FACE RECOGNITION BASED ON KERNEL RADIAL BASIS FUNCTION NETWORKS

Yuan Wang *Yunde Jia*

Computer Science Department
Beijing Institute of Technology
Beijing 100081, P.R.CHINA
yjiar@bit.edu.cn

Changbo Hu *Matthew Turk*

Computer Science Department
University of California
Santa Barbara, CA 93106, USA
{cbhu, mturk}@cs.ucsb.edu

ABSTRACT

Linear subspace analysis has been extensively applied to face recognition. However, a linear subspace can not describe the nonlinear variations of face images. Alternatively, a kernel feature space can reflect nonlinear information of faces. In this paper we present a new face recognition method based on Radial Basis Function (RBF) networks in kernel space. The face features are extracted in kernel space and then fed into an RBF network, resulting in a face recognition algorithm that is computationally simple and robust. The experimental results show that our algorithm performs better than a traditional RBF network for face recognition.

1 INTRODUCTION

Face recognition is a very challenging research problem due to variations in illumination, facial expression and pose. It has received extensive attention during the past 20 years, not only because of the potential applications in fields such as Human Computer Interaction, biometrics and security, but also because it is a typical pattern recognition problem whose solution would help in solving other classification problems.

A successful face recognition methodology depends largely on the particular choice of the features used by the (pattern) classifier. Linear subspace analysis, such as Principal Component Analysis (PCA) and Fisher Linear Discriminant Analysis (FLDA), has been used for face recognition by many researchers. But these methods only extract features from the input space without considering nonlinear information between the components of input data.

Kernel methods such as Kernel Principal Component Analysis (KPCA) [1,2] and Kernel Fisher Discriminant Analysis (KFDA) [1,4] show better results in face recognition than linear subspace analysis methods. They extract features from a feature space which contains nonlinear information of the input data through kernel methods. Since much of the important information may be contained in the feature space, such as the nonlinear relationships between two or more image pixels, methods for extracting features from kernel feature spbace are very important.

Radial Basis Function (RBF) neural networks [5,9] have been successfully applied to face recognition. Their main advantages are computational simplicity, supported by well-developed mathematical theory, and robust generalization, powerful enough for real-time real-life tasks [11,12]. RBF networks are considered as ideal for practical vision applications by Girosi [13], as they are good at handling problems with sparse, high-dimensional data, and because they use approximation to handle noisy, real-life data.

This paper introduces a new algorithm, called Kernel RBF Networks (KRBF Networks), using RBF networks to extract features from kernel feature space. This network shows better results than RBF networks in our face recognition experiments.

2 RELATED WORK

The idea of Radial Basis Function neural networks (RBF networks) derives from the theory of function approximation. It has been identified as a valuable model by a wide range of researchers [14, 9, 15, 16, 17]. Gutta et al. [5] also applied RBF networks to recognize partial faces.

Kernel-based nonlinear analysis has received much attention in pattern recognition, because the kernel approach can efficiently construct nonlinear relations of the input data in an implicit feature space obtained by the nonlinear kernel mapping. This depends only on inner products in the kernel feature space – the feature space does not need to be computed explicitly. The kernel method was first used in Support Vector Machines (SVM) [18] and then in kernel PCA [1] and kernel Fisher Discriminant Analysis [3].

The rest of this section will introduce RBF networks and kernel feature space.

2.1 RBF Networks

When an RBF network is used as a classifier, as depicted in Figure 1, it typically involves three layers: the input layer, the hidden layer and the output layer.

The input layer receives input data (a face image vector). The hidden layer is used to cluster the input data and extract features. The output nodes give the recognition result.

