Classification and Regression using Linear Networks, Multilayer Perceptrons and Radial Basis Functions

Vikas Chandrakant Raykar

***Abstract*— The aim of the assignment was to implement a Linear network (LN) for a 3 class pattern classification problem. Training was done using the LMS algorithm. A 3-h-1 Multi Layer Perceptron (MLP) was implemented for a two class problem. Back propagation was used for training. The network was then pruned using Optimal Brain Surgeon. A Radial Basis Function (RBF) network using Inverse multiquadratic basis function was implemented for function approximation. The training was done using LMS algorithm. Also an MLP was designed and implemented for printed numeral recognition.**

I. Input Standardization

Before using the training data for training the network the data needs to be standardized. This involves subtracting the mean from all the data and multiplying by the square root of the inverse of the covariance matrix. The square root can be found since the covariance matrix is symmetric and can be diagonalized.This basically performs a coordinate transformation of a distribution to one which has zero mean and the covariance matrix as the identity matrix. Let X be a random vector. Then the whitening transform of X i.e. Y is given by $Y = \sum^{-1/2}(X - m)$ where μ is the mean vector.

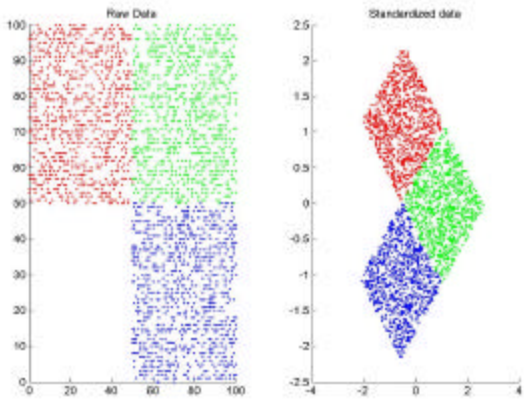

Figure 1 shows the raw data for 2 dimensional feature vector for a three class problem and also the standardized data. The mean is zero and the coordinates are transformed

This report was written for ENEE 739Q SPRING 2002 as a part of the course project. The author is a graduate student at Department of Electrical Engineering, University of Maryland, and College Park MD 40742 USA.

II. Error Measure

Once the classification is done the method needs to be evaluated. We use the average mean squared error as the error measure. Since most of the methods in this project are based on minimizing the mean squared error it is not necessary to make use of probability of error criterion to compare the method for various parameters. To evaluate the mean squared error we select a set of samples independent from the training set called as the *test set*. Since in our simulations the range of the components of the feature vector is small the test set is considered as the entire image itself i.e. 100x100 points.

III. Stopping Criterion

Most of the iterative algorithms used for minimizing the mean square error require some stopping criterion. The stopping criterion can be when the norm of the gradient or the mean square error itself goes below a certain preset threshold. This is calculated on the training set. Alternatively a validation set independent of the training set can be selected and training can be stopped when the mean squared error over the validation set starts increasing. In my simulations in order to set proper thresholds the network is over trained i.e. to observe the behavior of the error the use can enter the number of iterations. Once the threshold is set the stopping criterion can be implemented in the program.

IV. Pattern Classification Using a Linear Network

A 100x100 image is created and divided it into four quadrants, as shown in Figure 1. The problem is a 2D pattern classification problem in which the samples are drawn equiprobably from the three colored quadrants Red, Green and Blue. Each pattern is a set of (row,col) pixel coordinates. N/3 training samples were selected uniformly from each of the four quadrants. Using a total of *N* training samples, a linear *3-3* network was used to classify the samples into the three classes (R G B).

One way to classify a 3 class problem is to use two 2 class problems, where the ith problem is solved by a linear discriminant function that separates the points assigned to class i from those not assigned to class i. But this approach leads to regions where the classification is undefined. Figure 2 shows such a classifier. Each of the two class problem was

trained using the LMS algorithm. There is confusion where the decision regions of two classes overlap.

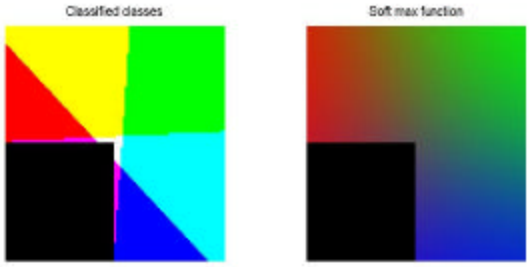

Figure 2. Classification based on class i and not class i approach. The first image is the actual output and the second is using the softmax function. The regions other than R G and B indicate the overlap between two adjacent classes. The white region is where all the three region overlap. The network was trained using N=600 alpha (n) =alpha (0)/n alpha (0) =0.01and 2000 iterations. The training sample at each iteration was chosen randomly.

