Available online at www.sciencedirect.com

ScienceDirect

Fuzzy Sets and Systems 305 (2016) 131–148

FUZZY sets and systems

www.elsevier.com/locate/fss

A comparative study of feature extraction methods and their application to P-RBF NNs in face recognition problem

Sung-Kwun Oh [a,*], Sung-Hoon Yoo [a], Witold Pedrycz [b,c,d]

[a] *Dept. of Electrical Engineering, The University of Suwon, San 2-2, Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea*
[b] *Department of Electrical & Computer Engineering, University of Alberta, Edmonton T6R 2V4 AB, Canada*
[c] *Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland*
[d] *Department of Electrical and Computer Engineering, Faculty of Engineering, King Abdulaziz University, Jeddah, 21589, Saudi Arabia*

Received 14 July 2014; received in revised form 3 September 2015; accepted 20 November 2015

Available online 27 November 2015

Abstract

This study is concerned with a design of a face recognition algorithm realized based on feature extraction with the aid of the 2-directional 2-dimensional linear discriminant analysis referred to as $(2D)^2$LDA. The 2DLDA algorithm yields high accuracy, but comes with an unresolved problem of handling large feature matrices present in face recognition problems. The proposed $(2D)^2$LDA algorithm is computationally more efficient and produces more powerful discrimination decision. The proposed P-RBF NNs is used as a recognition module. The architecture of this module consists of three functional components. The coefficients of the P-RBF NNs model are obtained by fuzzy inference method forming the inference part of fuzzy rules. The essential design parameters (including learning rate, momentum, fuzzification coefficient, and the feature selection mechanism) of the networks are optimized by means of differential evolution (DE). The experimental results reported for benchmark face datasets – the Yale, ORL database, and IC&CI dataset demonstrate the effectiveness and efficiency of $(2D)^2$LDA algorithm compared with other pre-processing approaches such as LPP, 2D-PCA, 2D-LPP, SR, PCA, $(2D)^2$PCA and fusion of PCA-LDA. The experimental results show that $(2D)^2$LDA based P-RBF NNs achieves higher performance than being reported by other methods.
© 2015 Elsevier B.V. All rights reserved.

Keywords: LDA (Linear Discriminant Analysis); $(2D)^2$LDA (Two-directional two-dimensional Linear Discriminant Analysis); P-RBF NNs (Polynomial-based Radial Basis Function Neural Networks); FCM (Fuzzy C-Means); DE (Differential Evolution)

1. Introduction

Machine recognition of human face exploiting still and video images has become an active research area in the communities of image processing, pattern recognition, neural networks, and computer vision. In many pattern recognition systems, the paradigm of neural classifiers has demonstrated many tangible advantages with regard to its learning abil-

* Corresponding author.
E-mail address: ohsk@suwon.ac.kr (S.-K. Oh).

http://dx.doi.org/10.1016/j.fss.2015.11.018
0165-0114/© 2015 Elsevier B.V. All rights reserved.

ities, generalization aspects, and robustness, cf. [1,2]. The results have shown that the neural networks are effective constructs in comparison with some conventional models [3]. These properties make neural networks an attractive tool for many pattern classification problems. In this paper, we present a concept of the polynomial-based radial basis function neural networks (P-RBF NNs) based on a fuzzy inference mechanism [4,5]. The main objective of this study is to propose an efficient learning algorithm for the P-RBF NNs and show its applications to face recognition. The P-RBF NNs is proposed as one of the recognition parts of overall face recognition system that consists of two parts such as the preprocessing part and recognition part.

In recent years several preprocessing algorithms which directly use 2D images without any transformation process have been proposed. Linear discriminant Analysis (LDA) and Principal component Analysis (PCA) are the most commonly considered models for feature representation and dimensionality reduction. However, they exhibit two disadvantages. One is its poor performance in case of a small size of the training sample. Another one concerns a large computational overhead due to matrix-to-vector conversion [6,7]. In order to overcome these problems, two-dimensional Linear Discriminant Analysis (2DLDA) and two-dimensional Principal Component Analysis have been proposed in which image covariance matrices can be constructed directly using original image matrices. As a result, it is easier to evaluate the covariance matrices accurately and the computation cost becomes reduced [8]. However, a drawback of these 2D approaches is that they require more coefficients than conventional approaches used to image representation. To alleviate this issue, a new concept of 2-directional 2DLDA ($(2D)^2$LDA) and 2-directional 2DPCA ($(2D)^2$PCA) has been proposed to deal with face recognition problem [9,10]. The main difference between 2DLDA and the existing $(2D)^2$LDA is that the latter works simultaneously for the rows and columns of face images, while the former only works in the row direction of face image. In the preprocessing part, we introduce the design of $(2D)^2$LDA for facial feature extraction.

