Google Colab Free GPU Tutorial

fuat Follow
Jan 26, 2018 · 8 min read

Now you can develop **deep learning** applications with Google Colaboratory -on the **free Tesla K80 GPU**- using Keras, Tensorflow and PyTorch.

Hello! I will show you how to use **Google Colab**, *Google's free cloud service* for **AI developers**. With Colab, you can develop deep learning applications on the **GPU for free**.

Thanks to KDnuggets!

I am happy to announce that this blog post was selected as KDnuggets Silver Blog for February 2018! Read this on KDnuggets.

What is Google Colab?

Google Colab is a free cloud service and now it supports free GPU!

You can;

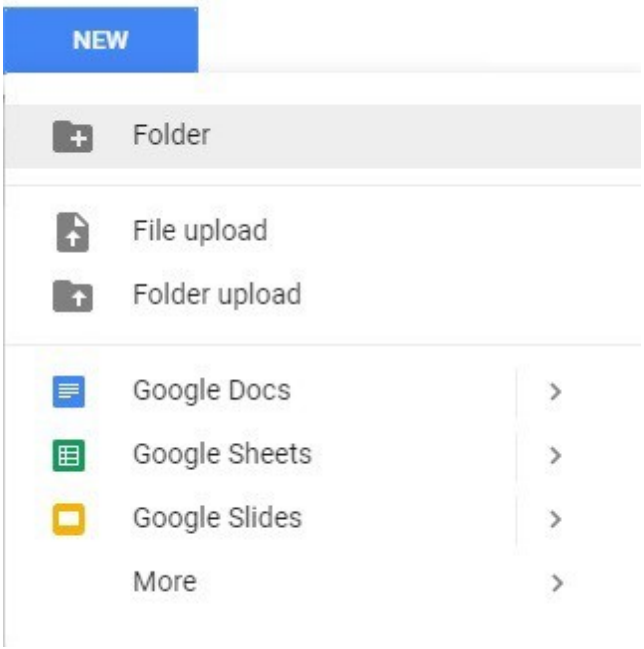
> improve your **Python** programming language coding skills.
>
> develop deep learning applications using popular libraries such as **Keras**, **TensorFlow**, **PyTorch,** and **OpenCV**.

The most important feature that distinguishes Colab from other free cloud services is; **Colab** provides GPU and is totally free.

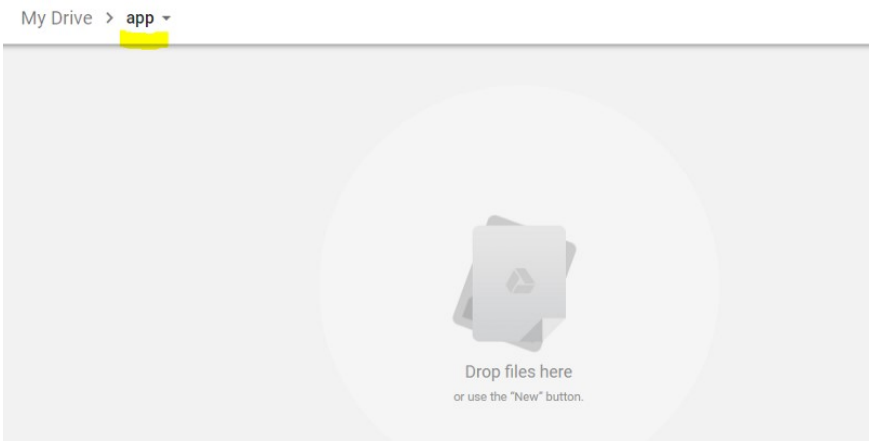
Detailed information about the service can be found on the faq page.

Getting Google Colab Ready to Use

Creating Folder on Google Drive



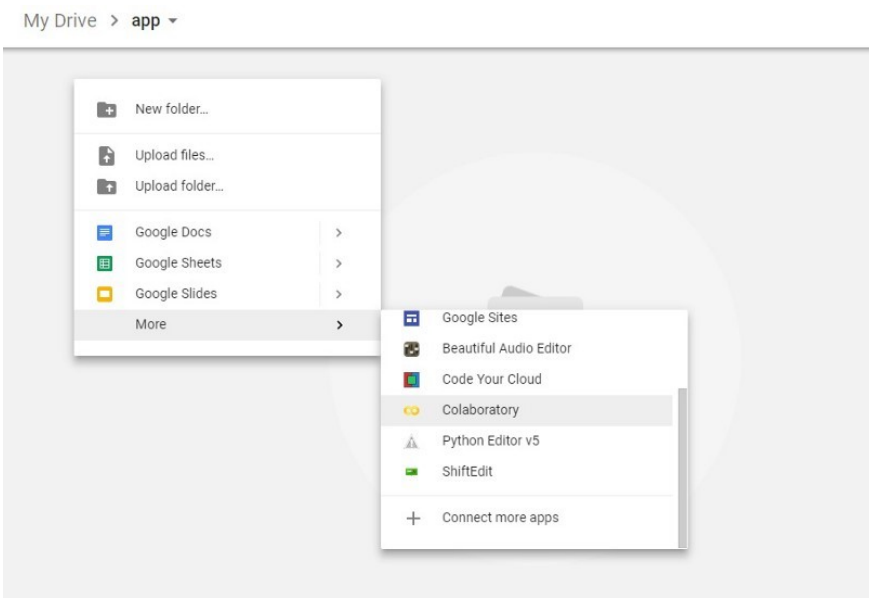
Since **Colab** is working on your own **Google Drive**, we first need to specify the folder we'll work. I created a folder named "**app**" on my **Google Drive**. Of course, you can use a different name or choose the default **Colab Notebooks** folder instead of **app folder**.



