Hierarchical Radial Basis Function Neural Networks for Classification Problems

Yuehui Chen[1], Lizhi Peng[1], and Ajith Abraham[1,2]

[1]School of Information Science and Engineering
Jinan University, Jinan 250022, P.R.China
Email: yhchen@ujn.edu.cn
[2]IITA Professorship Program, School of Computer Science and Engg.
Chung-Ang University, Seoul, Republic of Korea
Email: ajith.abraham@ieee.org

Abstract. Hierarchical neural networks consist of multiple neural networks assembled in the form of an acyclic graph. The purpose of this study is to identify the hierarchical radial basis function neural networks and select important input features for each sub-RBF neural network automatically. Based on the pre-defined instruction/operator sets, a hierarchical RBF neural network can be created and evolved by using tree-structure based evolutionary algorithm. This framework allows input variables selection, over-layer connections for the various nodes involved. The HRBF structure is developed using an evolutionary algorithm and the parameters are optimized by particle swarm optimization algorithm. Empirical results on benchmark classification problems indicate that the proposed method is efficient.

1 Introduction

Hierarchical Neural Network (HNN) is neural network architecture in which the problem is divided and solved in more than step [1]. Ohno-Machado divide hierarchical network into two architectures that are bottom-up and top-down architectures [1]. In bottom-up designs, several specialized network are used to classify the instances and a top-level network (triage network) aggregates the results. In this design, all instances are used in all networks. However, the specialized networks work only on certain features. In contrast, in top-down hierarchical architecture design, the top-level network divides the inputs to be classified in specialized networks.

Many version of HNN have been introduced and applied in various applications [1][3][4][5]. Erenshteyn and Laskov examine the application of hierarchical classifier to recognition of finger spelling [2]. They refer hierarchical NN as multi-stage NN. The approach aimed to minimize the network's learning time without reducing the accuracy of the classifier. Mat Isa et al. used Hierarchical Radial Basis Function (HiRBF) to increase RBF performance in diagnosing cervical cancer [3]. HiRBF cascading together two RBF networks, where both network have different structure but using the same algorithms. The first network classifies all data and performs a filtering process to ensure that only certain attributes to be fed to the second network. The study shows that HiRBF performs better compared to single RBF. HRBF has been proved effective in the reconstruction of smooth surfaces from sparse noisy data points [5].

In this paper, an automatic method for constructing HRBF network is proposed. Based on the pre-defined instruction/operator sets, a HRBF network can be created and evolved. HRBF allows input variables selection, over-layer connections for different nodes. In our previous work, the hierarchical structure was evolved using Probabilistic Incremental Program Evolution algorithm (PIPE) with specific instructions [6][7] and Ant Programming [8]. In this research, the hierarchical structure is evolved using the Extended Compact Genetic Programming, a tree-structure based evolutionary algorithm. The fine tuning of the parameters encoded in the structure is accomplished using particle swarm optimization (PSO). The proposed method interleaves both optimizations. Starting with random structures and corresponding parameters, it first tries to improve the structure and then as soon as an improved structure is found, it fine tunes its parameters. It then goes back to improving the structure again and, fine tunes the structure and rules' parameters. This loop continues until a satisfactory solution is found or a time limit is reached. The novelty of this paper is in the usage of flexible neural tree model for selecting the important features and for improving the accuracy.

2 The Hierarchical RBF Model

The function set F and terminal instruction set T used for generating a hierarchical RBF model are described as $S = F \bigcup T = \{+_2, +_3, \ldots, +_N\} \bigcup \{x_1, \ldots, x_n\}$, where $+_i (i = 2, 3, \ldots, N)$ denote non-leaf nodes' instructions and taking i arguments. x_1,x_2,...,x_n are leaf nodes' instructions and taking no other arguments. The output of a non-leaf node is calculated as a RBF neural network model (see Fig.1). From this point of view, the instruction $+_i$ is also called a basis function operator with i inputs.

