



Assignment #3

AATG013/ANT5013 Deep Learning and Computer Vision

Assigned: Tuesday, 4/25, 2023

Due: Tuesday, 5/2, 2023 (by midnight)

- Colab code 1 set을 사이버캠퍼스에 제출
 - assignment6-1.ipynb

code는 colab에서 수행되어 결과를 보여야 합니다.



Problem [1] PyTorch code: image classification using CNN and cfar10

Problem [1] [50 points]

Assignment 3에서는 CIFAR10 dataset을 사용하고 FC MLP (fully-connected multilayer perceptron) NN을 구성하고, image classification을 수행하였습니다. CIFAR10 dataset은 FashionMNIST와 마찬가지로 10 classes의 물체 images로 구성되어 있으나, 그 images의 내용에 있어서 FashionMNIST images 보다 약간 더 복잡도가 높습니다. 따라서 FC MLP로는 accuracy를 높이는 데 한계가 있어서 Assignment 3에서 주어진 model과 조건으로는 약 50%~60% 정도만 달성할 수 있었습니다.

- 본 과제에서는 같은 CIFAR10 dataset을 사용하나 이번엔 CNN으로 classification net을 구성하여 실험하고자 합니다.
- Assignment 3에서 chapter 3의 code를 개정했던 것처럼, 아래의 github chapter 4에 있는 code을 개정하여 CNN image classification network을 design, train, validate하는 pyTorch code를 작성하고 제출합니다. [40 points]

<https://github.com/PacktPublishing/Modern-Computer-Vision-with-PyTorch>

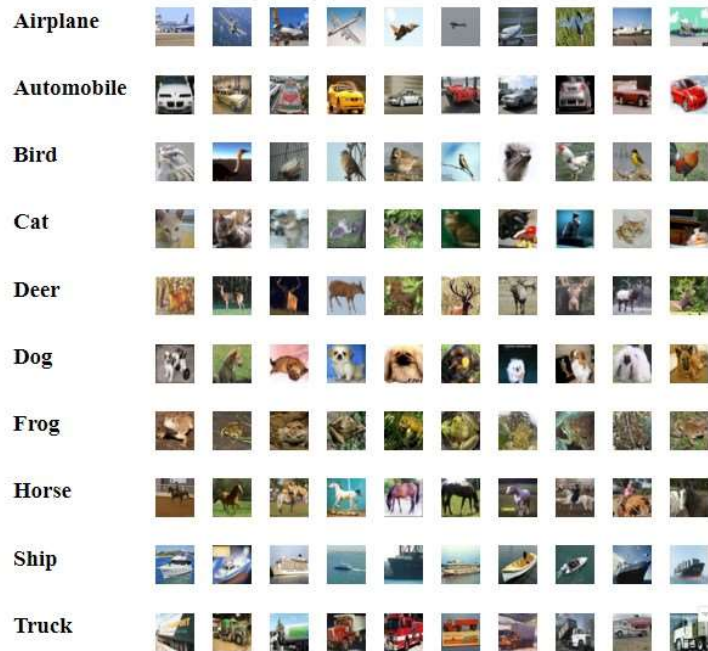


	image size	train images	validation images
CIFAR10	32x32x3	50,000	10,000

train dataset으로 train하고 train loss, validation loss, train accuracy, validation accuracy 등의 plot를 보이는 Colab code, assignment 6-1.ipynb 를 제출합니다. [40 points] .



Problem [1] : data handling (from Assignment 3)

```
import torch
import torch.nn as nn
device = "cuda" if torch.cuda.is_available() else "cpu"

from torch.utils.data import Dataset, DataLoader
from torchvision import datasets, transforms

import matplotlib.pyplot as plt
import numpy as np

transform = transforms.ToTensor()

data_path = '../data_cifar/'
cifar10_train = datasets.CIFAR10(data_path, train=True, download=True, transform=transform)
cifar10_val = datasets.CIFAR10(data_path, train=False, download=True, transform=transform)

print("Training: ", len(cifar10_train))    #Training: 50000
print("Validating: ", len(cifar10_val))    #Testing: 10000

type(cifar10_train)          # torchvision.datasets.cifar.CIFAR10
type(cifar10_val)           # torchvision.datasets.cifar.CIFAR10

type(cifar10_train[0])      # tuple

image, label = cifar10_train[1]
type(image)                 # torch.Tensor
image.shape                 # torch.Size([3, 32, 32])
```

