ELEC946 Intelligent System Design, Spring 2021
Homework Programming Assignment 4

Instructor: Gil-Jin Jang   Email: gjang@knu.ac.kr
School of Electronics Engineering, Kyungpook National University

1 Introduction

The purpose of programming assignment 4 is implementing neural network training on FashionMnist and digits data from scikit-learn, using train_neuralnet.py in Chapter 5.

The following code is a two-layer multilayer perceptron using backpropagation learning (two_layer_net.py from https://github.com/WegraLee/deep-learning-from-scratch/ch05 is required.

```
import sys, os
sys.path.append(os.pardir)

import numpy as np
from dataset.mnist import load_mnist
from two_layer_net import TwoLayerNet

# read MNIST data
(x_train, t_train), (x_test, t_test) = load_mnist(normalize=True, one_hot_label=True)

network = TwoLayerNet(input_size=784, hidden_size=50, output_size=10)

iters_num, train_size, batch_size, learning_rate = 10000, x_train.shape[0], 100, 0.1
train_loss_list, train_acc_list, test_acc_list = [], [], []
iter_per_epoch = max(train_size / batch_size, 1)

for i in range(iters_num):
    batch_mask = np.random.choice(train_size, batch_size)
    x_batch = x_train[batch_mask]
    t_batch = t_train[batch_mask]

    # computing gradients
    #grad = network.numerical_gradient(x_batch, t_batch) # numerical gradients
    grad = network.gradient(x_batch, t_batch) # backpropagation (faster)

    # update
    for key in ('W1', 'b1', 'W2', 'b2'):
        network.params[key] -= learning_rate * grad[key]

    loss = network.loss(x_batch, t_batch)
    train_loss_list.append(loss)

    if i % iter_per_epoch == 0:
        train_acc = network.accuracy(x_train, t_train)
        test_acc = network.accuracy(x_test, t_test)
        train_acc_list.append(train_acc)
        test_acc_list.append(test_acc)
        print(train_acc, test_acc)
```

The input argument hidden_sizes=(50,) of the class constructor TwoLayerNet() indicates that there is a single hidden layer of 50 nodes.

Assignment 4: Trying Other Data

4.1 Fashion MNIST

Modify the above code for Fashin MNIST, whose description is given in assignment 3.

Submission `hw4-1.py`

Using exisiting codes If your code uses `two_layer_net.py` from `https://github.com/WegraLee/deep-learning-from-scratch/ch05`, you do not need to include that file.

Using your own methods/classes If you added any extra methods or classes, include in `hw4-1.py` so that you do not need to add any extra code files.

Parameters If necessary, change the parameters.

4.2 sckit-learn's digits

Modify the above code for digits from scikit-learn.

Submission `hw4-2.py`

Using exisiting codes same as Assignment 4.1

Using your own methods/classes same as Assignment 4.1

Dataset splitting Split the dataset into training (80%) and test (20%) sets. Note: the split should be done per-class — for each class, split 80/20 for training/test.

Parameters If necessary, change the parameters.

4.3 wine.csv

Modify the code in Assignment 4.1 for the file `wine.csv`.

Submission `hw4-3.py`. Execution: `python3 hw4-3.py` *`filename`*. The *filename* can be the name of any file. Your code should read any csv-formatted file. Refer to assignment 1 for reading a csv file.

Using exisiting codes same as Assignment 4.1

Using your own methods/classes same as Assignment 4.1

Dataset splitting Split the dataset into training (80%) and test (20%) sets. Note: the split should be done per-class — for each class, split 80/20 for training/test.

Parameters If necessary, change the parameters.

Submission Guidelines and Grading Scheme

Common Requirements:

1. `wine.csv` is given for you to test `hw4-3`.
2. write or replace with ID and NAME of yours at the beginning of the code (10%).
3. Use python 3.7 or higher.
4. specify the names of used packages in your code in the first comment block. You may install new packages (libraries) locally by python3 command `‘‘pip3 install ...’’`

5. make sure that you have installed most recent version of scikit-learn (0.23.2, as of November 22, 2020) to properly run the example. Use the command `‘‘pip3 install sklearn>=0.23.2’’`
6. Make a zip file `hw4.zip` of all the necessary `.py` files, and upload it to `lms.knu.ac.kr`
7. This programming assignment is rougly 7-9% of total score.

Grading: The grading score is composed of

10% Basic score for submission

10% Name, ID, and other information is correct

50% Executability and correctness of the output

30% Code readability (subjective)

Plagiarism For copy and being-copied, all the assignment scores will become 0

Due and late submission see LMS.

Late submission deduction 10% deduction per hour afer the regular submission deadline.