Rule Extraction Based on Data Dimensionality Reduction Using RBF Neural Networks

Xiuju Fu and *Lipo Wang*[*]

School of Electrical and Electronic Engineering
Nanyang Technological University
Block S2, Nanyang Avenue
Singapore 639798
Email: {p146793114,elpwang}@ntu.edu.sg
http://www.ntu.edu.sg/home/elpwang
[*]Corresponding author

Abstract

Compact rules is desirable in the task of rule extraction. Since there are often redundant or irrelevant attributes in data sets, removing the redundant or irrelevant attributes from the data sets can lead to more compact rules. In this paper, firstly, a novel method, a separability-correlation measure (SCM), is used to rank the importance of attributes, and an RBF classifier is used to evaluate the best subset of attributes to be retained. Secondly, large overlaps between clusters of the same class are allowed in order to reduce the number of hidden units in the RBF network. Thirdly, rule extraction is carried out based on the retained subset of attributes used as the input to the RBF neural network. Simulations show that this procedure lead to more compact 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 [3][5], redundant inputs are removed before rule extraction. Huber [7] 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.

Our rule extraction algorithm is based on RBF neural networks. The simple architecture and the local approximation in kernel functions of RBF neural networks makes extracting the concept of data from RBF neural networks more desirable than from other neural networks. In order to obtain more compact rules, on the one hand, data dimensionality reduction (DDR) is needed before inputting data to RBF neural networks. On the other hand, reducing the number of hidden units of RBF neural networks should be carried out before extracting rules.

DDR is a preprocessing procedure to reduce the number of attributes while maintaining the concept of data sets. DDR has become an important aspect of pattern recognition, since operators and automated controllers are able to make better use of lower-dimensional data compared to higher-dimensional ones. Computation burden can be reduced for further processes by DDR. Reduced data dimensionality leads to less complicated network structure and thus increased efficiency in processing data.

In this paper, a separability-correlation measure (SCM) is proposed for determining the importance of the original attributes. The SCM is composed of the intraclass distance to interclass distance ratio and an attribute-class correlation measure. The magnitude SCM corresponding to a certain attribute gives the importance of the attribute. An RBF classifier is used to remove unimportant attributes using the ranking result from SCM. Large overlaps between clusters of the same class are allowed, which leads to a less number of hidden units. Then rule extraction is carried out based on the results of DDR.

We focus on reducing the dimensionality of data, simplifying the structure of RBF neural networks, and further extracting rules in this paper. The SCM measure for ranking the importance of attributes is proposed in Section 2. Section 3 introduces how to construct a RBF neural network classifier efficiently by allowing for large overlaps between clusters of the same class. Section 4 shows our rule extraction algorithm. Experimental results are shown in Section 5. Finally, we conclude the paper in Section 6.

2 Separability-Correlation Measure

Our attribute importance ranking is based on the class separability and the correlation between attributes and class labels. Class separability may be measured by the intraclass distance S_w and the interclass distance S_b [4]. The greater S_b is and the smaller S_w is, the better the separability of the data set is. The ratio of S_w and S_b is calculated and is used to measure the separability of the classes: the smaller the ratio, the better the separability. If omitting attribute k_1 from the data set leads to less class separability, i.e., a greater S_w/S_b, compared to the case where attribute k_2 is removed, attribute k_1 is more important for classification of the data set than attribute k_2, and vice versa. Hence the importance of the attributes can be ranked by computing the intraclass-to-interclass distance ratio with each attribute omitted in turn.

The correlation between the changes in attributes and their corresponding changes in class labels is considered while ranking the importance of attributes. We propose the following correlation between the k-th attribute and the class labels in the data set :

$$C_k = \sum_{i \neq j} |x_{ik} - x_{jk}| \cdot \mathrm{sign}|y_i - y_j| \quad , \qquad (1)$$

where for any y, $\mathrm{sign}|y| = 1$ if $|y| > 0$ and $\mathrm{sign}|y| = 0$ if $|y| = 0$. x_{ik} and x_{jk} are the k-th attribute of the i-th pattern and the j-th pattern, respectively. y_i and y_j are the class labels of the i-th pattern and the j-th pattern, respectively. A great magnitude of C_k shows that there are close correlation between class labels and the k-th attribute, which indicates great importance of attribute k in classifying the patterns, and vice versa.

