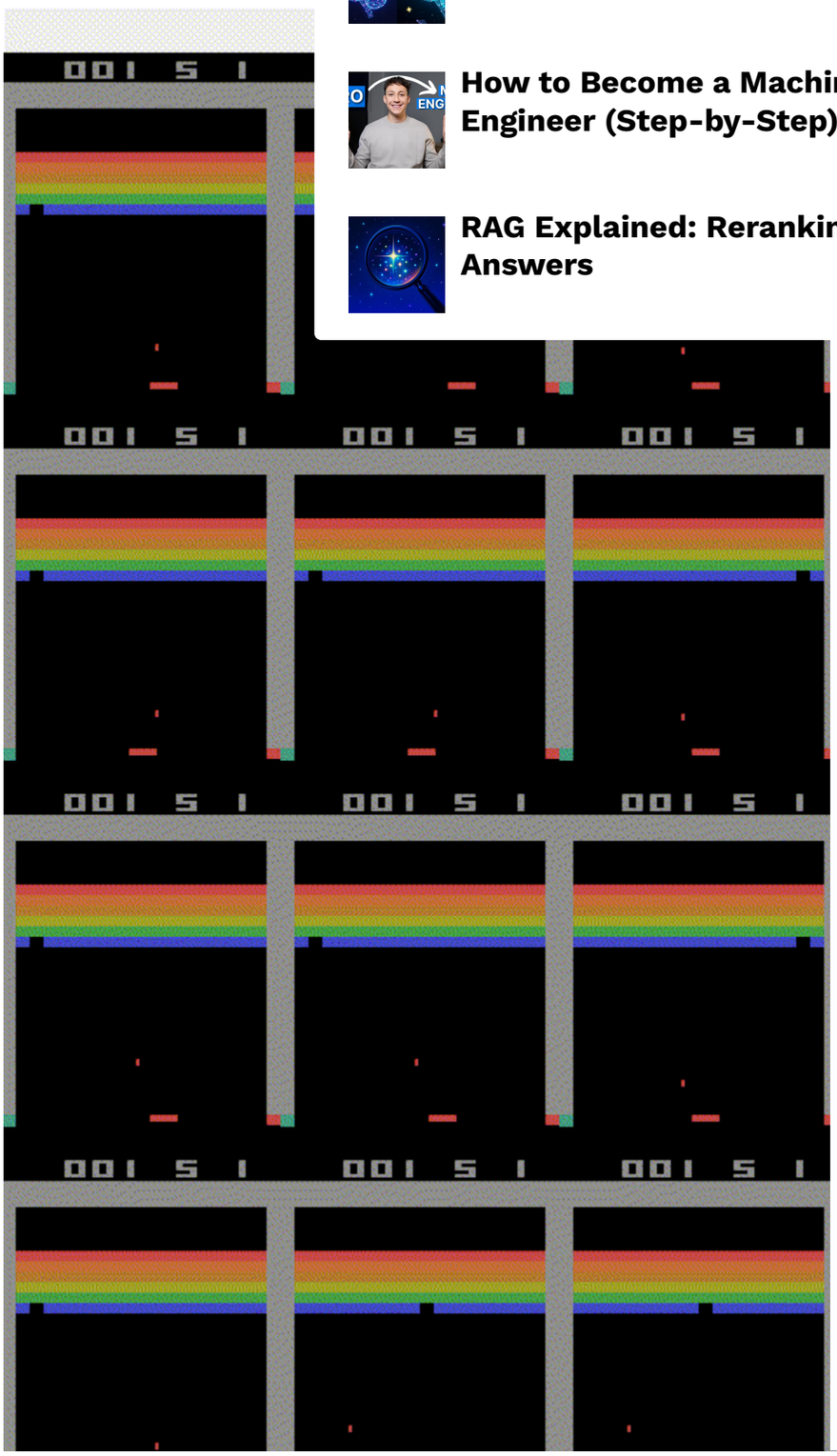
ARTIFICIAL INTELLIGENCE

Training R… baselines3

Motivation

Vincent Mueller

Feb 3, 2022 7 min read



(GIF by author)

Since about 2 years, reinforcement learning has become a hobby for me. I especially enjoy training agents on games. A huge problem for me during these years, was the lack of a reliable reinforcement learning library for python and I had to either program the state of the art algorithms by mysel
source on github.
stable-baselines3

What you can

I will walk you thro
baselines3 and the
can train an agent
runs of the trainec
about saving and l
training on multipl
will show you how
agent on the more complex LunarLander-v2 envi
A2C agent on the atari breakout environment.