We design a P-RBF NNs based on fuzzy inference mechanism. The essential design parameters (including learning rate, momentum, fuzzification coefficient, and feature selection) are optimized by means of differential evolution (DE). The proposed P-RBF NNs dwell upon structural findings about training data that are expressed in terms of partition matrix resulting from fuzzy clustering, in this case being Fuzzy C-Means (FCM). The network is of functional nature as the weights between the hidden layer and the output are some polynomials. The use of the polynomial weights becomes essential to reflect the nonlinear nature of data encountered in regression or classification problems. From the perspective of linguistic interpretation, the proposed network can be expressed as a collection of "if-then" fuzzy rules. The architecture of the networks discussed here embraces three functional components reflecting the three phases of input–output mapping realized in rule-based architectures.

This study is organized as follows: In Section 2, we propose an overall face recognition system and design method. Section 3 offers a brief review of conventional LDA, 2DLDA, and $(2D)^2$LDA. In Section 4, we propose a general architecture of P-RBF NNs. Sections 5 and 6 discuss the P-RBF NNs classifiers to realize learning and its optimization carried out by the DE. The results of experiment are reported in Section 7. Conclusions are presented in Section 8.

2. Structure of face recognition system

In this section, an overall structure of proposed face recognition system and design method is described. Face recognition system is constructed to data preprocessing and RBF NNs pattern classifier applying optimization techniques. Histogram equalization, AdaBoost, $(2D)^2$LDA realized a phase of data preprocessing. Histogram equalization compensates for illumination-distorted of images. The face area is detected by using AdaBoost. The features of face images are extracted with the use of the $(2D)^2$LDA method. The RBF NNs pattern classifier is constructed. The classifier comprises three functional modules. These modules realize the condition, conclusion and aggregation phase of the processing scheme. The input space of the condition phase is represented by fuzzy set formed by running the FCM clustering algorithm. The conclusion phase involves a certain polynomial. The output of the network is determined by running fuzzy inference. The proposed RBF NNs come with the fuzzy inference mechanism constructed with the use of the fuzzy rule-based network. In the construction of the classifier, Differential Evolution (DE) is used to optimize the momentum coefficient, learning rate, and the fuzzification coefficient used in the FCM algorithm. Fig. 1 portrays an overall architecture of the system.

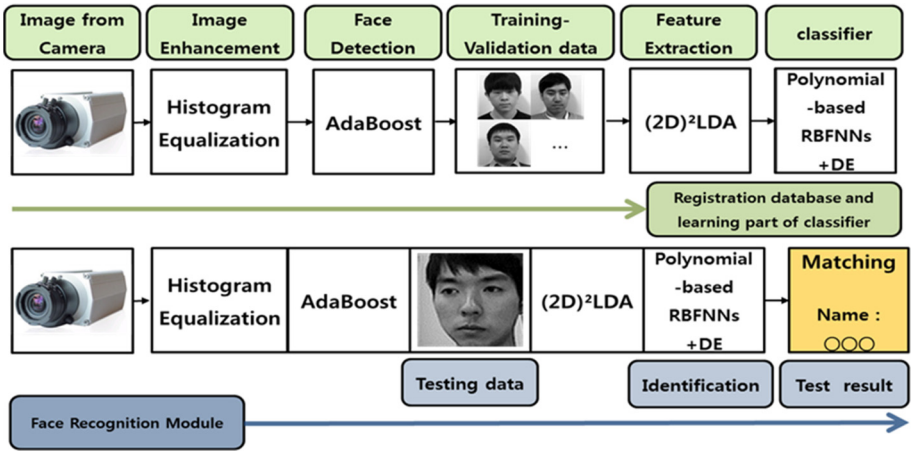

Fig. 1. An overall architecture of the face recognition system.

3. Extraction of facial features

The main issue in the design of face recognition algorithm is to extract meaningful facial features. In real world, there are very large number of features, and many features in the initial feature set are irrelevant to the classification task and correlated with each other, which increases the computational complexity and reduces the recognition rate.

3.1. Two-dimensional linear discriminant analysis: row 2DLDA

Linear discriminant analysis or Fisherfaces method [11–13] overcomes the limitations of eigenfaces method by applying the Fisher's linear discriminant criterion. This criterion maximizes the ratio of the determinant of the between-class scatter matrix of the projected samples to the determinant of the within-class scatter matrix of the projected samples. Unlike the PCA method that extracts features to best represent face images; the LDA method finds the subspace that best discriminates faces coming from different classes. The LDA exhibits two drawbacks when being directly applied to the original input space. First, some non-face information such as image background is misclassified when the face of the same subject is presented with different backgrounds. Secondly, when the Singular Small Sample (SSS) problem occurs, the within-class scatter matrix becomes singular. This yields a so-called singularity problem [25]. To overcome this drawback, the 2-dimensional LDA is based on 2D matrices rather than 1D vectors. This means that the image matrix does not need to be converted into a vector. As a result, the 2DLDA has two advantages: it is easier to evaluate the covariance matrix accurately, and it exhibits a lower computing overhead.

