# Linguistic Rule Extraction From a Simplified RBF Neural Network [1]

Xiuju Fu[1] and Lipo Wang[2]

[2] School of Electrical and Electronic Engineering
Nanyang Technological University
Block S2, Nanyang Avenue
Singapore 639798
Email: elpwang@ntu.edu.sg
http://www.ntu.edu.sg/home/elpwang

**Summary**

Representing the concept of numerical data by linguistic rules is often desirable. In this paper, we present a novel rule-extraction algorithm from the radial basis function (RBF) neural network classifier for representing the hidden concept of numerical data. When training the RBF neural network, we allow for large overlaps between clusters corresponding to the same class, thereby reducing the number of hidden units while improving classification accuracy. The weights connecting the hidden units with the output units are then simplified. The interval for each input in the condition part of each rule is adjusted in order to obtain high accuracy in the extracted rules. Simulations using some bench-marking data sets demonstrate that our approach leads to more accurate and compact rules compared to other methods for extracting rules from RBF neural networks.

**Keywords:** rule extraction, data mining, radial basis function neural network, classifier, concise rules

# 1 Introduction

Extracting rules to represent the concept of data is an important aspect of data mining. The task of extracting concise, i.e., a small number of, rules from a trained neural network is usually challenging due to the complicated architecture of a neural network.

Some research work has been carried out in extracting rules through RBF neural networks. In [2][4], redundant inputs are removed before rule extraction. Huber [6] selects rules according to importance; however, the accuracy is reduced with pruning. McGarry [10][11][12] extracts rules from RBF neural networks by considering the parameters of Gaussian kernel functions and weights which connect hidden units to the output layer. However, when the number of rules is small, the accuracy is low. When the accuracy is acceptable, the number of rules becomes large.

In this paper, we propose a novel technique to extract rules from the RBF neural network. First, the number of hidden units is reduced due to our modification when training the RBF neural network. The weights connecting hidden units with output units are simplified (pruned) subsequently. Then the interval for each input in the condition part of each rule is adjusted in order to obtain a high rule accuracy. We show that our technique leads to a compact rule set with desirable accuracy.

This paper is organized as follows. In Section 2, we proposed a way to obtain an efficient RBF classifier based on a modification in which large overlaps are permitted between clusters with the same class label. Our proposed algorithm to extract rules from the trained RBF classifier is described in Section 3. Experimental results are shown in Section 4. Finally, Section 5 presents the conclusions of this paper.

## 2 CONSTRUCTING AN EFFICIENT RBF CLASSIFIER

In the RBF neural network, the activation of a hidden unit is determined by the distance between the input vector and the center vector of the hidden unit. The weights connecting the hidden layer and the output layer can be determined by the linear least square (LLS) method [1][14], which is fast and free of local minima, in contrast to the multilayer perceptron neural network.

[1]Xiuju Fu, studying for Master degree in School of EEE, Nanyang Technological University, Block S2, Nanyang Avenue, Singapore 639798, Email: p146793114@ntu.edu.sg

There are three layers in the RBF neural network, i.e., the input layer, the hidden layer with Gaussian activation functions, and the output layer. It is desirable for an RBF classifier to have a small number of hidden units, and at the same time, a low classification error rate. We now discuss the effect of overlapped receptive fields of Gaussian kernel functions of the RBF neural network on the number of its hidden unit. Overlapped receptive fields of different clusters can improve the performance of the RBF classifier in rejecting noise when tackling with noisy data [9]. However, the overlaps between different classes will decrease the accuracy of classification when tackling with noise free data.

A modification in training RBF neural network for simplifying the construction of RBF neural networks is applied. As known, classification error rate is mainly determined by the degree of overlap between clusters for *different classes*, and is independent of the degree of overlap between clusters for the same class. In the above clustering algorithm, if cluster radii are increase by decreasing $\theta$ ($\theta$ is the ratio of in-class patterns and all the patterns in one cluster), the number of clusters will decrease; however, the classification error rate will also increase because of larger overlaps between clusters for *different classes*. But if we modify the clustering algorithm such that small overlaps between clusters for different classes are maintained, and large overlaps between clusters for the same class are allowed, the classification error rate should remain the same with the number of clusters decreased. That is, some clusters are enlarged in size, and at the same time, $\theta$-criterion is satisfied in all clusters.

