Rule Extraction from an RBF Classifier Based on Class-Dependent Features

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 - Rule extraction is a technique for knowledge discovery. Compact rules with high accuracy are desirable. Due to the curse of irrelevant features to classifiers, feature selection techniques are discussed widely. We propose to extract rules based on class-dependent features from an radial basis function (RBF) classifier by genetic algorithms (GA). Each Gaussian kernel function of the RBF neural network is active for only a subset of patterns which are approximately of the same class. Since each feature may have different capabilities in discriminating different classes, features should be masked differently for different classes. In our method, different feature masks are used for different groups of Gaussian kernel functions corresponding to different classes. The feature masks are adjusted by GA. The classification accuracy of the RBF neural network is used as the fitness function. Thus, the dimensionality of a data set is reduced. Concise rules with high accuracy are subsequently obtained based on the class-dependent features. We demonstrate our approach using computer simulations.

I. INTRODUCTION

Extracting rules from data sets has attracted more and more attention in recent years. Rule extraction is an important task of data mining. The popularity of rule extraction is due to its ability to represent the concept of data in linguistic words.

The amount of data available is increasing in volume and dimensionality. There is a need for preprocessing, such as cleaning uncompleted data and reducing the dimensionality of data. In most cases, there are redundant or irrelevant attributes in data sets. Hence, it is desirable to remove the redundant or irrelevant attributes from the data sets, which can facilitate practical applications in improving speed and relieving memory constraints. Data dimensionality reduction (DDR) is often used as one of the first steps for data mining tasks.

Techniques for DDR may be classified into two categories: class-independent (features selected are common to all classes) and class-dependent (different feature sets are selected for different classes). In class-independent DDR [1][6], all features selected are assumed to play equal roles in discriminating each class from the others.

Genetic algorithms (GA) were used to obtain class-independent features [3][4][5][7][11]. Each chromosome in the population pool represents a feature mask [3][4][7][11]. Assume there are n features in total. n bits are needed in a chromosome to represents the n features. The kth bit of a chromosome indicates the presence or absence of the kth feature, i.e., the feature is presented if the bit is 1, and is absent if the bit is 0. The classification accuracy of a classifier is used as the fitness function in GA. Fung *et al* [5] proposed to use fuzzy GA (FGA) to select class-independent features. In [3][5], the fitness evaluator is a nearest-neighbor (NN) classifier. Chaikla and Qi [4] also choose the NN classifier together with multiple correlation as the fitness function of GA to select class-independent features. Raymer *et al* [11] encode the number of NN classifiers into a chromosome together with the features. The classification accuracy is also used as the evaluation function. In [7], the classification result is determined by the vote of several different classifiers, i.e., the logistic classifier (LOG), the linear discriminant classifier (LDC) and the quadratic discriminant classifier (QDC), etc..

Class-dependent feature selection was proposed by Oh *et al* [9][10] to improve recognition performance. Class-dependent features were selected by considering class separation in conjunction with the recognition rate [10].

0-7803-7282-4/02/$10.00 ©2002 IEEE

Next, Oh *et al* constructed multiple MLP classifiers based on the class-dependent features obtained. For each class, a MLP classifier whose inputs were the features selected for this class was trained individually. Thus, if there are M classes in the data set, M MLP classifiers have to be trained, which is computationally expensive.

In the present work, different feature subsets are selected for different classes based on the possibility that a feature may have different capability in discriminating different classes. For different groups of hidden units corresponding to different classes, different feature subsets are selected as inputs. GA is used to search for the optimal feature masks. In other words, we incorporate Oh *et al*'s class-dependent feature selection [9][10] in the RBF classifier. In contrast to Oh *et al* [9][10], only a single such RBF network, rather than multiple MLPs, are required for a multi-class problem.

We use the RBF neural network as a classifier. The rule-extraction method extracts rules from the simplified RBF classifier whose inputs are class-dependent features. In an RBF classifier, the boundary of the receptive field of the kernel function is a hyper-sphere. The Euclidean distance between a pattern and the center of the cluster measures the probability that a pattern belongs to a class. Rules with hyper-rectangular decision boundaries are extracted based on the training result of an RBF neural network using gradient descent theory.

This paper is organized as follows. Our GA-based RBF classifier is presented in Section II. Section III shows our rule extraction method. Experimental results are shown in Section IV. Finally, we conclude the paper in Section V.