To overcome this problem we define 3 linear discriminant functions g1(x), g2(x) and g3(x) and assign x to class i if gi(x) > gj(x) for all j not equal to i. The resulting classifier is known as a Linear machine. The weights are got using the LMS procedure for minimizing the mean squared error

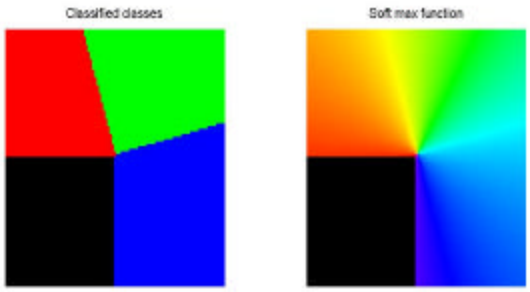

Figure 3 shows the decision regions for a Linear machine. The decision regions are convex and there is no region of confusion. Also note that since we are using our criterion function as minimizing the mean squared error even though if there exists a separating plane between two classed the LMS algorithm need not necessarily give the perfect separating plane. It just gives the plane based on the minimum mean squared error. The network was trained using N=600 alpha(n)=alpha(0)/n alpha(0)=0.0001and 2000 iterations. The training sample at each iteration was chosen randomly.

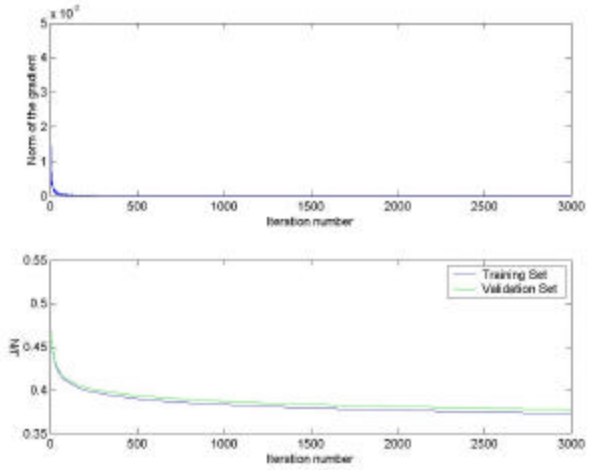

Figure 4 The first plot shows the norm of the gradient as a function of the iteration number. The second plot shows the average mean squared error averaged over all the training samples and also for the validation set as a function of the iteration number. It can be seen that the weights start to converge after about 1000 iterations. Also the curves for validation set and the training set follow each other The network was trained using N=600 alpha(n)=alpha(0)/n alpha(0)=0.0001and 3000 iterations. 90 samples were chosen in the validation set. The training sample at each iteration was chosen randomly .

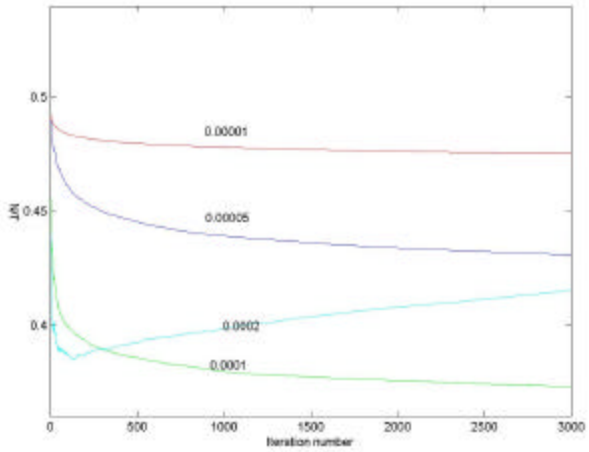

Figure 5 This plot shows the mean squared error averaged over all the training samples as a function of the iteration number for different values of alpha(0). It can be seen that if alpha(0) is greater than 0.0001 then it starts diverging The network was trained using N=600 alpha(n)=alpha(0)/n and 3000 iterations. The training sample at each iteration was chosen randomly .