We propose a separability-correlation measure (SCM) R_k as the sum of the class separability measure S_{wk}/S_{bk} and the correlation measure C_k (k refers to the k-th attribute), where S_{wk} and S_{bk} are intraclass and interclass distances calculated with the k-th attribute omitted from each pattern, respectively.

The importance level of attributes is ranked through the value of R_k. The greater the magnitude of R_k, the more important the k-th attribute.

3 A Simplified RBF Classifier

RBF neural networks are popular in various fields due to its simple architecture. Usually, there are three layers in the RBF neural network, the input layer, i.e., the hidden layer with Gaussian activation functions and the output layer. In this paper, we use the RBF network for classification. If there is M classes in the dataset, we write the m-th output of the network as follows:

$$y_m(\mathbf{X}) = \sum_{j=1}^{K} w_{mj} \o_j(\mathbf{X}) + w_{m0} b_m \quad . \qquad (2)$$

Here $\mathbf{X}$ is the input pattern vector (n-dimension). $m = 1, 2, ..., M$. K is the number of hidden units, M is the number of output. w_{mj} is the weight connecting the j-th hidden unit to the m-th output node. b_m is the bias, w_{m0} is the weight connecting the bias and the m-th output node. $\o_j(\mathbf{X})$ is the activation function of the j-th hidden unit:

$$\o_j(\mathbf{X}) = \mathrm{e}^{\frac{||\mathbf{X}-\mathbf{C_j}||^2}{2\sigma_\mathbf{j}^2}} \quad , \qquad (3)$$

where $\mathbf{C_j}$ and σ_j are the center and the width for the j-th hidden unit, respectively, and are adjusted during learning.

In the RBF neural network, the output of a hidden unit is based on 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 calculated by the linear least square (LLS) method [1], which is fast and free of local minima, in contrast to the multilayer perceptron neural network.

Once the centers, widths, and the weights are determined, the architecture of an RBF network is fixed. Both the dimensionality and the distribution of the input patterns affect the number of the hidden units. Dimensionality reduction will lead to the reduction in the number of hidden units.

Table 1: Reduction in the number of hidden units for Iris data set

Comparisons	Small overlap	Large overlap
Error rate in classification	0.0373	0.0467
Number of hidden units	5.2	4

[13] shows that overlapped receptive fields of different clusters can improve the performance of the RBF classifier in rejecting noise when tackling with noisy data. In [14] and [9], overlapping Gaussian kernel functions are created to map out the territory of each cluster with a less number of Gaussians. Small overlaps between the Gaussians for different classes are measured by the ratio between the number of the in-class patterns and the number of the out-class patterns in this cluster. A pre-defined value θ is set, i.e., the ratio should be not less than θ to guarantee a small classification error rate for the training data set (this is θ-criterion). When small overlaps among clusters for *different classes* are permitted, small overlaps among clusters for the *same class* may exist.

In this paper, by allowing for large overlaps between clusters for *the same class*, we can further reduce the number of clusters substantially. This will lead to a simplified RBF neural network.

Through the algorithm described in Section 2, the importance level of each attribute is obtained. The subset of attributes which includes k most important attributes is input to the RBF classifier, and for $k = 1, 2, ..., N$, classification error rate is calculated for each k. For small k, classification error decreases as k increases until all important attributes are included. If for some k_1, the classification rate for $k = k_1 + 1$ is greater than that for $k = k_1$, then attributes $(k_1 + 1)$, $(k_1 + 2), ..., N$ are considered irrelevant.

4 Rule Extraction

Obtaining the important attributes by the method stated in our previous section, the important attributes are used as the input of RBF neural networks. By training the RBF neural network, the concept of data is memorized in the construction of the RBF classifier. The concept of data is explained based on the RBF classifier by our rule extraction algorithm. 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)

$$W = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \multicolumn{4}{c}{\cdots\cdots\cdots\cdots\cdots\cdots\cdots} \\ x_{m1} & x_{32} & \cdots & x_{mn} \end{pmatrix} \quad .$$

The matrix will be converted into

$$W_1 = \begin{pmatrix} 0 & \cdots & x_{1k_1} & \cdots & 0 \\ 0 & \cdots & x_{2k_2} & \cdots & 0 \\ \multicolumn{5}{c}{\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots} \\ \cdots & 0 & \cdots & x_{mk_m} & 0 \end{pmatrix} \quad ,$$

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:

$$Upper(j, i) = \mu_i + \rho_{i,j} \quad , \tag{4}$$

$$Lower(j, i) = \mu_i - \rho_{i,j} \quad , \tag{5}$$

$$\rho_{i,j} = \eta_j * \overline{x}_{ik_i} * \sigma_i \quad . \tag{6}$$

where μ_i is the jth item of the center of the ith kernel function, initially, $\eta_j = 1$, σ_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.

