

Application of AI/ML techniques in Atmospheric Science

Summer School, Lahti, 2025

August 2025

Artificial Intelligence Tutorial 0. Introduction of Deep Learning for Applications

Module Leader: Zhi-Song Liu

Module Contributor: Zhi-Song Liu

Other Contributors: Ella Litjens, Wenqing Peng

Objectives

The objectives of this laboratory include

- (i) to familiarize the basic commands Python
- (ii) to familiarize the use of Google cloud server and deep learning platforms.
- (iii) to learn how to use own data for applications

Time: Approx. 1 hr

Equipment

PC with Internet connection

Google account (free)

USB flash driver storing the necessary source codes (provided and described in the following table)

Preparation

1. Register a google account if you do not have
2. Create a folder on the computer as the working directory, for example, ../AI_lab.
3. Copy all the materials from the flash driver to your working directory

In your working directory, you have the following files:

Folder	Files	Description
imagenet	images/polar_bear.jpeg	Testing images
	images/imagenet_classes.txt	Class number of ImageNet
	images/imagenet_synsets.txt	Synsets for different classes
	imagenet_demo.py	Executive code for testing
MNIST	image/test{1,2,3}.jpeg	Testing images
	image/mnist_train.cdv	MNIST training data
	image/mnist_test.cdv	MNIST testing data
	model/epoch_100.py	Pre-trained model for testing
	mnist_demo.py	Executive code for testing

Summary of this tutorial

This tutorial provides two demonstrations of image classification via deep learning algorithms: ImageNet and MNIST. As shown in Figure 1(a), ImageNet is a general image classification that can recognize 1000 different objects from the given RGB images. The 1000-class of objects are a subset of ImageNet, which is defined and organized by Li Fei-Fei group [1]. The deep learning algorithm we use is AlexNet [2]. Similarly, as shown in Figure 1(b), MNIST is a domain-specific classification. The task is to recognize the handwritten digits from different writing styles. It needs to recognize 0~9 from a black-and-white image. The definition of MNIST can be found at [3]. The deep learning algorithm we use is LeNet [4].

The objective of this tutorial is to allow users to learn how to use the deep learning platform to test on your own data for classification. Depending on what environment users have, we provide guidance using the online Google Cloud.

Prerequisite

For deep learning, the basic developing language we use is Python. Python is a great object-oriented, interpreted, and interactive programming language. Most of non-commercial or commercial deep learning frameworks are built based on Python, like Caffe, Pytorch, TensorFlow, CNTK and so on.

We choose Pytorch for this tutorial because it is an open source machine learning library, it can be efficiently used for development and applications. For users without proper computers, there is the good news that Google provides a free online platform that each user can access one GPU through Google cloud for deep learning programming.

In order to use Pytorch, let us first understand a few Linux commands:

Command	Description
<i>pwd</i>	Present working directory
<i>cd xxx</i>	Change directory to folder xxx
<i>cd ..</i>	Go back to the parent directory
<i>ls</i>	List directory contents of files and folders

For Python, it is like C++ language, you need to import dependent libraries for programming. For example:

```
import numpy as np # import library numpy and name it np for simplicity
a = [1,2,3,4,5] # input data
out = np.max(a) # call max function from numpy for computation
print('the maximum value is: ', out) # print out the result
```

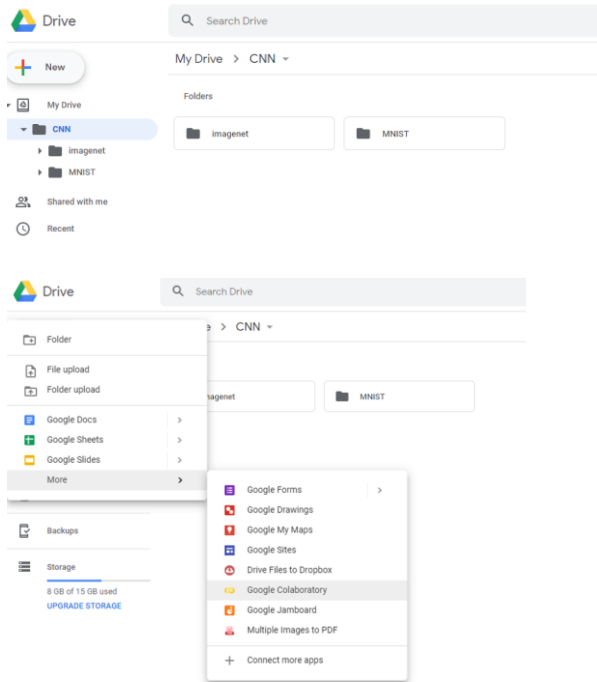
We have a vector $a=[1,2,3,4,5]$, we want to find the largest value. We can import numpy to recall its max function and use “print” to output the results. We save this code in a file named “compare_max.py” and run the code, we have

```
python compare_max.py
```