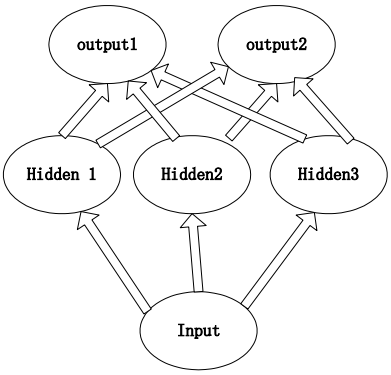

Figure 1. RBF Network

Every hidden-layer node of an RBF classifier represents a class and constructs a hypersurface for each class. These hypersurfaces can be viewed as discriminant functions, where each hidden layer node produces a high value for the class it represents and a low value for all other classes. A Gaussian Radial Basis Function could be a good choice for the hidden layers, because it is a very good similarity function:

$$\Phi(\vec{\mathbf{x}}) = \exp\left(\frac{-\|x_i - \mu_k\|^2}{2h \cdot \sigma_k^{\,2}}\right) \quad (1)$$

Suppose that each hidden layer node is a Gaussian Radial Basis Function (1) and μ_k is the center of the k th class. The closer x_i is to μ_k, the higher the value the Gaussian function will produce. The outputs of the hidden layer can be viewed as a set of features extracted from the input space. That is, the hidden layer is equal to a functional facial base producing some characteristics across the face space.

2.2 Kernel feature space

First, the input data is projected into feature space F by a nonlinear mapping $\Phi : R^N \to R^F$, F >> N ; then we can use the kernel method [1,6] to extract features from the feature space. This is done in a low-dimensional space by defining the inner product of high-dimensional vectors:

$$\Phi(x_i) \cdot \Phi(x_j) = k(x_i, x_j) \quad (2)$$

Here we use a polynomial kernel as an example to show how to project a vector into high-dimensional space and what nonlinear information it will produce.

$$\begin{aligned} &\Phi(x) \cdot \Phi(y) = k(x, y) = (x, y)^2 \\ &k(x, y) \ = (x_1^2, x_1x_2, x_2x_1, x_2^2)(y_1^2, y_1y_2, y_2y_1, y_2^2)^T \end{aligned} \quad (3)$$

where $x = (x_1, x_2)$ and $y = (y_1, y_2)$

Eq. (3) shows the procedure to project from low-dimensional space to high-dimensional space; the vector in high-dimension space contains nonlinear information such as x_1x_2, y_1y_2.

3 KRBF NETWORKS

3.1 The training of KRBF networks

The architecture of KRBF networks is just like that of RBF networks, but the training of these two networks is quite different.

The training of KRBF networks includes three steps:

- Use a kernel k-means algorithm to cluster the input data (instead of the standard k-means algorithm used in RBF networks).
- Train the parameters of the Radial Basis Functions (the hidden layer nodes) according to the training samples of each cluster.
- Train the weights between hidden layer and output layer.

3.1.1 Kernel k-means algorithm

For the input data $x_1, x_2, \cdots, x_N$, the k-means clustering algorithm aims to partition the N samples into *K* clusters, $C_1, C_2, \cdots, C_K$, according to the Euclidean distance, and return the center of each cluster. The kernel k-means algorithm aims to partition samples in the kernel feature space. To achieve the kernel k-means algorithm, the key point is computing Euclidean distance in the kernel feature space.

Suppose that x_i is a face vector, and $u_i = \phi(x_i)$ is the projection of x_i in the feature space. We define: $\phi(x_i) \cdot \phi(x_j) = k(x_i, x_j)$, then we can get a Euclidean distance formula (4) in feature space:

$$D^2(u_i, u_j) = \left\|\phi(x_i) - \phi(x_j)\right\|^2$$
$$= \phi^2(x_i) - 2\phi(x_i)\cdot\phi(x_j) + \phi^2(x_j) \quad (4)$$
$$= k(x_i, x_i) - 2k(x_i, x_j) + k(x_j, x_j)$$

If z_k are the centers of the classes in kernel space, then

$$z_k = \frac{1}{|C_k|}\sum_{i=1}^{N}\delta(u_i, C_k)u_i \quad (5)$$

where $\delta(u_i, C_K)$ is an indicator function and $|C_K|$ is the number of samples in C_K:

$$\delta(u_i, C_K) = \begin{cases} 1 & D(u_i, Z_k) < D(u_i, Z_j) \quad for \quad all \quad j \neq k \\ 0 & otherwise \end{cases} \quad (6)$$

$$|C_k| = \sum_{i=1}^{N}\delta(u_i, C_k)$$

then the distance between a vector, u_i , and a center , z_k ,in feature space is:

$$D^2(u_i, z_k) = \left\| u_i - \frac{1}{|C_k|}\sum_{j=1}^{N}\delta(u_j, C_k)u_j \right\|^2$$
$$= k(x_i, x_i) - \frac{2}{|C_k|}\sum_{j=1}^{N}\delta(u_j, C_k)k(x_i, x_j) + \quad (7)$$
$$\frac{1}{|C_k|^2}\sum_{j=1}^{N}\sum_{l=1}^{N}\delta(u_j, C_k)\delta(u_l, C_k)k(x_j, x_l)$$