II. THE RBF CLASSIFIER

A. A Traditional RBF Classifier

RBF neural networks [2][12] are widely used for function approximation, pattern classification, and so on. In an 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 a linear least square (LLS) method [2], which is fast and free of local minima, in contrast to the MLP neural network.

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. There are M classes in the data set and M output neurons. The m-th output of the network is written as follows:

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

Here $\mathbf{X}$ is the n-dimensional input pattern vector. $\mathbf{X} = \{x_1, x_2, ..., x_k, ..., x_n\}$. $m = 1, 2, ..., M$. K is the number of hidden units. 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 in the m-th output neuron. $\phi_j(\mathbf{X})$ is the activation function of the j-th hidden unit:

$$\phi_j(\mathbf{X}) = \mathrm{e}^{\frac{||\mathbf{X}-\mathbf{C_j}||^2}{2\sigma_j^2}} \quad , \tag{2}$$

where $\mathbf{C_j} = \{c_1, c_2, ..., c_k, ..., c_n\}$ and σ_j are the center and the width for the j-th hidden unit, respectively, which are adjusted during learning.

B. Architecture of a Novel RBF Classifier

We observe that the hidden neurons in an RBF network may be grouped according to classes. That is, if most of the patterns in the cluster represented by a hidden neuron belong to class i, we say that this hidden neuron belongs to the group for class i (Fig. 1). We add a class-dependent feature mask for each group of hidden neurons.

The m-th output of the network is as follows:

$$y_m(\mathbf{X}) = \sum_{i=1}^{M}\sum_{j=1}^{k_i} w_{mj}^i\phi_j^i(\mathbf{X}) + w_{m0}b_m \quad , \tag{3}$$

where $\phi_j^i(\mathbf{X})$ is the activation function of the j-th hidden unit which serves class i:

$$\phi^i{}_j(\mathbf{X}) = \mathrm{e}^{\frac{||\mathbf{X^i}-\mathbf{C_j^i}||^2}{2\sigma_j^{i\,2}}} \quad . \tag{4}$$

Here $\mathbf{X^i} = \{g_1^i x_1, g_2^i x_2, ..., g_k^i x_k, ..., g_n^i x_n\}$. $\{g_1^i, g_2^i, ..., g_k^i, ..., g_n^i\}$ is the feature mask for class i. $g_k^i = 0, 1$. σ_j^i is the width for the j-th hidden unit of class i and is obtained during training in the presence of the feature masks. $\mathbf{C_j^i} = \{g_1^i c_1, g_2^i c_2, ..., g_k^i c_k, ..., g_n^i c_n\}$.

C. Training the Classifier

Finding the centers, widths and the weights connecting hidden nodes to the output is the key to constructing and training the RBF classifier. Both the dimensionality and the distribution of the input patterns affect the number of the hidden units.

The notation used is as follows. δ_0^i is the maximum radius of the receptive field of the Gaussian kernel function

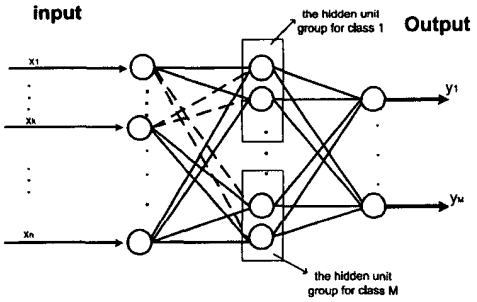

Fig. 1. Architecture of a new RBF neural network. The dashed lines connecting features with a group of hidden units indicate that these features are not input to that group of hidden units.

for class i. δ_0^i is the standard deviation of the distances from the patterns of class i to the centroid of class i [12][2]. θ is the percentage of in-class patterns (the patterns belongs to the current class processed) in a cluster. θ controls the nature of Gaussians [12]. For example, a cluster with $\theta = 60\%$ (a "fat" Gaussian, there are 60% in-class patterns in the cluster) can detect global features in the data set that might not be detected by the cluster with a larger θ (a "narrow" Gaussian) [12], e.g. , $\theta = 90\%$. V is the training set. α is the changing rate of radii of clusters. L is the training stage. N_{Least} is the least number of patterns required for a cluster. Q_i is the number of total patterns that belong to class i in the training set. K_s counts the number of trials intending to generate a cluster. K^i is the number of patterns of class i left in V.

