F-007

An Efficient Construction of RBF Network Based on Training by SOM

Kazuhiko Yamashita[†] **Goutam Chakraborty**[†] **Yuji Mizuno**[†]

1. Abstract

In this work we proposed to train both input to hidden as well as hidden to output layer weights of a RBF network using Self-Organizing Maps (SOM) training. Initially a two dimensional SOM is trained. Once SOM training is over, the SOM input to output weights determine the RBF hidden units' location in problem feature space. Next, for every individual sample the winner SOM output is identified, and the label (class) of the sample is tagged with SOM outputs. This tag information is used to decide the connection weights between RBF middle layer nodes to output nodes. Thus by just executing SOM the RBF network is constructed. The performance is compared with multilayer Perceptron trained with error back propagation.

2. Introduction

2.1 Self-Organizing Maps (SOM)

Artificial neural network (ANN) is a powerful tool for solving clustering and the pattern recognition problems. One of the stable clustering tool is Self-Organizing Maps (SOM). Moreover, it is easy to manipulate the number of outputs in case of SOM, and still have meaningful results, and it has the property to project the data distribution in multi-dimension space in two dimension.

In addition, SOM can be used as a supervised network, as classifier too. It is possible when the different classes are well separated and the class information is available. After SOM network training converges, we input the different samples once at a time, and check the winning output. We mark the winning output same as the class of the input sample. When different classes of samples are well separated in the feature space, and there is no overlapping, we can uniquely assign class label to different SOM output units. After this phase, when an unknown sample is input, we can determine the class from the winner output and its class-label.

In case, the distribution of samples in the feature space is overlapping, we can not uniquely label the SOM output units with class information. The same output unit can sometimes win a class and some other time win another class, if they were from the overlapping region. But, this confusion can be resolved using a RBF type neural network. We will elaborate this idea in next section.

2.2 RBF Network

RBF Network (RBFN) is a three-layer feed forward network. RBFN is a supervised artificial neural network used for classification of interpolation problems. The main important point of RBFN is that its training time is much faster than error-back propagation training. On the other hand, if the number of hidden units is large, RBFN tends to get over-trained.

The name, radial basis function (RBF) network comes from the type of functions used at the hidden layer nodes, to convert the node input to output. One of the most popular function has the form of normal distribution, as shown below.

$$\phi_j = exp\left(- \frac{\|x - \mu_j\|}{2\sigma_j^2} \right)^2 \tag{1}$$

Here, μ_j is center of function, σ_j^2 the variance and x is the input vector. ϕ_j is output of the j^{th} hidden units. Physically, different hidden units take care of the region surrounding them. They are like spline functions in interpolation problems.

Finally, the input to the output units is the dot product of output of hidden units and the hidden to output weights vectors. If there are C classes, let the output vector corresponding to x as input,

$$\overrightarrow{y(\overrightarrow{x})} = \sum_{j=0}^{M} w_{kj}\ \phi_j(\overrightarrow{x}) \tag{2}$$

$$\overrightarrow{y(\overrightarrow{x})} = \overrightarrow{W}\overrightarrow{\Phi} \tag{3}$$

where M is the number of hidden units. In general, for RBFN, the output units do not use any activation function. The input-output relation at the output units is linear. In that case, to minimize the training samples error, i.e.,

$$Error + \frac{1}{2}\sum_{N}\sum_{C}\{y_k(\overrightarrow{x}^n) - t_k^n\}^2 \tag{4}$$

There is a straight calculation of the connection weights. Here, W is the weight matrix, with C rows for C-classes and M columns for connections to M hidden units. W is the solution of the following linear equation:

$$\Phi^T \Phi W^T = \Phi^T T \tag{5}$$

Here, Φ is a matrix with N rows and M columns, where N is the number of samples and M is the number of hidden nodes. The elements of first row is, $\phi_1(\overrightarrow{x}^1)$, $\phi_2(\overrightarrow{x}^1)$, $\phi_3(\overrightarrow{x}^1)\ldots\phi_M(\overrightarrow{x}^1)$. Similarly, the second row is $\phi_1(\overrightarrow{x}^2)$, $\phi_2(\overrightarrow{x}^2)$, $\phi_3(\overrightarrow{x}^2)\ldots\phi_M(\overrightarrow{x}^2)$. Thus, we have N rows for N sample data. T is the class information matrix, having N rows and C columns. As it is a linear equation, and as in most of the cases $\Phi^T\Phi$ is invertible, in general there is a solution.

We want to avoid the complex computation of Φ matrix, multiplication and inversion of big matrices. We propose a method for setting the weight matrix using SOM.