The kernel k-means algorithm comprises the following steps:

1. Randomly initialize the samples to *K* clusters, assigning $\delta(u_i, C_K)$.
2. Compute the distances from each sample to each center with formula (7) and re-assign $\delta(u_i, C_K)$ to the new value.
3. Repeat step 2 until it converges.
4. For each cluster C_K, select the sample that is closest to the center as the representative of C_K.

3.1.2. Training the parameters of the Radial Basis Functions

Here we select the Gaussian RBF (1) with two parameters, μ_k and σ_k , for the hidden layer. μ_k are the centers of clusters, the mean values of the samples of each cluster. It is very difficult to compute μ_k in feature space, but we can easily compute $\|x_i - \mu_k\|^2$ by (7). σ_k can be achieved by computing the variance of samples of each cluster in the kernel feature space.

3.1.3. Training the weights between hidden and output layers

It is obvious that the outputs of the hidden units lie between 0 and 1. The closer the input is to the center of the Gaussian, the larger the response of the node will be. Z_j , an output unit of the output layer, is given by:

$$Z_j = \sum_i w_{ij} y_i + w_{0j} \quad (8)$$

where Z_j is the output of the j th output node, y_i is the output of the i th hidden layer node, and w_{ij} is the weight connecting the i th hidden layer node to the j th output node. In this paper, the number of nodes in the output layer corresponds to the number of people to be identified, and each node corresponds to one person. When training the network, we define the output in advance with the constraint that an input vector X is classified as belonging to the class associated with the output node j with the largest output, Z_j . Then we can obtain the weight w_{ij} through pseudoinverse techniques.

3.2 Face recognition with KRBF Networks

When the test face vector X is passed through the hidden layer, we can extract the features of the test face from the feature space. We then classify the feature by Eq. (8); the largest output layer node determines the class to which the input face belongs.

4 EXPERIMENTS

Our experiments are performed on two benchmarks: one is the ORL database and the other is a dataset of the FERET database [10]. On each benchmark, we reduce the face images from 112×92 to 28×23 in order to compute efficiently.

There is no theoretic selection scheme for selecting kernel function for kernel method. We chose a polynomial function as the kernel function for space projection.

In following experiments, KRBF networks will be compared with RBF networks on face recognition.

4.1 Experiment on Cambridge ORL database

The ORL face database used in this paper is composed of 400 images of size 112×92. There are 40 persons, with 10 images of each person. The images are taken at different times, lighting and facial expressions. The faces are in up-right position of frontal view, with slight left-right rotation.

Figure 2. ORL face samples

In the experiment of Figure3, each set of ten images for a person is randomly partitioned into a training subset of five images and a test set of the other five images. Both methods use the same training and test data.

This experiment makes a comparison of recognition rates between KRBF networks and RBF networks, using different numbers of hidden layer nodes. We can see the result in Figure 3. The recognition rate increases with the number of hidden layer nodes, and KRBF networks achieve higher recognition rates than RBF networks. With KRBF networks, the quadric polynomial kernel achieves only slightly better recognition rates than does the cubic polynomial kernel.

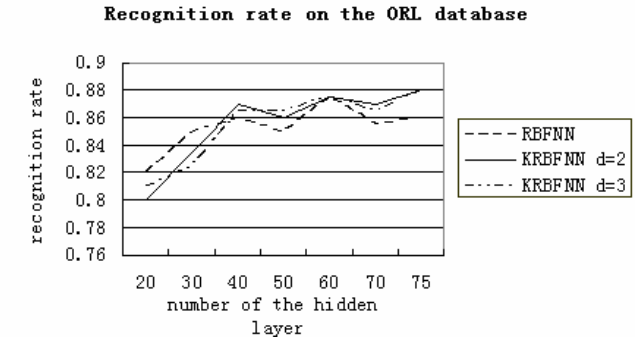

Figure 3. ORL database recognition rate

In the following experiment, each set of ten images for a person is randomly partitioned into training subsets of 2, 4, 6, and 9 images, and at the same time, test sets of the other 8, 6, 4, and 1 images. The same training and test data are used for both methods.