Installation

The **stable-baselines3** library provides the most
reinforcement learning algorithms. It can be inst
python package manager "pip".

```
pip install stable-baselines3
```

I will demonstrate these algorithms using the op
environment. Install it to follow along.

```
pip install gym
```

Testing algorithms with cartpole en

Training a PPO agent

The stable-baslines library contains many differe

learning algorithms. In the following Code, I will show, how you can train an agent that can beat the openai cartpole environment using the proximal policy optimization algorithm.

```
from stable_baselines3 import PPO
import gym

# Parallel environments
env = gym.make("CartPole-v1")

model = PPO(policy =
model.learn(total_tir

obs = env.reset()
for i in range(1000):
    action, _state =
    obs, reward, done
    env.render()
    if done:
      obs = env.reset
```

Cartpole_PPO1.py hosted with

You can easily exc
reinforcement lear
baselines3 library.
replace PPO with t

```
from stable_baselines3 import PPO
model = PPO(policy = "MlpPolicy",env =
```

Setting the policy to "**MlpPolicy**" means, that we
state vector as input to our model. There are on
options here. Use "**CnnPolicy**" if you provide **ima**
there is "MultiInputPolicy" for handling mutiple i
cartpole environment cannot output images, I w
usecase of "CnnPolicy" later on with other gym e

Saving and loading models

To save the model, use the following line of code

```
model.save("ppo_cartpole")  # saving the model to ppo_c
```

You can load the saved model back into python

```
model = PPO.load("ppo_cartpole")  # loading the model fr
```

The following code shows the whole process of

and loading a PPO model for the cartpole environment. Make sure, that you save your model only after training it.

```
from stable_baselines3 import PPO
import gym

env = gym.make("CartPole-v1")
model = PPO(policy = "MlpPolicy",env =  env, verbose=1)
model.learn(total_timesteps=25000)

model.save("ppo_cartpole")  # saving the model to ppo_cartpole.zip
model = PPO.load("ppo_cartpole")

obs = env.reset()
for i in range(1000):
    action, _state = model.predict(obs, deterministic=True)
    obs, reward, done, info = env.step(action)
    env.render()
    if done:
      obs = env.reset()
```

StableBaselinesSavingAndLoad

Parallel Training o

You can also very
at the same time
the training process of the agent.

We can create parallel environments using the m
function of the stablebaselines3 library.

```
from stable_baselines3.common.env_util impo
```

We use it in the same way, we used the openai g
create a new environment. But we tell the functi
parallel environments we want to create.

```
env = make_vec_env("CartPole-v1", n_envs=4)
```

Since you train a single agent, you can save the
same manner as before.

But one important difference to the previous cas
of the terminal state when we test our trained a
epsiode ends in one of the environments, it is au
reset. Before testing the agent looked like this:

```
obs = env.reset()
for i in range(1000):
```

```
    action, _state = model.predict(obs, deterministic=True)
    obs, reward, done, info = env.step(action)
    env.render()
    if done:
      obs = env.reset()
```

Now, it get shorter:

```
obs = env.reset()
for i in range(10
    action, _stat
    obs, reward,
    env.render()
```

The following code
same time:

```
1  from stable_baselines
2  from stable_baselines
3
4  # Parallel environmer
5  env = make_vec_env("C
6
7  model = PPO("MlpPolicy", env, verbose=1)
8  model.learn(total_timesteps=25000)
9  model.save("ppo_cartpole")
10
11 obs = env.reset()
12 for i in range(1000):
13     action, _state = model.predict(obs, deterministic=True)
14     obs, reward, done, info = env.step(action)
15     env.render()
```

ParallelTrainingSB3.py hosted with ❤ by GitHub

Using other gym environments

In order to run most of the other gym environme
install the Box2D library for python. This is pretty
and linux, but painstaking on windows.

Installing Box2D

Box2D is an open-source physics engine for 2D p
gym environments use it for handling collisions

Linux/OSX

To the extend of my knowledge, there are no problems with directly installing Box2D on linux and mac PCs.

```
pip install box2d-py
```