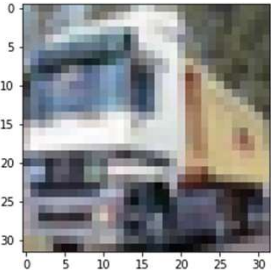


Problem [1] : data handling (from Asssignment 3)

```
classes = cifar10_train.classes
print (classes)                # ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog',
                                # 'frog', 'horse', 'ship', 'truck']

print(label)                   # 9
print(classes[label])          # truck

plt.imshow(image.permute(1, 2, 0))
```



```
torch.manual_seed(80)
train_loader = DataLoader(cifar10_train, batch_size=100, shuffle=True)
val_loader   = DataLoader(cifar10_test, batch_size=500, shuffle=False)

# . . .
```



Problem [1] : CNN guideline

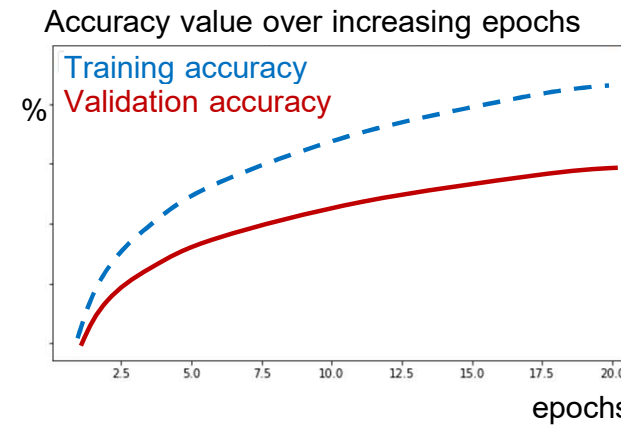
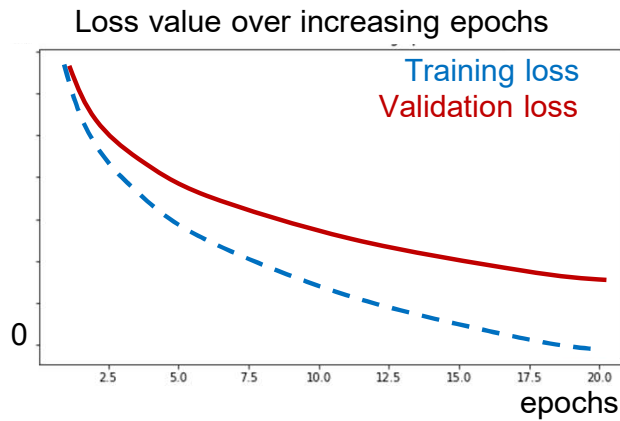
[CNN_on_FashionMNIST.ipynb](#)

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 26, 26]	640
MaxPool2d-2	[-1, 64, 13, 13]	0
ReLU-3	[-1, 64, 13, 13]	0
Conv2d-4	[-1, 128, 11, 11]	73,856
MaxPool2d-5	[-1, 128, 5, 5]	0
ReLU-6	[-1, 128, 5, 5]	0
Flatten-7	[-1, 3200]	0
Linear-8	[-1, 256]	819,456
ReLU-9	[-1, 256]	0
Linear-10	[-1, 10]	2,570

- 교재/강의노트/github의 CNN code는 위에 보인 바와 같이 2개의 convolution layer와 2개의 linear layer로 구성.
- 과제에서는 2~3개의 convolution layer와 2~3개의 linear layer로 독자적으로 구성.
 - 디자인의 결과를 반드시 위와 같은 torch_summary로 보임.
 - Filter 수 등은 조정
 - Transfer learning은 사용하지 않음. (VGG-16, ResNet-18 등은 사용하지 않음)
 - Image size 는 CIFAR10에 맞춤



Problem [1] : 결과 제출



- `train loss`, `validation loss`, `train accuracy`, `validation accuracy` 등을 plot
- Train batch size는 100, validation은 전체 10000
- `Tensor`, `array` 등은 `dimension`이 맞도록 주의.
- `torch_summary` 출력
- 주어진 상황에서 최대의 성능을 이끌어내도록 다음을 조정
 - Scaling, batch normalization
 - Regularization, dropout
 - Optimizer, Learning rate, . . .
 - Code의 text/comment cell에 어떠한 조정/hyperparameter 등이 사용되었는지 명확히 기술하시오.