Let A_i^j be an image of size $a \times b$ pixels representing the ith sample of class j, μ_j is the mean of class j, C is the number of classes, and N_j is the number of samples in class j. The between-class scatter matrix G_B and within-class scatter matrix G_W are computed as

$$G_B = \sum_{j=1}^{C} (\mu_j - \mu)^T (\mu_j - \mu) \tag{1}$$

$$G_W = \sum_{j=1}^{C} \sum_{i=1}^{N_j} \left(A_i^j - \mu_j\right)^T \left(A_i^j - \mu_j\right) \tag{2}$$

Where μ represents the mean of all classes. Once G_B and G_W have been formed, we compute the optimal projection axes, denoted by X, so that the total scatter of the projected samples of the training images becomes maximized. To maximize the total scatter of the projected images we use the Fisher's criterion:

$$J(X) = \frac{X^T G_B X}{X^T G_W X} \tag{3}$$

The projection of a training image onto these optimal projection axes results in a feature matrix of the respective training image. That is, we define the feature matrix of A_j^i in the form

$$Z_i^j = A_i^j X \tag{4}$$

where Z_i^j is the feature matrix of dimension $a \times q$. The proposed 2DLDA works in the row-wise direction as the image covariance matrix G_W is obtained by the outer products of the row vectors of the training images.

3.2. Alternative two-dimensional linear discriminant analysis: column 2DLDA

Eq. (2) reveals that the scatter matrix G_W can be obtained from the outer products of row vectors of images, assuming that the training images have a zero mean. For this reason, we claim that the original 2DLDA works in the row direction of images. Apparently, a natural extension is to use the outer product between column vectors of images to construct G_B and G_W. To devise an alternative 2DLDA, we present that the between-class scatter matrix H_B and the within-class scatter matrix H_W be computed as

$$H_B = \sum_{j=1}^{C} (\mu_j - \mu)(\mu_j - \mu)^T \tag{5}$$

$$H_W = \sum_{j=1}^{C} \sum_{i=1}^{N_j} \left(A_i^j - \mu_j\right)\left(A_i^j - \mu_j\right)^T \tag{6}$$

In this new formulation, H_B and H_W in (5) and (6) are obtained as outer products of column vectors, unlike G_B and G_W in (1) and (2) in the case of the original 2DLDA. Using these two scatter matrices, which are similar to the original 2DLDA, in the proposed model we also find the optimal projection axes $W(m \times q)$ given by

$$J(W) = \frac{W H_B W^T}{W H_W W^T} \tag{7}$$

Thus, the eigenvectors of $H_W^{-1} H_B$ are computed, and then q eigenvectors corresponding to the first q largest eigenvalues of $H_W^{-1} H_B$ are selected. Finally, projection of a training image onto these optimal projection axes results in a feature matrix of the respective training images. That is, if Z_i^j represents the feature matrix of A_i^j, then

$$Z_i^j = W^T A_i^j \tag{8}$$

Eq. (6) reveals that the images covariance matrix H_W can be obtained from the outer products of the column vectors of the training images, assuming that they have a zero mean. Therefore, the presented alternative 2DLDA works in the column direction of images.

3.3. Two-directional two-dimensional linear discriminant analysis: $(2D)^2$LDA

Let X denote the $n \times d$ optimal projection matrix obtained in the original 2DLDA method as explained in section 3.1, and let W denote the $m \times q$ matrix obtained by an alternative 2DLDA method as explained in section 3.2. For $(2D)^2$LDA method, each training image A_i^j is projected onto both X and W simultaneously to obtain the respective feature matrix F_i^j, which is of dimensionality of $q \times d$:

$$F_i^j = W^T A_i^j X \tag{9}$$

The matrix F is also called the coefficient matrix in image representation. When used for face recognition, the matrix F is also called the feature matrix. After projecting each training image A_i $(i = 1, 2, 3, \ldots, N)$ onto X and W, we obtain the feature matrices F_i $(i = 1, 2, 3, \ldots, N)$.

In a dataset, distance or similarity relationships between pairs of classes is the important information to be used for classification purposes. Distance or similarity relationships, usually expressed as Euclidean distance, Mahalanobis

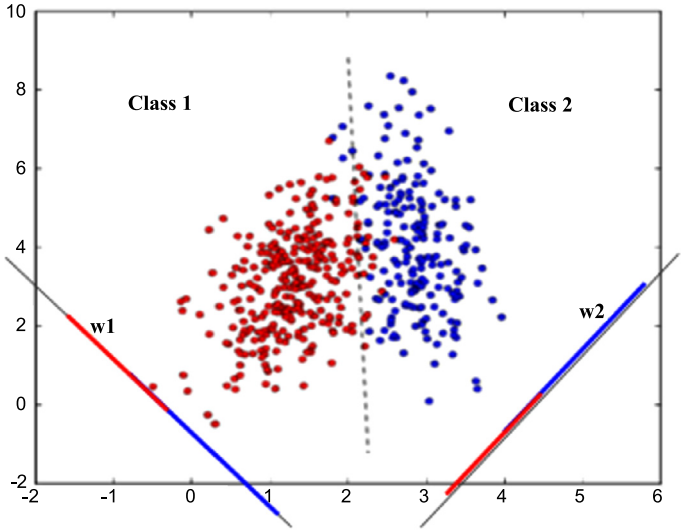

Fig. 2. 2DLDA projection technique.