Windows

On windows there
process of the Box
separately using s
install swig using a

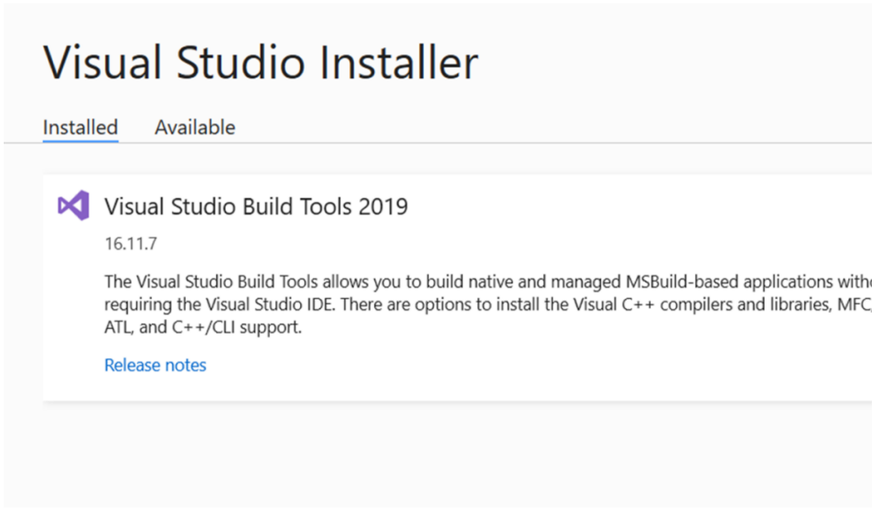
```
conda install swig
```

If you don't use an

Microsoft Visual C·
have it installed, t
download the new

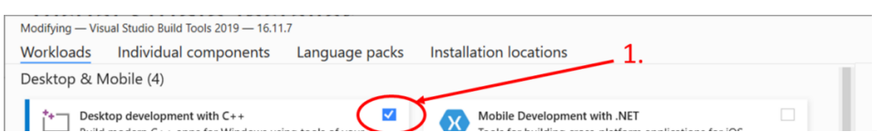
Microsoft C++ Build Tools – Visual Studio

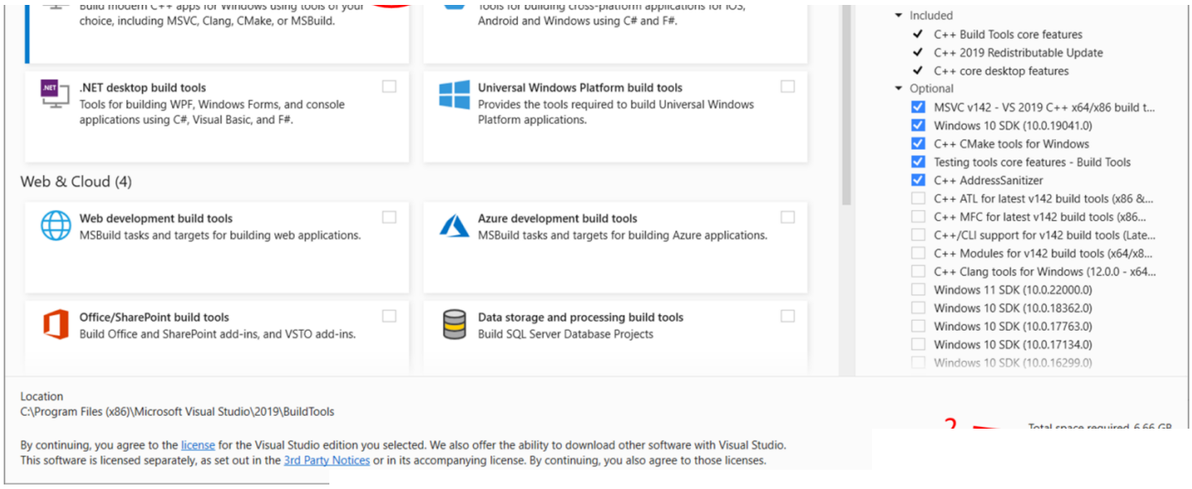
Here you can install the Buildtools.



(Image by author)

When you have installed the "buildtools", open t
installer (it is probably already open after install
"buildtools").



