Proceedings of the 38th
Conference on Decision & Control
Phoenix, Arizona USA • December 1999

Face Recognition Using Radial Basis Function (RBF) Neural Networks

Meng Joo Er, *Member, IEEE* , Shiqian Wu and Juwei Lu
Intelligent Machine Research Laboratory
School of Electrical and Electronic Engineering
Nanyang Technological University
Block S1, Nanyang Drive, Singapore 639798.
Tel : (65) 7904529 , FAX : (65) 7912687
Email: emjer@ntu.edu.sg

Abstract — This paper presents some new results on face recognition using Radial Basis Function (RBF) Neural Networks. First, face features are extracted by discriminant eigenfeatures. Then, a general approach, which determines the initial structure and parameters of RBF neural networks, is presented. A hybrid learning algorithm is used to dramatically decrease the dimension of the search space in the gradient method, which is crucial on optimization of high-dimension problem. Experimental results conducted on the ORL database image of Cambridge University show that the error rate is 1.5 % which is a tremendous improvement over the best existing result of 3.83 %.

1 Introduction

Recently, face recognition has received significant attention from the communities of computer vision, neural networks and signal processing [1]. Although many researchers have investigated a number of issues related to face recognition by human beings and machines, it is still difficult to design an automatic system for the task because face images are highly variable. As stated by Moeses et al [2], "Variations between images of the same face due to illumination and viewing direction are almost always larger than image variations due to changes in face identity". This makes face recognition a challenging task. In our opinion, two issues are central:

(1). What features can be used to represent a face under environmental changes?

(2). How to classify a new face image using the chosen representation?

For (1), many successful face detection and feature extraction methods have been developed [3]-[5]. Our research in this paper focuses on issue (2), i.e. we aim to find a general and efficient way for the task of face classification based on conventional feature extraction methods.

In many face recognition systems, the Nearest Neighbor (NN) is widely used for classification. Yet, NN is short of generalization and the efficiency of NN greatly depends on the number of available training samples. Neural-networks-based method has been proven to have many advantages for classification such as incredible generalization and good learning ability. However, as large numbers of vectors are invariably created to represent various face features, the neural system used for face recognition possesses the following characteristics:

- High dimension.
- Small sample sets.

The frequently used approach to deal with the high-dimension problem is to employ tree-structured adaptive networks [6]-[7]. The idea therein is that for most of the data, only a few dimensions of the input are necessary to compute the desired output and additional input dimensions are incorporated only when needed. The main advantages of this topology are that the learning algorithm is fast [6] and it can find a network structure size suitable for the classification of complex patterns through structure adaptation so as to overcome the complex input problem [7].

In this paper, face recognition is implemented via RBF neural networks. In order to reduce the structure complexity and control the computational burden, two strategies are adopted: First, face features are extracted by discriminant eigenfeatures through two processing modules, i.e. eigenface and Fisher's Linear Discriminant (FLD). Next, a hybrid learning algorithm is presented to train the RBF neural networks so that the dimension of the search space is significantly decreased in the gradient method. Furthermore, an approach whereby some domain knowledge is encapsulated in the choice of structure and parameters of RBF neural networks before learning takes place is presented.

In the following analysis, RBF neural networks are briefly introduced in Section 2. Feature extraction using discriminant eigenfeature is proposed in Section 3. Following structure determination and initialization in Section 4, a hybrid learning algorithm is presented in Section 5. Experimental results based on the ORL database are reported in Section 6. Finally, conclusions are drawn in Section 7.

2 Radial Basis Function (RBF) Neural Networks

RBF neural networks have recently attracted extensive research interests in the community of neural networks

0-7803-5250-5/99/$10.00 © 1999 IEEE

because: (1) They are universal approximators [8]; (2) Their learning speed is fast because of local-tuned neurons [9]; (3) They have more compact topology than other neural networks [10]. (4) They possess the best approximation property [11].
The basic structure of RBF neural networks is shown in Fig. 1.