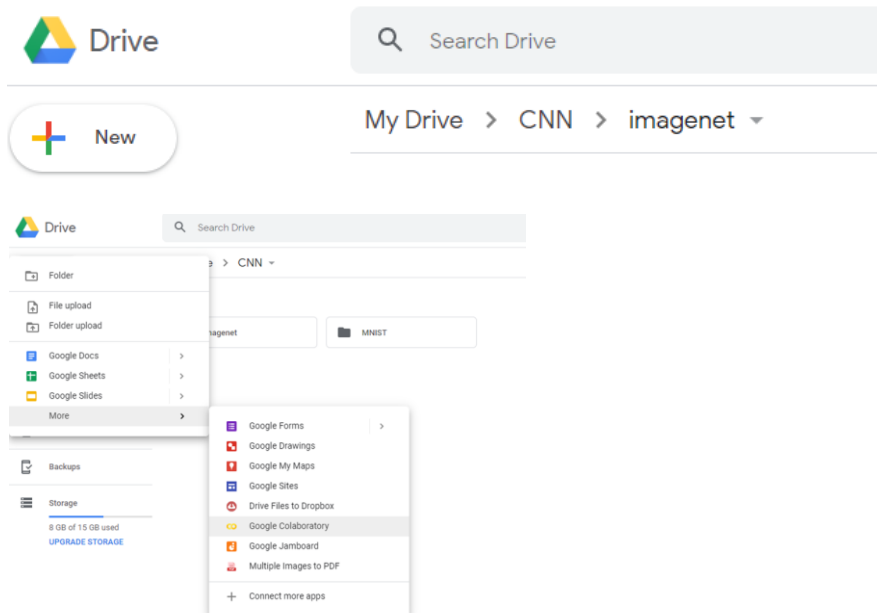
```
the maximum value is: 5
```

where `python xxx.py` means running the python file.

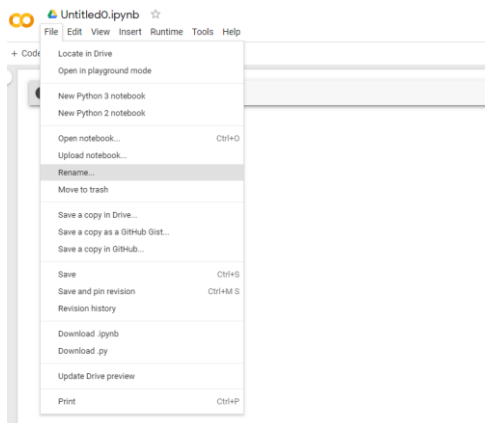
1. Log on your google drive and upload the materials to your google drive



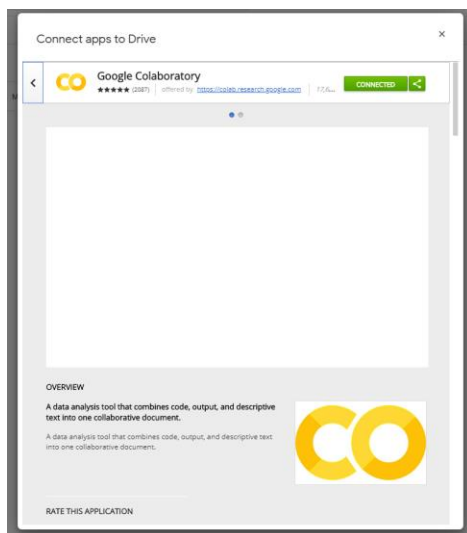
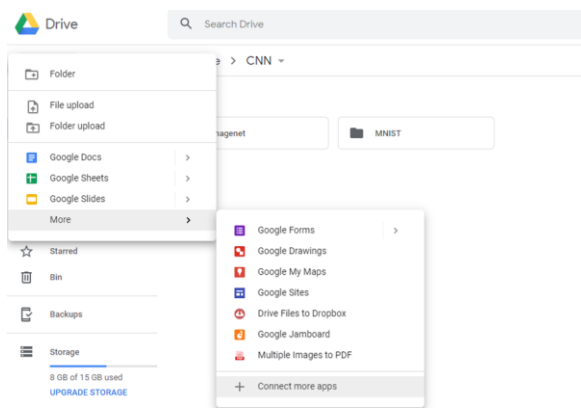
2. Click on the “New” button → “Google Colaboratory” to create a new file for executing codes