I created an empty "app" folder

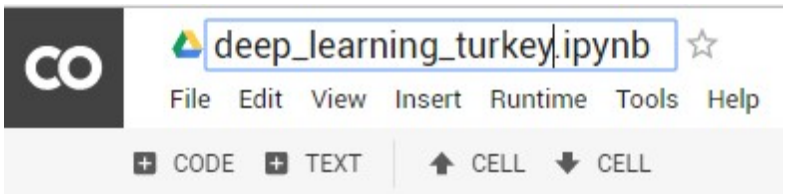
Creating New Colab Notebook

Create a new notebook via **Right click > More > Colaboratory**



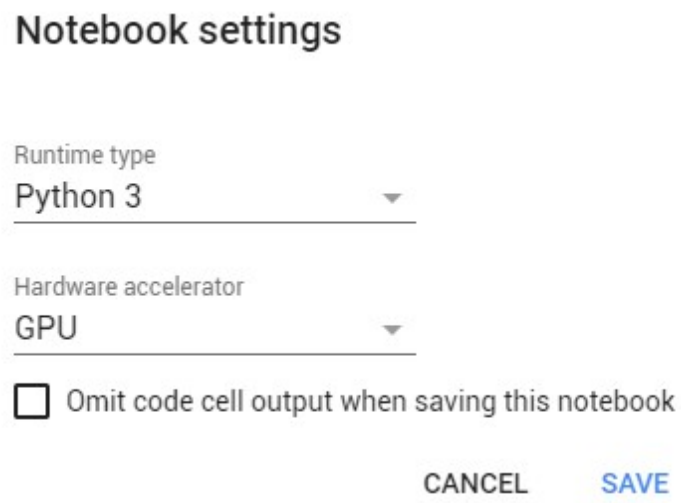
Right click > More > Colaboratory

Rename notebook by means of clicking the file name.



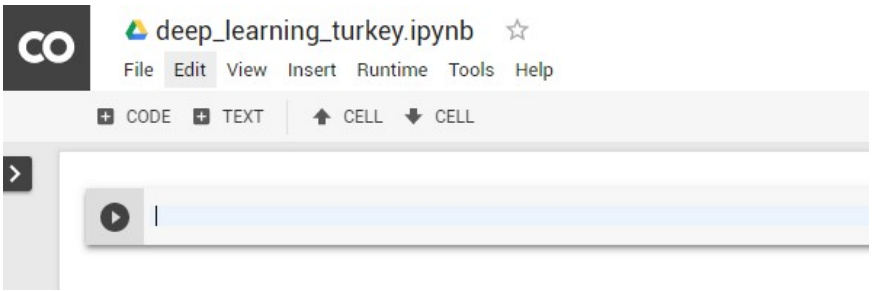
Setting Free GPU

It is so simple to alter default hardware **(CPU to GPU or vice versa)**; just follow **Edit > Notebook settings** or **Runtime>Change runtime type** and **select GPU** as **Hardware accelerator**.



Running Basic Python Codes with Google Colab

Now we can start using **Google Colab**.



I will run some **Basic Data Types** codes from Python Numpy Tutorial.

```
[1]  x = 3

[2]  print(type(x)) # Prints "<class 'int'>"

     <type 'int'>

[3]  print(x)       # Prints "3"

     3

     print(x + 1)   # Addition; prints "4"

     4
```

It works as expected :) If you do not know **Python** which is the **most popular programming language for AI**, I would recommend this simple and clean tutorial.

Running or Importing .py Files with Google Colab

Run these codes first in order to install the necessary libraries and perform authorization.

```
1  from google.colab import drive
2  drive.mount('/content/drive/')
```

When you run the code above, you should see a result like this:

```
from google.colab import drive
drive.mount('/content/drive/')

...  Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?c

     Enter your authorization code:
     |
```

Click the link, **copy** verification code and **paste** it to text box.

