VLSI & System Lab.

딥러닝 프레임워크 기반 인공지능의 이해

한양대 IDEC 강좌 – Day2

숙명여자대학교 ICT융합공학부 전자공학과 최웅

Outline

- 1: ~50분 강의 / ~30분 실습
  - Google Colab
  - Modules Import & Configuration
  - Preparing Data
  - Data Loader
- 2: ~60분 강의 / ~30분 실습
  - Model Build-up & Customizing
  - Model I/O (Input/Output)
  - Loss Function
- 3: ~40분 강의 / ~30분 실습
  - Optimizer
  - Training & Test (Inference)
  - Model Save & Load

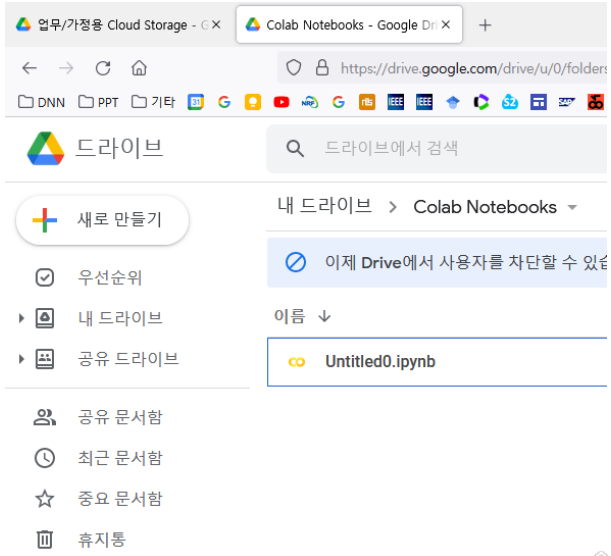
Google Colab.

❑ Link

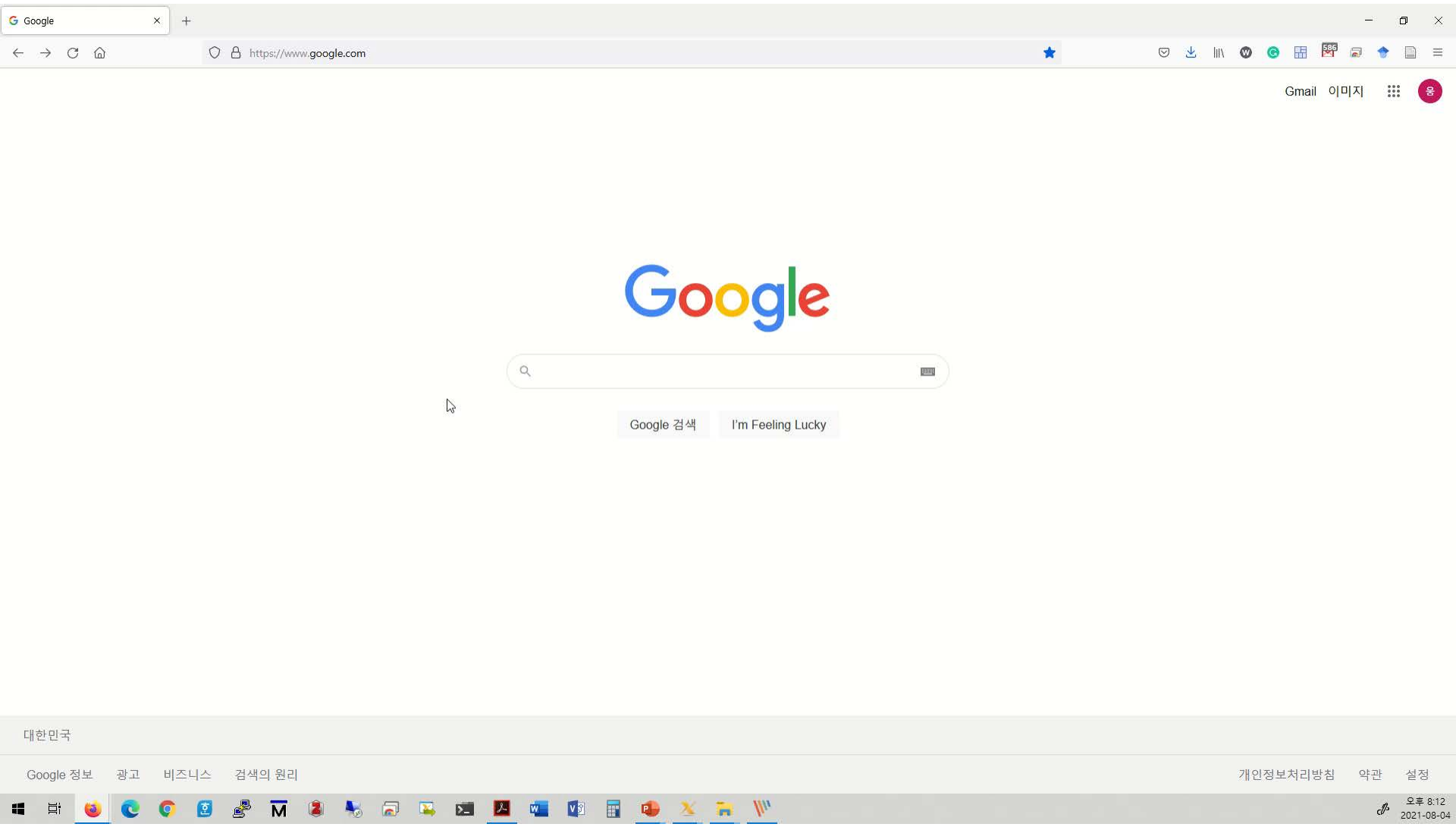
- https://colab.research.google.com/
  - 따로 구성이 필요하지 않음
  - GPU 무료 액세스 및 간편한 공유