The results of this experiment on the ORL database are shown in Figure 4. The horizontal axis represents different numbers of training and testing images. We chose the best recognition rates of KRBF networks and RBF networks. The results show that KRBF networks achieve better recognition rates than RBF networks as the amount of training data increases and when the degree of polynomial kernel function is equal to 2,3,4 respectively, these KRBF networks achieve very close recognition rates on ORL database.

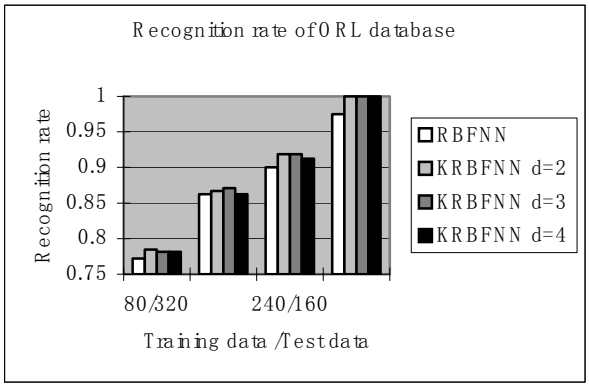

Figure 4. ORL database recognition rate

4.2 Experiment on FERET dataset

There are 70 persons in this dataset of the FERET face database. Each person has six different frontal-view images. There are three different illuminations and two different facial expressions in each illumination. Figure 5 shows some samples.

Figure 5. FERET dataset face samples

Each set of six images for a person is randomly partitioned into a training subset of four images and a test set of the other two images. Both methods use the same training and test data.

This experiment makes a comparison of recognition rates between KRBF networks and RBF networks, using different numbers of hidden layer nodes. We can see the result in Figure 6. The recognition rate increases with the number of hidden layer nodes, and KRBF networks achieve higher recognition rates than RBF networks. With KRBF networks, the cubic polynomial kernel

achieves better recognition rates than does the quadric and biquadratic polynomial kernels.

In the next experiment, each set of six images for a person is randomly partitioned into training subsets of 1, 2, 3, 4, and 5 images, and at the same time, test sets of the other 5, 4, 3, 2, and 1 images. The same training and test data are used for both methods.

The results of this experiment on the FERET dataset are shown in Figure 5. We select cubic polynomial kernel for KRBF networks. The horizontal axis represents different numbers of training and testing images. We chose the best recognition rates of KRBF networks and RBF networks. The results show that KRBF networks achieve better recognition rates than RBF networks as the amount of training data increases.

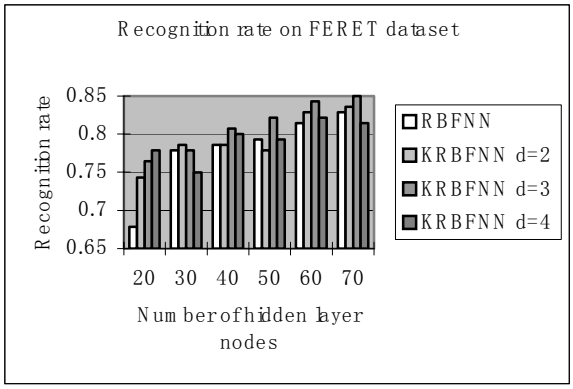

Figure 6. FERET face recognition rates

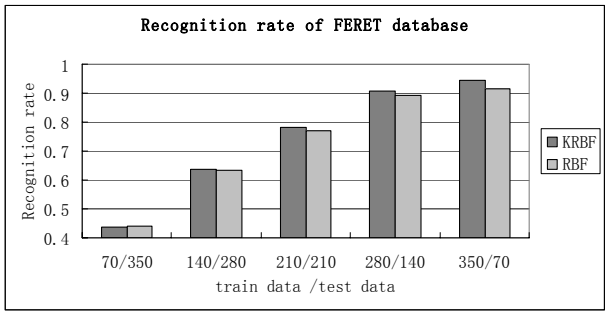

Figure 7. FERET face recognition rates

4.3 Discussion

Our experiments show that KRBF networks perform better than RBF networks in face recognition because the kernel feature space contains nonlinear information of components of the input vectors. When we select different values for the degree of the polynomial kernel function, KRBF networks shows different recognition rates, but d=2 and d=3 have almost the same rates. The recognition rates of KRBF networks increase rapidly at first and then smoothly as the number of hidden layer nodes increases. In addition, the recognition rates are closely related to the quantity of training data: the more training data are supplied, the higher recognition rates we achieve.