distance, and alike, reflect how well two classes are separated in the feature space. In multi-class LDA, the relationships between pairs of classes are likely to vary from one pair to another. The classes that are closer to each other are potentially more confusing (difficult to discriminate) and they should be given more attention during the feature extraction stage (see Fig. 2). The weighted LDA works by using the pairwise scatter matrix approach which only points at the between-class scatter matrix. Generally, the S_B of this pairwise scatter us computed as

$$S_B = \sum_{j=1}^{C-1} \sum_{k=j+1}^{C} (\mu_j - \mu_k)(\mu_j - \mu_k)^T \tag{10}$$

Therefore, for the 2DLDA and the alternative 2DLDA method we rewrite the between-class matrix (1) and (5) as follows:

$$G_B = \frac{1}{N} \sum_{j=1}^{C-1} \sum_{k=j+1}^{C} N_j N_k (\mu_j - \mu_k)^T (\mu_j - \mu_k) \tag{11}$$

$$H_B = \frac{1}{N} \sum_{j=1}^{C-1} \sum_{k=i+1}^{C} N_j N_k (\mu_j - \mu_k)(\mu_j - \mu_k)^T \tag{12}$$

C is the number of classes, N_j, N_k are the number of samples in each class j and k, and N represents the number of samples in $a \times b$ pixels images. (11) expresses the between-class scatter matrix of 2DLDA which essentially works in the row-direction of images, and (12) denotes the between-class scatter matrix of an alternative 2DLDA which works in the column direction of images.

The within-class scatter matrix is still not changed because the pairwise scatter approach analyzes how the classes are discriminated from each other pairwise. Thus, is does not measure discrimination inside the within-class scatter matrix. However, we can rewrite the within-class scatter matrices (2) and (6) as

$$G_W = \frac{1}{N} \sum_{j=1}^{C} \sum_{i=1}^{N_j} N_j \left(A_i^j - \mu_j\right)^T \left(A_i^j - \mu_j\right) \tag{13}$$

$$H_W = \frac{1}{N} \sum_{j=1}^{C} \sum_{i=1}^{N_j} N_j \left(A_i^j - \mu_j\right)\left(A_i^j - \mu_j\right)^T \tag{14}$$

Similarly to (11) and (13) are the within-class scatter matrix of 2DLDA which essentially works in the row-direction of images, and (12) and (14) is the within-class scatter matrix of an alternative 2DLDA, which works in the column direction of images.

4. Polynomial-based radial basis function networks: a general topology

In what follows, we discuss a general topology of polynomial-based radial basis function networks (P-RBF NNs).

4.1. Architecture of polynomial-based RBF networks

The general architecture of the radial basis function neural networks (RBF NNs) consists of the three layers [24]. RBF NNs exhibit a single hidden layer. Each node in the hidden layer determines a level of activation of the receptive field (radial basis function) $\Theta(\mathbf{x})$ given some incoming input $\mathbf{x}$. The jth output $y_j(\mathbf{x})$ is a weighted linear combination of the activation levels of the receptive fields:

$$y_j(\mathbf{x}) = \sum_{i=1}^{c} w_{ji}\Theta_i(\mathbf{x}) \tag{15}$$

$j = 1, \ldots, s$. 's' stands for the number of outputs (the number of classes encountered in classification problems). In the case of the Gaussian type of RBFs, we have

$$\Theta_i(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{v}_i\|^2}{2\sigma_i^2}\right) \tag{16}$$

Where $\mathbf{x}$ is the n-dimensional input vector $[x_1, \ldots x_n]^{\mathrm{T}}$, $\mathbf{v}_i = [v_{i1}, \ldots v_{in}]^{\mathrm{T}}$ is the center of ith basis function $\Theta_i(\mathbf{x})$ while c is the number of the nodes of the hidden layer. Typically the distance $\|.\|$ used in (16) is the Euclidean one [14].

The proposed P-RBF NNs exhibit a similar topology as the one encountered in RBF NNs. However the functionality and the associated design process exhibit evident differences. In particular, the receptive fields do not assume any explicit functional form (say, Gaussian, ellipsoidal, etc.), but are directly reflective of the nature of the data and come as the result of fuzzy clustering. Given the prototypes formed by the FCM clustering method [15,16], the receptive fields are expressed in the following way

$$\Theta_i(\mathbf{x}) = A_i(\mathbf{x}) = \frac{1}{\sum_{j=1}^{c}\left(\frac{\|\mathbf{x}-\mathbf{v}_i\|^2}{\|\mathbf{x}-\mathbf{v}_j\|^2}\right)} \tag{17}$$