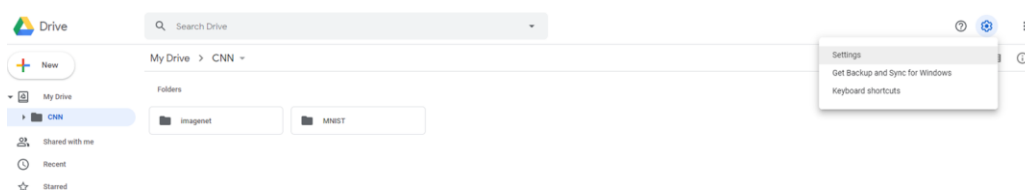
You will have a new Jupyter notebook file open as follows, and you can rename it by yourself, for example, “CNN-demo.ipynb”

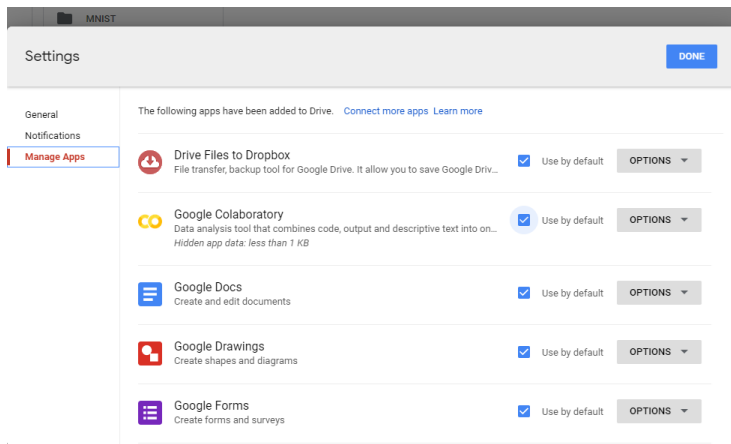


If you cannot find “Google Colaboratory”, click “More” → connect more apps → search “Google Colaboratory” and click “connect”.



Then click “settings” → “Manage Apps” and choose “Use by default”

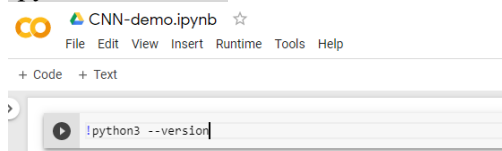





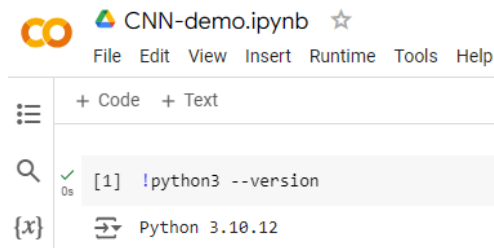
3. The created file is built on Jupyter notebook which is a web-based interactive development environment. On the top of the window, there is “+Code” for writing codes and “+Text” for writing comments or annotation. For each command, you can click “+Code” to write it and press “**shift + enter**” to execute it.

For example, you can check the python version by typing

!python3 --version

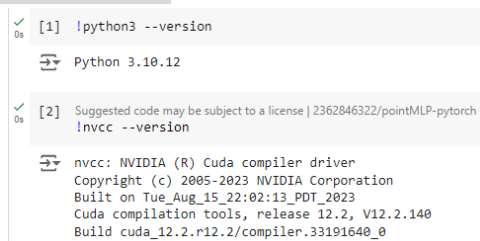


and click the  on the left, you will have the following output.