The basis function operator is shown in Fig.1(left). In general, the basis function networks can be represented as

$$y = \sum_{i=1}^{m} \omega_i \psi_i(x; \theta) \tag{1}$$

where $x \in R^n$ is input vector, $\psi_i(x; \theta)$ is ith basis function, and ω_i is the corresponding weights of ith basis function and θ is the parameter vector used in the basis functions. In this research, Gaussian radial basis functions are used,

$$\psi_i(x; \theta) = \prod_{j=1}^{n} exp(-\frac{\parallel x_j - b_j \parallel^2}{{a_j}^2}) \tag{2}$$

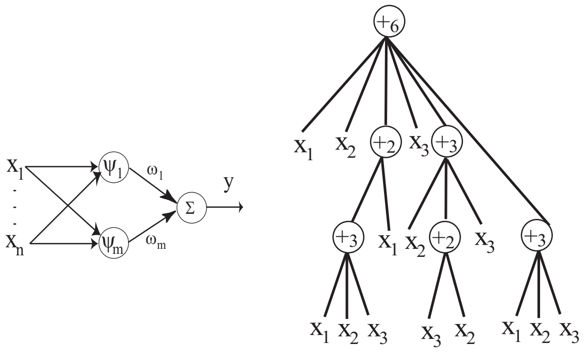

Fig. 1. A basis function operator (left), and a tree-structural representation of a hierarchical RBF neural network with function instruction set $F = \{+_2, +_3, +_4, +_5, +_6\}$, and terminal instruction set $T = \{x_1, x_2, x_3\}$ (right)

and the number of basis functions used in hidden layer is same with the number of inputs, that is, $m = n$.

In the creation process of HRBF tree, if a nonterminal instruction, i.e., $+_i (i = 2, 3, 4, \ldots, N)$ is selected, i real values are randomly generated and used for representing the connection strength between the node $+_i$ and its children. In addition, $2 \times n^2$ adjustable parameters a_i and b_i are randomly created as Gaussian radial basis function parameters. The output of the node $+_i$ can be calculated by using (1). The overall output of HRBF tree can be computed from left to right by depth-first method, recursively.

2.1 Tree Structure Optimization.

Finding an optimal or near-optimal neural tree is formulated as a product of evolution. In our previously studies, the Genetic Programming (GP), Probabilistic Incremental Program Evolution (PIPE) have been explored for structure optimization of the FNT [6][7]. In this paper, the Extended Compact Genetic Programming (ECGP) [9] is employed to find an optimal or near-optimal HRBF structure.

ECGP is a direct extension of ECGA to the tree representation which is based on the PIPE prototype tree. In ECGA, Marginal Product Models (MPMs) are used to model the interaction among genes, represented as random variables, given a population of Genetic Algorithm individuals. MPMs are represented as measures of marginal distributions on partitions of random variables. ECGP is based on the PIPE prototype tree, and thus each node in the prototype tree is a random variable. ECGP decomposes or partitions the prototype tree into sub-trees, and the MPM factorises the joint probability of all nodes of the prototype tree, to a product of marginal distributions on a partition of its sub-trees. A greedy search heuristic is used to find an optimal MPM mode under the framework of minimum encoding inference. ECGP can represent the probability distribution for more than one node at a time. Thus, it extends PIPE in that the interactions among multiple nodes are considered.

2.2 Parameter Optimization with PSO.

The Particle Swarm Optimization (PSO) conducts searches using a population of particles which correspond to individuals in evolutionary algorithm (EA). A population of particles is randomly generated initially. Each particle represents a potential solution and has a position represented by a position vector $\mathbf{x_i}$. A swarm of particles moves through the problem space, with the moving velocity of each particle represented by a velocity vector $\mathbf{v_i}$. At each time step, a function f_i representing a quality measure is calculated by using $\mathbf{x_i}$ as input. Each particle keeps track of its own best position, which is associated with the best fitness it has achieved so far in a vector $\mathbf{p_i}$. Furthermore, the best position among all the particles obtained so far in the population is kept track of as $\mathbf{p_g}$. In addition to this global version, another version of PSO keeps track of the best position among all the topological neighbors of a particle. At each time step t, by using the individual best position, $\mathbf{p_i}$, and the global best position, $\mathbf{p_g(t)}$, a new velocity for particle i is updated by

$$\mathbf{v_i(t+1)} = \mathbf{v_i(t)} + c_1\phi_1(\mathbf{p_i(t)} - \mathbf{x_i(t)}) + c_2\phi_2(\mathbf{p_g(t)} - \mathbf{x_i(t)}) \tag{3}$$