❑ Code Cell & Markdown

구글 드라이브에 저장됨



Google Colab.



Google Colab.

❑ 단축키

- ctrl + m + d: 코드 셀 삭제
- ctrl + m + a: 코드 셀 위에 새로운 코드 셀 삽입
- ctrl + m + b: 코드 셀 아래에 새로운 코드 셀 삽입
- shift + Enter: 현재 선택된 코드 셀 실행
- ctrl + / (코드의 일부가 선택된 상황에서): 주석 처리

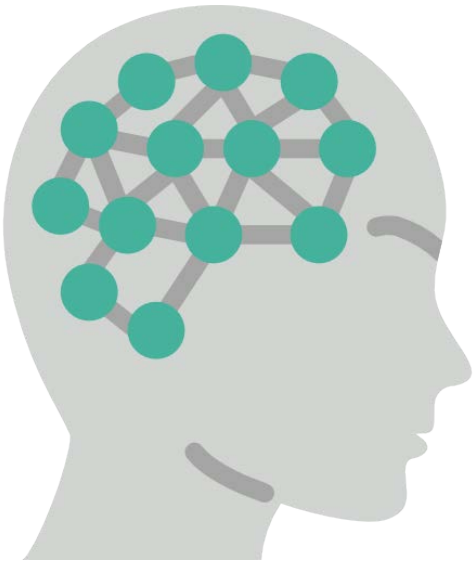
❑ GPU 사용

- 런타임 > 런타임 유형 변경
  - !nvidia-smi 로 확인 가능

MNIST Dataset

❑ MNIST Example

- 28 x 28 = 784 pixels Images



Module Import & Config.

01_module&preparing_dataset.ipynb

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

Module Import & Configuration

```python
import torch
import numpy as np
from torch import nn
from torch.utils.data import DataLoader
from torchvision import datasets
from torchvision.transforms import ToTensor
import matplotlib.pyplot as plt
import warnings
# select multiline -> 'ctrl + /' -> multiline comment
# print option
torch.set_printoptions(precision=4, linewidth=50000, sci_mode=None)
# Control Warning Message
warnings.filterwarnings(action='ignore')
```

Module & Configuration Test

```python
1
```

Preparing Data

```python
dataset_dir = '~/dataset'
train_data = datasets.MNIST(root=dataset_dir, train=True,  download=True, transform=ToTensor())
test_data  = datasets.MNIST(root=dataset_dir, train=False, download=True, transform=ToTensor())
```

0초 오전 9:19에 완료됨

Module Import & Config.

Module Import & Configuration

```python
import torch
import numpy as np
from torch import nn
from torch.utils.data import DataLoader
from torchvision import datasets
from torchvision.transforms import ToTensor
import matplotlib.pyplot as plt
import warnings
# select multiline -> 'ctrl + /' -> multiline comment
# print option
torch.set_printoptions(precision=4, linewidth=50000, sci_mode=None)
# Control Warning Message
warnings.filterwarnings(action='ignore')
```

Preparing Data

01_module&preparing_dataset.ipynb

Preparing Data

[6]
```
dataset_dir = '~/dataset'
train_data = datasets.MNIST(root=dataset_dir, train=True,  download=True, transform=ToTensor())
test_data  = datasets.MNIST(root=dataset_dir, train=False, download=True, transform=ToTensor())
train_data
```

```
Dataset MNIST
    Number of datapoints: 60000
    Root location: /root/dataset
    Split: Train
    StandardTransform
Transform: ToTensor()
```

```
test_data
```

```
Dataset MNIST
    Number of datapoints: 10000
    Root location: /root/dataset
    Split: Test
```

