PROGRAMMING ASSIGNMENT 2

Utkarsh Pratap Singh Jadon, jadon.1

CSE 5526: PA-2: Report

1. INTRODUCTION

Programming Assignment 2 required the implementation of an RBF network for a one-dimensional output variable, and Gaussian basis functions. A set of 75 data points had to be generated by sampling the function 0.5 + 0.4sin(2πx) with added uniform noise in the interval [-0.1, 0.1], and x values taken randomly from uniform distribution in the interval [0.0, 1.0]. Gaussian centers had to be determined by K-means algorithm, and the variances are set accordingly. Least Means Square (LMS) rule is used to update the network weights.

Number of bases were varied in the range of 2, 4, 7, 11, 16. Two values of learning rates were used, which were 0.01 and 0.02. Training for each of the above 10 combinations was stopped after 100 epochs. Later, similar implementation was carried out, but with universal variance for all the clusters.

2. METHODOLOGY

Firstly, the 75 data points are generated by sampling the given function. This data is then used to find the center of the clusters, and the variances, using K-means algorithm. Variances can be set in two ways; determine variance for each gaussian center, or set same universal variance to all gaussian centers. Different values of bases and learning rates are taken, and training in each case is stopped after 100 epochs.

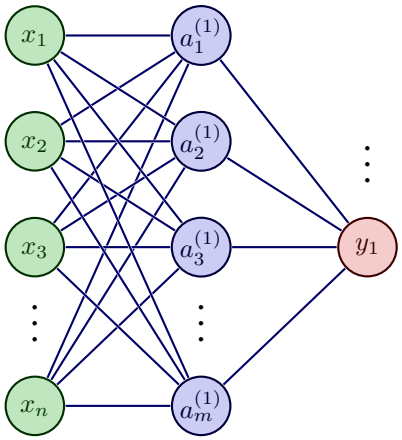


Fig. 1. RBF Network

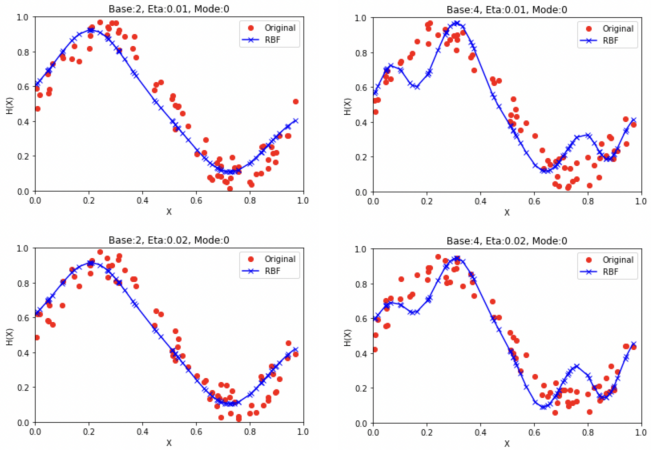
Figure 1 shows the structure of a Radial Basis Function Network Architecture. There are 75 neurons in the input layer, corresponding to 75 input data points. There is one hidden layer, which consists of neurons depending on the number of bases required, which in our case varies between 2, 4, 7, 11, and 16. Output layer has one neuron, connected to the hidden layer, since we need a one dimensional output variable. There are no weights in the hidden layer, but output layer has weights, which are updated using LMS rule.

$$y = \sum_{n=1}^{m} w_n * \varphi(x) \tag{1}$$

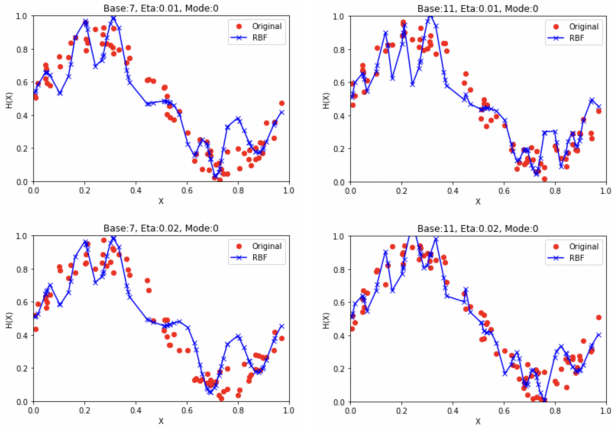
Two models were trained; one with different variance for each cluster, and one with universal variance for all clusters. Once the basic RBF neural network was created using random initialization of weights and biases for the output layer, centers of the gaussian functions were calculated. In the first case, variance for each cluster was calculated separately, whereas in the second case, a universal variance was assigned to all the clusters. These two values were used in the forward and backward propagation for weight update. Results section shows the performance of both models, with different number of bases, and different choices of learning rates.