where c_1 and c_2 are positive constant and ϕ_1 and ϕ_2 are uniformly distributed random number in [0,1]. The term $\mathbf{v_i}$ is limited to the range of $\pm\mathbf{v_{max}}$. If the velocity violates this limit, it is set to its proper limit. Changing velocity this way enables the particle i to search around its individual best position, $\mathbf{p_i}$, and global best position, $\mathbf{p_g}$. Based on the updated velocities, each particle changes its position according to the following equation:

$$\mathbf{x_i(t+1)} = \mathbf{x_i(t)} + \mathbf{v_i(t+1)}. \tag{4}$$

2.3 Procedure of the general learning algorithm.

The general learning procedure for constructing the HRBF network can be described as follows.

1) Create an initial population randomly (HRBF trees and its corresponding parameters);
2) Structure optimization is achieved by using ECGP algorithm;
3) If a better structure is found, then go to step 4), otherwise go to step 2);
4) Parameter optimization is achieved by the PSO algorithm as described in subsection 2. In this stage, the architecture of HRBF model is fixed, and it is the best tree developed during the end of run of the structure search. The parameters (weights and flexible activation function parameters) encoded in the best tree formulate a particle.
5) If the maximum number of local search is reached, or no better parameter vector is found for a significantly long time then go to step 6); otherwise go to step 4);
6) If satisfactory solution is found, then the algorithm is stopped; otherwise go to step 2).

Table 1. Comparison of results on Iris data

Model	Recognition rate on total data set (%)
RBF [12]	95.3
HRBF (this paper)	99.5

Table 2. Results of ten runs on Iris data

	1	2	3	4	5	6	7	8	9	10	Average
Misclassification	1	1	0	1	0	1	1	1	1	0	0.7
Recognition rate (%)	99.3	99.3	100	99.3	100	99.3	99.3	99.3	99.3	100	99.5
Features	3	4	3	4	4	3	4	4	3	4	3.6
Parameters	64	60	84	108	108	60	64	84	104	108	84.4
Training time (minutes)	2	6	15	7	8	9	7	4	5	11	7.4

Table 3. Five-Fold cross validation for Iris data

	1	2	3	4	5	Average (%)
Training patterns	120	120	120	120	120	120
Misclassification (training)	0	0	0	0	0	0
Recognition rate (training)(%)	100	100	100	100	100	100
Testing patterns	30	30	30	30	30	30
Misclassification (testing)	0	0	1	0	1	0.4
Recognition rate (testing)(%)	100	100	96.7	100	96.7	98.68

3 Feature Selection and Classification Using Hierarchical RBF Paradigms

3.1 Iris Data Classification

The iris data is a common benchmark in classification and pattern recognition studies [10]. It contains 50 measurements of four features from each of the three species Iris setosa, Iris versicolor, and Iris virginica [11]. We label the species 1, 2, and 3, respectively, which gives a 5×150 pattern matrix of observation vectors

$$Z_k^T = [x_1^k, x_2^k, x_3^k, x_4^k, c_k], c_k \in 1, 2, 3, k = 1, 2, \ldots, 150 \tag{5}$$

where $x_1^k, x_2^k, x_3^k, x_4^k$ are the sepal length, sepal width, petal length, and petal width, respectively.

The iris data is a famous benchmark to test the performance of classifier systems. In our computer simulations, we normalized each attribute value into a real number in the unit interval. Table 1 shows the classification results of conventional RBF and the proposed HRBF network. For the iris example, we also used 150 patterns to design a HRBF classifier system via the proposed algorithm. The used instruction set is $F = \{+_2, +_3, x_1, x_2, x_3, x_4\}$. Table 2 shows the results of ten runs (i.e. ten different initializations of parameters). To estimate the performance of the proposed method on unseen data, the five-fold cross-validation was performed on the iris data. In the five-fold cross-validation

