23362052 -0.00669059 -0.27063072]
[ 0.02928148  0.4288373  -0.0121032  -0.56541636]
[ 0.03785822  0.23388722 -0.02341153 -0.27657092]
[ 0.04253597  0.03910697 -0.02894295  0.00863704]
[ 0.04331811  0.23463179 -0.02877021 -0.2930354 ]
[ 0.04801074  0.43015187 -0.03463091 -0.59465143]
[ 0.05661378  0.23553131 -0.04652394 -0.31307507]
[ 0.06132441  0.04110194 -0.05278544 -0.0354197 ]
[ 0.06214644 -0.15322488 -0.05349384  0.24015282]
[ 0.05908195  0.04261881 -0.04869078 -0.06891219]
[ 0.05993432 -0.15177247 -0.05006903  0.20801973]
[ 0.05689887  0.04402834 -0.04590863 -0.10002757]
[ 0.05777944 -0.15040662 -0.04790918  0.17782518]
[ 0.05477131 -0.34481139 -0.04435268  0.45501783]
[ 0.04787508 -0.53927907 -0.03525232  0.73339708]
[ 0.0370895  -0.73389668 -0.02058438  1.01478007]
[ 2.24115654e-02 -5.38506337e-01 -2.88779504e-04  7.15705271e-01]
[ 0.01164144 -0.34338039  0.01402533  0.42293146]
[ 0.00477383 -0.5386982   0.02248396  0.72000267]
[-0.00600013 -0.34389443  0.03688401  0.43448059]
[-0.01287802 -0.53951861  0.04557362  0.73855874]
[-0.02366839 -0.73523928  0.0603448   1.04522893]
[-0.03837318 -0.93110816  0.08124937  1.35622807]
[-0.05699534 -1.1271501   0.10837394  1.67318193]
```

```
[-0.07953834 -1.32335078  0.14183757  1.99755597]
[-0.10600536 -1.51964239  0.18178869  2.33059719]
Episode finished after 33 timesteps
[-0.01530096 -0.01757203  0.00454691  0.0099562 ]
[-0.0156524  -0.21275889  0.00474603  0.30407025]
[-0.01990758 -0.40794816  0.01082744  0.59824618]
[-0.02806654 -0.60321992  0.02279236  0.89431986]
[-0.04013094 -0.40841435  0.04067876  0.60888778]
[-0.04829922 -0.21388398  0.05285651  0.32928987]
[-0.0525769  -0.40971698  0.05944231  0.6381617 ]
[-0.06077124 -0.60561522  0.07220554  0.94895583]
[-0.07288355 -0.41153573  0.09118466  0.67980516]
```

Spaces Dalam contoh di atas, kami telah mengambil sampel tindakan acak dari ruang tindakan lingkungan. Tapi apa sebenarnya tindakan itu? Setiap lingkungan dilengkapi dengan action_space dan observasi_space. Atribut-atribut ini bertipe Space, dan mereka menggambarkan format tindakan dan pengamatan yang valid:

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

```
Discrete(2)
Box(-3.4028234663852886e+38, 3.4028234663852886e+38, (4,), float32)
```

Ruang Diskrit memungkinkan rentang tetap angka non-negatif, jadi dalam kasus ini tindakan yang valid adalah 0 atau 1. Ruang Kotak mewakili kotak n-dimensi, jadi pengamatan yang valid akan berupa larik 4 angka. Kami juga dapat memeriksa batas Kotak:

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

```
[4.8000002e+00 3.4028235e+38 4.1887903e-01 3.4028235e+38]
[-4.8000002e+00 -3.4028235e+38 -4.1887903e-01 -3.4028235e+38]
```

Introspeksi ini dapat membantu untuk menulis kode generik yang bekerja untuk banyak lingkungan yang berbeda. Kotak dan Diskrit adalah Ruang yang paling umum. Anda dapat mengambil sampel dari Space atau memeriksa apakah ada sesuatu yang menjadi miliknya:

```
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
```

Untuk CartPole-v0 salah satu tindakan menerapkan gaya ke kiri, dan salah satunya menerapkan gaya ke kanan. (Bisakah Anda mencari tahu yang mana?)

Untungnya, semakin baik algoritme pembelajaran Anda, semakin sedikit Anda harus mencoba menafsirkan angka-angka ini sendiri.

Available Environments Gym hadir dengan rangkaian lingkungan yang beragam mulai dari yang mudah hingga yang sulit dan melibatkan berbagai jenis data. Lihat daftar lengkap lingkungan untuk mendapatkan pandangan mata burung.

Kontrol klasik dan teks mainan: selesaikan tugas skala kecil, sebagian besar dari literatur RL. Mereka di sini untuk membantu Anda memulai. Algoritma: melakukan perhitungan seperti menambahkan angka multi-digit dan membalik urutan. Orang mungkin keberatan bahwa tugas-tugas ini mudah dilakukan oleh komputer. Tantangannya adalah mempelajari algoritme ini murni dari contoh. Tugas-tugas ini memiliki properti bagus yang mudah untuk memvariasikan kesulitan dengan memvariasikan panjang urutan.

Atari: mainkan game Atari klasik. Kami telah mengintegrasikan Lingkungan Pembelajaran Arkade (yang berdampak besar pada penelitian pembelajaran penguatan) dalam bentuk yang mudah dipasang. Robot 2D dan 3D: mengontrol robot dalam simulasi. Tugas-tugas ini menggunakan mesin fisika MuJoCo, yang dirancang untuk simulasi robot yang cepat dan akurat. Termasuk beberapa lingkungan dari benchmark terbaru oleh peneliti UC Berkeley (yang kebetulan akan bergabung dengan kami musim panas ini). MuJoCo adalah perangkat lunak berpemilik, tetapi menawarkan lisensi percobaan gratis.

The registry tujuan utama gym adalah untuk menyediakan banyak koleksi lingkungan yang mengekspos antarmuka umum dan diversi untuk memungkinkan perbandingan. Untuk membuat daftar lingkungan yang tersedia di instalasi Anda, tanyakan saja gym.envs.registry

```
from gym import envs
print(envs.registry.all())
#> [EnvSpec(DoubleDunk-v0), EnvSpec(InvertedDoublePendulum-v0), EnvSpec(BeamRider-v0), Env
```

```
dict_values([EnvSpec(Copy-v0), EnvSpec(RepeatCopy-v0), EnvSpec(ReversedAddition-v0),
```

Ini akan memberi Anda daftar objek EnvSpec. Ini menentukan parameter untuk tugas tertentu, termasuk jumlah percobaan yang harus dijalankan dan jumlah langkah maksimum. Misalnya, EnvSpec(Hopper-v1) mendefinisikan lingkungan di mana tujuannya adalah untuk membuat robot simulasi 2D melompat; EnvSpec(Go9x9-v0) mendefinisikan game Go pada papan 9x9.

ID lingkungan ini diperlakukan sebagai string buram. Untuk memastikan perbandingan yang valid untuk masa depan, lingkungan tidak akan pernah diubah dengan cara yang memengaruhi

kinerja, hanya diganti dengan versi yang lebih baru. Saat ini kami menambahkan setiap lingkungan dengan v0 sehingga penggantian di masa mendatang dapat secara alami disebut v1, v2, dll.

Sangat mudah untuk menambahkan lingkungan Anda sendiri ke registri, dan dengan demikian membuatnya tersedia untuk gym.make(): cukup daftarkan() mereka pada waktu buka.

Background: Why Gym? (2016)

Pembelajaran penguatan (RL) adalah subbidang pembelajaran mesin yang berkaitan dengan pengambilan keputusan dan kontrol motorik. Ini mempelajari bagaimana agen dapat belajar bagaimana mencapai tujuan dalam lingkungan yang kompleks dan tidak pasti. Ini menarik karena dua alasan:

> .RL is very general, encompassing all problems that involve making a sequence of decisions: misalnya, mengendalikan motor robot agar dapat berlari dan melompat, membuat keputusan bisnis seperti penetapan harga dan manajemen inventaris, atau bermain video game dan board game. RL bahkan dapat diterapkan pada masalah pembelajaran terawasi dengan keluaran sekuensial atau terstruktur. .RL algorithms have started to achieve good results in many difficult environments. RL memiliki sejarah panjang, tetapi hingga kemajuan terbaru dalam pembelajaran mendalam, RL membutuhkan banyak rekayasa khusus masalah. Hasil Atari DeepMind, BRETT dari grup Pieter Abbeel, dan AlphaGo semuanya menggunakan algoritme RL dalam yang tidak membuat terlalu banyak asumsi tentang lingkungannya, dan dengan demikian dapat diterapkan di pengaturan lain.

Namun, penelitian RL juga diperlambat oleh dua faktor: Kebutuhan akan benchmark yang lebih baik. Dalam pembelajaran yang diawasi, kemajuan telah didorong oleh kumpulan data berlabel besar seperti ImageNet. Di RL, padanan terdekat adalah kumpulan lingkungan yang besar dan beragam. Namun, koleksi lingkungan RL open-source yang ada tidak memiliki variasi yang cukup, dan seringkali sulit untuk diatur dan digunakan. Kurangnya standarisasi lingkungan yang digunakan dalam publikasi. Perbedaan halus dalam definisi masalah, seperti fungsi hadiah atau serangkaian tindakan, dapat secara drastis mengubah kesulitan tugas. Masalah ini membuat sulit untuk mereproduksi penelitian yang diterbitkan dan membandingkan hasil dari makalah yang berbeda.

Gym adalah upaya untuk memperbaiki kedua masalah tersebut.

✓ 4 d selesai pada 13.50