3. Proposed System

As mentioned earlier, the proposed algorithm is based on SOM. We first give a brief description of SOM training algorithm. It is one of the unsupervised learning methods. The output nodes are self organized after training. This network is also an important tool for visualization of multi-dimensional data. This network is two-layer structure and, in this paper, the output layer is two dimensional. The input to output layer nodes

[†]Dept. of Software and Information Science, Iwate Prefectural University

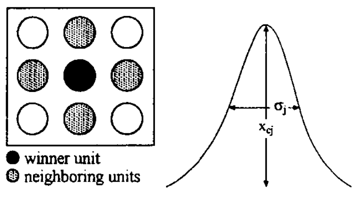

Figure 1: Decision of RBF variance

connection weights modify according to the following formula [1].

$$m_i(t+1) = m_i(t) + \alpha_i(t)h_{ci}[x(t) - m_i(t)] \qquad (6)$$

m_i is input to hidden weights, x is input value and h_{ci} is neighborhood function. t is learning epoch number, and then, α_i is learning rate. The learning rate is decreased gradually as the training proceeds. Not only the winner unit but also its neighboring units, as defined by h_{ci} are made to learn.

Once SOM training is over, the structure is retained as it is as the input and hidden layer of RBFN. The SOM outputs no longer retain its two-dimensional positional information as the middle-layer units of RBFN. After SOM training is converged, we present the training samples one by one, and note which SOM output wins. We keep a counter at all these nodes for every class. For the winner node, we add the corresponding counter by one. Once all the training samples are presented, these counter values determine the middle to output layer weights of the RBFN. If the counter corresponding to a class (output unit) is zero, there is no connection. Otherwise, there is a connection with weight equal to the counter value. At the end, the weights of all the connections to an output unit is normalized. The value of σ at a middle layer unit is also determined from the SOM result as shown in Fig. 1. Physically σ_j is decided from the average Euclidean distance of four neighboring units and x_{cj} is decided from the number of winning samples of class c.

Thus the whole RBFN is constructed from SOM result alone.

4. Experiments and Results

4.1 Data sets

We used vowel data. There are ten different types of vowel, and two formant frequencies as two features. The data is described in Table 1. There are many overlapping areas due to similarity in pronunciation of nearby vowels. We used 511 samples as training data. The number of test data of each class is sixteen.

Vowel Data (training samples: 511, test samples: 160)	
Input	Output
1. First formant 2. Second formant	1.~10. Vowel

Table 1: Vowel data

4.2 RBFN using SOM

The parameters used for SOM training and the experiment result are shown Table 2.

The total of recognition rate for test samples using the proposed RBFN is 77.5% and the training time was 25.72 second. Low recognition rate for class1 and class8 is due to strong overlapping.

SOM parameters	
Learning rate	0.05
Radius	3
Training times	10000
Result of recognition	
Class	Recognition rate (%)
1	43.75
2	87.5
3	93.75
4	75.0
5	68.75
6	100.0
7	100.0
8	50.0
9	93.75
10	62.5
Total of recognition rate	77.5
Processing Time	25.72sec

Table 2: SOM parameters and recognition rate

4.3 Results using Multilayer Perceptron

Parameter values and results using Multilayer Perceptron trained by error back-propagation is shown Table 3.

BP parameters	
Learning rate	0.1
Hidden units	30
Training times	50000
Result of recognition	
Class	Recognition rate (%)
1	87.5
2	87.5
3	56.25
4	87.5
5	62.5
6	68.75
7	100.0
8	75.0
9	87.5
10	0.0
Total of recognition rate	71.25
Processing time	9min 25.703sec

Table 3: BP parameters and recognition rate

The total recognition rate for test samples is 71.25 % and the training time is 9 minutes 25.703 second. The learning rate is decrease gradually as training progressed. Here the worst recognition was for class10. Of course, there is overlapping. Yet, different values of the recognition rate shows that there are still ways to improve the recognition rate. For BP, the training may be stuck in local minimum. But similar results were obtained when it is repeated.

5. Conclusion

In this paper, we proposed construction of RBF network based on SOM. Our proposed method gives better recognition rate than BP and it stands head and shoulders above BP in training time.

Consequently, this method is better than BP as learning machine. However, it is possible to use our proposed RBFN as the initial state, and then further training is possible for the middle to output connection weights as well as the different σ values to improve the recognition rate. Due a good starting point, we conjecture that this training will be finished quickly.

References

[1] T. Kohonen: "Self-Organizing Maps", Springer-Verlag Tokyo, pp.116, 2005.