In addition, the weights between the output layer and the hidden layer are not single numbers but come in the form of polynomials of input variables (hence the term of functional links that becomes present in this architecture)

$$w_{ji} = f_{ji}(\mathbf{x}) \tag{18}$$

The neuron located at the output layer realizes a linear combination of the activation levels of the corresponding receptive fields hence (15) can be rewritten as follows,

$$y_j(\mathbf{x}) = \sum_{i=1}^{c} f_{ji}(\mathbf{x})A_i(\mathbf{x}) \tag{19}$$

The above structure of the classifier can be represented through a collection of fuzzy rules

$$\text{If } \mathbf{x} \text{ is } A_i \text{ then } f_{ji}(\mathbf{x}) \tag{20}$$

where, the fuzzy sets A_i is the i-cluster (membership function) of the ith fuzzy rule, $f_{ji}(\mathbf{x})$ is a polynomial function generalizing a numeric weight used in the standard form of the RBF NNs, and c is the number of fuzzy rules (clusters), and $j = 1, \ldots, s$.

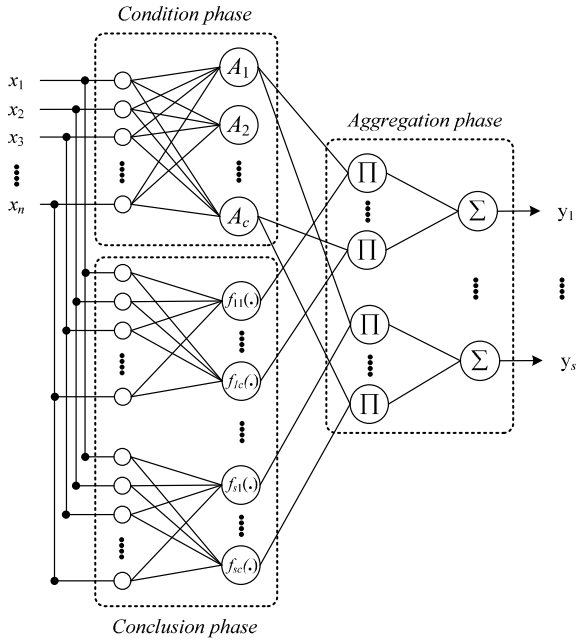

Fig. 3. Topology of P-RBF NNs exhibiting three functional modules of condition, conclusion and aggregation phases.

4.2. Three processing phases of polynomial-based RBF neural networks

The proposed P-RBF NN is implemented by realizing three processing phases that is, condition, conclusion and aggregation phases. The condition and conclusion phases relate to the formation of the fuzzy rules and their ensuing analysis. The aggregation phase is concerned with a fuzzy inference (mapping procedure).

4.2.1. Condition phase of networks

The condition phase of P-RBF NNs is handled by means of the Fuzzy C-Means clustering method. In this section, we briefly review the objective function-based fuzzy clustering with intent of highlighting it key features pertinent to this study. The FCM algorithm is aimed at the formation of 'c' fuzzy sets (relations) in $\mathbb{R}^n$.

The objective function Q guiding the clustering is expressed as a sum of the distances of individual data. Consider the set X, which consists of N patterns treated as vectors located in some n-dimensional Euclidean space, that is, $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_k \in \mathbb{R}^n$, $1 \leq k \leq N$. In clustering we assign patterns $\mathbf{x}_k \in X$ into c clusters, which are represented by its prototypes $\mathbf{v}_i \in \mathbb{R}^n$, $1 \leq i \leq c$. The assignment to individual clusters is expressed in terms of the partition matrix $U = [u_{ik}]$ where

$$\sum_{i=1}^{c} u_{ik} = 1, \quad 1 \leq k \leq N \tag{21}$$

and

$$0 < \sum_{k=1}^{N} u_{ik} < N, \quad 1 \leq i \leq c \tag{22}$$

The minimization of Q is realized in successive iterations by adjusting both the prototypes and entries of the partition matrix, that is min $Q(U, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c)$. The corresponding formulas used in an iterative fashion read in the form

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\mathbf{x}_k - \mathbf{v}_i\|}{\|\mathbf{x}_k - \mathbf{v}_j\|}\right)^{\frac{2}{m-1}}}, \quad 1 \leq k \leq N, \ 1 \leq i \leq c \tag{23}$$

and

$$\mathbf{v}_i = \frac{\sum_{k=1}^{N} u_{ik}^m \mathbf{x}_k}{\sum_{k=1}^{N} u_{ik}^m}, \quad 1 \le i \le c \tag{24}$$

The properties of the optimization algorithm are well documented in the literature, cf. [15]. In the context of our investigations, we note that the resulting partition matrix produces 'c' fuzzy relations (multivariable fuzzy sets) with the membership functions $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_c$ forming the corresponding rows of the partition matrix U, that is $U = [\mathbf{u}_1^{\mathrm{T}} \mathbf{u}_2^{\mathrm{T}} \ldots \mathbf{u}_c^{\mathrm{T}}]$. From the design standpoint, there are several essential parameters of the FCM that impacts the usage of the produced results. These parameters concern the number of clusters, the values of the fuzzification coefficient and a form of the distance function. The fuzzification coefficient exhibits a significant impact on the form (shape) of the developed clusters.

