EE412 Foundation of Big Data Analytics, Fall 2021
HW4

Due date: 12/19/2021 (11:59pm)

Submission instructions: Use KAIST KLMS to submit your homework. Your submission should be a single gzipped tar file with the file name `YourStudentID_hw4.tar.gz`. For example, if your student ID is 20211234, the file name should be `20211234_hw4.tar.gz`. You can also use these extensions: tar, gz, zip, tar.zip. Do not use other options besides the ones mentioned above.

For the programming assignment, you are free to use either python 2.x or 3.x. Make sure to add p2 or p3 to the end of the file names. For example, if you used python 2, the file names should be `hw4_1_p2.py` and `hw4_2_p2.py`

Your zip file should contain a total four files; one PDF file for writeup answers (`hw4.pdf`), two python files and the Ethics Oath pdf file. Do not use any Korean letters in file names or directory names.
If you do not follow this format, we will **deduct** 1 point of total score per mistake.

Submitting writeup: Write down the solutions to the homework questions in a single PDF file. You can use the following template. If you use Python in the exercises, **make sure you attach your code to your document**(`hw4.pdf`). Please write as succinctly as possible.

Submitting code: Each problem is accompanied by a programming section. Put all the code for each question into a single file. For homework 4, we will only allow you to to import **numpy, sys, math, csv, and os** in your code. **You are not allowed to use any other libraries**. Please make sure your code is well structured and has descriptive comments.
You will receive credit for each correct output line. We will **deduct** 1 point per line if you print any other line except the answer (i.e., elapsed time, description, and etc.).

Ethics Oath: For every homework submission, please fill out and submit the **PDF** version of this document that pledges your honor that you did not violate any ethics rules required by this course and KAIST. You can either scan a printed version into a PDF file or make the Word document into a PDF file after filling it out. Please sign on the document and submit it along with your other files.

Discussions with other students are permitted and encouraged. However, for writing down the solutions, such discussions (except with course staff members) are no longer

acceptable: you must write down your own solutions independently. If you received any help, you must specify on the top of your written homework any individuals from whom you received help, and the nature of the help that you received. *Do not, under any circumstances, copy another person's solution.* We check all submissions for plagiarism and take any violations seriously.

1 Neural Nets and Deep Learning (40 points)

(a) [20 pts] Compute the gradients using the chain rule

Figure 1 shows a simple neural network which consists of two input nodes, two hidden nodes and two output nodes. Given input data x_1, x_2, we want the network to predict the target label y_1, y_2 correctly. In other words, the network outputs o_1, o_2 are equal to the target labels y_1, y_2. In order to train the network using gradient descent, we have to compute the gradient of the loss function with respect to the parameters of the network using the backpropagation algorithm.

Suppose we use the sigmoid activation function in the hidden and output layers. Also say we use the mean squared error for the loss function. Express the answer using the parameters in Figure 1.

- Compute the gradient of the loss with respect to w^2_{ij} $(i, j = 1, 2)$
- Compute the gradient of the loss with respect to w^1_{ij} $(i, j = 1, 2)$

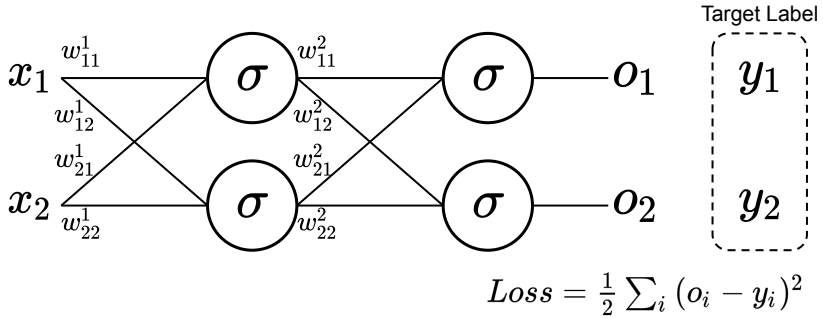

Figure 1: A simple neural network

(b) [20 pts] Implement a fully-connected network to distinguish digits using Python.

The dataset can be downloaded from this link:
http://www.di.kaist.ac.kr/∼swhang/ee412/mnist_sample.zip[1]

The dataset is sampled from the MNIST dataset, which is one of the most common datasets used for image classification and contains 70,000 grayscale images of handwritten digits (0 to 9). Each image has 28 x 28 resolution and is transformed to a one-dimensional vector of length 784.

[1] THE MNIST DATABASE of handwritten digits.

- `training.csv`: Used for the model training and contains 100 data points for each digit. Each line contains a vector of length 784 and the corresponding label.
- `testing.csv`: Used for the model testing. Uses the same format as `training.csv`.

