**data.py (program file1 to be used in Lab1)**

**#Input packages to be used**

import numpy as np # to handle matrix and data operation

import pandas as pd # to read csv and handle dataframe

import torch # to load pytorch library

import torch.utils.data # to load data processor

**# load training and testing datasets**

train = pd.read\_csv('mnist\_train.csv', header=None)

# read a csv file called 'mnist\_train.csv'

test = pd.read\_csv('mnist\_test.csv', header=None)

# read a csv file called 'mnist\_train.csv'

print(train)

print(train.info())

print(test)

print(test.info())

**# split image and label**

train\_label = train.iloc[:, 0].values

# first : gets all rows, second 0 gets the first column (label)

train\_img = train.iloc[:, 1:]

# first : gets all rows, second 1: gets the second to last columns (data)

print(train\_label)

print(train\_img)

test\_label = test.iloc[:, 0].values

# first : gets all rows, second 0 gets the first column (label)

test\_img = test.iloc[:, 1:]

# first : gets all rows, second 1: gets the second to last columns (data)

print(test\_label)

print(test\_img)

print(train\_img.shape)

print(test\_img.shape)

**# reshape data to be [samples][channel][width][height]**

train\_img = train\_img.values.reshape(-1,1,28,28)

test\_img = test\_img.values.reshape(-1,1,28,28)

**# convert image pixel value from [0, 255] to [0, 1]**

train\_img = train\_img / 255.0

test\_img = test\_img / 255.0

print(train\_img.shape)

print(test\_img.shape)

**# convert the data to tensor format for training**

torch\_X\_train = torch.from\_numpy(train\_img).float()

# training images

torch\_y\_train = torch.from\_numpy(train\_label)

# training labels

torch\_X\_test = torch.from\_numpy(test\_img).float()

**# testing images**

torch\_y\_test = torch.from\_numpy(test\_label)

# testing labels

# print data dimension

print('training image dimension: ', torch\_X\_train.shape)

print('training label dimension: ', torch\_y\_train.shape)

print('testing image dimension: ', torch\_X\_test.shape)

print('testing label dimension: ', torch\_y\_test.shape)

**# pack image and label into one class**

def get\_train\_set():

return torch.utils.data.TensorDataset(torch\_X\_train,torch\_y\_train)

def get\_test\_set():

return torch.utils.data.TensorDataset(torch\_X\_test,torch\_y\_test)