After completion of the authorization process, you should see this:

Now you can reach you Google Drive with:

```
1  !ls "/content/drive/My Drive/"
```

install **Keras:**

```
!pip install -q keras
```

upload mnist_cnn.py file to **app** folder which is located on your **Google Drive**.

```
from __future__ import print_function
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras import backend as K

batch_size = 128
num_classes = 10
epochs = 12
```

mnist_cnn.py file

run the code below to train a simple convnet on the MNIST dataset.

```
!python3 "/content/drive/My Drive/app/mnist_cnn.py"
```

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [==============================] - 1s 0us/step
x_train shape: (60000, 28, 28, 1)
60000 train samples
10000 test samples
Train on 60000 samples, validate on 10000 samples
Epoch 1/12
2018-01-25 23:47:16.992173: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:892] successful NUMA node read from SysFS had negative value (-1), but th
2018-01-25 23:47:16.992422: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1030] Found device 0 with properties:
name: Tesla K80 major: 3 minor: 7 memoryClockRate(GHz): 0.8235
pciBusID: 0000:00:04.0
totalMemory: 11.17GiB freeMemory: 505.38MiB
2018-01-25 23:47:16.992455: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:0) -> (device: 0, name: Tesla K
22912/60000 [==========>...................] - ETA: 9s - loss: 0.4608 - acc: 0.855160000/60000 [==============================] - 13s 214us/step - loss: 0.2
Epoch 2/12
60000/60000 [==============================] - 11s 188us/step - loss: 0.0866 - acc: 0.9747 - val_loss: 0.0377 - val_acc: 0.9876
Epoch 3/12
58112/60000 [============================>.] - ETA: 0s - loss: 0.0656 - acc: 0.980260000/60000 [==============================] - 11s 187us/step - loss: 0.0
Epoch 4/12
60000/60000 [==============================] - 11s 188us/step - loss: 0.0533 - acc: 0.9840 - val_loss: 0.0297 - val_acc: 0.9911
Epoch 5/12
60000/60000 [==============================] - 11s 188us/step - loss: 0.0469 - acc: 0.9860 - val_loss: 0.0305 - val_acc: 0.9895
Epoch 6/12
 3584/60000 [>.............................] - ETA: 9s - loss: 0.0331 - acc: 0.990860000/60000 [==============================] - 11s 186us/step - loss: 0.0
Epoch 7/12
60000/60000 [==============================] - 11s 187us/step - loss: 0.0393 - acc: 0.9877 - val_loss: 0.0256 - val_acc: 0.9918
Epoch 8/12
52352/60000 [=========================>....] - ETA: 1s - loss: 0.0355 - acc: 0.989460000/60000 [==============================] - 11s 187us/step - loss: 0.0
Epoch 9/12
60000/60000 [==============================] - 11s 186us/step - loss: 0.0317 - acc: 0.9902 - val_loss: 0.0268 - val_acc: 0.9919
Epoch 10/12
60000/60000 [==============================] - 11s 187us/step - loss: 0.0297 - acc: 0.9915 - val_loss: 0.0275 - val_acc: 0.9922
Epoch 11/12
 2304/60000 [>.............................] - ETA: 10s - loss: 0.0325 - acc: 0.989160000/60000 [==============================] - 11s 187us/step - loss: 0.
Epoch 12/12
60000/60000 [==============================] - 11s 189us/step - loss: 0.0272 - acc: 0.9917 - val_loss: 0.0254 - val_acc: 0.9923
Test loss: 0.02544446041899282
Test accuracy: 0.9923
```

As you can see from the results, each epoch lasts only **11 seconds**.

Download Titanic Dataset (.csv File) and Display First 5 Rows

If you want to download **.csv file** from **url** to “**app**” **folder**, simply run:

```
!wget https://raw.githubusercontent.com/vincentarelbundock
/Rdatasets/master/csv/datasets/Titanic.csv -P "/content
/drive/My Drive/app"
```

You may upload your .csv files **directly** to “app” folder instead of **wget method.**

Read **.csv file** in “**app**” folder and display **first 5 rows**:

```
import pandas as pd
titanic = pd.read_csv("/content/drive/My Drive/app
/Titanic.csv")
titanic.head(5)
```