The following steps is applied to permit large overlaps between clusters with the same class label. A copy $V_c$ of the original data set $V$ is made first. When a qualified cluster, e.g., cluster A in Fig.1(b) (same as in Fig.1(a)), is generated, the members in this cluster are "removed" from the copy data set $V_c$, but the patterns in the original data set $V$ is unvaried. Subsequently, the initial center of the next cluster is selected from the *copy data set* $V_c$, but the candidate members of this cluster are patterns in the *original data set* $V$, thus include the patterns in the cluster A. Subsequently when pattern 2 is selected as an initial cluster center, a much larger cluster B, which combines clusters B, C, and D in Fig.1(a) can still satisfy the $\theta$-criterion and can therefore be created.

In order to obtain fewer number of clusters with high accuracy in classification at the same time, a dynamic $\theta$ is applied. The initial value of $\theta$ is 100%. There are two conditions to determine whether reduce $\theta$ or not. One condition is if the ratio between the number of remaining patterns in $V_c$ and the number of patterns in $V_c$ in last epoch is fewer than a certain ratio $\alpha_1$, the other is if the epoch number for searching one suitable cluster exceeds one certain number $R_Count$. If either of the two conditions is satisfied, the $\theta$ will be changed to a certain ratio of its previous value (we set the ratio as 90%). Of course, the

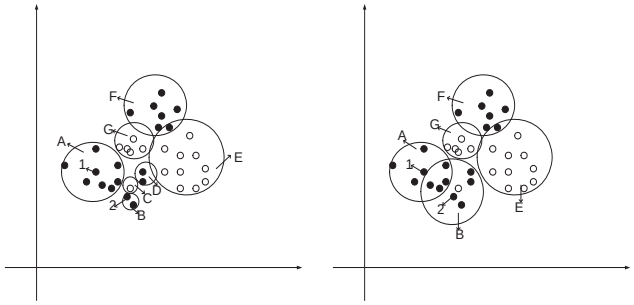


Figure 1: (a) no large overlap (b)large overlap

minimal $\theta$ is constrained too (we set it as 75%).

When training RBF neural networks, the initial centers for clusters are chosen randomly (without considering the order of classes), which gives an equal chance for each class to form clusters, in contrast to the Gaussian Masking (GM) algorithm [13]. In addition, the last a few patterns left will not be considered as candidates of initial cluster centers, i.e., isolated patterns in each class will be omitted in the clustering process, in order to minimize the number of clusters.

By allowing for large overlaps between clusters for the same class, we can further reduce the number of clusters substantially. This will lead to more efficient construction of RBF networks, i.e., the number of hidden units will be reduced. The experimental results will be shown in Section 4.

## 3 RULE EXTRACTION

After training the RBF neural network, the concept of data is memorized in the construction of the RBF classifier. We try to explain the concept of data through the RBF classifier. Since each hidden unit of the RBF neural network is responsive to a subset of patterns (instances), the weights connecting the hidden unit with output units can reflect for which output the hidden unit serves. Our rule extraction algorithm is directly based on the widths, centers of Gaussian kernel functions, and weights connecting hidden units and the output layer.

First, determine the corresponding output unit which each hidden unit serves for through simplifying the weights between hidden units and output units: consider the weight matrix (assume there are m hidden units, and n output units)

$$W = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{m1} & x_{32} & \cdots & x_{mn} \end{pmatrix} \quad .$$

The matrix will be converted into

$$W_1 = \begin{pmatrix} 0 & \cdots & x_{1k_1} & \cdots & 0 \\ 0 & \cdots & x_{2k_2} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & 0 & \cdots & x_{mk_m} & 0 \end{pmatrix} \quad ,$$

where $x_{jk_j}$ is the maximum value of each row j (j=1, ... m) of matrix $W$. Thus, $W_1$ reflects the corresponding output which each hidden unit mainly serves for. In addition $x_{jk_j}$s are normalized to 1, i.e., all $x_{jk_j}$ are divided by the largest of them.