Our algorithm is based on Roy *et al* [12]'s algorithm, however, we have made some modifications. Compared to Roy *et al*'s original algorithm [12], we have made following changes:

1. In Step 1 (b), class-dependent features are used instead of total features [12] with the introduction of the feature masks.
2. In Step 1 (c), δ_0^i (maximum neighborhood radius) is calculated as the standard deviation of class i since different classes are with different feature masks eq. 4. In [12], the maximum neighborhood radius is the maximum of the class standard deviations multiplying a certain constant.
3. In Step 1 (e), N_{Least} is set as the least number of patterns required for a cluster. Since the initial center is selected randomly, we set N_{Least} in order to increase the possibility generating larger-size clusters. The N_{Least} is adjusted automatically according to the condition of training (how many patterns of the class concerned are left). In [12], a general number is predefined as the minimum number of patterns required for forming a Gaussian kernel function.
4. In [12], six training stages were used corresponding to θ={50%, 60%, ..., 100%}, respectively. In each stage, θ is unchanged and all clusters of this stage should meet the θ-criterion. A classification result is obtained for this θ. The classification error rate from a stage is compared to that in the previous stage. Whether to continue to the next training stage is determined by the comparison in the classification results (If the classification accuracy is better than the previous stage, i.e., it is possible to obtain a higher accuracy if increasing θ. Thus, the data is trained again using a larger θ. Else, the training is stopped). Although a larger θ leads to a higher accuracy in the training data set, it may lead to poorer generalization in the testing data set. Hence, in our algorithm (Step 2), θ is automatically adjusted according to the training condition (how many patterns of the class concerned are left). With the decreasing number of patterns, θ is decreased by a certain factor, say 0.9 in our simulations.

From the algorithm stated above, it can be seen that if a new class is added into the data set, there is no need to train the entire RBF classifier. All the hidden units in the old classifier can be maintained. The training procedure can begin from Step 2 above. After the hidden units for the new class are obtained, the new hidden units can be combined with the previous hidden units and the weights between the hidden layer and the output layer can be calculated using the LLS method. Similarly, if some new patterns of an existed class are added to the data set, there is no need to train the entire RBF classifier either. All the hidden units corresponding to other classes in the old classifier can be maintained. The training procedure can begin from Step 2 above to search for clusters of the class concerned. After the hidden units for the class are obtained, the new hidden units can be combined with those maintained hidden units. The weights between the hidden layer and the output layer can then be calculated using the LLS method.

D. GA Encoding

Suppose n is the total number of the original features and M is the number of classes. A binary string representing a possible solution in GA is shown in Fig.2. The length of each individual is $n \cdot M$ bits. A chromosome G is presented as follows:

$$G = \{(g_1^1, ...g_i^1, ..., g_n^1), ..., (g_1^k, ...g_i^k, ..., g_n^k) , ..., (g_1^M, ...g_i^M, ..., g_n^M)\} \quad (5)$$

Here $g_i^k = 0, 1.\ k = 1, 2, ..., M.\ i = 1, 2, ..., n.$

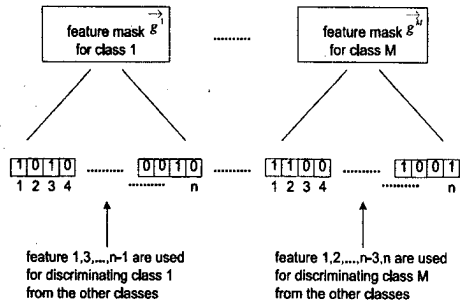

Fig. 2. An encoding string presenting an exemplar solution.

We use the roulette wheel selection to select chromosomes in each generation. Two-point crossover is used. Mutation can prevent the fixation at some particular loci. A locus in the parent chromosome is selected randomly and the bit at the position is replaced (if the original bit is 0, it is replaced by 1, and vice versa). Usually, the mutation rate is relatively small to avoid too much variation. However, at later generations, the number of identical members increases, which leads to a stagnant state. In order to break stagnant states to search for optimal results, we use a dynamic mutation rate, i.e., if the number of identical members in a population exceeds a certain percentage, the mutation rate is increased by a certain amount.

Our fitness function is:

$$F(G) = 1 - E_v(G) \tag{6}$$

where $E_v(G)$ is the classification error rate of the validation data set for chromosome G.