```
import pandas as pd
titanic = pd.read_csv("drive/app/Titanic.csv")
titanic.head(5)
```

	Unnamed: 0	Name	PClass	Age	Sex	Survived	SexCode
0	1	Allen, Miss Elisabeth Walton	1st	29.00	female	1	1
1	2	Allison, Miss Helen Loraine	1st	2.00	female	0	1
2	3	Allison, Mr Hudson Joshua Creighton	1st	30.00	male	0	0
3	4	Allison, Mrs Hudson JC (Bessie Waldo Daniels)	1st	25.00	female	0	1
4	5	Allison, Master Hudson Trevor	1st	0.92	male	1	0

Cloning Github Repo to Google Colab

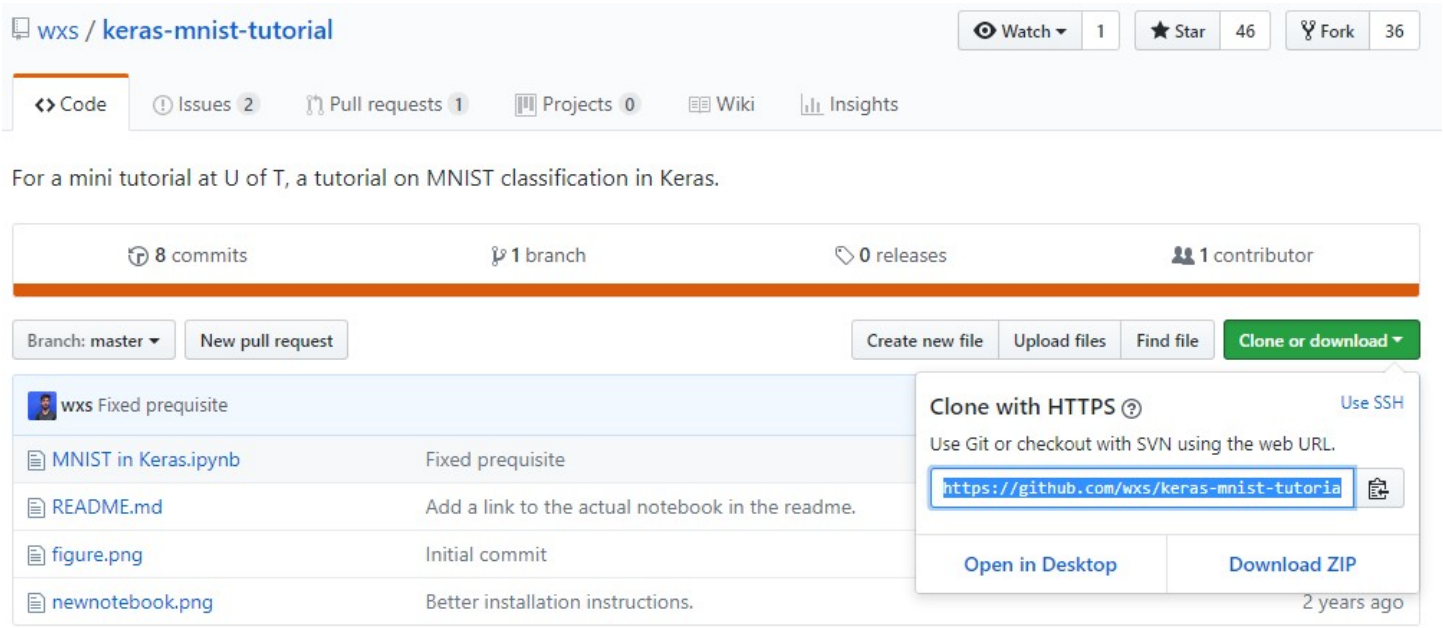
It is easy to clone a Github repo with Git.

Step 1: Find the Github Repo and Get "Git" Link

Find any Github repo to use.

For instance: https://github.com/wxs/keras-mnist-tutorial

Clone or download > Copy the link!



2. Git Clone

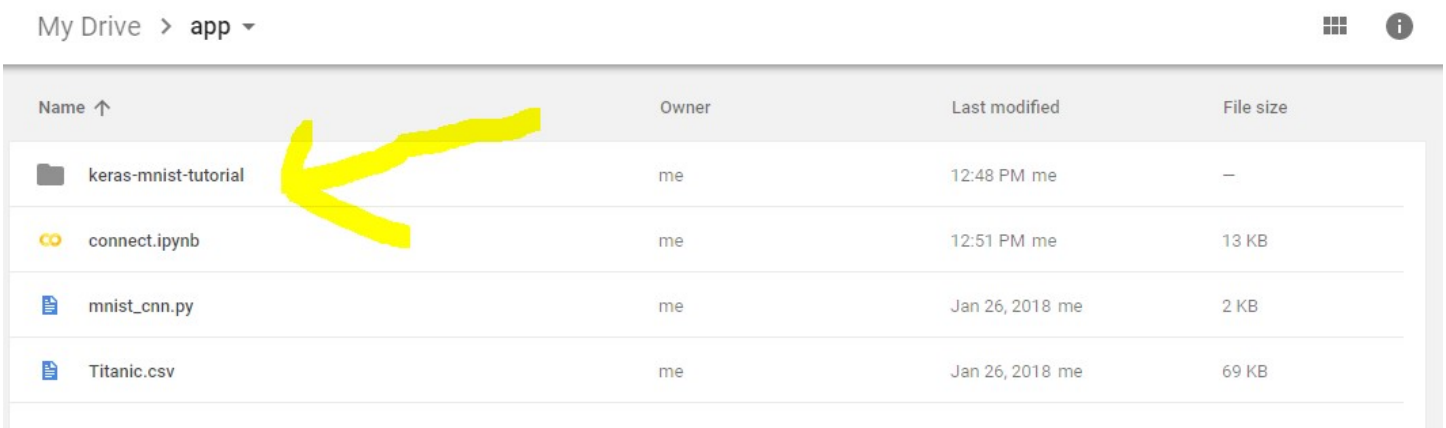
Simply run:

```
!git clone https://github.com/wxs/keras-mnist-tutorial.git
```

```
!git clone https://github.com/wxs/keras-mnist-tutorial.git

Cloning into 'keras-mnist-tutorial'...
remote: Counting objects: 26, done.
remote: Total 26 (delta 0), reused 0 (delta 0), pack-reused 26
Unpacking objects: 100% (26/26), done.
Checking out files: 100% (4/4), done.
```

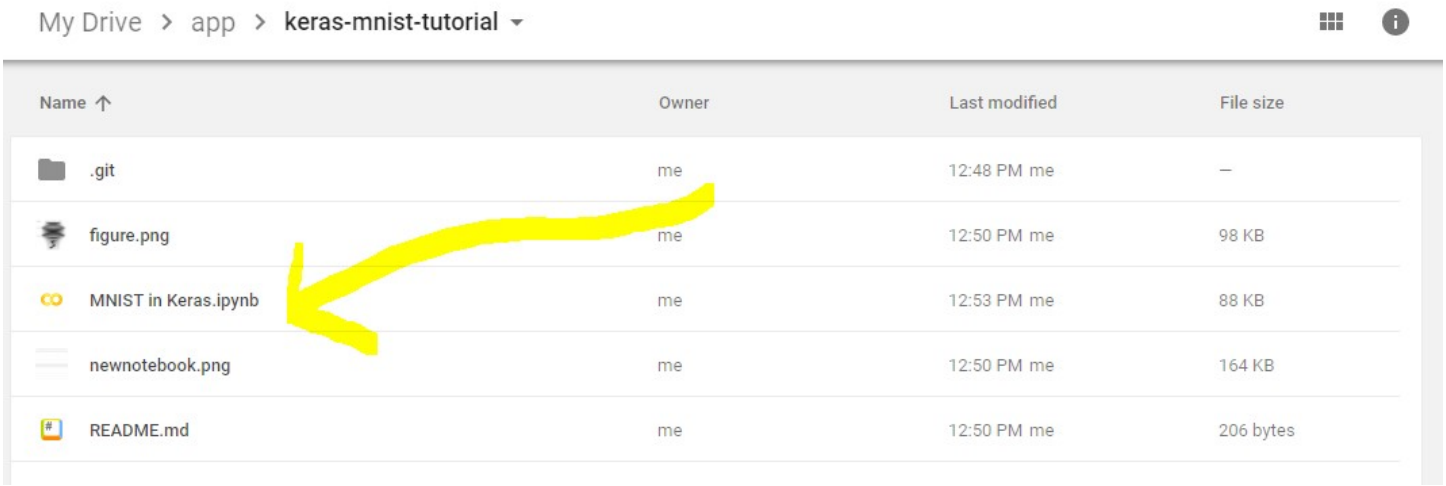
3. Open the Folder in Google Drive

Folder has the same with the Github repo of course :)



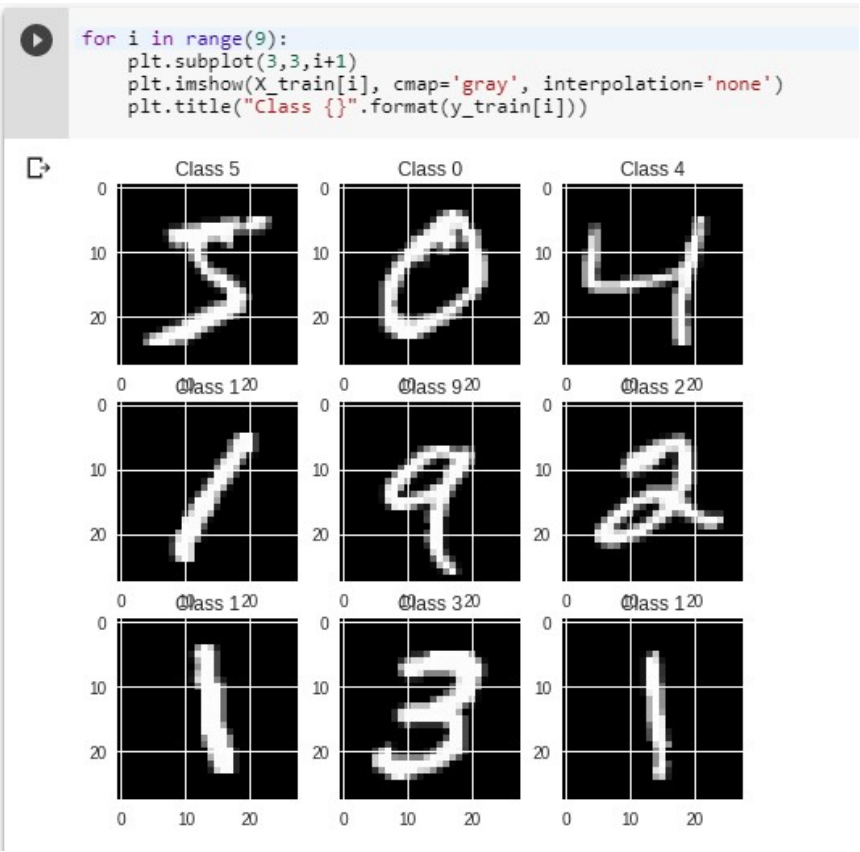
4. Open The Notebook

Right Click > Open With > Colaboratory



5. Run

Now you are able to run Github repo in Google Colab.



Some Useful Tips

1. How to Install Libraries?

Keras