5 CONCLUSION

RBF networks use a k-means algorithm to cluster sample vectors. In the testing process, the distances from each test data to the center of each cluster reflect the features we want to extract from the test data. KRBF networks apply RBF networks in kernel feature space and extract features from the feature space. We apply KRBF networks in whole face recognition, achieving better results than with RBF Networks.

6 REFERENCES

[1] B.Scholkopf, A.Smola and K.R.Muller. “Nonlinear Component Analysis as a Kernel Eigenvalue Problem”. *Neural Computation*, Vol. 10, pp. 1299-1319, 1998.

[2] M.-H. Yang. N. Ahuja, and D. Kriegman, “Face recognition using Kernel Eigenfaces.” *Advances in NIPS*, Vol. 14, 2002.

[3] Q. Liu and S. Ma. “Face Recognition Using Kernel Based Fisher Discriminant Analysis.” *IEEE Conf. on Automatic Face and Gesture Recognition*, 2002.

[4] M.-H. Yang. “Kernel Eigenfaces vs. Kernel Fisherfaces: Face Recognition Using Kernel Methods” *IEEE Conf. on Automatic Face and Gesture Recognition*, 2002.

[5] S. Gutta, V. Philomin and M. Trajkovic. “An Investigation into the use of Partial-Faces for Face Recognition” *IEEE Conf. on Automatic Face and Gesture Recognition*, 2002.

[6] B. Scholkopf, “Statistical Learning Kernel Methods”. *NIPS'00*, 2000.

[7] R. Zhang and A. I. Rudnicky. “A large Scale Clustering Scheme for Kernel K-Means” *Proc. of ICPR*, 2002

[8] G. Baudat and F. Anouar. “Generalized Discriminant Analysis Using a Kernel Approach”. *Neural Computation*, 12(10):2385-2404, 2000

[9] T. Poggio and F. Girosi, “Networks for approximation and learning,” *Proc. IEEE*, Vol.78, No.9, pp.1481-1497, 1990.

[10] P. J. Phillips, H. Wechsler, J. Huang, and P. Rauss, “The FERET database and evaluation procedure for face recognition algorithms,” *Image and Vision Computing J*, Vol. 16, No. 5. pp 295-306, 1998.

[11] D. Pomerleau, *Neural Network Perception for Mobile Robot Guidance*, Kluwer Academic Publishing, Boston, MA, 1994.

[12] M. Rosenblum and L.S. Davis, “An improved radial basis function network for visual autonomous road following,” *IEEE Transactions on Neural Networks*, Vol. 7, No. 5, pp. 1111-1120, 1996.

[13] F. Girosi. “On some extensions of Radial Basis Functions and their applications in artificial intelligence.” *Int. Journal of Computer and Mathematics with Applications*, 24(12):61-80, 1992.

[14] J. Moody and C. Darken, Learning with localized receptive fields. In: D. Touretzky, G. Hinton, T. Sejnowski, eds, *Proceedings of the 1988 Connectionist Models Summer School*, Morgan Kaufmann, CA.

[15] M.T. Musavi, , W. Ahmed, K.H. Chan, K. Faris, and D.M. Hummels, "On the Training of Radial Basis Function Classifiers," *Neural Networks*, Vol. 5, No. 4, pp. 595-603, Aug. 1992.

[16] S. Ahmad and V. Tresp, Some solutions to the missing feature problem in vision, in S.J. Hanson, J.D. Cowan and C.L. Giles, eds., *Advances in Neural Information Processing Systems*, Vol. 5, Morgan Kaufmann, 1993.

[17] C..M.Bishop, *Neural Networks for Pattern Recognition*, Clarendon Press, 1995.

[18] Vladimir N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 1995.