III. RULE EXTRACTION

The rule extraction algorithm proposed here is based on the widths and the centers of the Gaussian kernel functions, and the weights connecting the hidden neurons to the output layer. For each class, a subset of features are selected in order to discriminate the class from other classes. A group of kernel functions are generated for the class based on the selected feature subset. Each hidden neuron of the RBF neural network is responsive to a subset of input patterns (instances). The rules extracted in our paper are in an IF-THEN form.

The objective of tuning the rule premises is to determine the boundaries of rules so that a high rule accuracy is obtained for the testing data set. Before starting the tuning process, all of the premises of the rules must be initialized. Let us assume that the number of attributes is n. The number of rules equals to the number of hidden neurons in the trained RBF network. The number of the premises of rules equals to n. The upper limit U_{ji} and the lower limit L_{ji} of the jth premise in the ith rule are initialized according to the trained RBF classifier as:

$$U_{ji}^{(0)} = \mu_{ji} + \sigma_i \quad , \tag{7}$$

$$L_{ji}^{(0)} = \mu_{ji} - \sigma_i \quad , \tag{8}$$

where μ_{ji} is the jth item of the center of the ith kernel function. σ_i is the width of the ith kernel function.

We introduce the following notations. Suppose $\eta^{(t)}$ is the tuning rate at time t. Initially $\eta^{(0)} = 1/N_I$, where N_I is the number of iteration steps for adjusting a premise. N_I is set to be 20 in our experiments. E is the rule error rate.

$$Q_{ji}^{(t)} \equiv \frac{\partial E}{\partial U_{ji}}|_t \quad , \tag{9}$$

$$A_{ji}^{(t)} \equiv \frac{\partial E}{\partial L_{ji}}|_t \quad . \tag{10}$$

$U_{ji}^{(t)}$ and $L_{ji}^{(t)}$, the upper and lower limits at time t, are tuned as follows.

$$U_{ji}^{(t+1)} = U_{ji}^{(t)} + \Delta U_{ji}^{(t)} \quad , \tag{11}$$

$$L_{ji}^{(t+1)} = L_{ji}^{(t)} + \Delta L_{ji}^{(t)} \quad . \tag{12}$$

Initially, we let

$$\Delta U_{ji}^{(0)} = \eta^{(0)} \quad . \tag{13}$$

$$\Delta L_{ji}^{(0)} = -\eta^{(0)} \quad . \tag{14}$$

Subsequent $\Delta U_{ji}^{(t)}$ and $\Delta L_{ji}^{(t)}$ are calculated as follows.

$$\Delta W_{ji}^{(t)} = \begin{cases} \eta^{(t)} & , \text{ if } Q_{ji}^{(t-1)} < 0 \\ -\eta^{(t)} & , \text{ if } Q_{ji}^{(t-1)} > 0 \\ \Delta W_{ji}^{(t-1)} & , \text{ if } Q_{ji}^{(t-1)} = 0 \\ -\Delta W_{ji}^{(t-1)} & , \text{ if } Q_{ji}^{(t-1)} = 0 \text{ for} \\ & , \ \frac{1}{3}N_I \text{ consecutive} \\ & , \text{ iterations,} \end{cases} \tag{15}$$

where $W = U, L$.

The gradient information of the rule error rate is used to determine the subsequent trend in adjusting premises, i.e., the direction of change is always opposite to the error gradient, so that the error decreases during the premise adjustment, as indicated by the first two lines in eq.15. When $Q_{ji}^{(t)} = 0$ consecutively for $\frac{1}{3}N_I$ time steps, this means that the current direction of premise adjustment is fruitless. $\Delta W_{ji}^{(t)}$ changes its sign as shown in the 4th line of eq. 15. In this situation, we also let $\eta^{(t)} = 1.1\eta^{(t-1)}$, which helps to keep the progress from being trapped. Otherwise $\eta^{(t)}$ remains unchanged.

TABLE I
FEATURE MASK FOR THYROID DATA SET.

Classes	Feature masks
Class 1	0 1 1 1 1
Class 2	0 1 1 0 0
Class 3	0 1 0 0 0

IV. THE EXPERIMENTAL RESULTS

Thyroid data set from the UCI Repository of Machine Learning Databases [8] is used in this paper to test our algorithm. There are 5 attributes in Thyroid data set, i.e., (1) T3-resin uptake test (A percentage), (2) total Serum thyroxin as measured by the isotopic displacement method, (3) total serum triiodothyronine as measured by radioimmuno assay, (4) basal thyroid-stimulating hormone (TSH) as measured by radioimmuno assay, and (5) maximal absolute difference of TSH value after injection of 200 micro grams of thyrotropin-releasing hormone as compared to the basal value. The 3 classes are normal, hyper and hypo. There are 215 patterns in Thyroid data set, 115 patterns for training, 50 for validation, and 50 for testing.