```
!pip install -q keras
import keras
```

PyTorch

```
from os import path
from wheel.pep425tags import get_abbr_impl, get_impl_ver,
get_abi_tag
platform = '{}{}-{}'.format(get_abbr_impl(), get_impl_ver(),
get_abi_tag())

accelerator = 'cu80' if path.exists('/opt/bin/nvidia-smi')
else 'cpu'

!pip install -q http://download.pytorch.org
/whl/{accelerator}/torch-0.3.0.post4-{platform}-
linux_x86_64.whl torchvision
import torch
```

or try this:

```
!pip3 install torch torchvision
```

MxNet

```
!apt install libnvrtc8.0
!pip install mxnet-cu80
import mxnet as mx
```

OpenCV

```
!apt-get -qq install -y libsm6 libxext6 && pip install -q -U
opencv-python
import cv2
```

XGBoost

```
!pip install -q xgboost==0.4a30
import xgboost
```

GraphViz

```
!apt-get -qq install -y graphviz && pip install -q pydot
import pydot
```

7zip Reader

```
!apt-get -qq install -y libarchive-dev && pip install -q -U
libarchive
import libarchive
```

Other Libraries

`!pip install` or `!apt-get install` to install other libraries.

2. Is GPU Working?

To see if you are currently using the GPU in Colab, you can run the following code in order to cross-check:

```
import tensorflow as tf
tf.test.gpu_device_name()
```

```
import tensorflow as tf
tf.test.gpu_device_name()
```

```
''
```

only CPU

```
'/device:GPU:0'
```

GPU

3. Which GPU Am I Using?

```
from tensorflow.python.client import device_lib
device_lib.list_local_devices()
```

Currently, **Colab only provides Tesla K80**.

```
from tensorflow.python.client import device_lib
device_lib.list_local_devices()
```

```
[name: "/device:CPU:0"
 device_type: "CPU"
 memory_limit: 268435456
 locality {
 }
 incarnation: 11501092526971672 4, name: "/device:GPU:0"
 device_type: "GPU"
 locality {
   bus_id: 1
 }
 incarnation: 2835578110136398533
 physical_device_desc: "device: 0, name: Tesla K80, pci bus id: 0000:00:04.0, compute capability: 3.7"]
```

4. What about RAM?

```
!cat /proc/meminfo
```

```
!cat /proc/meminfo
```

```
MemTotal:       13342000 kB
MemFree:         4059676 kB
MemAvailable:   11139900 kB
Buffers:          637980 kB
Cached:          6078588 kB
SwapCached:            0 kB
Active:          4728852 kB
Inactive:        3296644 kB
Active(anon):    1468368 kB
Inactive(anon):   121888 kB
```

5. What about CPU?

```
!cat /proc/cpuinfo
```

```
!cat /proc/cpuinfo

processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 79
model name      : Intel(R) Xeon(R) CPU @ 2.20GHz
stepping        : 0
microcode       : 0x1
cpu MHz         : 2199.998
cache size      : 56320 KB
physical id     : 0
siblings        : 2
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm const
bugs            :
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:
```

- ## 6. Changing Working Directory

Normally when you run this code:

```
!ls
```

You probably see **datalab and drive** folders.

Therefore you must add **drive/app** before defining each filename.

To get rid of this problem, you can simply change the working directory. (In this tutorial I changed to **app folder**) with this simple code:

```
import os
os.chdir("drive/app")
```

After running code above, if you run again

```
!ls
```

You would see **app folder content** and don't need to add **drive/app** all the time anymore.

7. "No backend with GPU available" Error Solution

If you encounter this error:

```
Failed to assign a backend
No backend with GPU available. Would you like to use a
runtime with no accelerator?
```

Try again a bit later. A lot of people are kicking the tires on GPUs right now, and this message arises when all GPUs are in use.

Reference

8. How to Clear Outputs of All Cells

Follow **Tools>>Command Palette>>Clear All Outputs**

9. "apt-key output should not be parsed (stdout is not a terminal)" Warning

If you encounter this warning:

```
Warning: apt-key output should not be parsed (stdout is not
a terminal)
```

That means authentication has already done. You only need to mount Google Drive:

```
!mkdir -p drive
!google-drive-ocamlfuse drive
```

10. How to Use Tensorboard with Google Colab?

I recommend this repo:

https://github.com/mixuala/colab_utils

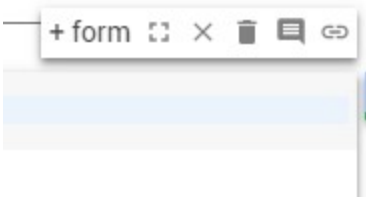
11. How to Restart Google Colab?

In order to restart (or reset) your virtual machine, simply run:

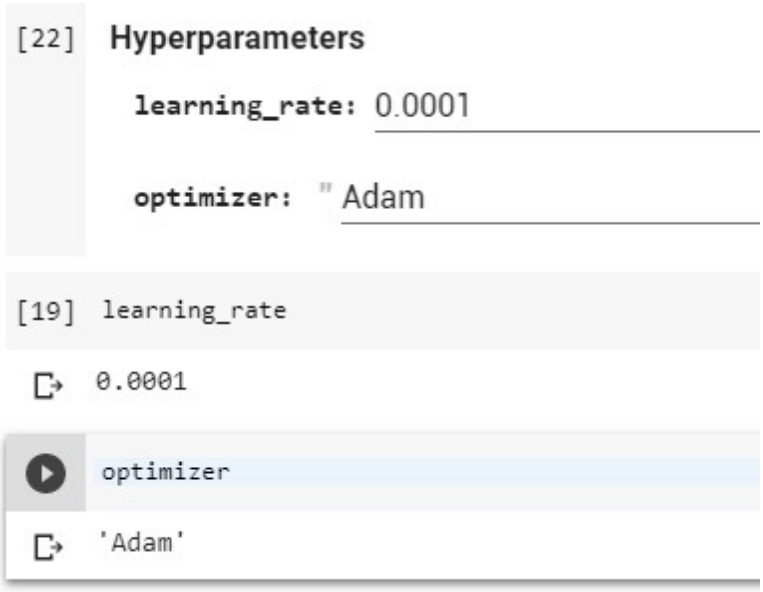
```
!kill -9 -1
```

12. How to Add Form to Google Colab?

In order not to change hyperparameters every time in your code, you can simply add form to Google Colab.

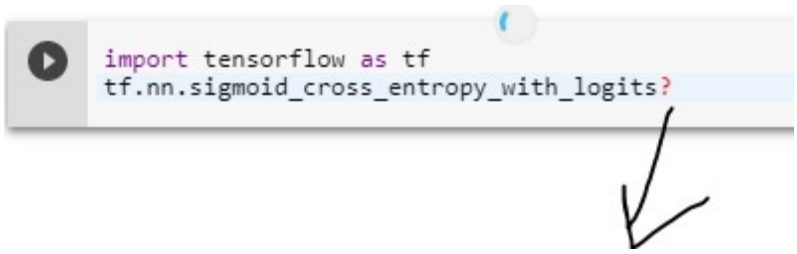


For instance, I added form which contain `learning_rate` variable and `optimizer` string.

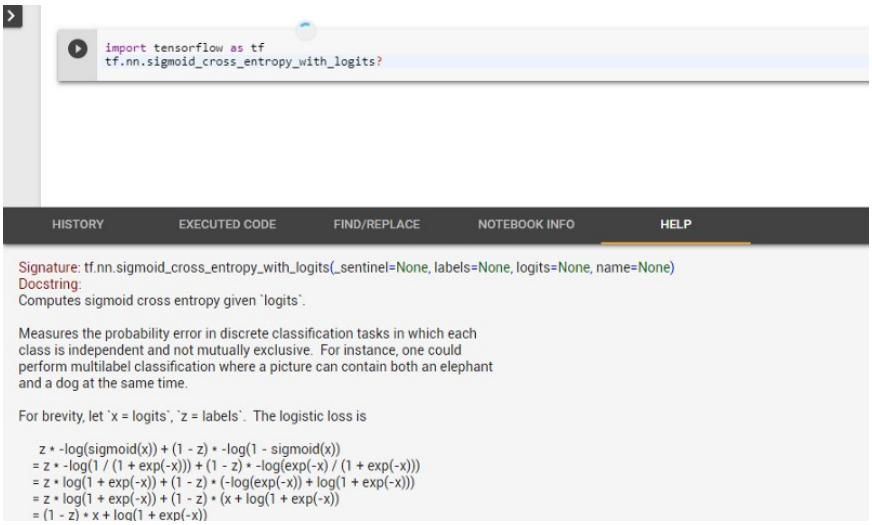


13. How to See Function Arguments?

To see function arguments in TensorFlow, Keras etc, simply **add question mark (?)** after function name:



Now you can see original documentation without clicking TensorFlow website.



14. How to Send Large Files From Colab To Google Drive?