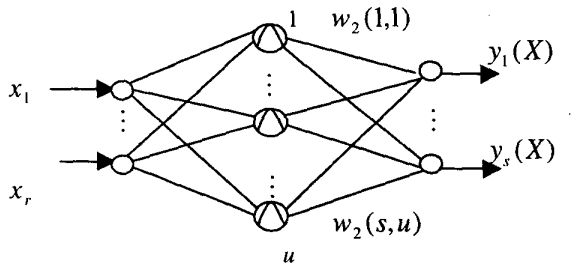


Fig. 1 RBF neural networks

The output of the *i*th RBF unit is

$$R_i(X) = R_i(\frac{\|X - C_i\|}{\sigma_i}) \quad i=1,2,\cdots u \qquad (1)$$

where X is an r-dimensional input vector, C_i is a vector with the same dimension as X, u is the number of hidden units and $R_i(.)$ is the ith RBF unit. Typically, $R_i(.)$ is chosen as a Gaussian function

$$R_i(X) = \exp[-\frac{\|X - C_i\|^2}{\sigma^2}] \qquad (2)$$

The jth output $y_j(X)$ of RBF neural networks is

$$y_j(X) = b(j) + \sum_{i=1}^{u} R_i(X) \times w_2(j,i) \qquad (3)$$

where $w_2(j,i)$ is the weight of the ith receptive field to the jth output and $b(j)$ is the bias of the jth output.
In the following analysis, the bias is not considered in order to reduce the network complexity. Hence, the jth output $y_j(X)$ of an RBF neural network is

$$y_j(X) = \sum_{i=1}^{u} R_i(X) \times w_2(j,i) \qquad (4)$$

3 Feature Extraction Using Discriminant Eigenfeatures

Here, we use the fisherface or discriminant eigenfeature method proposed by Belhumeur [4] for feature extraction. This method aims at overcoming the drawback of eigenface method by integrating FLD criteria into the eigenface method, while retaining the advantages of eigenface method to project faces from the high-dimensional image space to a significantly lower dimensional feature space. Consequently, the fisherface-based features are the most discriminant eigenfeatures and are well suitable for classification purpose. The most discriminant eigenfeature can be obtained through two processing modules, namely eigenfaces and FLD.

3.1 Eigenfaces

Denote the training set of N face images by $\{z_1, z_2, \cdots, z_N\}$. The Principle Components Analysis (PCA) is applied to the set of training images to find the N eigenvectors of the covariance matrix $\frac{1}{N}\sum_{n=1}^{N}(z_n - \bar{z})(z_n - \bar{z})^T$ where $\bar{z} = \frac{1}{N}\sum_{n=1}^{N} z_n$ is the average of the ensemble. The eigenvalues of the covariance matrix are then calculated. Let ϕ_k be the eigenvector corresponding to the kth largest eigenvalue. The set of the first $N'(\leq N)$ orthonormal vectors, $\Phi = [\phi_1, \cdots, \phi_{N'}]$, forms a basis of an eigenface space. Thus, for any face image z_i, its corresponding eigenface-based feature x_i can be obtained by projecting z_i into the eigenface space.

3.2 Fisherfaces

The FLD is applied to the projection of the set of training samples in eigenface space $\{x_1, x_2, \cdots x_N\}$. The method finds an optimal subspace for classification in which the ratio of the between-class scatter and the within-class scatter is maximized. Let the between-class scatter matrix be defined as:

$$S_B = \sum_{i=1}^{c} N_i(x_i - \bar{x})(x_i - \bar{x})^T \qquad (5)$$

and the within-class scatter matrix be defined as:

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in N_i} (x_k - \bar{x}_i)(x_k - \bar{x}_i)^T \qquad (6)$$

where $\bar{x}_i$ is the mean of the ith class, and N_i is the number of samples in the ith class. The optimal subspace for classification, $E_{optimal}$ can be obtained by FLD as follows:

$$E_{optimal} = \arg\max_{E} \frac{|E^T S_B E|}{|E^T S_W E|} = [e_1, e_2, \cdots, e_M] \qquad (7)$$

where $[e_1, e_2, \cdots, e_M]$ is the set of generalized eigenvectors of S_B and S_W corresponding to the set of decreasing generalized eigenvalues, and $M \leq N'$. Thus, the feature vectors p for any query face images z and training face images in the most discriminant sense can be calculated as:

$$p = E_{optimal}^{T} \ \Phi^{T} \ (z - \bar{z}) \qquad (8)$$

For all of the face images to be studied in the sequel, we will use the feature vectors instead of their corresponding original intensity data.

