

Google Colab을 써보자

이제 CPU로 학습이 어려워요...

CPU? GPU? FPGA? ASIC?

- CPU – 범용적인 작업을 처리(CISC)
- GPU – 전용작업을 빠르게 처리(RISC)
- 속도: CPU < GPU < FPGA < ASIC(Google TPU)
- 처리범위: CPU > GPU > FPGA > ASIC

- 예: CPU – Intel, AMD(우리가 쓰는 CPU)
GPU – 게임, 채굴(범용), 딥러닝
FPGA – 채굴, 딥러닝(전용, 변경가능),
ASIC – 채굴(전용, 변경불가), TPU(Google자체 딥러닝 유닛)

Google이 TPU를 만든 이유

- GPU는 CPU보다 수 십배 이상 빠른 딥러닝 학습 시간 제공
- TPU는 GPU보다 수 십배 이상 빠른 딥러닝 학습 시간 제공
- 속도만 빠른 것이 아닌 전기도 절약
- Tensorflow를 구글이 지원하고 있으므로, 딥러닝에 최적화된 레퍼런스 필요

Colab?

- Google에서 무료로 제공하는 딥러닝 학습 공간
- Python + Jupyter 환경 지원
- Tensorflow, Keras 등 딥러닝 패키지 기본 지원
- 개인용PC 그 이상의 스펙 제공
(메모리 10GB 이상, 디스크 50GB ~ 300GB, GPU, TPU지원)
- 무엇보다, GPU를 제공하는데 일반 개인용이 아닌 서버용

Colab의 장점

- Chrome브라우저 + 인터넷 연결만 되면 사용이 가능하다.
- 내 컴퓨터의 자원을 사용하지 않는다.
- GPU, TPU를 가지고 있을 필요가 없다.
- 보통의 실습용 분석은 처리 가능할 정도의 사양을 제공.

Colab을 사용해봅시다 - 접속



The image shows a Google search interface. At the top, the Google logo is displayed in its multi-colored font. To the right of the logo is a search bar containing the text "google colab". Below the search bar, there are navigation links: "전체" (All), "동영상" (Videos), "뉴스" (News), "도서" (Books), "이미지" (Images), and "더보기" (More). To the right of these links are "설정" (Settings) and "도구" (Tools). Below the search bar, a dropdown menu shows search suggestions for "google colab", including "google colab", "google colab laboratory", "google colab 사용법" (Usage), "google colab 가격" (Price), "google colab 파일 업로드" (File Upload), "google colab tensorboard", "google colab tutorial", "google colab r", "google colab laboratory gpu", and "google colab tpu". Below the suggestions are two buttons: "Google 검색" (Google Search) and "I'm Feeling Lucky". At the bottom right of the suggestions box, there is a small link: "부적절한 예상 검색어 신고" (Report inappropriate suggested search terms).

google colab

google colab
google colab laboratory
google colab 사용법
google colab 가격
google colab 파일 업로드
google colab tensorboard
google colab tutorial
google colab r
google colab laboratory gpu
google colab tpu

Google 검색 I'm Feeling Lucky

부적절한 예상 검색어 신고

google colab

전체 동영상 뉴스 도서 이미지 더보기 설정 도구

검색결과 약 2,200,000개 (0.32초)

Google Colab
<https://colab.research.google.com/> ▼ 이 페이지 번역하기

These are a few of the notebooks from Google's online Machine Learning course. See the full course website for more. Intro to Pandas · Tensorflow concepts ...

[GPU](#) · [Colaboratory](#) · [Get Started with TensorFlow](#) · [Intro to pandas](#)

이 페이지를 19. 7. 7에 방문했습니다.