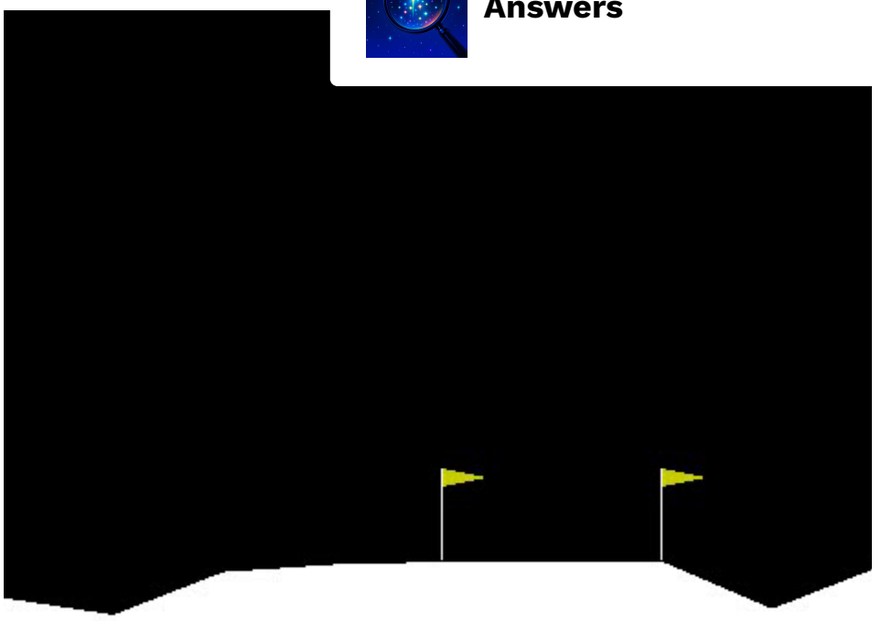


Then you can insta

```
pip install box2d-py
```

Beating LunarLanc

I will now show yo
using the stable_b
landing module be



(GIF by author)

It is a more complex task than the cartpole envi
agent is given the following information in form

- (Continuous): X distance from target site
- (Continuous): Y distance from target site
- (Continuous): X velocity

- (Continuous): Y velocity
- (Continuous): Angle of ship
- (Continuous): Angular velocity of ship
- (Binary): Left leg is grounded
- (Binary): Right leg is grounded

So we have to use the **MlpPolicy** as well.

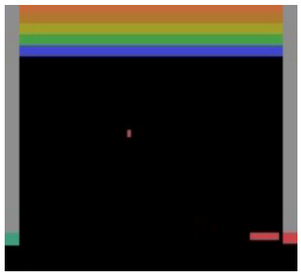
I chose the PPO al
be learning very fa
agent 2 million tra
The game is consi


```
import gym
from stable_baselines

# Parallel environmer
#env = make_vec_env('

# Create environment
env = gym.make('Lunar

# Instantiate the age
model = PPO('MlpPolicy', env, verbose=1)
# Train the agent
model.learn(total_timesteps=int(2e6))
# Save the agent
model.save("ppo_lunar2")

# Load the trained agent
#model = PPO.load("ppo_lunar", env=env)

# Enjoy trained agent
obs = env.reset()
for i in range(10000):
    action, _states = model.predict(obs, deterministic=True)
    obs, rewards, dones, info = env.step(action)
    env.render()
    if dones:
        obs = env.reset()
```

LunarLander-v2_PPO.py hosted with ❤ by GitHub

Atari breakout from pixels

Now it is time for our agent to tackle "atari breal
the pixels on the screen.



(GIF by author)

The breakout environment is not included in the standard installation of gym so you have to install a gym v
the atari collectio


```
pip install gym[atar
```