4 Structure Determination and Initialization

4.1 Structure Determination

In order to use the RBF neural networks for face recognition, we set the number of inputs equal to that of features (i.e., dimension of the input space), while the number of outputs is set to that of classes (see Fig. 1). The selection of RBF nodes is as follows:

(1) Initially, we set the number of RBF units equal to that of output, i.e., we assume that each class has just one subclass.

(2) For each class k, k=1,2$\cdots$ s, the center of RBF nodes is selected as the mean value of the sample data belonging to the class, i.e.:

$$C^k = \frac{\sum_{i=1}^{N^k} p^k(r,i)}{N^k} \qquad (9)$$

where $p^k(r,i)$ is the ith sample with r-dimension belonging to class k and N^k is the number of patterns in class k.

(3) For any class k, compute the distance d_k from the mean to the furthest point p_f^k belonging to class k:

$$d_k = | p_f^k - C^k | \qquad (10)$$

(4) For any class k:

- Calculate the distance $dc(k,j)$ between the mean of class k and the mean of other classes as follows:

$$dc(k,j) = | C^k - C^j | \quad j=1,2...s, j \neq k \qquad (11)$$

- Find

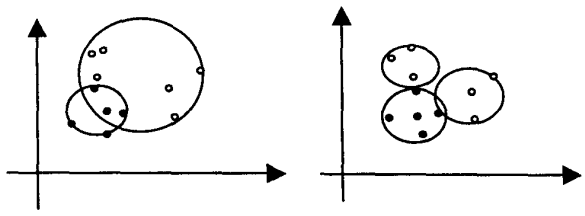
$$d_{\min}(k,l) = \arg \ \min(dc(k,j)) \qquad (12)$$

- Check the relationship between $d_{\min}(k,l)$ and d_k, d_l:

Case 1: If $d_k + d_l \leq d_{\min}(k,l)$, class k has no overlapping with other classes.

Case 2: If $d_k + d_l > d_{\min}(k,l)$, class k has overlapping with other classes and misclassifications may occur in this case.

(5) For all the training data, check how the data are classified. There is no problem if two functions belonging to the same class overlap significantly. If two classes are misclassified greatly, we should consider splitting one or two of the classes into two, see Fig. 2.



(a) (b)

Fig. 2 One class splits into two classes

(6) Repeat (2) ~ (5), until all the training sample patterns are classified satisfactorily.

(7) The mean values of the classes are selected as the centers of RBF units.

4.2 Width Estimation

Essentially, RBF classifiers utilize overlapping localized regions formed by simple kernel functions to create complex decision regions while the amount of overlapping is controlled by the widths of RBF units [12]. If no overlapping occurs, the system will not generalize well [13]. However, if the widths are too large, the interaction of different classes will be great and the output belonging to the class will not be so significant, while the output of other classes may be large so that it will lead to misclassification greatly. Here, we present two criteria for the choice of widths of RBF units.

1) *Majority Criterion*: In any class, each datum should have more than 50% confidence level for the class it belongs to. The detailed calculations are as follows:

First, d_k, the distance from the mean to the furthest point belonging to class k, is calculated according to Eq. (10). Next, define the width σ_1^k of class k considering the confidence level as

$$\sigma_1^k = d_k / \sqrt{\beta} \qquad (13)$$

where β is called confidence coefficient.

We can see from Eq. (2) that if the furthest sample in class k should have output 0.5 (we say that the confidence level is 50%), then β is 0.7.

The choice of β is determined by the distribution of sample patterns. If the data are scattered greatly, but the centers are close, a small β should be selected.

(2) *Overlapping Criterion*: For any class k, the choice of width σ_2^k considering the overlapping of the neighboring class l is determined by

$$\sigma_2^k = \eta \times d_{\min}(k,l) \qquad (14)$$

where η is an overlapping factor that controls the overlap of different classes.

