Rule Extraction by Genetic Algorithms Based on a Simplified RBF Neural Network

Xiuju Fu and *Lipo Wang**

School of Electrical and Electronic Engineering
Nanyang Technological University
Block S2, Nanyang Avenue
Singapore 639798
Email: elpwang@ntu.edu.sg
* Corresponding author

Abstract - As an important task of data mining, extracting rules to represent the concept of numerical data is attracting much attention. In this paper, we propose a novel algorithm to extract rules using genetic algorithms (GA) and the radial basis function (RBF) neural network classifier. The interval for each input in the condition part of each rule is adjusted using GA. The fitness of a chromosome is determined by the accuracy of extracted rules. The decision boundary of rules extracted is in the form of hyper-rectangular. During the training of an RBF neural network, large overlaps between clusters corresponding to the same class is allowed in order to decrease the number of hidden units while maintaining classification accuracy. The weights connecting the hidden units with the output units are then pruned. Our simulations demonstrate that our approach leads to more accurate and concise rules.

1 INTRODUCTION

Knowledge discovery in databases (KDD) [19][24] is very useful in economic and scientific domains. Huge amount of data have been stored in documents or in hard disks of computers. KDD techniques are used to reveal critical information hidden in the data sets.

Rule extraction is a common task of KDD. The goal of a rule extraction system is to obtain insights for numeric data sets. Generally, a knowledge discovery system includes the follow components:

1. *Data collection*

   For example, large volumes of data are produced in transactions on the Internet, in supermarkets, and in banks.

2. *Data preprocessing*

   There may be some irrelevant or unimportant data. Dimensionality reduction is desirable for both memory constraint and speed limitation. Data collected from different sources may be in different forms. Hence, the follow preprocessing for data is needed.

   - *Feature selection*: Much research work [13][14][16] has been carried out in choosing the optimal feature subset to represent the concept of data.
   - *Normalization*: For a neural network, input values are usually normalized between [0,1]. Some nominal inputs should be transformed into numeric ones.

3. *The selection of rule extraction tools*

   Decision trees, neural networks, and genetic algorithms, etc., are often used as powerful tools for rule extraction.

   - *Neural networks*: Since neural networks are excellent at predicting, learning from experiences, and generalizing from previous examples, many researchers focus on applying neural networks in the area of rule extraction [10][17][26][27]. However, a disadvantage of neural networks is that it is difficult

0-7803-6657-3/01/$10.00 ©2001 IEEE

to determine network architecture and training parameters. Explaining the operation of a trained network can also be difficult.

- *Decision trees*: Decision trees [15][30] can form concise rules in contrast to neural networks. However, the accuracy of decision trees is often lower than neural networks, especially for noisy data. It is difficult for decision trees to tackle dynamic data. Tsang et al [29] and Umano et al [30] combined neural networks with decision trees to obtain better performance in rule extraction.
- *GA*: Due to its ability to search globally for the optimal solution to a problem, GA has often been combined with neural networks in rule extraction tasks. Fukumi and Akamatsu [4] used GA to prune the connections in neural networks before extracting rules. Hruschka and Ebecken [5] proposed clustering genetic algorithm (CGA) to cluster the activation values of the hidden units of a trained neural network. Rules are then extracted based on the results from CGA. Ishibuchi et al [11] used GA to obtain concise rules by selecting important members from the rules obtained from a neural network.

4. *The expression of the extracted rules*

   Usually, the rules are in IF-THEN forms. The premise parts of rules are composed of different combinations of inputs. There are 3 kinds of decision boundary of rules.

   - *hyper-rectangular*
   - *hyper-plane*
   - *hyper-ellipse*

Some research work has been carried out in extracting rules from RBF neural networks. The architecture of the RBF neural network is simple and it can meet the need of local tuning and global approximation. In [2][7], before inputing attributes to the rule extraction system based on an RBF neural network, redundant inputs were removed, however, some information was lost with the removal. Huber [9] selected rules according to importance; however, the accuracy was reduced with pruning. McGarry [20]-[22] extracted rules from the parameters of Gaussian kernel functions and weights in RBF neural networks. However, when the number of rules was small, the accuracy was low. When the accuracy was acceptable, the number of rules became large.