Given only a single
and direction of th
give the agent a nu
he can learn the m

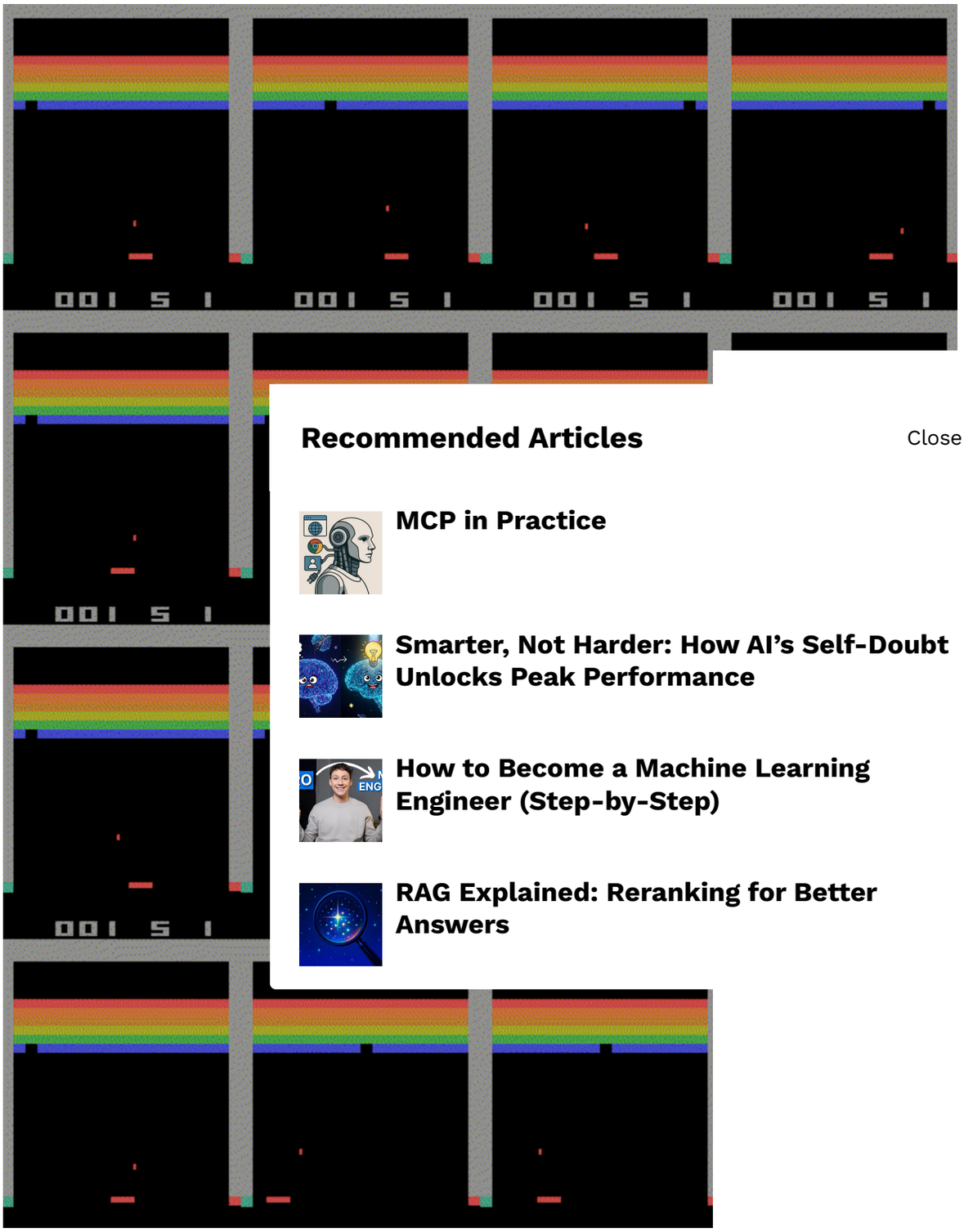
```
from stable_base
```

```
env = VecFrameStacl
```

Also be aware of t
now. I trained the agent for **5 million timesteps** v
algorithm and used **16 parallel environments**.

```
from stable_baselines3.common.env_util import make_atari_env
from stable_baselines3.common.vec_env import VecFrameStack
from stable_baselines3 import A2C

# There already exists an environment generator
# that will make and wrap atari environments correctly.
# Here we are also multi-worker training (n_envs=4 => 4 environmer
env = make_atari_env('BreakoutNoFrameskip-v4', n_envs=16)
# Frame-stacking with 4 frames
env = VecFrameStack(env, n_stack=4)

model = A2C("CnnPolicy", env, verbose=1)
model.learn(total_timesteps=int(5e6))
obs = env.reset()
#model = A2C.load("A2C_breakout") #uncomment to load saved model
model.save("A2C_breakout")
while True:
    action, _states = model.predict(obs)
    obs, rewards, dones, info = env.step(action)
    env.render()
```

breakout_A2C.py hosted with ❤ by **GitHub**



(GIF by author)

As you can see, the agent has learned the trick c
in the bricks and shooting the ball behind the wa
has difficulties with shooting the last few bricks
not able to finish the game. This is most likely ca
the agent was not trained sufficiently on situatic
bricks. By increasing the training duration, the ag
able to beat the environment.

Conclusion

I want to take a brief moment the most importa
information about the stable-baselines3 library.
environment provides a vector with information

then use the **MlpPolicy**. If it instead gives whole images, then use the **CnnPolicy**. You can use multiple environments in parallel to speed up the training. But they all train the same **one agent**. The cartpole environment can be beaten easily with a few thousand time steps of data. The LunarLander-v2 environment is more complex required 2 million timesteps to beat with PPO. Atari breakout will be solved with pixels and this makes it an even harder task. With 5 million timesteps, I was almo
the environment u

Want to conn

Linkedin https://w
m%C3%BCller-6b3
profile.php?id=100
Vincent02770108 M
Become medium r
membership fees

Join Medium wi

Related stories

Deep Q learning is no rocket science

Snake with Policy Gradients Deep Reinforcem

Backpropagation in Neural Networks

Other stories

How you can use GPT-J

Eigenvalues and eigenvectors in PCA

Support Vector Machines, Illustrated