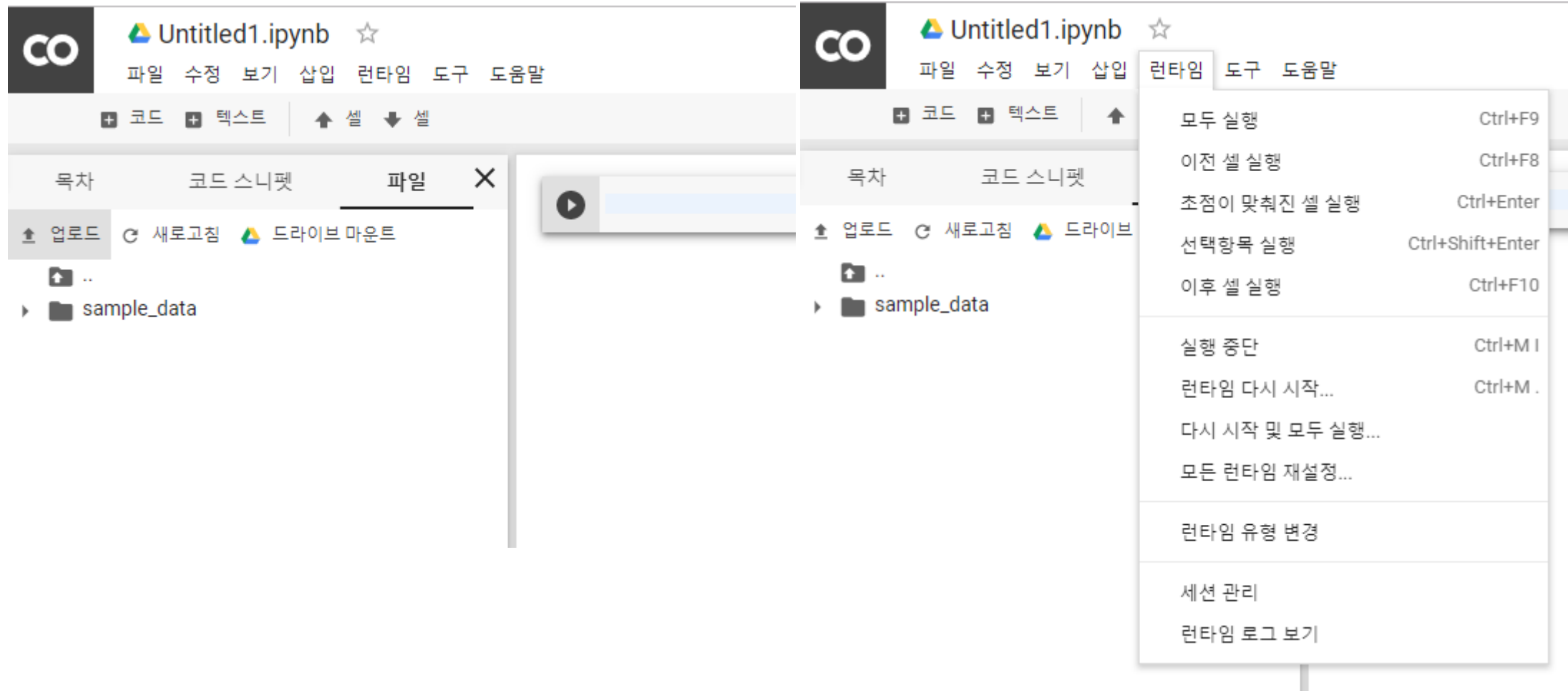
Colab을 사용해봅시다 - 첫 시작

The screenshot displays the Google Colaboratory web interface. The top navigation bar is orange and contains links for '메' (Home), '최근 사용' (Recent), 'GOOGLE 드라이브' (Google Drive), 'GITHUB', and '업로드' (Upload). Below this, a search bar labeled '노트 필터링' (Filter notes) is visible. The main area shows a list of notebooks:

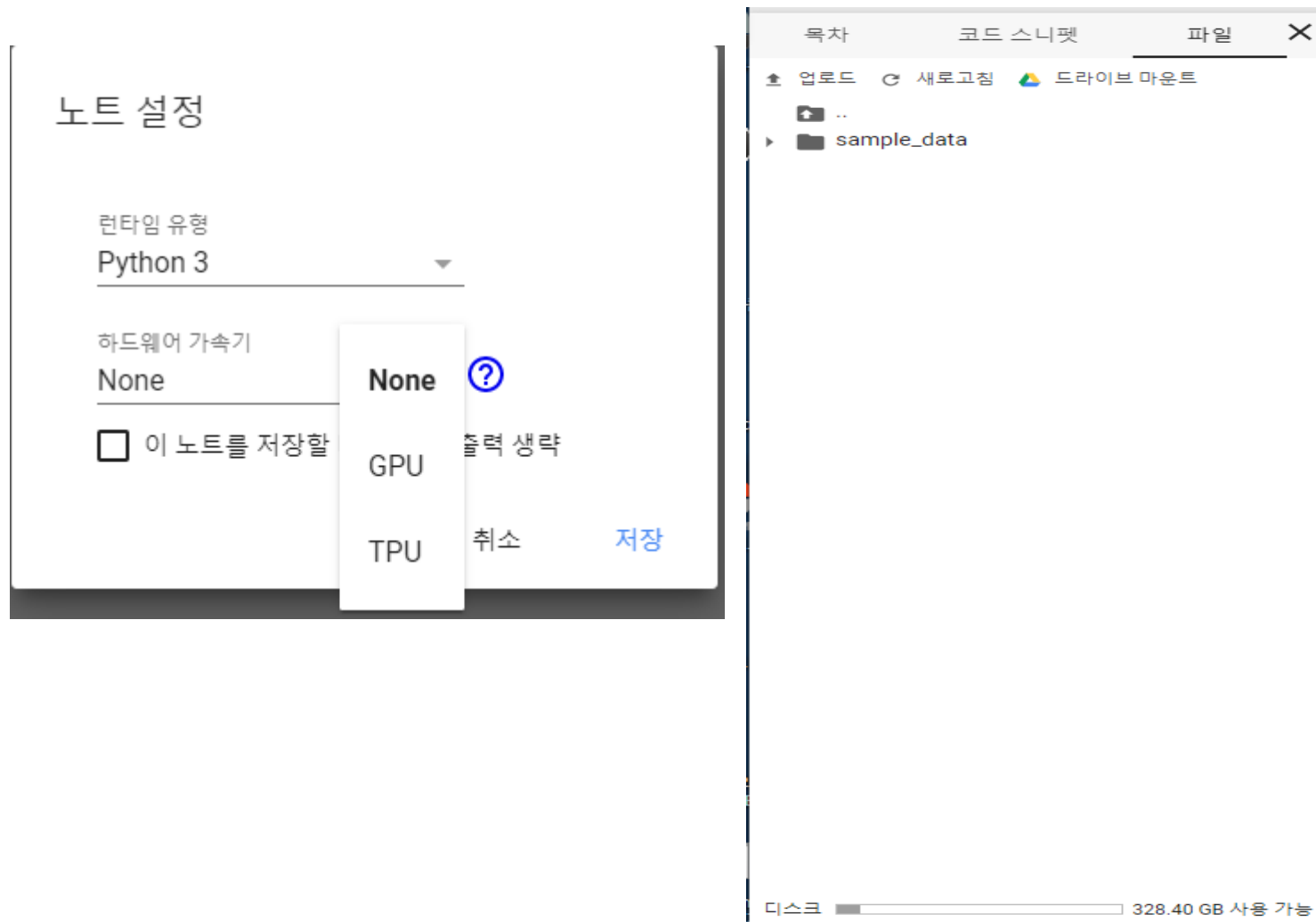
제목	처음 연 시간	마지막 연 시간	
Colaboratory에 오신 것을 환영합니다	2019년 4월 9일	0분 전	
5.2-using-convnets-with-small-datasets.ipynb	20시간 전	2시간 전	
Untitled0.ipynb	2019년 4월 9일	21시간 전	

At the bottom of the list, there is a link to '새 PYTHON 3 노트' (New PYTHON 3 notebook) and a '취소' (Cancel) button. On the right side, a sidebar shows the 'Untitled1.ipynb' file with a star icon. Below the file name, there are tabs for '코드' (Code) and '텍스트' (Text), and buttons for '셀' (Cell) operations. A play button icon is also visible in the sidebar.