In this paper, we propose a novel technique to extract rules from the RBF neural network using GA. First, a modification is carried out for reducing the number of hidden units when training the RBF neural network (Section 2). The weights connecting hidden units with output units are simplified (pruned) subsequently. In Section 3, GA is used for tuning the interval for each input attribute in the condition part of each rule in order to obtain a high rule accuracy. We show that our technique leads to a compact rule set with desirable accuracy with experimental simulations (Section 4). Finally, Section 5 presents the conclusions of this paper.

2 CONSTRUCTING AN EFFICIENT RBF CLASSIFIER

In an RBF neural network, the distance between the input pattern and the center of the hidden unit determines the activation of a hidden unit. The weights between the hidden layer and the output layer can be determined by the linear least square (LLS) method [1][28].

In [12], overlapped kernel functions are created to map out the territory of each cluster with a reduced number of Gaussians. The performance of the RBF classifier can be improved by overlapping between different classes which can reject noise when tackling noisy data [18]. The degree of overlaps between different classes is measured by θ, which is the ratio between the numbers of in-class and out-class patterns in a cluster. A θ-criterion is used to avoid large overlaps between different classes. Thus there are small overlaps between the Gaussians for different classes and a small classification error rate is obtained. Usually, θ is determined empirically and is related to an acceptable classification error rate.

In this paper, we propose to reduce the number of clusters, and thus the number of hidden units, by allowing for large overlaps between clusters of the same class. Since classification error rate is mainly determined by the degree of overlaps between different classes, and is independent of the degree of overlap between clusters for the same class, 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. Thus, the classification error rate would remain the same with the number of clusters decreased. This leads to more efficient construction of RBF networks, i.e., the number of hidden units will be reduced.

3 GA FOR RULE EXTRACTION

3.1 Simplifying weights

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 mainly serves. Our rule extraction algorithm is directly based on the widths and the centers of the Gaussian kernel functions, and the weights connecting the hidden units to the output layer.

First, let us determine the output unit for which each hidden unit mainly serves by simplifying the weights between hidden units and output units. Assume there are m hidden units and n output units. The weight matrix W, which is a matrix with m rows and n columns, is converted to W_1 by maintaining only the maximum value of each row to indicate the output unit that each hidden unit mainly serves.

3.2 Encoding rule premises using GA

The rules extracted in our paper are in an IF-THEN form. Symbolic rule i is written as follows:

IF input 1 is within the interval (L_{1i}, U_{1i})
and input 2 is within the interval (L_{2i}, U_{2i})
and
.
.
.
and input n is within the interval (L_{ni}, U_{ni})
THEN the class label of the input pattern is k_i.

Here U_{ji} and L_{ji} are the upper limit and the lower limit of interval j in rule i, respectively.

Let us write:

$$U_{ji} = \mu_{ji} + R^{(1)}{}_{ji} \quad , \tag{1}$$

$$L_{ji} = \mu_{ji} - R^{(2)}{}_{ji} \quad , \tag{2}$$

where μ_{ji} is component j of the center of interval i, $R^{(1)}{}_{ji}$ and $R^{(2)}{}_{ji}$ are the distances from the center to the upper limit and the lower limit, respectively.

Through training an RBF neural network, the input data space is separated into several subspaces. One subspace is represented by a hidden unit of the RBF neural network. Since our kernel function is the Gaussian function, each subspace is hyper-ellipse. Since the decision boundary of our rules is hyper-rectangular, we use GA for searching the premise parts of rules. There are as many rules as hidden units. Hence, efficient architecture of RBF neural networks leads to compact rules. We set the initial value of interval center μ_{ji} in equations 1 and 2 to be the center of hidden unit i. We select the initial values of $R^{(1)}{}_{ji}$ and $R^{(2)}{}_{ji}$ randomly.