The suitable interval of attributes composes of the premise parts of extracted rules. Let us assume that the number of attributes is $N$. The upper limit $Upper(j,i)$ and the lower limit $Lower(j,i)$ of the jth attribute in the ith rule are initialized as:

$$Upper(j,i) = \mu_i + \rho_{i,j} \quad , \tag{1}$$

$$Lower(j,i) = \mu_i - \rho_{i,j} \quad , \tag{2}$$

$$\rho_{i,j} = \eta_j * \overline{x}_{ik_i} * \sigma_i \quad . \tag{3}$$

where $\mu_i$ is the jth item of the center of the ith kernel function, initially, $\eta_j = 1$, $\sigma_i$ is the width of the ith kernel function. $\overline{x}_{ik_i}$ is the corresponding element in $W_1$. $\rho_{i,j}$ will be adjusted according to our iteration steps as follows.

$\eta_j$ is modified according to:

$$\eta_j = \eta_j + Sign * 0.025. \tag{4}$$

$Sign$ has two value $+1$ and $-1$, which is determined by the trend of change in the rule accuracy when adjusting. The initial $Sign$ is $+1$. If the rule accuracy becomes lower with the adjusted $\eta$, $Sign$ is changed to $-1$. Otherwise, $Sign$ remains the same. The stop-criterion for the iteration is a predefined rule error rate. When adjusting the intervals to obtain high accuracy, the validation set is used for determining the direction of adjusting, which can help the extraction rules not too fit to the training data set, and obtain good results in the testing data set.

Table 1: Reduction in the number of hidden units

| Comparisons | Iris | | Thyroid | |
|---|---|---|---|---|
| | Small overlap | Large overlap | Small overlap | Large overlap |
| Error rate in classification | 0.0491 | 0.0387 | 0.048 | 0.0385 |
| Number of hidden units | 5.2 | 3.4 | 13.8 | 7.4 |

As compared with the technique proposed by McGarry [10][11][12], a higher accuracy with concise rules is obtained in our method. In [10][12], the input intervals in rules are expressed in the following equations:

$$X_{upper} = \mu_i + \sigma_i - S \quad , \tag{5}$$

$$X_{lower} = \mu_i - \sigma_i + S \quad , \tag{6}$$

$S$ is feature "steepness", which was discovered empirically to be about 0.6 by McGarry. Obviously the empirical parameter will not be suitable to all data sets. The experimental results are shown in the next section.

## 4 THE EXPERIMENTAL RESULTS

Two data sets, Iris and Thyroid from the UCI Repository of Machine Learning Databases, are used for testing our methods. There are 4 attributes and 3 classes in Iris data set. There are 5 attributes and 3 classes in Thyroid data set. Each data set is divided into 3 parts, i.e., training, validation, and test sets. 150 patterns of Iris data set is divided into 50 patterns for each set. There are 215 patterns in Thyroid data set. 115 patterns are for training, 50 patterns for validation and 50 patterns for testing. We set $\alpha_1 = 0.8$ and we set the maximum epoch number of searching one cluster is $R_Count$=20. in our experiments. The results shown in Table 1 are the average value of 5 independent experiments. The smallest number of hidden units in constructing an RBF neural network classifier is 3 for Iris data set. For Thyroid, at least 6 hidden units are needed.

Table 1 shows that when large overlaps among clusters of the same class are permitted, both the number of hidden units and the classification error rate are decreased.

After the learning procedure and using our proposed rule extraction method, we obtain 3 symbolic rules for Iris data set.

The accuracy of the symbolic rules that we obtain through the proposed method is 90%-93% for Iris data set. For Thyroid data set, 6 rules are obtained, with 5 conditions in each rule, and the accuracy is 80%-85%.

We now compare our results with rule-extraction results using RBF neural networks. In [4], 5 or 6 rules are needed to represent the concept of Iris data (the accuracy is not available). Huber [6] extracted 8 rules to represent Iris data set (the accuracy is not available). In order to get a small rule base, unimportant rules are pruned according ranking [6]. However, the accuracy of rules is reduced [6] at the same time. McGarry [10][11][12] extracted rules from RBF neural networks directly from the parameters of Gaussian kernel functions and weights. In [10], the accuracy reaches 100%, but the number of rules is large (for the Iris data set, 53 rules are needed). In [11] and [12], the number of rules for the Iris data set is small, i.e., 3, but the accuracy of the extracted rules is only 40% and around 80%, respectively. For Thyroid data set, we obtain 8 symbolic rules, and there are 5 conditions in each rules. The accuracy of extracted rules is 80%. The results of extracted rules for Thyroid data set using other methods are not available.