Similarly, you can also check the GPU driver version as follows,

!nvcc --version



4. In order to install Pytorch and its relevant libraries for this experiment, you need to run the following code,
!pip3 install torch torchvision

```
CNN-demo.ipynb
File Edit View Insert Runtime Tools Help

Code + Text

[5] !python3 --version
Python 3.6.9

[6] !nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2018 NVIDIA Corporation
Built on Sat_Aug_25_21:08:01_CDT_2018
Cuda compilation tools, release 10.0, V10.0.130

!pip3 install torch torchvision
Requirement already satisfied: torch in /usr/local/lib/python3.6/dist-packages (1.3.1)
Requirement already satisfied: torchvision in /usr/local/lib/python3.6/dist-packages (0.4.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from torch) (1.17.4)
Requirement already satisfied: pillow>4.1.1 in /usr/local/lib/python3.6/dist-packages (from torchvision) (4.3.0)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from torchvision) (1.12.0)
Requirement already satisfied: olefile in /usr/local/lib/python3.6/dist-packages (from pillow>4.1.1->torchvision) (0.46)
```

5. Now, we have built a virtual environment that supports Pytorch and GPU for development, the next step is to mount your Google drive to this created virtual environment. You need to run the following code,

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

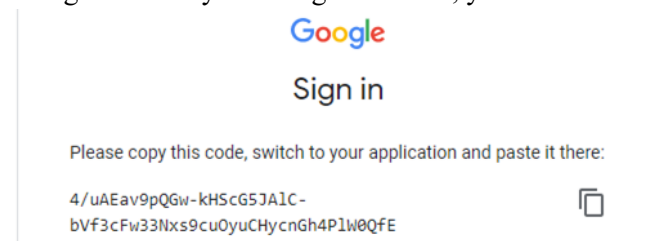
it means that we mount the google drive to “/content/drive” path, you should have the output like the following,

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

Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6b60k8ndgf4ndg3f6ee6491hc0hrc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3A...

Enter your authorization code:

Click the generated URL link and you may be asked to give permission to access your personal Google drive. By allowing the access, you will have an authorization code as,



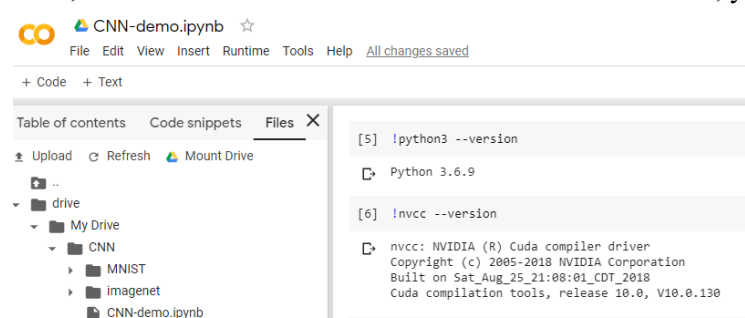
Copy the code paste it back, you will have the following result.

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

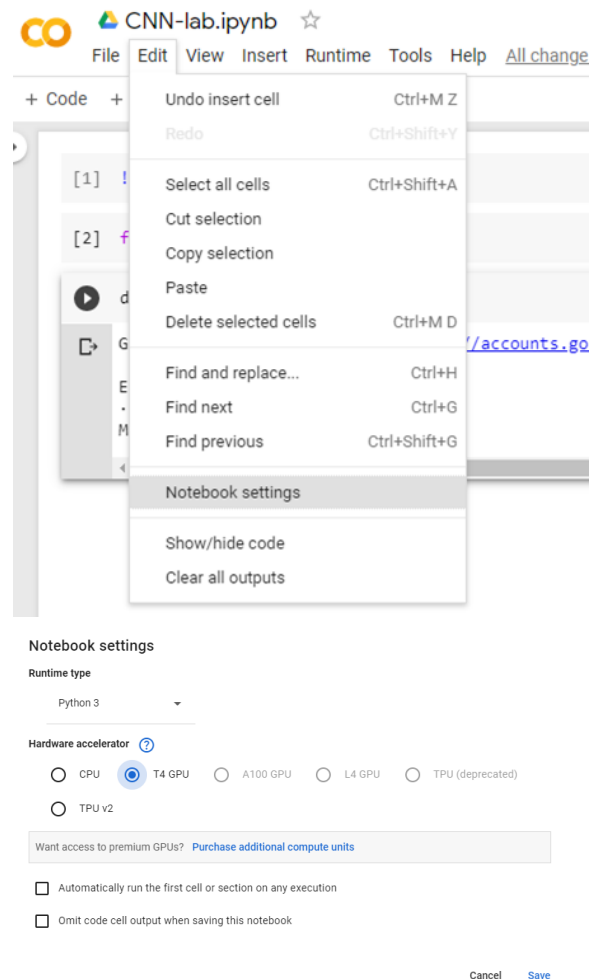
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6b60k8ndgf4ndg3f6ee6491hc0hrc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3A...