```
# Which file to send?
file_name = "REPO.tar"

from googleapiclient.http import MediaFileUpload
from googleapiclient.discovery import build

auth.authenticate_user()
drive_service = build('drive', 'v3')

def save_file_to_drive(name, path):
  file_metadata = {'name': name, 'mimeType': 'application/o
  media = MediaFileUpload(path, mimetype='application/octet
```

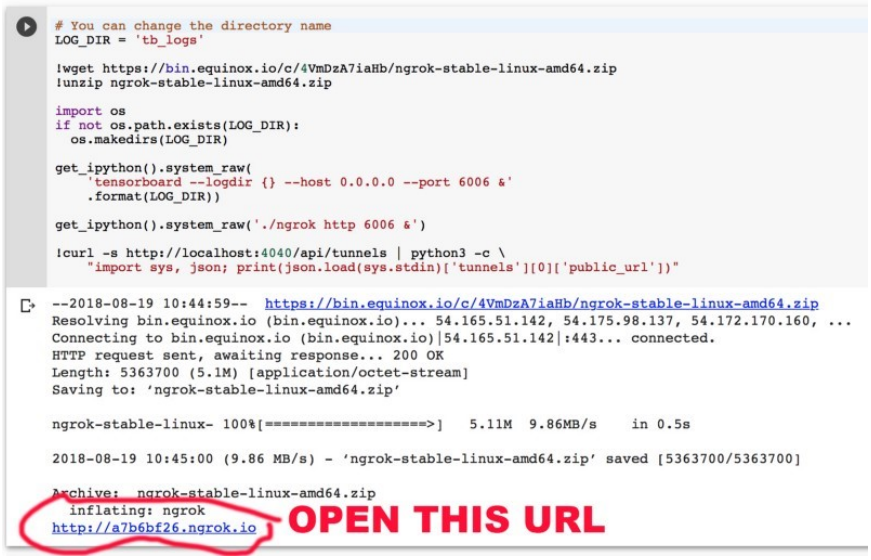
15. How to Run Tensorboard in Google Colab?

If you want to runt Tensorboard in Google Colab, run the code below.

```
# You can change the directory name
LOG_DIR = 'tb_logs'

!wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-lin
!unzip ngrok-stable-linux-amd64.zip

import os
if not os.path.exists(LOG_DIR):
  os.makedirs(LOG_DIR)

get_ipython().system_raw(
    'tensorboard --logdir {} --host 0.0.0.0 --port 6006 &'
```

You can track your **Tensorboard** logs with created ***ngrok.io*** URL. You will find the URL at the end of output.

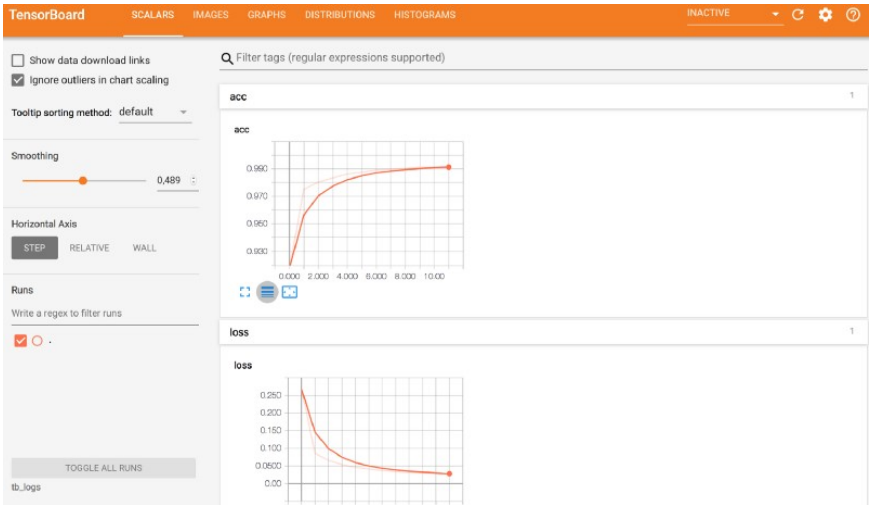
Note that your **Tensorboard** logs will be save to **tb_logs** dir. Of course, you can change the directory name.



After that, we can see the Tensorboard in action! After running the code below, you can track you Tensorboard logs via ngrok URL.

```
from __future__ import print_function
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras import backend as K
from keras.callbacks import TensorBoard

batch_size = 128
num_classes = 10
epochs = 12

# input image dimensions
img_rows, img_cols = 28, 28

# the data, shuffled and split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, i
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, i
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_
    input_shape = (img_rows, img_cols, 1)

x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)

model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
```

Tensorboard :)



Conclusion

I think **Colab** will bring a new breath to Deep Learning and AI studies all over the world.

If you found this article helpful, it would mean a lot if you gave it some applause👏 and shared to help others find it! And feel free to leave a comment below.

You can find me on Twitter .

Last Note

This blog post will be **constantly updated**.

Changelog

26-01–2018

> “insert app folder to path” *removed*
>
> “downloading, reading and displaying .csv file” *added*
>
> “Some Useful Tips” *added*