4.2.2. Conclusion phase of networks

Polynomial functions are dealt with in the conclusion phase. For convenience, we omit the suffix j from the expression $f_{ji}(\mathbf{x})$ shown in (20). Several classes of polynomials are worth noting

Constant: $$f_i(\mathbf{x}) = a_{i0} \tag{25}$$

Linear: $$f_i(\mathbf{x}) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j \tag{26}$$

Quadratic: $$f_i(\mathbf{x}) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j + \sum_{j=1}^{n} \sum_{k=j}^{n} a_{ijk} x_j x_k \tag{27}$$

These functions are activated by partition matrix and lead to local regression models located at the condition phase of the individual rules.

In case of the quadratic function, the dimensionality of the problem increases quite quickly especially when dealing with problems of high dimensionality. The reduced quadratic function

$$f_i(\mathbf{x}) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j + \sum_{k=1}^{n} a_{ijk} x_k^2 \tag{28}$$

is also discussed with an intent to reduce computational burden.

4.2.3. Aggregation phase of networks

Let us consider the P-RBF NNs structure by considering the fuzzy partition realized in terms of FCM as shown in Fig. 3. The node denoted by Π is realized as a product of the corresponding fuzzy set and the polynomial function. The family of fuzzy sets A_i forms a partition (so that the sum of membership grades sum up to one at each point of the input space). The "$\sum$" neuron is described by a linear sum as shown in (19). The output of P-RBF NNs can be obtained by following a standard inference mechanism used in rule-based systems [20,26],

$$y_j = g_j(\mathbf{x}) = \sum_{i=1}^{c} \frac{u_i f_{ji}(\mathbf{x})}{\sum_{k=1}^{c} u_k} = \sum_{i=1}^{c} u_i f_{ji}(\mathbf{x}) \tag{29}$$

where, $u_i = A_i(\mathbf{x})$. All the entries sum up to 1 as indicated by (21). $g_j(\mathbf{x})$ describes here the discriminant function used for discerning jth class.

Based on the local polynomial-like representation, the global characteristics of the P-RBF NNs result through the composition of their local relationships.

5. Polynomial-based radial basis function neural networks classifiers and its learning using gradient descent method

In this section, we consider the use P-RBF NNs as classifiers and show how their functionality gives rise to highly nonlinear discriminant functions.

5.1. The discriminant function

There are many different ways to describe pattern classifiers. One of the most useful ways is the one realized in terms of a set of discriminant functions $g_i(\mathbf{x})$, $i = 1, \ldots, m$ (where m stands for the number of classes). The classifier is said to assign a input vector $\mathbf{x}$ to class ω_i if

$$g_i(\mathbf{x}) > g_j(\mathbf{x}) \quad \text{for all } j \neq i. \tag{30}$$

Thus, the classifiers are viewed as networks that compute m discriminant functions and select the category corresponding to the largest value of the discriminant.

In this paper, the proposed P-RBF NNs classifier is used as a discriminate function for two-class or multi-class. If a classification problem is multi-class one then we use (30) as discriminant function, otherwise, we use the following decision rule defined commonly as a single discriminant function $g(\mathbf{x})$ in two-class problem.

$$\text{Decide } \omega_1 \text{ if } g(\mathbf{x}) > 0; \text{ otherwise decide } \omega_2 \tag{31}$$

The final output of networks, (31), is used as a discriminant function $g(\mathbf{x})$ and can be rewritten in a form of the linear combination

$$g(\mathbf{x}) = \mathbf{a}^{\mathrm{T}}\mathbf{fx} \tag{32}$$

where $\mathbf{a}$ is a vector of coefficients of the polynomial functions used in the conclusion phase of the rules in (25)–(28) and $\mathbf{fx}$ is a matrix of U and $\mathbf{x}$. These can be defined for each of polynomial as follows.

i) Constant:

$$\mathbf{a}^{\mathrm{T}} = [a_{10}, \ldots, a_{c0}], \mathbf{fx} = [u_1, \ldots, u_c]^{\mathrm{T}}$$

ii) Linear:

$$\mathbf{a}^{\mathrm{T}} = [a_{10}, \ldots, a_{c0}, a_{11}, \ldots, a_{c1}, \ldots, a_{cn}]$$
$$\mathbf{fx} = [u_1, \ldots, u_c, u_1x_1, \ldots, u_cx_1, \ldots, u_cx_n]^{\mathrm{T}}$$

iii) Quadratic:

$$\mathbf{a}^{\mathrm{T}} = [a_{10}, \ldots, a_{c0}, a_{11}, \ldots, a_{c1}, \ldots, a_{cn}, \ldots, a_{cnn}]$$
$$\mathbf{fx} = [u_1, \ldots, u_c, u_1x_1, \ldots, u_cx_1, \ldots, u_cx_n, \ldots, u_cx_nx_n]^{\mathrm{T}}$$

iv) Reduced quadratic:

$$\mathbf{a}^{\mathrm{T}} = [a_{10}, \ldots, a_{c0}, a_{11}, \ldots, a_{c1}, \ldots, a_{cn}, \ldots, a_{cnn}]$$
$$\mathbf{fx} = \left[u_1, \ldots, u_c, u_1x_1, \ldots, u_cx_1, \ldots, u_cx_n, \ldots, u_cx_n^2\right]^{\mathrm{T}}$$