The following table shows the mean square error averaged over the test set(.i.e. the entire image) for different initial learning rates. The network was trained using N=600 alpha(n)=alpha(0)/n and 3000 iterations. The training sample at each iteration was chosen randomly . As the step size decreases the number of iterations required to reach the

minimum increases and so for a fixed number of iterations the error increases as the initial learning rate decreases. However it cannot be increased beyond a limit whence it starts diverging.

Alpha(0)	Average MSE
0.0002	0.5158
0.0001	0.1754
0.00005	0.2182
0.00001	0.3206

V. Pattern Classification Using A Multi Layer Perceptron

In this problem we use an MLP to classify two classes as shown in Figure6. The feature vector is a two dimensional vector corresponding to the row and column pixel coordinates. The white region in the shape of an F corresponds to one class and the black region(background) corresponds to the second class. A 3 layered network (input layer, hidden layer and the output layer.) was designed. The input layer had 3 units the first unit corresponding to the bias and the other two to the feature vector. The hidden layers have h hidden units with a bias.(i.e. h+1 units). The output layer has one unit. -1 and +1 were used as the target values. -1 corresponds to the background and +1 corresponds to the foreground. The network was trained using back propagation algorithm using N training samples. Momentum was also incorporated in the training. The training can be made faster by adding a momentum term. Momentum simply adds a fraction m of the previous weight update to the current one. When the gradient keeps pointing in the same direction, this will increase the size of the steps taken towards the minimum. When the gradient keeps changing direction, momentum will smooth out the variations The N training samples were presented in a random fashion. One presentation of N training samples is called a epoch. The training was done over several epochs. The parameters for the a tanh(ßx) activation function were chosen as a=1.716 and ß=2/3.

Figure 6 shows the decision regions for N=1000 training samples learning rate alpha=0.01 no momentum and 50 hidden layers. The weights converged to a local minima in 300 epochs

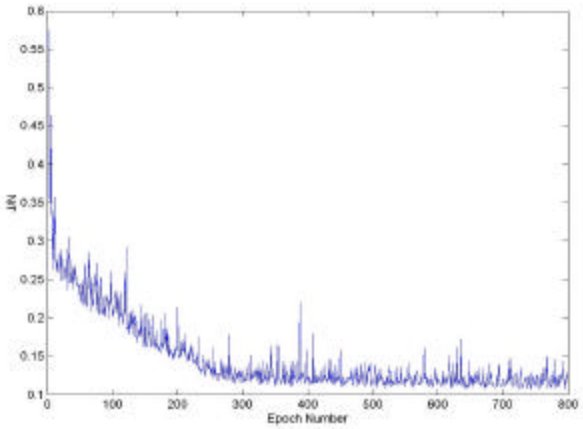

Figure 7 shows the corresponding mean square error averaged over the training samples as a function of epoch number . It can be seen that the local minimum is reached after 300 epochs. (N=1000 alpha=0.01 no momentum and 50 hidden layers)

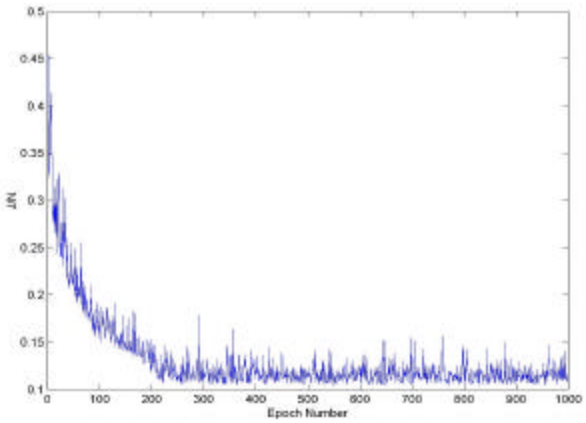

Figure 8 The following plot shows the error as a function of number of epochs with momentum of 0.4 N=1000 h=50 learning rate alpha=0.01. The error reaches a local minima in around 200 epochs. The convergence is faster with momentum.
If the number of samples used reduces then more number of epochs is needed for convergence.