η_j is modified according to:

$$\eta_j = \eta_j + Sign * 0.025. \tag{7}$$

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

Table 2: Comparison of the number of hidden-units before and after the irrelevant attributes are removed for Iris data set

Input attributes	Before removal	4,3,1,2
	After removal	4,3
Number of hidden units	Before removal	4
	After removal	3
Classification error rate	Before removal	0.0467
	After removal	0.0333

becomes lower with the adjusted η, $Sign$ 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.

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:

$$X_{upper} = \mu_i + \sigma_i - S \quad , \tag{8}$$

$$X_{lower} = \mu_i - \sigma_i + S \quad , \tag{9}$$

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.

5 Experimental Results

Iris data set is used for testing our method. There are 4 attributes in Iris data set. The data set is divided into 3 parts, i.e., training, validation, and test sets. In 150 patterns of Iris data set, 90 patterns are for training, 30 for validation, 30 for testing. We set $\alpha = 0.1$ and $\theta = 7$ in our experiments. We set $\alpha_1 = 0.8$ and we set the maximum epoch number of searching one cluster is R_Count=20. The experiment is repeated 5 times with different initial conditions and the average results are recorded.

The rank of importance of the attributes according to our SCM is: 4, 3, 1, 2 for Iris data set. Table 2 shows the classification error rate of the RBF classifier for various subsets of attributes in the order of importance. We see from Table 2 that as the number of attributes used increases the validation error first decreases, reaches minimum when 2 attributes are used, and then increases. Hence in the Iris data set, attributes 1 and 2 are irrelevant attributes for classification and are then removed. This improves the classification performance and decreases the number of inputs and the number of hidden units of the RBF neural network. Table 2 summarizes the advantages of removing irrelevant attributes.

Table 3: Comparisons with other rule extraction algorithms

comparison	rule number	average number of premises	rule accuracy
McGarry1 [10]	53	4	100%
McGarry2 [11]	3	4	40%
McGarry3 [12]	3	4	85%
Halgamuge [5]	5-6	-	-
Huber [7]	8	-	-
Our method	3	2	93.3%

"-" represents that the value is not available

The attributes left after our DDR procedure are used as inputs to an RBF neural network. It is shown in Table 2 that, after removing unimportant attributes, the accuracy of the RBF classifier becomes better. The architecture of the RBF neural network is simplified by DDR, and at the same time, DDR lead to more compact rules, i.e., a less number of premises in each rule.

By the rule-extraction algorithm described in Section 3, the rule set for Iris is obtained. There are 3 rules in the rule set. There are two premises corresponding to the two attributes left in each rule. The accuracy of obtained rules is 93.3%.

In Table 3, we compare our rule extraction algorithm with other rule extraction algorithms based on RBF neural networks. It shows that our method obtain better performance with a less number of premises, a less number of rules, and higher accuracy.

On the other hand, 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., [2][6][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.

6 Conclusion

In this paper, a SCM is proposed to rank the importance of attributes. Unimportant attributes are removed from the inputs to the RBF classifier according to the ranking results from SCM. Iris data set is used to test the method. Experimental results show that the method proposed is effective in reducing the size of data sets and reducing the structural complexity of the RBF neural network. The RBF neural network combined with the SCM saves time for DDR for that only a limited number of candidate subsets need to be evaluated. Constructing RBF neural networks, large overlaps between clusters of the same class are allowed in order to decrease the complexity of architecture of RBF neural networks by reducing the number of hidden units. Then the results obtained are used in our rule extraction task. More compact rules are obtained, i.e., the number of rules is decreased, the premises of rules are simplified and the accuracy is desirable, after removing the unimportant attributes in data and simplifying the architecture of RBF neural networks.

References

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

[2] 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.

[3] 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.

[4] Devijver, P. A. and Kittler, J., *Pattern recognition: a statistical approach*, Prentice-Hall International, Inc. London, 1982.

[5] 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.

[6] 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.

[7] 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.

[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] Kaylani, T. and Dasgupta, S., "A new method for initializing radial basis function classifiers systems", IEEE International Conference on Man, and Cybernetics, Vol.3, pp.2584-2587, 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] Maffezzoni, P. and Gubian, P., "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.

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