Fashion-MNIST Classification

PROJECT 3.

Computer Vision 2021 Spring

Due date May 28th

1. Introduction

1. Problem : Classification

2. Dataset : Fashion-MNIST

3. This project consists of two parts.

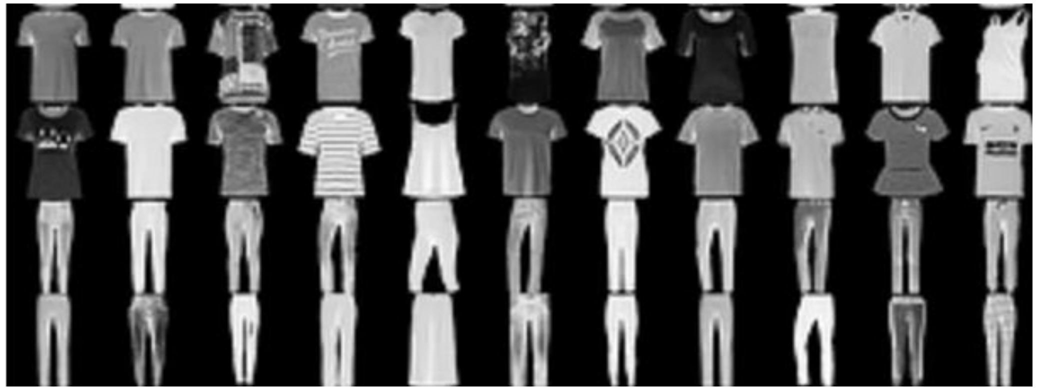
- Task1 : Let's analyze and Compare the difference between

  'Fully Connected Model' and 'LeNet-5'

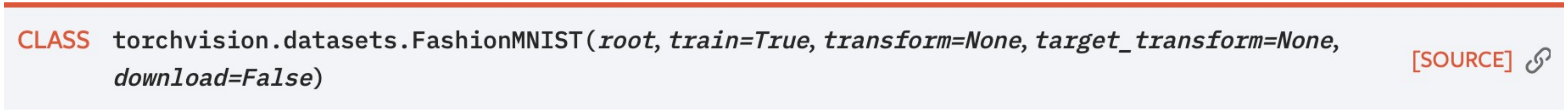
- Task2 : Let's analyze the element of training process
  - Adjusting Learning rate, Batch Size, Loss Function, Optimizer parameter and so on.

2. Dataset

1. **Fashion-MNIST Dataset** : Fashion-MNIST is a dataset of [Zalando]'s article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes



2. **Download** : You can get FashionMNIST dataset using torchvision.dataset.



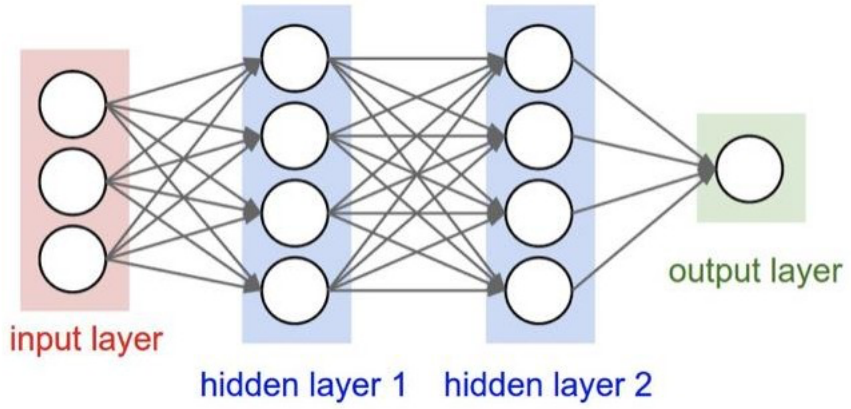
```
CLASS torchvision.datasets.FashionMNIST(root, train=True, transform=None, target_transform=None,
      download=False)                                                                          [SOURCE]
```