For the discriminant function coming in the form of (33), a two-class classifier implements the decision rule expressed by (31). Namely, $\mathbf{x}$ is assigned to ω_1 if the inner product $\mathbf{a}^{\mathrm{T}}\mathbf{fx}$ is greater than zero and to ω_2 otherwise. The equation $g(\mathbf{x}) = 0$ defines the decision surface that separates two classes. Otherwise, a multi-class classifier implements the decision rule expressed by (30). Each output node generates a discriminant function corresponding to each class. If the ith output is larger than all remaining outputs, pattern $\mathbf{x}$ is assigned to ith class.

5.2. Learning of P-RBFNNs using gradient descent method

In order to estimate the coefficients of the polynomials standing in the conclusion phase of the P-RBF NNs classifier, we consider a gradient descent method. For convenience, we explain the learning process considering the case where the discriminant function is governed by (31). The learning of P-RBF NNs uses gradient descent method augmented by a momentum term. The coefficients are adjusted by taking the negative gradient of the error function E_q expressed as:

$$E_q = \frac{1}{2}(t_q - y_q)^2 \quad q = 1, \ldots, N \tag{33}$$

where N is the size of training data set. t_q is the target output for the qth pattern. y_q is the network output when exposed to the qth pattern:

$$y_q = \sum_{i=1}^{c} u_{qi} f_i \tag{34}$$

The learning of the P-RBF NNs involves four cases given a specific form of f_i as given by (25)–(28). Those are Constant, Linear, Quadratic, Reduced Quadratic type, respectively. The detailed formulas come as follows.

If the polynomial is constant then $f_i = a_{i0}$. Here we have:

$$\frac{\partial E_q}{\partial a_{i0}} = \frac{\partial E_q}{\partial y_q} \cdot \frac{\partial y_q}{\partial f_i} \cdot \frac{\partial f_i}{\partial a_{i0}} \tag{35}$$

More specifically the above expressions read as

$$\frac{\partial E_q}{\partial y_q} = -(t_q - y_q) \tag{36}$$

$$\frac{\partial y_q}{\partial f_i} = u_i \tag{37}$$

$$\frac{\partial f_i}{\partial a_{i0}} = 1 \tag{38}$$

Therefore, the error rate $\Delta a_{i0}(l)$ in lth learning iteration can be written

$$\Delta a_{i0}(l) = -\eta \cdot \frac{\partial E_q}{\partial a_{i0}} = \eta(t_q - y_q)u_i \tag{39}$$

In the sequel, when considering momentum, each $a_{i0}(l)$ is updated according to the following rule:

$$a_{i0}(l+1) = a_{i0}(l) + \Delta a_{i0}(l) + \alpha\big(a_{i0}(l) - a_{i0}(l-1)\big) \tag{40}$$

Here, η and α are represent learning rate and momentum coefficient respectively.

Besides this "constant" polynomial of the conclusion phase explained previously, the learning of polynomial types such as linear, quadratic, and reduced quadratic are also carried out in the same way.

6. Optimization of parameters of the P-RBFNNs with the aid of differential evolution

6.1. Differential evolution (DE)

EAs form a class of direct search algorithms. Once a new potential solution based on some given solution has been generated, the new parameter vectors are accepted in case it reduces the value of its objective function. This method realizes the greedy search.

This search converges fast but it could be trapped in local minima. In order to overcome this pitfall, the evolutionary strategy called Differential Evolution (DE) [17,18] has been proposed.

The DE algorithm can be outlined as the following sequence of steps:

Step 1: Set up the values of the parameters of the method such as a crossover rate (CR), scaling factor (SF), and mutation type (one out of the 5 types available) and then randomly generate "NP" population in search space. Each variable (or vector) in the n-dimensional search space is denoted by $D_i(t) = [x_1(t), x_2(t), \ldots, x_n(t)]$ and the population is $\mathrm{NP}(t) = \{D_1(t), D_2(t), \ldots, D_s(t)\}$ which is composed of the elements $D(t)$. Evaluate each individual using the objective function.

Step 2: Perform mutation to generate a mutant vector $D_{mutant}(t+1)$. For the target vector, randomly choose distinct vectors $D_a(t)$, $D_b(t)$, $D_c(t)$ etc. $(a, b, c \in \{1, 2, \ldots, NP\})$.