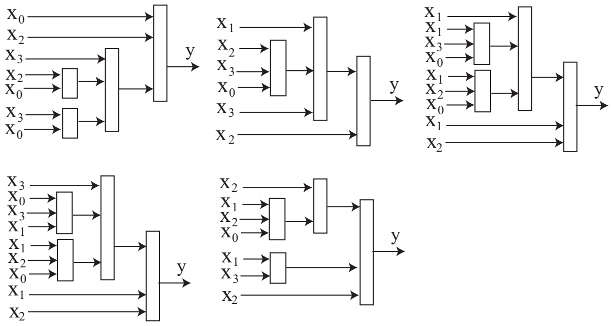

Fig. 2. The evolved optimal HRBF architectures for five-fold cross-validation (Iris data)

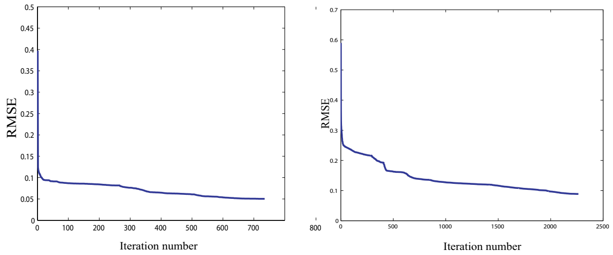

Fig. 3. The convergence performance of five-fold cross validation test 3 for Iris data (left), and for Wine data (right).

experiment, the normalized iris data were divided into five disjoint groups containing 30 different patterns each, with ten patterns belonging to each class. Then we derived HRBF models via the proposed method on all data outside one group and tested the resulting HRBF classifier on the data inside that group. Finally, five HRBF classifiers were derived. The evolved hierarchical architectures for five-fold cross-validation are shown in Fig.2. The convergence performance of five-fold cross validation test 3 is shown in Fig.3(left). Table 3 reports the results of five-fold cross validation. The average classification result is 100.0% correct (no misclassifications) on the training data and 98.68% correct (about 0.4 misclassification) on the test data.

3.2 Wine Data Classification

In this section, the proposed HRBF model is applied to wine data. The wine data set is a 13-dimensional problem with 178 samples from three classes. We chose this data set because it involves many continuous attributes. In our computer simulations, we normalized each attribute value into a real number in the unit interval.

Table 4. Comparison of results on Wine data

Model	Recognition rate on total data set (%)
RBF [12]	98.89
HRBF (this paper)	99.6

Table 5. Results of ten runs on Wine data

	1	2	3	4	5	6	7	8	9	10	Average
Misclassification	1	1	1	0	1	0	1	1	0	1	0.7
Recognition rate (%)	99.4	99.4	99.4	100	99.4	100	99.4	99.4	100	99.4	99.6
Features	5	4	4	5	5	6	4	6	6	4	4.9
Parameters	85	60	64	107	84	113	64	108	79	84	84.8
Training time (minutes)	6	5	13	12	9	10	7	8	12	11	10.3

Table 6. Five-Fold cross validation for Wine data

	1	2	3	4	5	Average (%)
Training patterns	136	144	144	144	144	142.4
Misclassification (training)	0	0	0	0	0	0
Recognition rate (training)(%)	100	100	100	100	100	100
Testing patterns	42	34	34	34	34	35.6
Misclassification (testing)	0	0	1	0	0	0.2
Recognition rate (testing)(%)	100	100	97.1	100	100	99.4

Table 4 shows the classification results of conventional RBF and the proposed HRBF model. For the wine example, we also used 178 patterns to design a HRBF classifier system via the proposed algorithm. The used instruction set is $F = \{+_2, +_3, +_4, x_1, x_2, \ldots, x_{13}\}$. Table 5 shows the results of ten runs (i.e. ten different initializations of parameters). To estimate the performance of the proposed method on unseen data, the five-fold cross-validation was performed on the Wine data. In the five-fold cross-validation experiment, the normalized Wine data were divided into five disjoint groups. Then we derived HRBF models via the proposed method on all data outside one group and tested the resulting HRBF classifier on the data inside that group.

Finally, five HRBF classifiers were derived. The evolved hierarchical architectures for five-fold cross-validation are shown in Fig.4. The convergence performance of five-fold cross validation test 3 is shown in Fig.3 (right). Table 6 reports the results of five-fold cross validation. The average classification result is 100.0% correct (no misclassifications) on the training data and 99.4% correct (about 0.2 misclassification) on the test data.