Construct a fully-connected network, which has an input layer, one hidden layer, and an output layer. First, convert the label to a vector of size 10 using **one-hot encoding** where the vector element corresponding to the label has a 1 while all the other elements are 0. For example, 0 is encoded as [1, 0, 0, 0, 0, 0, 0, 0, 0, 0] while 5 is encoded as [0, 0, 0, 0, 0, 1, 0, 0, 0, 0]. One-hot encoding is commonly used to deal with categorical values. Then, implement the gradient descent algorithm using forward and backward propagation to train the network. Use the sigmoid activation function and mean squared error loss function as in (a).

We provide a simple skeleton code (`hw4_1_sample.py`) for better understanding, but feel free to modify it as needed:
http://www.di.kaist.ac.kr/∼swhang/ee412/hw4_1_sample.py

You will have to try different numbers of iterations and learning rates while monitoring the loss and accuracy to find a hyperparameter setting that results in high accuracy. We define accuracy as the fraction of predictions the network got right (range in [0, 1]).

Please **use command-line** arguments to obtain the file path of the dataset. (Do not fix the path in your code.) For example, run:

```
python hw4_1.py path/to/training.csv path/to/testing.csv
```

After training the network, your code (`hw4_1_p2.py` or `hw4_1_p3.py`) should `print` the accuracy on both training and test data as well as the the number of iterations and learning rate η. Typically, the test accuracy is lower than the training accuracy. The output format is the following:

```
<TRAINING ACCURACY>
<TEST ACCURACY>
<NUMBER OF ITERATIONS>
<η>
```

For example,

```
0.941
0.820
5000
0.001
```

We will give full credit as long as the test accuracy is reasonably high (> 0.7).

2 Mining Data Streams (40 points)

(a) [20 pts] Solve the following problems, which are based on the exercises in the Mining of Massive Datasets 3rd edition (MMDS) textbook.

- Exercises 4.4.1 and 4.4.2
  Suppose our stream consists of the integers 3, 1, 4, 1, 5, 9, 2, 6, 5. Our hash functions will all be of the form $h(x) = ax + b \bmod 32$ for some a and b. You should treat the result as a 5-bit binary integer. Determine the tail length for each stream element and the resulting estimate of the number of distinct elements if the hash function is:
  (a) $h(x) = 2x + 1 \bmod 32$.
  (b) $h(x) = 3x + 7 \bmod 32$.
  (c) $h(x) = 4x \bmod 32$.
  Do you see any problems with the choice of hash functions? What advice could you give someone who was going to use a hash function of the form $h(x) = ax + b \bmod 2^k$?

- Exercise 4.5.3
  Suppose we are given the stream 3, 1, 4, 1, 3, 4, 2, 1, 2 and apply the Alon-Matias-Szegedy Algorithm to estimate the surprise number. For each possible value of i, if X_i is a variable starting position i, what is the value of $X_i.value$?

(b) [20 pts] Implement the DGIM algorithm.

Implement the DGIM algorithm described in MMDS chapter 4.6 using Python.

The dataset can be downloaded from this link:
http://www.di.kaist.ac.kr/∼swhang/ee412/stream.txt

The stream is randomly generated and contains a sequence of 10 million 0's and 1's. Each line contains a 0 or 1. Suppose that this sequence is the window (i.e., $N =$ 10M) and that we need to summarize the window due to its large size. The goal is to estimate the number of 1's in the last k bits.

Please **use command-line** arguments to obtain the path for data file and multiple ks. (Do not fix the paths in your code.) For example, run:

`python hw4_2.py path/to/stream.txt` k_1 k_2 ... k_M

After the run, your code (`hw4_2_p2.py` or `hw4_2_p3.py`) should `print` M estimated numbers of 1's in the last k_i bits of the window. For example, the first line is the estimated number of 1's in the last k_1 bits:

`python hw4_2.py path/to/stream.txt 1 10 100`

0.5
2.0
23.0

The output type is not important if the answer is correct. For example, both 2 and 2.0 are allowed. You should make your algorithm as efficient as possible since M can be arbitrary.

Answers to Frequently Asked Questions

- About using **external libraries**: We will only allow you to import **numpy, sys, math, csv, and os** in your code. You are not allowed to use any other libraries.
- Please use the print **built-in function** in Python to produce outputs.
- **No late submission** allowed as we need to submit grades by 12/23.
- There is no time limit for HW4. However, both the MNIST classification task and the DGIM algorithm do not take much time if you have implemented them correctly.
- Questions about problem 1-(b)
  - You can add more hidden layers.
  - You can add a bias vector.
  - You can use variants of gradient descent algorithm (e.g., mini-batch gradient descent, stochastic gradient descent).
  - You can report the number of epochs instead of the number of iterations.