(3) According to Eq. (13) and Eq. (14), the width of class k is finally determined by

$$\sigma^k = \max\ (\sigma_1^k,\ \sigma_2^k) \qquad (15)$$

5 Hybrid Learning Algorithm

The adjustment of RBF unit parameters is a nonlinear process while the identification of weight w_2 is a linear one. Though we can apply the gradient method to find the entire set of optimal parameters, the method is generally slow and likely to become trapped in local minima. So, we propose a hybrid learning algorithm, which combines the gradient method and the Linear Least Squared (LLS) method to adjust the parameters, see [13] and [14] for details.

5.1 Weight Adjustment

Let r and s be the number of inputs and outputs respectively, and suppose that u RBF units are generated for all training patterns according to Section 4.1. For any input $P_i(p_{1i}, p_{2i}, \cdots p_{ri})$, the jth output y_j of the system can be calculated according to Eq. (4). Rewriting Eq. (4) in a more compact form:

$$W_2 \times R = Y \qquad (16)$$

Our objective is the following: Given $R \in \Re^{u \times N}$ and $T = (t_1, t_2, \cdots t_s)^T \in \Re^{s \times N}$, where N is the total number of sample patterns, T is the target matrix consisting of 1's and 0's with exactly one per column that identifies the processing unit to which a given exemplar belongs. Furthermore, given the following relationship:

$$E = \| T - Y \| \qquad (17)$$

find an optimal coefficient matrix $W_2^* \in \Re^{s \times u}$ such that the error energy $E^T E$ is minimized. This problem can be solved by the well-known LLS method by approximating

$$W_2^* \times R = T \qquad (18)$$

The optimal W_2^* is given by:

$$W_2^* = T \times R^+ \qquad (19)$$

where R^+ is the pseudoinverse of R and is given by:

$$R^+ = (R^T R)^{-1} R^T \qquad (20)$$

5.2 Modification of Parameters of RBF Units

Here, the parameters (center and width) are adjusted by taking the negative gradient of the error function, E^n which is given by:

$$E^n = \frac{1}{2}\sum_{k=1}^{s}(t_k^n - y_k^n)^2 \qquad n = 1,2....N \qquad (21)$$

where y_k^n and t_k^n represent the kth real output and target output at the nth pattern respectively.

The error rate for each output y_k^n can be calculated readily from Eq. (21):

$$\frac{\partial E^n}{\partial y_k^n} = -(t_k^n - y_k^n) \qquad (22)$$

For the internal nodes (center C and width σ), the error rate can be derived by the chain rule as follows:

$$\Delta C^n(i,j) = -\xi \frac{\partial E^n}{\partial C(i,j)^n} \qquad i=1,2....r,\ \ j=1,2\ ....u$$

$$= -\xi \frac{\partial E^n}{\partial y_k^n}\frac{\partial y_k^n}{\partial R_j^n}\frac{\partial R_j^n}{\partial C(i,j)^n}$$

$$= 2\xi \sum_{k=1}^{s} y_k^n \cdot w_2(k,j) \cdot R_j^n \cdot (P_i^n - C(i,j)) / \sigma_j^2 \qquad (23)$$

$$\Delta \sigma_j{}^n = -\xi \frac{\partial E^n}{\partial \sigma_j{}^n}$$

$$= -\xi \frac{\partial E^n}{\partial y_k^n}\frac{\partial y_k^n}{\partial R_j^n}\frac{\partial R_j^n}{\partial \sigma_j^n}$$

$$= 2\xi \sum_{k=1}^{s} y_k^n \cdot w_2(k,j) \cdot R_j^n \cdot (P_i^n - C(i,j))^2 / \sigma_j^3 \qquad (24)$$

where $\Delta C^n(i,j)$ is the error rate of the center of the ith input variable of the jth prototype at the nth training pattern, $\Delta \sigma_j^n$ is the error rate of the width of the jth prototype at the nth pattern, P_i^n is the ith input variable at the nth pattern and ξ is the learning rate.