Enter your authorization code:
.....
Mounted at /content/drive/

Now, check on the left side of the window and click “Files”, you should have the following result.



Congratulations! Now you can see your Google drive's files in this working environment. Our last step is to allow the GPU to be used in this environment, click "Edit" → "Notebook settings" and set the mode as GPU as follows,



- Now, assuming that you have copied the provided files to folder "CNN", you need to go to that directory by running
Google colab used: `%cd /xxx/CNN`

where XXX is the absolute path which depends on the device you use. For example,
Google colab used: `%cd /content/drive/My Drive/CNN`

To check whether you are at the correct working directory, run
Google colab used: `!ls`

You may have to rerun the mount command if you can't navigate to your drive after switching from CPU to GPU.

You should see all the files as following



Experiment 1. ImageNet Classification

The first experiment will be ImageNet classification. We have provided a demo code for testing the ability of image classification, that can be found in the directory /CNN/imagenet . Navigate to this folder using the magic command %cd and the file path as previously learnt.

Then run the following code

Google colab used: `!python3 imagenet_demo.py images/polar_bear.jpg --network='alexnet'`

Let us take “polar_bear.jpg” as an example,



you should have the output as follows,

```
!python3 imagenet_demo.py images/polar_bear.jpg --network='alexnet'
```

```
alexnet is used for classification
The best prediction:

'images/polar_bear.jpg': 99.87646484375% is a 'ice bear, polar bear, Ursus Maritimus, Thalarctos maritimus'

Top 5 prediction:

99.87646484375% is a 'ice bear, polar bear, Ursus Maritimus, Thalarctos maritimus'
0.10136964172124863% is a 'white wolf, Arctic wolf, Canis lupus tundrarum'
0.019349906593561172% is a 'Arctic fox, white fox, Alopex lagopus'
0.002569720847532153% is a 'Samoyed, Samoyede'
0.00015759306552354246% is a 'kuvasz'
```

This demo can classify the input image by using specific convolutional neural networks. The image “polar_bear.jpg” is under the folder “image”. The network can be ‘alexnet’, ‘vgg19’ or ‘resnet152’. The result gives the best prediction and also top 5 predictions. In this example, we can see that using AlexNet, we can achieve over 99% confidence that the input image is a polar bear.

Exercise 1:

1. Try to upload your own image and test it using alexnet, show your results to tutors.
2. Try to use different convolution neural networks, and compare their performances.

Experiment 2. MNIST recognition

The second experiment is recognizing handwritten digits using LeNet network. We provide a demo code for testing the ability of image classification, you need to go to the directory /CNN/mnist as,

Google colab used: `!cd /mnist`

Then run the following code

Google colab used: `!python3 mnist_demo.py`

Let us take three images as examples,



then you have the following results:

```
!python3 mnist_demo.py

pretrained model is loaded
load image/test1.jpg
prediction: 1
load image/test2.jpg
prediction: 8
load image/test3.jpg
prediction: 4
```

Check the corresponding images and see if the predictions are correct.

Exercise 2:

1. Use your cell phone take a photo of your own written digit, upload it to the google drive and see if the model make correct prediction.
2. See if you can find a RGB color image with digit number and see the prediction result.

References:

[1] <http://image-net.org/index>

[2] A. Krizhevsky, I. Sutskever, and G. Hinton. “Imagenet classification with deep convolutional neural networks,” In NIPS, 2012.

[3] <http://yann.lecun.com/exdb/mnist/>

[4] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. “Backpropagation applied to handwritten zip code recognition,” Neural computation, 1989.