Now we encode the solutions for the rules. We encode real value $R^{(p)}{}_{ji}$ $(p = 1, 2)$ using k binary bits $(G^{(p)}{}_{ji} = \{g_k, g_{k-1}, ...g_i, ..., g_2, g_1\},\ g_i = 0, 1)$ as follows:

$$R^{(p)}{}_{ji} = B^{(p)}_{ji}/(2^k - 1) \tag{3}$$

where $B^{(p)}{}_{ji}$ is the decimal value corresponding to $G^{(p)}{}_{ji}$:

$$B^{(p)}{}_{ji} = g_k * 2^{k-1} + g_{k-1} * 2^{k-2} + ... + g_2 * 2^1 + g_1 * 2^0 \quad . \tag{4}$$

A chromosome *Chrom* in the population pool can be represented as a one dimensional binary string:

$$Chrom = (G^{(1)}{}_{11}, G^{(2)}{}_{11}, ..., G^{(1)}{}_{n1}, G^{(2)}{}_{n1}, ..., G^{(p)}{}_{ji}, ..., G^{(1)}{}_{1m}, G^{(2)}{}_{1m}, ..., G^{(1)}{}_{nm}, G^{(2)}{}_{nm}) \quad . \tag{5}$$

3.3 Crossover and mutation

Before genetic operators are applied, parent chromosomes should be selected from the population pool. Roulette wheel selection is used in our algorithm. In roulette wheel selection, the probability of selection is proportional to each chromosome's fitness. However, roulette wheel selection does not guarantee the global optimum and elitism is used to maintain the best members.

Two-point crossover is used. Two points are randomly located in one parent. The two parts of parent chromosomes between the two points are then exchanged to generate new offsprings. The probability of crossover is around 80%.

Mutation can help GA escape from local minima. A locus in the parent chromosome is selected randomly, the bit on the position is replaced (if the original bit is 0, then it is replaced by 1, and vice versus). In our algorithm, in order to break the stagnant state for searching optimal result, we use a dynamic mutation rate. If the number of identical members in a population exceeds a certain percentage, the mutation rate is increased by a certain amount.

3.4 Fitness function

Our fitness function is:

$$F(\{(L_{ji}, U_{ji})\}) = 1 - E(\{(L_{ji}, U_{ji})\}) \tag{6}$$

where $E(\{(L_{ji}, U_{ji})\})$ is the error rate of extracted rules. Each chromosome in the population pool corresponds to a rule set. The accuracy of the rule set is calculated. The better the fitness of the chromosome, the lower the error rate of its corresponding rule set.

4 THE EXPERIMENTAL RESULTS

Iris and Thyroid data sets in the UCI Repository of Machine Learning Databases are often used to test classification methods. There are 4 attributes and 3 classes in Iris data set. The four input attributes of Iris data is: sepal length, sepal width, petal length, and petal width. The 3 classes are Setoda, Versicolor, and Virginica. 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 $\theta = 7$ in our experiments.

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 allowing large overlaps among clusters with the same class label, for Iris, the average number of hidden units reduced from 4.6 to 3.4. The error rate in testing data remained the same. For Thyroid, the average number of hidden units reduced from 14.4 to 8, and the error rate in testing data reduced from 0.06 to 0.048. 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.

We use GA to search optimal solutions of rules. Population size is 300 individuals. The number of bits for each $G_{jp}^{(i)}$, is $k = 6$. Crossover probability is 90%. Mutation rate is dynamic. If the number of identical members in a generation exceeds a certain percentage (we set it to be 75%), the mutation rate is increased by a certain amount. In this paper, the initial mutation rate is 8/1000, and the increment amount is 1/1000. Elitist strategy is used. The number of elite chromosomes, which are maintained unchanged and live from one generation to the next, is 4. The GA searching procedure is stopped if there are 1000 generations produced or the accuracy of extracted rules is higher than a predefined value.

After the searching procedure of GA based on the trained RBF neural network, we obtain 3 symbolic rules for Iris data set. The accuracy of the symbolic rules that we obtain through the proposed method is 97.33% for Iris data set. For Thyroid data set, 6 rules are obtained, with 5 conditions in each rule, and the accuracy is 85%. Halgamuge et al [7] extract rules based on RBF neural networks, however, 5 or 6 rules are needed to represent the concept of Iris data (the accuracy is not available). Huber and Berthold [9] use 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 [9], however, the accuracy of rules is reduced at the same time. McGarry et al [20][21][22] extracted rules from RBF neural networks directly based on the parameters of Gaussian kernel functions and weights. In [20], the accuracy reaches 100%, but the number of rules is large (for the Iris data set, 53 rules are needed). In [21] and [22], 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. The results of extracted rules for Thyroid data set using other methods are not available.