4 Conclusions

Based on a novel representation and calculation of the hierarchical RBF models, an approach for evolving the HRBF was proposed in this paper. The hierarchical architecture and inputs selection method of the HRBF were accomplished using

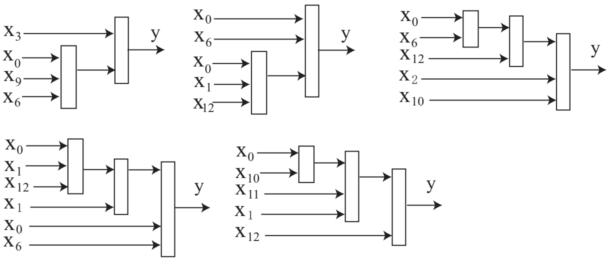

Fig. 4. The evolved optimal HRBF architectures for five-fold cross-validation (Wine data)

PIPE algorithm, and the free parameters embedded in the HRBF model were optimized using a PSO algorithm. Simulation results shown that the evolved HRBF models are effective for the classification of Iris and Wine data. Our future works will concentrate on,

- Improving the convergence speed of the proposed method by parallel implementation of the algorithm;
- Applying the proposed approach for more complex problems.

Acknowledgment

This research was partially supported the Natural Science Foundation of China under contract number 60573065, and The Provincial Science and Technology Development Program of Shandong under contract number SDSP2004-0720-03.

References

1. L. Ohno-Machado, "Medical Applications of Artificial Neural Networks: Connectionist Model of Survival". Ph.D Dissertation, Stanford University, 1996.
2. R. Erenshteyn, P. Laskov, "A multi-stage approach to fingerspelling and gesture recognition", Proceedings of the Workshop on the Integration of Gesture in Language and Speech, 185-194, 1996.
3. N. A. Mat Isa, Mashor, M. Y., and Othman, N. H., "Diagnosis of Cervical Cancer using Hierarchical Radial Basis Function (HiRBF) Network", In Sazali Yaacob, R. Nagarajan, Ali Chekima (Eds.), Proceedings of the International Conference on Artificial Intelligence in Engineering and Technology, 458-463, 2002.
4. S. Ferrari, M. Maggioni, N. Alberto Borghese, "Multiscale Approximation With Hierarchical Radial Basis Functions Networks", IEEE TRANSACTIONS ON NEURAL NETWORKS, 15(1):178-188, 2004.
5. S. Ferrari, I. Frosio, V. Piuri, and N. Alberto Borghese, "Automatic Multiscale Meshing Through HRBF Networks", IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, 54(4):1463-1470, 2005.
6. Y. Chen, B. Yang and J. Dong, "Nonlinear System Modeling via Optimal Design of Neural Trees", International Journal of Neural Systems, 14(2):125-137, 2004.

7. Y. Chen, B. Yang, J. Dong, and A. Abraham, "Time-series Forecasting using Flexible Neural Tree Model", *Information Science*, Vol.174, Issues 3/4, pp.219-235, 2005.
8. Y. Chen, B. Yang, J. Dong, "Automatic Design of Hierarchical TS-FS Models using Ant Programming and PSO algorithm", The Eleventh International Conference on Artificial Intelligence: Methodology, Systems, Applications, LNCS, 3192, 285-294, 2004.
9. K. Sastry and D. E. Goldberg. "Probabilistic model building and competent genetic programming", In R. L. Riolo and B. Worzel, editors, Genetic Programming Theory and Practise, chapter 13, pp. 205-220. Kluwer, 2003.
10. H. Ishibuchi, T. Murata, and I. B. Turksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems", Fuzzy Sets Syst., vol. 89, 135-150, 1997.
11. E. Anderson, "The Irises of the Gaspe peninsula," Bull. Amer. Iris Soc., vol. 59, 2-5, 1935.
12. Y.-J. Oyang, S.-C. Hwang, Y.-Y. Ou, C.-Y. Chen, and Z.-W. Chen, "Data Classification with the Radial Basis Function Network Based on a Novel Kernel Density Estimation Algorithm", IEEE Transactions on Neural Networks, 225-236, 2005.