The GA parameters are as follows. In the population pool, there are $4n = 20$ (n is the number of features) chromosomes. The initial mutation rate is 40%. If the number of identical members in a population exceeds 25%, the mutation rate is increased by 1%. The number of elite chromosomes, which remain unchanged and live from one generation to the next, is 2. The generation number is 50.

TABLE II
RULE ACCURACY FOR THYROID DATA SET.

rule accuracy	full features	class-dependent features
training set	94.57%	94.57%
validation set	95.35%	93.02%
testing set	90.7%	93.02%

It is shown in the feature masks (Table I) that feature 1 does not play any role on discriminating classes. Hence, the T3-resin uptake test can be unnecessary in this type of Thyroid diagnosis. For class 3, feature 2 can discriminate class 3 from other classes. Feature 2 and 3 are used to classify class 2 from other classes. Feature 2, 3, 4, 5 is used to discriminate class 1 from other classes.

Two rules are extracted for Thyroid data set. The rule accuracy (in Table II) is: 94.57% for training data set, 93.02% for validation data set, and 93.02% for testing data set. The accuracy in testing data set for 3 classes are: 97.67% for class 1, 100.0% for class 2, and 95.35% for class 3. With full features as inputs, 3 rules are obtained, and the rule accuracy in testing data set for 3 data subsets are: 94.57% for training data set, 95.35% for validation set, and 90.7% for testing set. Thus, higher rule accuracy and more concise rules are obtained when using class-dependent features.

V. CONCLUSIONS

In this paper, we have proposed a rule extraction method from a novel RBF classifier based on class-dependent features. The discriminatory power of each feature for discriminating classes is considered for each class. Different feature subsets are selected for different classes individually based on its ability in discriminating the class with other classes, which show the relationships between the feature subset and the class concerned. The class-dependent feature selection results obtained above provide a new direction for analyzing the relationships between features and classes. The reduction in dimensionality can lead to compact rules in rule extraction task. Thyroid data set is used to test the algorithm. Experimental results show that the algorithm proposed are effective in reducing the number of feature input and leads to compact and accurate rules simultaneously.

References

[1] R. Battiti, "Using mutual information for selecting features in supervised neural net learning", *IEEE Transactions on Neural Networks*, vol. 54, pp.537-550, July 1994.

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

[3] F. Z. Brill, D. E. Brown, and W. N. Martin, "Fast generic selection of features for neural network classifiers", *IEEE Transactions on Neural Networks*, vol. 3, no. 2, pp. 324 - 328, March 1992.

[4] N. Chaikla and Y. L. Qi, "Genetic algorithms in feature selection", *1999 IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, pp. 538 - 540, 1999.

[5] G. S. K. Fung, J. N. K. Liu, K. H. Chan, and R. W. H. Lau, "Fuzzy genetic algorithm approach to feature selection problem", *Proceedings of the Sixth IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 441 - 446, 1997.

[6] N. Kambhatla and T. K. Leen, "Fast non-linear dimension reduction", *IEEE International Conference on Neural Network*, vol. 3, pp.1213 - 1218, 1993.

[7] L. I. Kuncheva and L. C. Jain, "Designing classifier fusion systems by genetic algorithms", *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 327 - 336, Nov. 2000.

[8] P. M. Murphy, & D. W. Aha, (1994). *UCI Repository of machine learning databases* [www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

[9] Il-Seok Oh, Jin-Seon Lee, and C. Y. Suen, "Using class separation for feature analysis and combination of class-dependent features", *Fourteenth International Conference on Pattern Recognition*, vol. 1, pp. 453 - 455, 1998.

[10] Il-Seok Oh, Jin-Seon Lee, and C. Y. Suen, "Analysis of class separation and combination of class-dependent features for handwriting recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 1089 - 1094, Oct. 1999.

[11] M. L. Raymer, W. F. Punch, E. D. Goodman, L. A. Kuhn, and A. K. Jain, "Dimensionality reduction using genetic algorithms", *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 2, pp. 164 - 171, July, 2000.

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