Many rule-extraction methods have been explored based on the MLP. Desirable results have been obtained both in accuracy and numbers of rules (e.g., [3][5][7][8]). Compared with the rule extraction techniques using MLP, the accuracy of the rules extracted from the RBF neural networks is lower, however, the training of the RBF neural network can escape from local minima, which is very important for large data sets. The architecture of the RBF neural network is simpler and the training time is usually shorter in comparison with the MLP.

## 5 CONCLUSIONS

A useful modification in constructing and training the RBF network by allowing for large overlaps among clusters of the same class is applied, which make the number of hidden units reduce while maintaining the classification performance. The reduced number of hidden units can facilitate us in searching concise rules. Rule extraction is carried out from the simplified RBF classifier in order to explain and represent the concept of data. In order to make clearly which hidden unit serves for which class, the weights between the hidden layer and the output layer are simplified first. Then the interval for each input as the premise of each rule is determined by iteration steps, the validation data set is used for making sure that the result is fit for training data set and testing data set at the same time. Our rule extraction technique is simple to implement, which is shown through our experimental results, and concise rules with high accuracy are obtained based on the RBF classifiers.

## References


[1] Christopher M. Bishop, *Neural network for pattern recognition*, Oxford University Press, New York, 1995.

[2] T. Brotherton, G. Chadderdon, and P. Grabill, "Automated rule extraction for engine vibration analysis", *Proc. 1999 IEEE Aerospace Conference*, vol. 3, pp. 29-38, 1999.

[3] G. Bologna and C. Pellegrini, "Constraining the MLP power of expression to facilitate symbolic rule extraction", *Proc. IEEE World Congress on Computational Intelligence*, vol. 1, pp. 146-151, 1998.

[4] S. K. Halgamuge, W. Poechmueller, A. Pfeffermann, P. Schweikert, and M. Glesner, "A new method for generating fuzzy classification systems using RBF neurons with extended RCE learning Neural Networks", *Proc. IEEE World Congress on Computational Intelligence*, vol. 3, pp. 1589-1594, 1994.

[5] E. R. Hruschka and N. F. F. Ebecken, "Rule extraction from neural networks: modified RX algorithm", *Proc. International Joint Conference on Neural Networks*, Vol. 4, pp. 2504-2508, 1999.

[6] K.-P. Huber and M. R. Berthold, "Building precise classifiers with automatic rule extraction", *Proc. IEEE International Conference on Neural Networks*, vol. 3, pp. 1263-1268, 1995.

[7] H. Ishibuchi, M. Nii, and T. Murata, "Linguistic rule extraction from neural networks and genetic-algorithm-based rule selection", *Proc. International Conference on Neural Networks*, vol. 4, pp. 2390-2395, 1997.

[8] H. Ishibuchi and T. Murata, "Multi-objective genetic local search for minimizing the number of fuzzy rules for pattern classification problems", *Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence*, vol. 2, pp. 1100-1105, 1998.

[9] P. Maffezzoni and P. Gubian, "Approximate radial basis function neural networks (RBFNN) to learn smooth relations from noisy data", *Proceedings of the 37th Midwest Symposium on Circuits and Systems*, vol. 1, pp. 553-556, 1994.

[10] K. J. McGarry, S. Wermter, and J. MacIntyre, "Knowledge extraction from radial basis function networks and multilayer perceptrons", *Proc. International Joint Conference on Neural Networks*, vol. 4, pp. 2494-2497, 1999.

[11] K. J. McGarry, J. Tait, S. Wermter, and J. MacIntyre, "Rule-extraction from radial basis function networks", *Proc. Ninth International Conference on Artificial Neural Networks*, vol. 2, pp. 613-618, 1999.

[12] K. J. McGarry and J. MacIntyre, "Knowledge extraction and insertion from radial basis function networks", *IEE Colloquium on Applied Statistical Pattern Recognition (Ref. No. 1999/063)*, pp. 15/1-15/6, 1999.

[13] A. Roy, S. Govil, and R. Miranda, "An algorithm to generate radial basis function (RBF)-like nets for classification problems", *Neural networks*, vol. 8, no. 2, pp. 179-201, 1995.

[14] N. Sundararajan , P. Saratchandran, and Y. W. Lu , *Radial basis function neural networks with sequential learning: MRAN and its applications*, Singapore , River Edge, N.J.: World Scientific, 1999.