ECE 657 Assignment 1

Spring 2022

Problem 1

Download and read the technical paper titled *On Convergence Proofs for Perceptrons* (link) by Albert Novikoff which provides a proof for the convergence of perceptrons in a finite number of steps. After understanding the paper, restate the proof and explain each step in your way (be clear and concise).

Problem 2

The three-layer network in Fig. 1 divides the plane with three lines forming a triangle. Calculate the weights that will give a triangle with its vertices at (x,y) coordinates (0,0), (1,3), and (3,1). Remember that a single perceptron can act as a linear separator. Using this idea, create the triangle with the vertices given. Assign a class to be inside the triangle and the other one to be outside the triangle and try to separate them using 3 perceptrons as shown in the figure below.

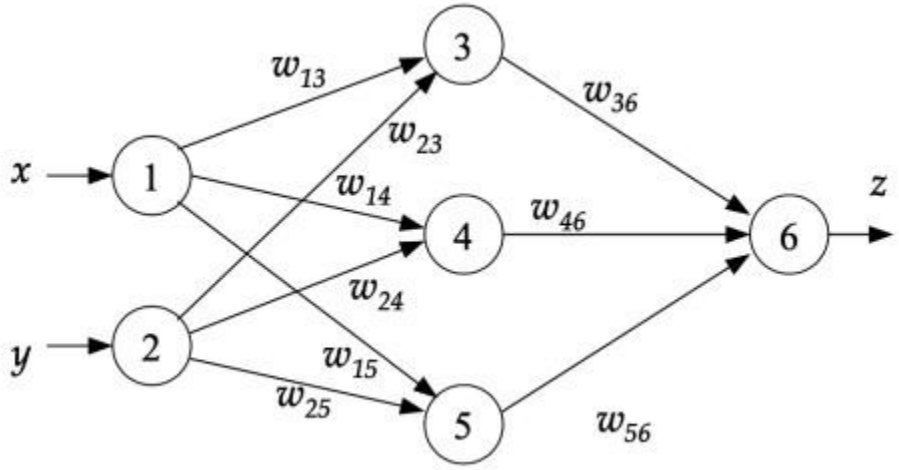

Figure 1: Structure of the three-layered network of problem 2

Problem 3

The normalized Widrow Hoff learning rule also known as the normalized least mean square learning rule is expressed as:

$$\Delta w^{(k)} = \eta(t^{(k)} - w^{(k)}x^{(k)})\frac{x^{(k)}}{||x^{(k)}||^2}$$

Where $\Delta w^{(k)} = w^{(k+1)} - w^{(k)}$ represents the weight vector change from iteration (k) to iteration (k+1), $||x^{(k)}||$ is the Euclidean norm of the input vector $x^{(k)}$ at iteration (k), $t^{(k)}$ is the target at iteration (k) and η is a positive number ranging from 0 to 1: $0 < \eta < 1$.

Show that if the same input vector $x^{(k)}$ is presented at iteration $(k + 1)$, then the weight vector decreases by factor $(1 - \eta)$ going from iteration (k) to iteration (k+1). That is: $\Delta w^{(k+1)} = (1 - \eta)\Delta w^{(k)}$

Problem 4

In this problem, you will implement the backpropagation algorithm and train your first multi-layer perceptron to distinguid between 4 classes. You should implement everything on your own **without** using existing libraries like Tensorflow, Keras, or PyTorch.

You are provided with two files "train data.csv" and "train labels.csv" The dataset contains 24754 samples, each with 784 features divided into 4 classes (0,1,2,3). You should divide this into training, and validation sets (a validation set is used to make sure your network did not overfit). You will then provide your model which will be tested with an unseen test set.

Use one input layer, one hidden layer, and one output layer in your implementation. The labels are one-hot encoded. For example, class 0 has a label of [1, 0, 0, 0] and class 2 has a label of [0,0,1,0]. Make sure you use the appropriate activation function in the output layer. You are free to use any number of nodes in the hidden layer. You need to provide one single function that allows us to use the network to predict the test set. This function should output the labels one-hot encoded in a numpy array.

Your work will be graded based on the the successful implementation of the backpropagation algorithm, the multi-layer perceptron, and the test set accuracy. You may want to thoroughly comment your implementation to allow us to easily understand it.