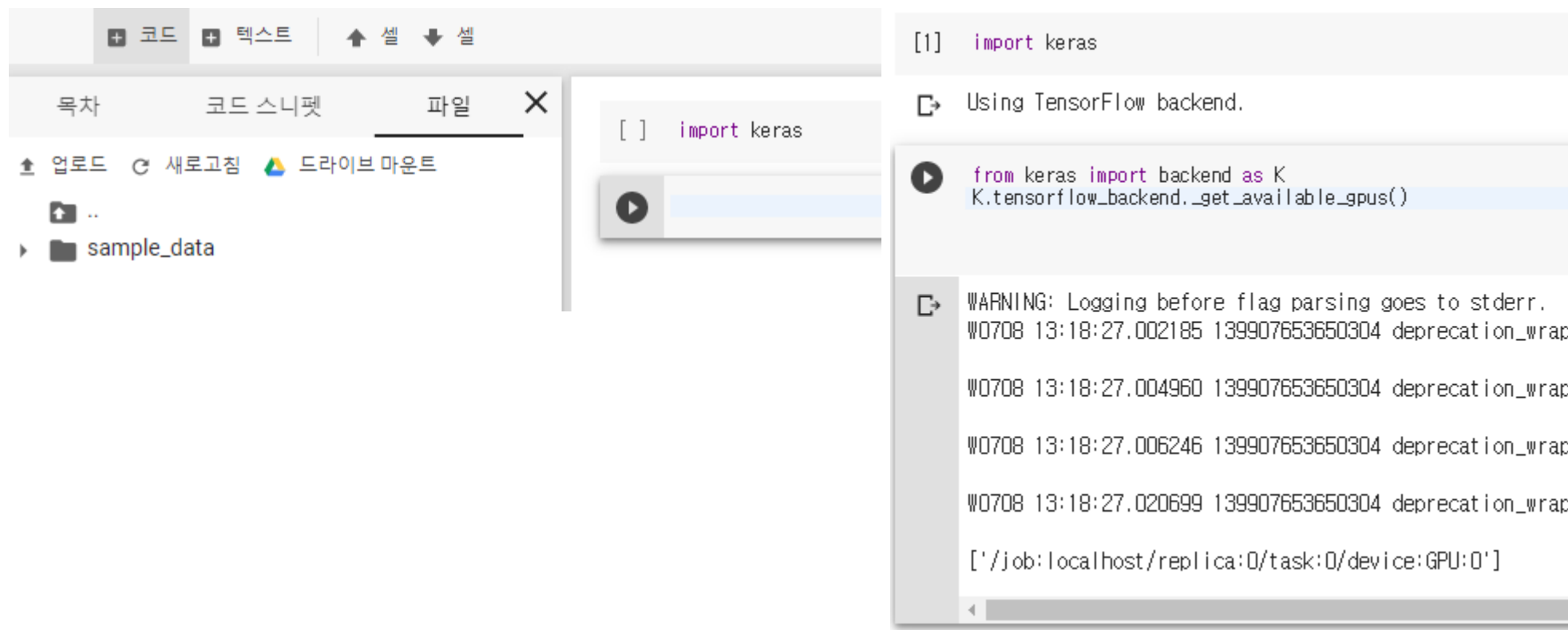
Colab을 사용해봅시다 - 살펴보기



Colab을 사용해봅시다 - GPU사용



Colab을 사용해봅시다 - GPU확인



The screenshot displays the Google Colab web interface. On the left, the file explorer shows a folder named 'sample_data'. The top navigation bar includes tabs for '코드' (Code), '텍스트' (Text), and '셀' (Cells). The main code editor contains the following code:

```
[ ] import keras
```

The output console on the right shows the execution results:

```
[1] import keras
```

Using TensorFlow backend.

```
from keras import backend as K
K.tensorflow_backend._get_available_gpus()
```

WARNING: Logging before flag parsing goes to stderr.
W0708 13:18:27.002185 139907653650304 deprecation_wrap
W0708 13:18:27.004960 139907653650304 deprecation_wrap
W0708 13:18:27.006246 139907653650304 deprecation_wrap
W0708 13:18:27.020699 139907653650304 deprecation_wrap
['/job:localhost/replica:0/task:0/device:GPU:0']