Much work has been carried out in extracting rules using the MLP. Good results have been obtained both in accuracy and numbers of rules (e.g., [8][10]). Compared with the rule extraction techniques using MLP, the accuracy of the rules extracted from GA combined with 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.

5 CONCLUSIONS

In this paper, a novel rule-extraction algorithm combining GA and RBF networks is proposed. Rule extraction is carried out from a simplified RBF classifier in order to explain and represent the concept of data in a concise way. The weights between the hidden layer and the output layer are simplified first. Then the interval for each input as the condition part of one rule is determined by genetic algorithms. Experimental results show that our rule extraction technique is simple to implement, and concise rules with high accuracy are obtained. The proposed method needs to be tested using larger and more realistic data, which will be the subject of future work.

References

[1] C. 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] Y. Davidor, *Genetic algorithms and robotics: a heuristic strategy for optimization*, World Scientific , 1990.

[4] M. Fukumi and N. Akamatsu, "Rule extraction from neural networks trained using evolutionary algorithms with deterministic mutation", *The 1998 IEEE International Joint Conference on on Computational Intelligence*, vol. 1, pp. 686-689, 1998.

[5] E. R. Hruschka and N. F. F. Ebecken, "Applying a clustering genetic algorithm for extracting rules from a supervised neural network", *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, vol. 3, pp. 407-412, 2000.

[6] J. J. Grefenstette, *Genetic algorithms for machine learning*, Kluwer Academic Publishers, 1993.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] T. Kaylani and S. Dasgupta, "A new method for initializing radial basis function classifiers systems", *Proc. IEEE International Conference on Man, and Cybernetics*, vol. 3, pp. 2584-2587, 1994.

[13] T. Kawatani and H. Shimizu, "Handwritten kanji recognition with the LDA method", *Fourteenth International Conference on Pattern Recognition*, vol. 2, pp.1301-1305, 1998.

[14] D. B. Kurt and Joydeep Ghosh, "Mutual information feature extractors for neural classifiers", *IEEE International Conference on Neural Networks* vol. 3, pp.1528-1533, 1996.

[15] A. Gupta, Park Sang, S. M. Lam, "Generalized Analytic Rule Extraction for feedforward neural networks", *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 6, pp. 985-991, Nov.-Dec. 1999.

[16] C. J. Liu and H. Wechsler, "Enhanced fisher linear discriminant models for face recognition", *Fourteenth International Conference on Pattern Recognition*, vol. 2, 1998, pp. 1368-1372.

[17] H. J. Lu, R. Setiono, and H. Liu, "Effective data mining using neural networks", *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, Dec. 1996.

[18] 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.

[19] C. J. Matheus, P. K. Chan, and G. Piatetsky-Shapiro, "Systems for knowledge discovery in databases", *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, no. 6, pp. 903-913, Dec. 1993.

[20] 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.

[21] 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.

[22] 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.

[23] S. K. Pal and P. P. Wang, *Genetic algorithms for pattern recognition*, CRC Press, 1996.

[24] M. J. Pazzani, "Knowledge discovery from data?", *IEEE Intelligent Systems*, vol. 15, no. 2, pp. 10-12, March-April, 2000.

[25] 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.

[26] T. Saito and Y. Takefuji, "Logical rule extraction from data by maximum neural networks Intelligent Processing and Manufacturing of Materials", *Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials*, vol. 2, pp. 723-728, 1999.

[27] R. Setiono, "Extracting rules from neural networks by pruning and hidden-unit splitting", *Neural Computation*, vol. 9, no.1, pp. 205-225, Jan. 1997.

[28] 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.

[29] E. C. C. Tsang, X. Z. Wang, and D. S. Yeung, "Improving learning accuracy of fuzzy decision trees by hybrid neural networks", *1999 IEEE International Conference on Systems, Man, and Cybernetics*, vol.3, pp. 337-342, 1999.

[30] M. Umano, T. Okada, I. Hatono, and H. Tamura, "Extraction of quantified fuzzy rules from numerical data", *The Ninth IEEE International Conference on Fuzzy Systems*, vol. 2, pp. 1062-1067, 2000.