5.3 Learning Procedure

The hybrid learning procedure is summarized as follows (see [13] and [14] for details): Each epoch of the learning procedure is composed of a forward pass and a backward pass. In the forward pass, we supply input data and functional signals to calculate the output R_j of the jth RBF unit. Then, the weight w_2 is modified according to Eqs.(19) and (20). After identifying the weight, the functional signals keep moving forward till the error measure is calculated. In the backward pass, errors propagate from the output end towards the input end. Keeping the weight fixed, the centers and widths of RBF nodes are modified by the BP learning algorithm according to Eqs.(23) and (24). The learning procedure is shown in Table 1.

Table 1 Two passes in the hybrid learning procedure

	Forward pass	Backward pass
Weight	Linear Least Squared	Fixed
Parameters of RBF units	Fixed	Gradient descent
Signals	Node outputs	Error rate

For given fixed parameters of RBF units, the weights found are guaranteed to be the global optimum due to the choice of the squared error measure [14]. It has much faster convergence speed than the BP method. Thus, for the modification of centers and widths by the BP algorithm, the feedback error always seems to be very small so that the learning rate ξ should be much bigger than usual.

In order to improve the convergence speed and avoid oscillating around the optimum value, ξ is gradually reduced according to the following equation:

$$\xi = \max(lr_{\max} \cdot \gamma^i, lr_{\min}) \qquad (25)$$

where $lr_{\max}$ and $lr_{\min}$ are maximum and minimum learning rates respectively, i is the number of epochs, and γ is a descent coefficient which lies in the range $0 < \gamma < 1$.

Our experiments show that this hybrid learning algorithm can not only decrease the dimension of the search space in the gradient method, but also cut down substantially the convergence time.

6 Experimental Results

Experimental studies are carried out on the ORL database image of Cambridge University. Like the experiment of [15], we also use a database of 400 images of 40 individuals, which contain quite a high degree of variations in expression, pose and facial details. A total of 200 images are used to train and another 200 are used to test, where each person has 5 images. The results are shown in Table 2 and Table 3 respectively.

Table 2 Error rate and parameters *

Number of feature vectors	Training							Testing	
	β	η	$lr_{\max}$	$lr_{\min}$	γ	Epochs	RMSE♠	NOM♣	Error rate ♦
50	0.225 ~0.51	1.0~1.5	1.5×10^5	10^4	0.99	50~80	0.044~ 0.034	1	0.5%
40	0.225 ~0.51	0.9~1.5	8.0×10^4	1.5×10^4	0.95	30~50	0.031~ 0.021	0	0
30	0.225 ~0.51	0.9~1.5	4.0×10^4	2.0×10^3	0.99	40~50	0.049~ 0.04	0	0
20	0.51	0.75~0.9	9.0×10^3	500	0.99	20~50	0.042~ 0.041	0	0
10	0.51	0.75~0.9	5.0×10^3	500	0.99	30	0.047~ 0.046	0	0

* Training data used as training patterns and testing data as testing patterns

♠ RMSE— Root Mean Squared Error

♣ NOM— Number of Misclassifications

♦ Error rate = Number of Misclassifications / Number of Total Testing Patterns

Table 3 Error rate and parameters**

Number of feature vectors	Training							Testing	
	β	η	$lr_{\max}$	$lr_{\min}$	γ	Epochs	RMSE	NOM	Error rate
50	0.51~ 0.357	1.8~ 2.2	5.5×10^5	5.0×10^4	0.99	70~150	0.037~ 0.028	6	3%
40	0.1	0.75	3.0×10^5	2.0×10^4	0.96	40	0.0089	6	3%
30	0.225	0.75 ~0.9					0.0053 ~0.007	6	3%
20	0.225 ~0.1	0.75 ~0.9					0.0035 ~0.003	8	4%
10	0.225 ~0.1	0.7~ 0.9					0.0025 ~0.001	18	9%

** Testing data used as training patterns and training data as testing patterns

Define the average error rate E_{ave} as

$$E_{ave} = \frac{\sum_{i=1}^{m} N_m^i}{mN_t} \quad (26)$$

where m is the number of experimental runs, each one being performed on random partition of the database into two sets, N_m^i is the number of misclassifications for the ith run, and N_t is the number of total testing patterns of each run. The comparison with Convolutional Neural Networks (CNN) approach [15] using the same ORL database in terms of E_{ave} is shown in Table 4.

