Homework 2

Ex 1: (Computer Vision: Algorithms and Applications (2020 draft) Ex5.1)

Activation and weight scaling Consider the two hidden unit network shown in Figure 5.52 which uses ReLU activation functions and has no additive bias parameters. Your task is to find a set of weights that will fit the function

$$y = |x_1 + 1.1x_2|.$$

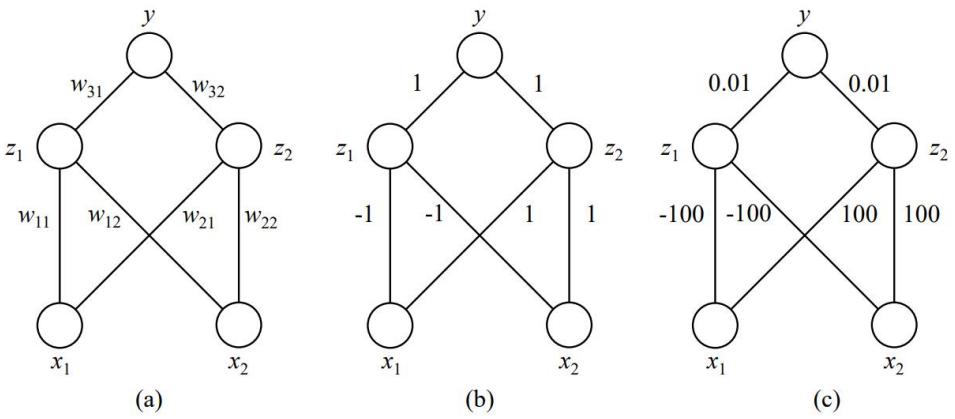

Figure 5.52 *Simple two hidden unit network with a ReLU activation function and no bias parameters for regressing the function $y = |x_1 + 1.1x_2|$: (a) can you guess a set of weights that would fit this function? (b) a reasonable set of starting weights; (c) a poorly scaled set of weights.*

1. Can you guess a set of weights that will fit this function?
2. Starting with the weights shown in column b, compute the activations for the hidden and final units as well as the regression loss for the nine input values $(x_1, x_2) \in \{-1, 0, 1\} \times \{-1, 0, 1\}$.
3. Now compute the gradients of the squared loss with respect to all six weights using the backpropagation chain rule equations (5.65–5.68) and sum them up across the training samples to get a final gradient.

$$e_i = \frac{\partial E_n}{\partial s_i} = h'(s_i)\frac{\partial E_n}{\partial y_i}, \tag{5.65}$$

$$\frac{\partial E_n}{\partial w_{ij}} = x_{ij}\frac{\partial E_n}{\partial s_i} = x_{ij}e_i, \tag{5.66}$$

$$\frac{\partial E_n}{\partial b_i} = \frac{\partial E_n}{\partial s_i} = e_i, \quad \text{and} \tag{5.67}$$

$$\frac{\partial E_n}{\partial x_{ij}} = w_{ij}\frac{\partial E_n}{\partial s_i} = w_{ij}e_i. \tag{5.68}$$

4. What step size should you take in the gradient direction, and what would your update squared loss become?
5. Repeat this exercise for the initial weights in column (c) of Figure 5.52.

6. Given this new set of weights, how much worse is your error decrease, and how many iterations would you expect it to take to achieve a reasonalbe solution?
7. Would batch normalization help in this case?

Ex2： Choose one of the deep learning frameworks in Python (e.g. PyTorch, Tensorflow, etc.) to build a convolutional neural network (CNN) for the classification task. Make observations on the effect of different choices of hyperparameter on the performance of the model on the mnist dataset. The choices include CNN structure, kernel size, stride, loss function, optimization method, learning rate, batch size, etc. Oroginal data were downloaded from http://yann.lecun.com/exdb/mnist/.

Ex3： Use Python, MATLAB, or other programming languages to reproduce Harris algorithm for corner detection，with input images of your choice. Varying the value of α and observing the effect of the parameter α on the effect of corner point detection, provided that other conditions remain unchanged.