Introduction:

The design of the artificial neuron is based upon the nerve cell of brain. The general structure of the neural network is as shown below:

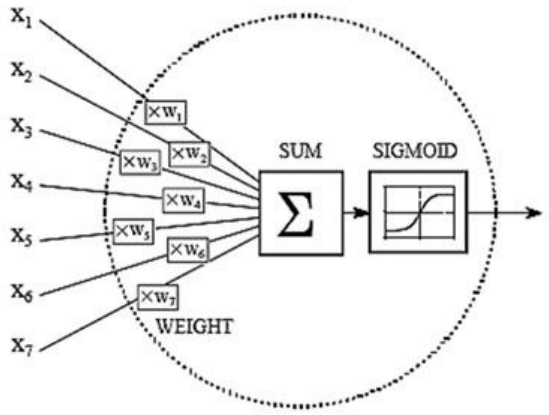

Fig: Diagram Showing General Structure of a Neural Network

The structure has the following parts:

- The inputs in the form of X's
- The weights as W's
- Summation function to perform the weighted sum
- The sigmoid function shown here represents the activation function which can be any other relation depending upon the requirement
- There can be biases associated with the input if it is required that the sigmoid function does not pass through origin

Levenberg-Marquardt Method:

Without going into much detail of the method, a general idea about how the LM method works, has been presented in this topic. It is supervised method of training.

- The LM method involves the solution of the following equation:

$$(\boldsymbol{J^t J} + \boldsymbol{\lambda I})\boldsymbol{\delta} = \boldsymbol{J^t E}$$

Here J is the Jacobian matrix for the system, λ is the Levenberg's damping factor, δ is the weight update vector that is to be calculated and E is the error vector. The J^tJ matrix can be approximated as the Hessian matrix.

- The damping factor is adjusted at each iteration to carry on the optimization process. If E is decreasing rapidly, λ can be reduced, thus making LM method equivalent to Gauss-Newton Algorithm. On the other hand, if E does not decrease rapidly, then λ can be increased and the method gets closer to Gradient Descent.

Back-Propagation (BP) Method:

It is a supervised method. The Backpropagation method involves two phases of propagation and updation. The propagation phase involves the following:

- The propagation of the input to generate the output.
- The backward propagation of the output through the layers in order to generate the weight update vector for all the layers involved.

Scaled Conjugate Gradient (SCG) Method:

This is one of the BP method. It is also supervised method of training. In the case where large number of weight vectors are involved, only the optimization methods can be used for the adjustments.

Levenberg-Marquardt Algorithm for Training Used in the Project:

The LM method is slower than backpropagation method in the sense that the number of operations to be performed are quite large and it consumes a large amount of memory. In order to solve this problem, it is suggested to make different neurons perform different tasks. Thus the network is divided into different neighbouhoods. Thus the LM method can be applied to each neighbourhood and thus the memory and time can be saved.

a. Training Steps:

Thus in order to modify the weights of a given neuron, we only need to have the information about its neighbourhood. This process is carried out in the same steps as in LM method with one extra step of defining the neighbourhoods.

- Define the network structure
- Initialize the weights
- Define the neighbourhoods
- Apply LM method on the neighbourhoods to train the network
- Evaluate the network error with the modified weights
- Stop the learning process when the terminating condition is met otherwise apply LM method again for training

b. Problems:

The problems faced in this method are:

- How to define the neighbourhoods?
- If the neighbourhoods can overlap or not and if yes, then how much
- How to select the neighbourhood to be trained?

c. Parameters for LM Method in MATLAB:

The values of the parameters used for the LM method are the default values already defined in MATLAB. They can be changed for further analysis. Some of these values are:

net.trainParam.epochs	1000	Maximum Number of Epochs to Train
net.trainParam.goal	0	Performance Goal
net.trainParam.max_fail	6	Maximum Validation Failures
net.trainParam.min_grad	1e-7	Minimum Performance Gradient
net.trainParam.mu	0.001	Initial mu
net.trainParam.mu_dec	0.1	mu Decrease Factor
net.trainParam.mu_inc	10	mu Increase Factor
net.trainParam.mu_max	1e10	Maximum mu
net.trainParam.time	inf	Maximum Time to Train in Seconds

Table: Default Parameters for LM Method in MATLAB