Table 4 Comparison of CNN and RBF approaches

Approach	E_{ave}
CNN	3.83%
RBF	1.5%

Here, the lowest error rate achieved by RBF neural networks is based on the following conditions: m=2 and the number of feature vectors is 30 and 40 respectively. The way to partition the training set and query set is the same as [15], whereas the value E_{ave} obtained by CNN in Table 4 is based on 3 runs and the size of self-organizing map is 8 and 9 respectively.

7 Conclusions

In [9], an assumption was made that RBF neural networks are best suited for learning to approximate continuous or piecewise continuous, real-valued mapping where the input dimension is sufficiently small. Our experimental results show that RBF neural networks are also excellent in face recognition to cope with the high-dimension problem. In this paper, a feature optimization approach by discriminant eigenfeatures through eigenface and FLD is presented. A general approach is put forward to determine the neural structure and select the initial parameters of RBF units. A hybrid learning algorithm is developed to decrease the dimension of the search space in the gradient method dramatically. Our results show that the error rate is 1.5%. This is a tremendous improvement over the result of [15], which is the best result to our knowledge based on the ORL database.

References

[1] R. Chellappa, S. Sirohey, C. L. Wilson and C. S. Barnes, "Human and Machine Recognition of Faces: A Survey," *CAR-TR-731, CS-TR-3339*, p1-83, 1994.

[2] Y. Moses, Y. Adini and S. Ullman, "Face Recognition: The Problem of Compensating for Changes in Illumination Direction," *Proceedings of the European Conference on Computer Vision*, Vol. A, pp. 286-296, 1994.

[3] M. A. Turk and A. P. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, Vol. 3, No.1, pp. 71-86, 1991.

[4] P. N. Belhumeur, J. P. Hespanha and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No.7, pp. 711-720, 1997.

[5] B. S. Manjunath, R. Chellappa and C. V. D. Malsburg, "A Feature Based Approach to Face Recognition," *Proceedings* of *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 373-378, 1992.

[6] T. D. Sanger, "A Tree-Structured Adaptive Network for Function Approximation in High-Dimensional Space," *IEEE Trans* on *Neural Networks*, Vol. 2, No. 2, pp.285-293, 1991.

[7] H. H. Song and S. W. Lee, "A Self-Organizing Neural Tree for Large-Set Pattern Classification," *IEEE Trans* on *Neural Networks*, Vol. 9, No. 3, pp. 369-380, 1998.

[8] J. Park and J. Wsandberg, "Universal Approximation Using Radial Basis Functions Network," *Neural Computation*, Vol. 3, pp. 246-257, 1991.

[9] J. Moody and C. J. Darken, "Fast Learning in Network of Locally-Tuned Processing Units," *Neural Computation*, Vol. 1, pp. 281-294, 1989.

[10] S. Lee and R. M. Kil, "A Gaussian Potential Function Network with Hierarchically Self-Organizing Learning," *Neural Networks*, Vol. 4, pp. 207-224, 1991.

[11] F. Girosi and T. Poggio, " Networks and the Best Approximation Property," *Biological Cybernetics*, Vol. 63, pp. 169-176, 1990.

[12] M. T. Musavi et al, "On the Training of Radial Basis Function Classifiers," *Neural Networks*, Vol. 5, pp.595-603, 1992.

[13] S. Q. Wu and M.J. Er, "A Novel Learning Algorithm for Dynamic Fuzzy Neural Networks," in *Proc. 1999 American Control Conference.*

[14] J-S. R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference System," *IEEE Trans Syst, Man, Cybern*, Vol. 23, No. 3, pp. 665-684, 1993.

[15] S. Lawrence, C. L. Giles, A. C. Tsoi and A. D. Back, "Face Recognition: A Convolutional Neural Networks Approach," *IEEE Trans* on *Neural Networks*, Special Issue on Neural Networks and Pattern Recognition, Vol.8, No.1, pp.98-113, 1997.