Machine Learning HW2

TA: 蔡馥任 (Handwriting) qwe846132@gapp.nthu.edu.tw

李明峯 (Programming) li871030@gmail.com

Deadline: 2023/04/17 (Mon.) 23:59

Grading Policy :

1. In the handwriting assignment, please submit the pdf file. (HW2_student_id_ Handwriting.pdf)
2. In the programming assignment, the code and report (HW2_student_id_ Programming.pdf) should be compressed into a ZIP file and uploaded to eeclass website. Also, please write a README file to explain how to run your code and describe related characteristics used in your report. The report format is not limited.
3. You are required to finish this homework with Python 3. Moreover, built-in machine learning libraries or functions (like sklearn.linear_model) are NOT allowed to use. But you can use dimension reduction functions such as *sklearn.decomposition.PCA*.
4. Discussions are encouraged, but plagiarism is strictly prohibited (changing variable names, etc.). You can use any open source with clearly mentioned in your report. If there is any plagiarism, you will get 0 in this homework.

Submission :

Please follow the following format and naming rules when submitting files.

1. HW2_student_id_ Handwriting.pdf
2. HW2_student_id.zip
   |----HW2_student_id_ Programming.pdf
   |----README.txt
   |----HW2.py (only .py)

- You need to upload both HW2_student_id_ Handwriting.pdf and HW2_student_id.zip to eeclass website.

Part 1. Handwriting assignment : (30%)

1. (10%)

Show that the logistic sigmoid function $y = \sigma(x) = \frac{1}{1+e^{-x}}$ satisfies the following properties:

(a) $\frac{\partial\sigma(x)}{x} = \sigma(x)(1 - \sigma(x))$ (3%)

(b) $\sigma(-x) = 1 - \sigma(x)$ (3%)

(c) $x = \sigma^{-1}(y) = ln(\frac{y}{1-y})$ (4%)

2. (10%)

Given a binary classification network $\hat{y}_n = \sigma(W^T\phi_n)$, where ϕ_n, W, and σ are input, weights, and sigmoid function. We use Binary Cross-Entropy Loss function with a label $y_n \in \{0, 1\}$ to optimize it as

$$L(W) = -\sum_{n=1}^{N}\{y_n ln(\hat{y}_n) + (1 - y_n)ln(1 - \hat{y}_n)\}$$

By making use of the result $\frac{\partial\sigma(x)}{x} = \sigma(x)(1 - \sigma(x))$. Show that the derivative of L(W) is given by

$$\nabla L(W) = \sum_{n=1}^{N}(\hat{y}_n - y_n)\phi_n$$

3. (10%)

(a) In question 2, we show the loss function $L(W)$ for binary classification problems with a logistic-sigmoid output $\hat{y}_n = \sigma(W^T\phi_n)$ so that $0 \leq \hat{y}_n \leq 1$. Please derive an appropriate loss function if we consider a network with an output $-1 \leq \hat{y}_n \leq 1$ with a label $y_n \in \{-1, 1\}$. (5%)

(b) What would be a good choice of activation function for the output (5%)

Part 2. Programming assignment : (70%)

In this part, you need to build a neural network to identify three different types of fruit images and classify them into three classes. Therefore, you need to implement a feedforward network and utilize the backpropagation algorithm to update the weights through loss functions. Note that you have to implement the "network" and "backpropagation" by writing code yourself instead of using machine learning libraries, such as Tensorflow and PyTorch. For reference, you can see the Colab notebook for expected results and search Github resources for related implementation about backpropagation algorithms. But please don't plagiarize.

Fig. 1: Illustration of the dataset about three kinds of fruits.

Data

Please refer to Google drive. The dataset is encapsulated in Data.zip, which contains Data_train and Data_test for training and testing. Three folders named Carambola, Lychee, and Pear under two folders contain three classes of fruit as shown in Fig. 1. In the partition for training, there are 490 images per class, you can partition this data to form the validation set yourself, and test the performance of your model through the testing data. Note that the testing data may not be used to train your model.

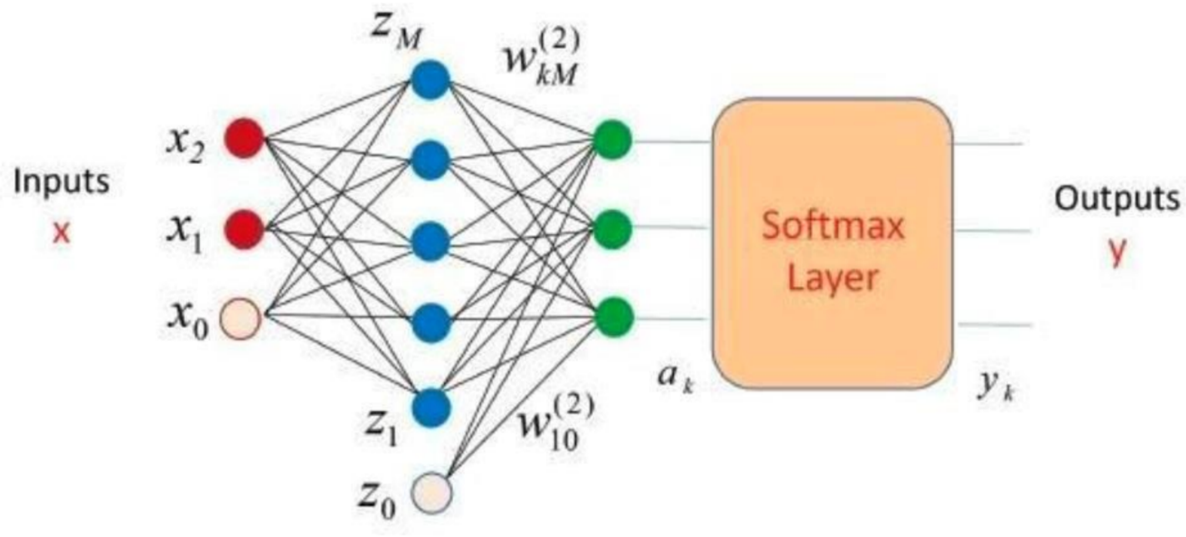

Fig. 2: Two-layer neural network with three inputs nodes and output nodes.

Problem Description

Please build a two-layer neural network (as in Fig. 2) to classify three types of images in the given dataset. More detailed steps are as follows:

1. Firstly, do dimensionality reduction using principal component analysis (PCA), which was introduced in Sec. 12.1 from the PRML textbook. Then, please utilize PCA to map the images in the data down to 2 dimensions, i.e. each image have only two floats as its principal components. You you can use any toolkits to do PCA. Finally, build a two-layer neural network with any number of hidden units chosen yourself and Train the weights using stochastic gradient descent. You need to implement the backpropagation algorithm to evaluate the gradient. Note that the number of input nodes in the neural network here is three, corresponding to the two principal components and the bias.
2. In the 2nd part, please build a three-layer neural network to do the same task. In addition, show the decision regions and discuss the performance difference compared with the network in part (a).

About Report

For both two-layer and three-layer neural network:

1. (30 points) Describe your model architecture details and PCA method.
2. (10 points) Show your test accuracy.
3. (10 points) Plot training loss curves.
4. (20 points) Plot decision regions and discuss the training / testing performance with different settings designed by yourself.