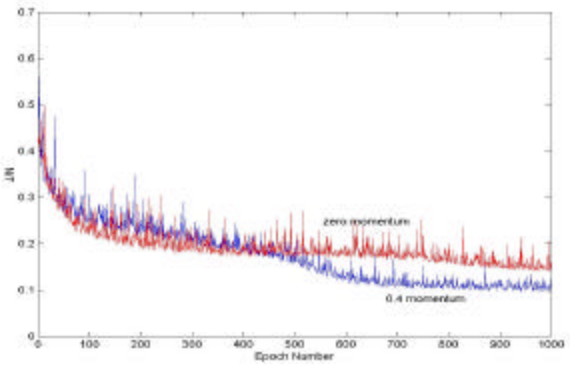

Figure 9 The following plot shows the error as a function of number of epochs with momentum of 0.4 N=200 h=30 learning rate alpha=0.01. The error reaches a local minima in greater than 700 epochs. With momentum we get a lower error tan without momentum in this case.

Figure 10 shows the decision regions for N=200 training samples learning rate alpha=0.01 0.4 momentum and 30 hidden layers. The weights converged to a local minima in 700 epochs

Figure 11 shows the decision regions for N=1000 training samples learning rate alpha=0.01 no momentum and 10 hidden layers. The weights converged to a local minima in 300 epochs . Even with 10 hidden layers we get pretty good results.

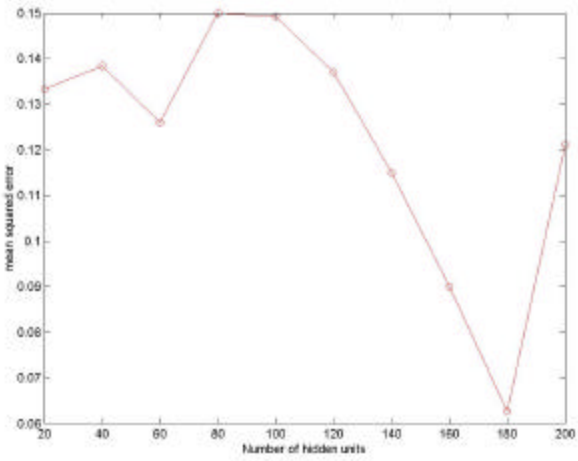

Figure 12 shows the average mean square error averaged over the test data i.e. the entire image for different number of hidden units and fixed learning rate N=1000 alpha=0.01 momentum=0.4. It is observed that as the number of hidden units increase the error decreases.

Optimal Brain Surgeon:

Once the MLP has been trained to convergence the number of hidden units can then be reduced using the optimal brain surgeon algorithm. The basic idea being to eliminate the weight which causes the least increase in the mean squared error when the weight is made zero. As can be seen from the network any hidden unit can be eliminated if all the weights connecting the hidden unit to the output are zero or all the weights connecting the input to the hidden unit are zero or both. The OBS was implemented but there was no significant decrease in the weights. Since running the program takes a lot of time I am still experimenting with the program.

VI. Function Approximation Using An RBF Network

In this problem we need to approximate the function given in the previous case(i.e an F) using radial basis function network. We have N training samples chosen randomly. We select h of them as function centers arbitrarily. The input is the row and the column coordinates of the image. Used inverse multiquadratic basis functions of the form $1/(x^2+s^2)$. s was chosen suitably as $s = d_{\max} / \sqrt{2h}$ where $d_{\max}$ is the maximum distance between the centers. The weights were determined using the LMS algorithm. For training the inputs were given in order i.e. first samples belonging to foreground were presented and then samples belonging to the background were presented.