https://pytorch.org/docs/stable/torchvision/datasets.html#fashion-mnist

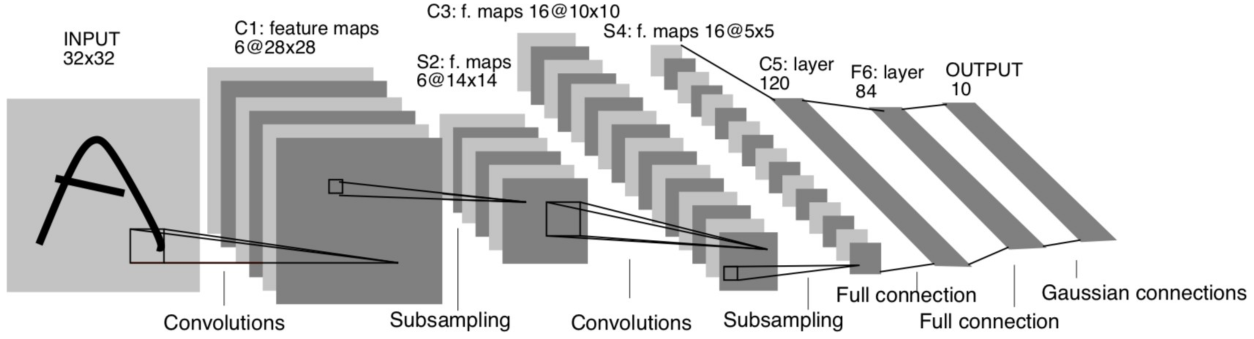
3. Task 1

Implement two networks :

1. Fully Connected Network



2. LeNet-5



- input shape: [channel, height, width] of input image
- output shape : vector(dimension=#class)
- You can adjust the number of hidden layers and units per layer.

- The default network architecture is above.
- In case of LeNet-5, the input size is [32x32x1]. So you need to use padding. You don't have to do feature map selection at Layer 3

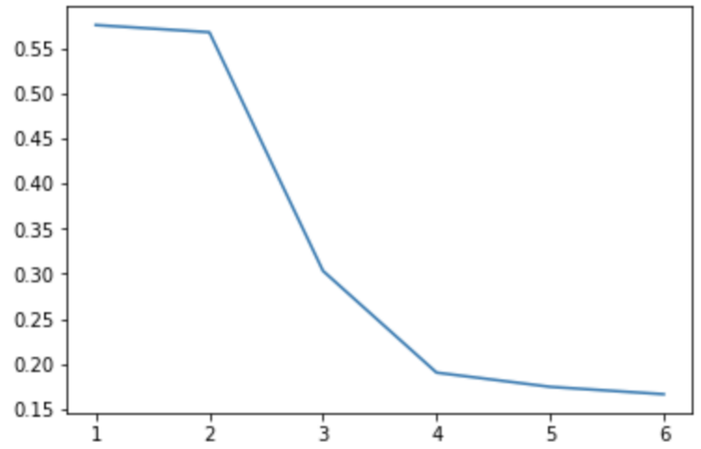
3. Task 1

Train and test two networks respectively.

1. **Train** : Train the two networks, fully connected network and LeNet-5. Define your loss function and optimizer using the PyTorch library and train the networks. To check your training state, print the loss periodically. There is no restriction on the printing format.

2. **Test** : With your trained network above, test the network using the test dataset. You have to print the accuracy of the test dataset to measure how well you trained your network. There is no restriction on the printing format.

```
[1, 500] loss: 0.5754264345169068
[1, 1000] loss: 0.5674753073453903
[1, 1500] loss: 0.3031841230094433
[2, 500] loss: 0.19046795634925365
[2, 1000] loss: 0.17470250104367732
[2, 1500] loss: 0.16652968794107437
```



print example

```
Accuracy of the network on the 10000 test images: 83 %
```

4. Task 2

Now, let’s analyze the elements of training process. From now on, we’ll use ‘LeNet-5’ network to analyze their influences.

1. **Optimizer parameter(including learning rate)** : Adjust parameter related with optimizer and compare the results before and after doing those
2. **Loss function** : Use different loss function and compare the results
3. **Batch size** : Use different batch size and compare the results
4. **Be creative and analytical!** : Change other hyper-parameters or model elements.

5. Submission : Report + Code

1. You should submit your report(.pdf) with your code(.py or .ipynb)

2. Your code should run without any errors.

Ex)

```
20XX123456_project3.zip
 └ report.pdf
 └ code.py (or code.ipynb)
```

5. Submission : Report

1. Environment : OS, version(pytorch, python), etc…
2. Network architecture description
   (Layer, Feature Map, Kernel Size, Stride, Activation)
3. Explain the concept and the function of the loss function and the optimizer you use
4. Screenshot of outputs with analysis
5. Analysis of the difference between Fully Connected Network and LeNet-5
6. Analysis of the effects of loss function, optimizer parameters, batch size and so on.

6. Grading Policy

1. Code ： 30 pts
2. Reports ： 70 pts

Total : 100 pts

7. Due Date

5/28 23:59

For late submission, -10% per each day.