ELE888 Intelligent Systems
Lab 3: Multilayer Neural Networks

Objective

To implement multi-layer neural network using the back propagation algorithm to classify linearly non-separable data.

Background

Multi-layer neural network (MNN) implements the linear discriminant functions however in a feature space where the input patterns are mapped non-linearly. Neural networks are quite powerful and easily realizable using simple algorithms where the form of non-linearity can be learned from the training data. One of the most popular methods for training the MNN is based on the gradient descent in error commonly known as backpropagation algorithm. Figure 1 shows an example of a 3 layer neural network.

The task is to estimate the weight vectors (w_{ji} and w_{kj}) between the input-hidden layer and the hidden-output layer for a given activation function. The pseudo code (11 steps) of the batch backpropogation algorithm is outlined on the following page.

Where: The sensitivity of unit k is given by

$$\delta_k = (t_k - z_k) \cdot f'(net_k) \tag{1}$$

and the sensitivity for a hidden unit is given by

$$\delta_j = f'(net_j) \sum_{k=1}^{c} w_{kj} \delta_k \tag{2}$$

The (net_j) and (net_k) are the net activation and $f(net_j)$ and $f(net_k)$ are the non-linear activation functions. The derivative of the function $f(\cdot)$ is shown as $f(\cdot)$.

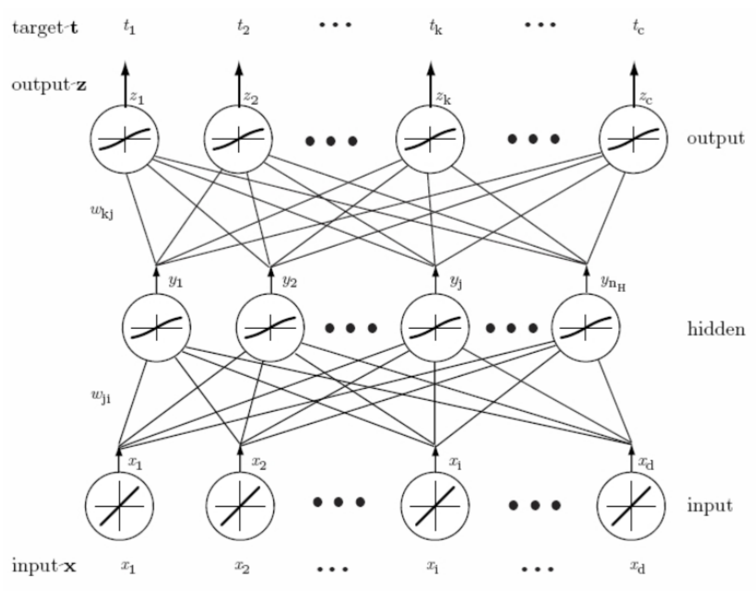

Figure 1: An $d - n_H - c$ fully connected 3 layer NN (Bias not shown)

Back Propagation Algorithm:

```
1. begin initialize network topology(# hidden units), $\mathbf{w}$, criterion $\theta$, $\eta$, $r \leftarrow 0$
2. do $r \leftarrow r + 1$ (increment epoch)
3. $m \leftarrow 0$; $\Delta w_{ij} \leftarrow 0$; $\Delta w_{jk} \leftarrow 0$
4. do $m \leftarrow m + 1$
5. $x_m \leftarrow$ select pattern
6. $\Delta w_{ij} \leftarrow \Delta w_{ij} + \eta \delta_j x_i$; $\Delta w_{jk} \leftarrow \Delta w_{jk} + \eta \delta_k \ y_i$
7. until $m = n$
8. $w_{ij} \leftarrow w_{ij} + \Delta w_{ij}$; $w_{jk} \leftarrow w_{jk} + \Delta w_{jk}$
9. until $\|\nabla J(\mathbf{w})\| < \theta$
10. return $\mathbf{w}$
11. end
```

Laboratory Exercises

- Construct a 2-2-1 neural network using the batch backpropogation algorithm for solving the classical XOR problem. The two inputs are given as $x_1 = [\ -1 \quad -1 \quad 1 \quad 1\ ]$ and $x_2 = [\ -1 \quad 1 \quad -1 \quad 1\ ]$. The targets (correct output) are $t = [\ -1 \quad 1 \quad 1 \quad -1\ ]$. Use a $\eta = 0.1$ and threshold $\theta = 0.001$. Assume the following sigmoid activation function (for both hidden and output units)

$$f(x) = a\mathbf{tanh}(bx) \tag{3}$$

  where $a = b = 1$.

- Verify that the computed final weight vectors satisfy the XOR operation. Plot the learning curve and note the number of epochs needed for convergence.

- Repeat the above for the **'Wine'** data set from the UCI repository: `http://archive.ics.uci.edu/ml/datasets/Wine`, using test samples from Class 1 (ω_1) and Class 3 (ω_2) - using only features $x_1 = Alcohol$ and $x_2 = MalicAcid$. Use class target labels ($\omega_1 = 1$, $\omega_2 = -1$).

- Compute the classification accuracy for the given data. Plot the learning curve and note the number of epochs needed for convergence

What to Submit?

- The programs should be demonstrated to the TA.

- A report including all the required plots, discussions, and your final observations and conclusions.