Colab을 사용해봅시다 - 예제 동기화

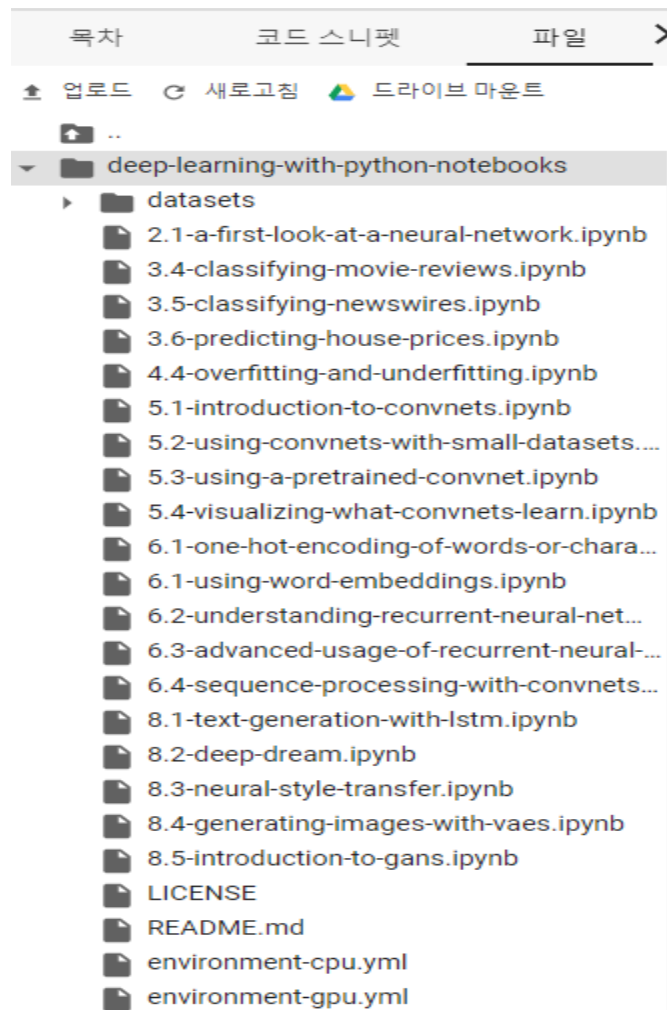
```
from tensorflow.python.client import device_lib  
print(device_lib.list_local_devices())
```

```
[{name: "/device:CPU:0"  
  device_type: "CPU"  
  memory_limit: 268435456  
  locality {  
  }  
  incarnation: 776877190252135024  
  , name: "/device:XLA_CPU:0"  
  device_type: "XLA_CPU"  
  memory_limit: 17179869184  
  locality {  
  }  
  incarnation: 2137097019529257532  
  physical_device_desc: "device: XLA_CPU device"  
  , name: "/device:XLA_GPU:0"  
  device_type: "XLA_GPU"  
  memory_limit: 17179869184  
  locality {  
  }  
  incarnation: 6860569131657014274  
  physical_device_desc: "device: XLA_GPU device"  
  , name: "/device:GPU:0"  
  device_type: "GPU"  
  memory_limit: 14892338381  
  locality {  
    bus_id: 1  
    links {  
    }  
  }  
  incarnation: 13757903569031965471  
  physical_device_desc: "device: 0, name: Tesla T4,  
  ]
```

```
!git clone https://github.com/rickiepark/deep-learning-with-python-notebooks
```

```
Cloning into 'deep-learning-with-python-notebooks'...  
remote: Enumerating objects: 123888, done.  
remote: Total 123888 (delta 0), reused 0 (delta 0), pack-reused 123888  
Receiving objects: 100% (123888/123888), 685.24 MiB | 39.45 MiB/s, done.  
Resolving deltas: 100% (126/126), done.  
Checking out files: 100% (104042/104042), done.
```