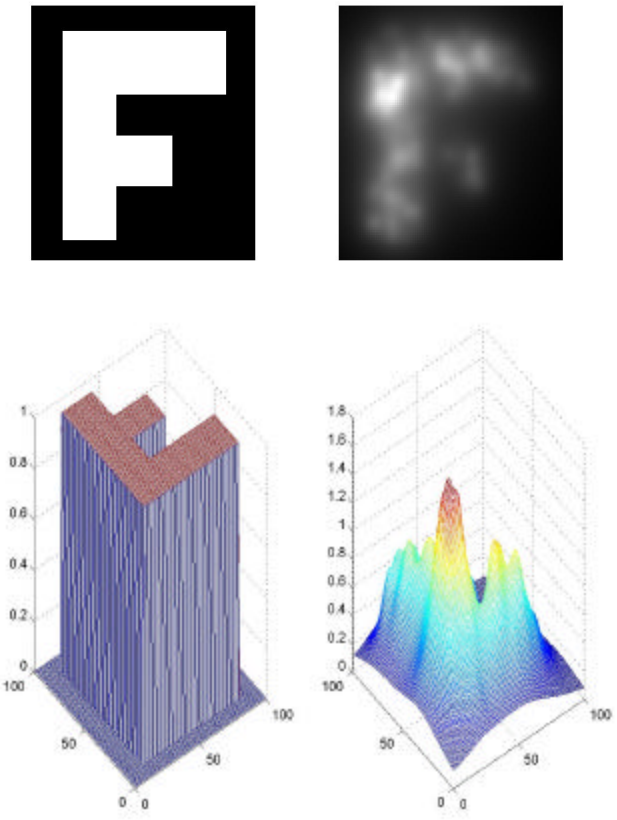

Figure 13 shows the function approximated for N=500 training samples 100 centers and learning rate of 0.01 for the LMS algorithm. The plot is or centers chosen to lie within the F region.

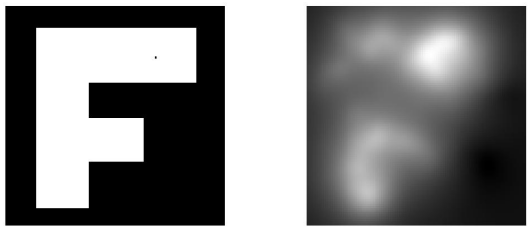

Figure 14 the following plot shows the result for N=500 training samples 100 centers and learning rate of 0.01 for the LMS algorithm. But the centers are chosen arbitrarily throughout the image.

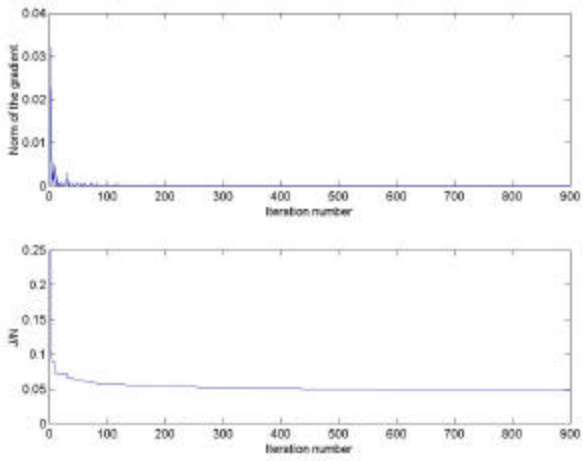

Figure 14 shows the norm of the gradient and the mean square error for the LMS algorithm as a function of iteration number.

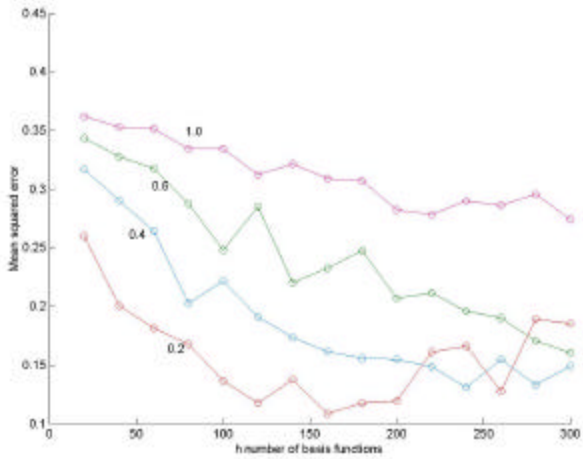

Figure 15 shows the mean squared error averaged over the test data i.e. the entire data set as a function of the number of basis functions used N=600 training samples.

VII. Optical Character Recognition

I have implemented a OCR based on a simple 3 layer MLP with (256+1 bias)(16x16 image) units in the input, h hidden layers and 10 outputs. The network still needs to be trained. The data set has to be generated first. I am still working on generating the data set for training.

References

[1] *Pattern Classification (2nd ed)*Richard O. Duda, Peter E. Hart and David G. Stork Wiley, New York, NY 2000
[2] Haykin, S. *"Neural Networks: A Comprehensive Foundation"*
[3] ENEE739 Q SPRING 2002 class notes

Vikas Chandrakant Raykar is a graduate student at the University of Maryland College Park, MD 20742 USA (telephone: 301-405-1207, e-mail: vikas@umiacs.umd.edu).