3. RESULTS

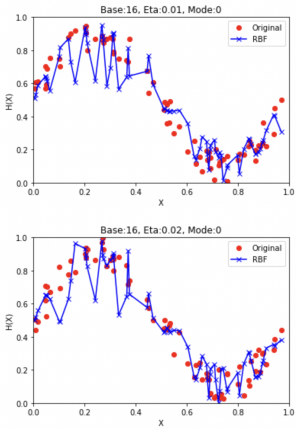
3.1. RBF with different variances



(a) (b)



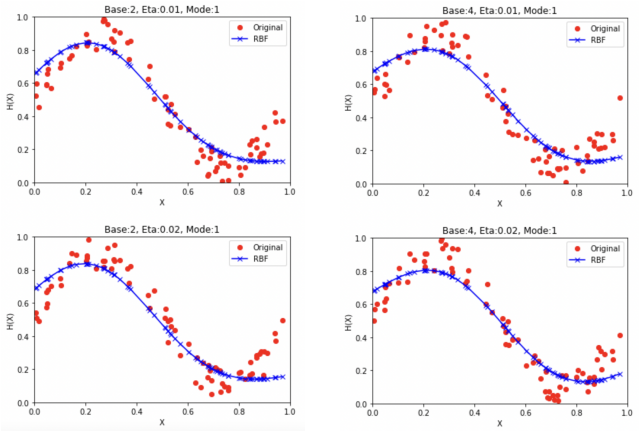
(c) (d)



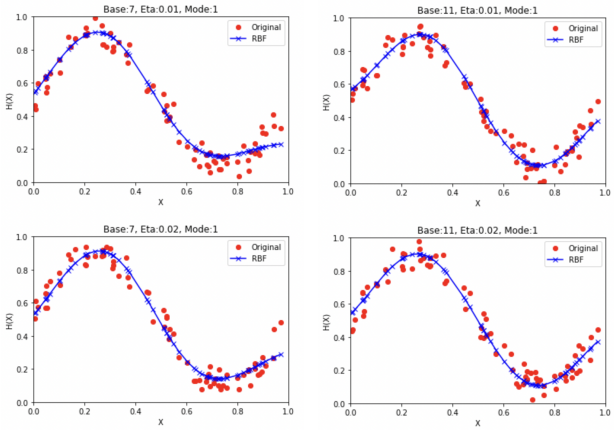
(e)

Fig. 2. (a)-(e) shows RBF approximation for K = 2, 4, 7, 11, 16 using different variances and $\eta = 0.01$ and 0.02 respectively

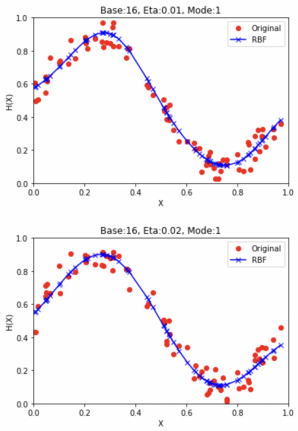
3.2. RBF with universal variance



(a) (b)



(c) (d)



(e)

Fig. 3. (a)-(e) shows RBF approximation for K = 2, 4, 7, 11, 16 using universal variance and $\eta = 0.01$ and 0.02 respectively

4. CONCLUSION

A two-layer RBF network model was developed with one hidden layer and one output layer. Number of neurons in the hidden layer depended on the number of bases functions. Center and variances of these functions were calculated using K-means algorithm. After performing forward and backward propagation, LMS rule was used to update the weights of output layer. Models were trained for 100 epochs. From Figure 2, it can be seen that increase in the number of bases function leads to overfitting of data. High learning rate yielded more smooth results as compared to lower learning rate. RBF network model with universal variance have similar approximation distribution for different bases, as shown in Figure 3. From these plots, it can be inferred that having higher number of bases function leads to better approximation (sometimes overfitting), and higher learning rate yields more smoothed out apporximated function.