Colab을 사용해봅시다 - 예제코드 사용



Colab을 사용해봅시다 - 예제파일 확인

목차

코드 스니펫

파일

×

5.2 - 소규모 데이터셋에서 컨브넷 사용하기

작은 데이터셋 문제에서 딥러닝의 타당성

데이터 내려받기

네트워크 구성하기

데이터 전처리

데이터 증식 사용하기

▶

```
import keras
keras.__version__
```

[]

```
#!pip3 install Pillow
from IPython.display import display
from PIL import Image
```

5.2 - 소규모 데이터셋에서 컨브넷 사용하기

이 노트북은 [케라스 창시자에게 배우는 딥러닝](#) 책의 5장 2절의 코드 예제입니다. 책에는 더
함합니다. 이 노트북의 설명은 케라스 버전 2.2.2에 맞추어져 있습니다. 케라스 최신 버전이
다를 수 있습니다.

Colab을 사용해봅시다 - CPU실행

```
In [*]: history = model.fit_generator(  
        train_generator,  
        steps_per_epoch=100,  
        epochs=30,  
        validation_data=validation_generator,  
        validation_steps=50)
```

WARNING:tensorflow:From c:\program files\python37\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.

Instructions for updating:

Use tf.cast instead.

Epoch 1/30

100/100 [=====] - 141s 1s/step - loss: 0.6878 - acc: 0.5405 - val_loss: 0.6767 - val_acc: 0.5840

Epoch 2/30

100/100 [=====] - 131s 1s/step - loss: 0.6549 - acc: 0.6165 - val_loss: 0.6479 - val_acc: 0.6260

Epoch 3/30

5/100 [>.....] - ETA: 1:38 - loss: 0.5759 - acc: 0.7300

Colab을 사용해봅시다 - GPU실행

```
[16] history = model.fit_generator(  
    train_generator,  
    steps_per_epoch=100,  
    epochs=30,  
    validation_data=validation_generator,  
    validation_steps=50)
```

W0708 13:32:17.919582 140272722179968 deprecation_wrapper.py:119] From /usr/local/lib/python3.6/dist-packages/keras/backend/

```
Epoch 1/30  
100/100 [=====] - 14s 143ms/step - loss: 0.6902 - acc: 0.5435 - val_loss: 0.6704 - val_acc: 0.6060  
Epoch 2/30  
100/100 [=====] - 8s 83ms/step - loss: 0.6614 - acc: 0.5980 - val_loss: 0.6864 - val_acc: 0.5700  
Epoch 3/30  
100/100 [=====] - 8s 83ms/step - loss: 0.6334 - acc: 0.6495 - val_loss: 0.7272 - val_acc: 0.5430  
Epoch 4/30  
100/100 [=====] - 8s 84ms/step - loss: 0.5976 - acc: 0.6755 - val_loss: 0.6043 - val_acc: 0.6760  
Epoch 5/30  
100/100 [=====] - 8s 84ms/step - loss: 0.5533 - acc: 0.7210 - val_loss: 0.5849 - val_acc: 0.6840
```

Colab을 사용해봅시다 - 결론

```
[21] # 이미지 전처리 유틸리티 모듈
      from keras.preprocessing import image

      fnames = sorted([os.path.join(train_cats_dir, fname) for fname in os.listdir(train_cats_dir)])

      # 증식할 이미지 선택합니다
      img_path = fnames[3]

      # 이미지를 읽고 크기를 변경합니다
      img = image.load_img(img_path, target_size=(150, 150))

      # (150, 150, 3) 크기의 넘파이 배열로 변환합니다
      x = image.img_to_array(img)

      # (1, 150, 150, 3) 크기로 변환합니다
      x = x.reshape((1,) + x.shape)

      # flow() 메서드는 랜덤하게 변환된 이미지의 배치를 생성합니다.
      # 무한 반복되기 때문에 어느 지점에서 중지해야 합니다!
      i = 0
      for batch in datagen.flow(x, batch_size=1):
          plt.figure(i)
          imgplot = plt.imshow(image.array_to_img(batch[0]))
          i += 1
          if i % 4 == 0:
              break

      plt.show()
```