[8]
```
print('[Print] Type of trainData :', type(train_data))
print('[Print] Type of trainData.data :', type(train_data.data))
print('[Print] Type of trainData.targets :', type(train_data.targets))
print('[Print] Type of trainData.classes :',type(train_data.classes))
```

```
[Print] Type of trainData : <class 'torchvision.datasets.mnist.MNIST'>
[Print] Type of trainData.data : <class 'torch.Tensor'>
[Print] Type of trainData.targets : <class 'torch.Tensor'>
[Print] Type of trainData.classes : <class 'list'>
```

[12]
```
print(train_data.targets)
```

0초 오전 11:28에 완료됨

Preparing Data

❑ Dataset

- 불러오기: torchvision의 하위 모듈인 datasets을 이용

```
dataset_dir = '~/dataset'
train_data = datasets.MNIST(root=dataset_dir, train=True,  download=True, transform=ToTensor())
test_data  = datasets.MNIST(root=dataset_dir, train=False, download=True, transform=ToTensor())
train_data
```

- targets, classes

```
print(train_data.targets)
print(train_data.classes)
```

```
tensor([5, 0, 4,  ..., 5, 6, 8])
['0 - zero', '1 - one', '2 - two', '3 - three', '4 - four', '5 - five', '6 - six', '7 - seven', '8 - eight', '9 - nine']
```

- ‘train_data.data’ vs ‘train_data’
  - Raw Data를 UINT8 Max값인 255로 나누어 Normalize

```
# trainData is nomalized using /255
print(f'trainData.data: {train_data.data[0][24][4]}')
print(f'trainData.data/255: {train_data.data[0][24][4]/255}')
print(f'trainData: {train_data[0][0][0][24][4]}')
```

```
trainData.data: 136
trainData.data/255: 0.5333333611488342
trainData: 0.5333333611488342
```

Data Loader

01_Pytorch_Basic.ipynb - Colab

https://colab.research.google.com/drive/1caHYei7pcxG4MYoP1FoFJP25eBsDZo9C?hl=ko#scrollTo=WSJRR2IM8TcO

01_Pytorch_Basic.ipynb

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트

```
trainData: 0.5333333611488342
```

Data Loader

```python
# train_dataset, valid_dataset = torch.utils.data.random_split(train_data, [50000, 10000])
# print(f'trainDataset: {len(train_dataset)}')
# print(f'validDataset: {len(valid_dataset)}')

batch_size = 64

# train_dataloader = DataLoader(train_dataset, batch_size=batch_size)
# valid_dataloader = DataLoader(valid_dataset, batch_size=batch_size)
# test_dataloader  = DataLoader(test_data,  batch_size=batch_size)

train_dataloader = DataLoader(train_data, batch_size=batch_size)
test_dataloader  = DataLoader(test_data,  batch_size=batch_size)

for imgs, labs in train_dataloader:
    print("Shape of image [N, C, H, W]: ", imgs.shape)
    print("Shape of label             : ", labs.shape, labs.dtype)
    break
```

```
Shape of image [N, C, H, W]:  torch.Size([64, 1, 28, 28])
Shape of label             :  torch.Size([64]) torch.int64
Shape of image [N, C, H, W]:  torch.Size([64, 1, 28, 28])
Shape of label             :  torch.Size([64]) torch.int64
```

[72]

```python
imgs[0]*255
```

0초 오후 12:41에 완료됨

Data Loader

❑ Mini-Batch

```python
# train_dataset, valid_dataset = torch.utils.data.random_split(train_data, [50000, 10000])
# print(f'trainDataset: {len(train_dataset)}')
# print(f'validDataset: {len(valid_dataset)}')

batch_size = 64

# train_dataloader = DataLoader(train_dataset, batch_size=batch_size)
# valid_dataloader = DataLoader(valid_dataset, batch_size=batch_size)
# test_dataloader  = DataLoader(test_data,  batch_size=batch_size)

train_dataloader = DataLoader(train_data, batch_size=batch_size)
test_dataloader  = DataLoader(test_data,  batch_size=batch_size)

for idx, (imgs, labs) in enumerate(train_dataloader):
    print("Shape of image [N, C, H, W]: ", imgs.shape)
    print("Shape of label              : ", labs.shape, labs.dtype)
    if idx == 1:
        break
```