There are five types of mutation:

Fig. 4. The structure of parameter vectors for optimization of P-RBF NNs.

$$DE/Rand/1/\beta : D_{mutant}(t+1) = D_c(t) + \beta\big(D_a(t) - D_b(t)\big) \tag{41}$$

$$DE/Best/1/\beta : D_{mutant}(t+1) = D_{best}(t) + \beta\big(D_a(t) - D_b(t)\big) \tag{42}$$

$$DE/Rand/2/\beta : D_{mutant}(t+1) = D_e(t) + \beta\big(D_a(t) - D_b(t) - D_c(t) - D_d(t)\big) \tag{43}$$

$$DE/best/2/\beta : D_{mutant}(t+1) = D_{best}(t) + \beta\big(D_a(t) - D_b(t) - D_c(t) - D_d(t)\big) \tag{44}$$

$$DE/Rand\ To\ Best/1 : D_{mutant}(t+1) = D_c(t) + \beta\big(D_{best}(t) - D_c(t)\big) + \beta\big(D_a(t) - D_b(t)\big) \tag{45}$$

Produce a mutant vector using one of the mutation methods shown above. Generally, the strategy *DE/Rand/1/*β are used.

Step 3: Perform crossover to obtain a trial vector for each target vector using its mutant vector in the following equation:

$$D^j_{\text{trial}}(t+1) = \begin{cases} D^j_{mutant}(t+1) & \text{if (rand} < \text{CR) or } j = j^{randindex} \\ D^j_{target}(t) & \text{Otherwise} \end{cases} \tag{46}$$

$j = 1, 2, \ldots, n$; r and $\in [0, 1]$ is a random number drawn from a uniform distribution over [0, 1]; CR is the crossover rate $\in [0, 1]$; and $j^{randindex} \in (1, 2, \ldots, n)$ is randomly selected index.

Step 4: Evaluate the trial vectors $D_{trial}(t+1)$. If a trial vector comes with the better fitness than that of individual in the current generation, it is transferred to the next generation.

Step 5: Until the termination criterion has been satisfied, repeat Steps 2–4.

In this study, essential design parameters are optimized *by DE/rand/1/bin* method.

6.2. Architecture of vectors in P-RBF NNs

Through the optimization of these parameters such as learning rate, momentum coefficient, fuzzification coefficient and the feature selection by using DE, P-RBF NNs structure exhibits better convergence in the generation process of the network from the viewpoint of performance. Fig. 4 shows the structure of parameter vectors used in optimization of the P-RBF NNs.

Individual vectors of the P-RBF NNs include entries that represent optimized learning rate, momentum, fuzzification coefficient, and feature selection. The learning rate and momentum of vectors are applied to optimize the connections (weight) standing in (38) and (40). The fuzzification coefficient changes the shape of the membership functions produced by the Fuzzy C-Means. The values of the membership function depend on the location of the center point and the fuzzification coefficient to be adjusted. In the feature selection part, a feature whose value is greater than 0.5 becomes selected.

7. Experimental studies

The complete face recognition system includes two stages. First stage requires an extraction of pertinent features from the facial images and the formation of the feature vectors. The second stage involves a classification of facial images based on the derived feature vector obtained in the first stage.

In this experiment, we have used $(2D)^2$LDA algorithm as a feature domain that uses global data to form the feature vector at the first stage. We tested the recognition rate in row and column feature dimensions. The number of feature dimensions was 49(7×7). The designed P-RBF NNs with the proposed learning algorithm has been used as a classifier in the second stage of the face recognition system. Our objective is to quantify the performance of the proposed P-RBF NNs classifier. We compare and analyze the performance of the PCA, $(2D)^2$PCA and PCA-LDA fusion algorithm. In the assessment of the performance of the classifier, we report the percentage of correctly classified

Table 1
Description of Yale face image database used for experiment.

Yale database
It contains 165 face images of 15 individuals. There are 11 images per subject, one for each facial expression or configuration: center-light, glasses/no glasses, happy, normal, left-light, right-light, sad, sleepy, surprised and wink. Samples of Yale face database are shown in Fig. 5.

Fig. 5. Samples of face images in the Yale face database.

Table 2
Description of ORL face image database used for experiment

ORL database
It contains 400 face images from 40 individuals in different states. The total number of images for each person is 10. None of the 10 samples is identical to any other sample. They vary in position, rotation, scale and expression. The changes in orientation have been accomplished by rotating the person a maximum of 20° in the same plane; also each person has changed his/her facial expression in each 10 samples (open/close eye, smiling/not smiling). The changes in scale have been achieved by changing the distance between the person and the video camera. For some individuals, the images were taken at different times, varying facial details (glasses/ no glasses). Each image was digitized and presented by a 112×92 pixel array whole gray levels between 0 and 255. Samples of the ORL database are shown in Fig. 6.

patterns (classification rate). In the experiments, we consider two well-known face database, namely Yale and AT&T. The Yale database is used to examine the performance of the algorithms under the condition of varied facial expression and lighting configuration. The AT&T database is used to test the performance of the face recognition algorithms under the condition of minor variation of scaling and rotation. All experiments we carried out on a 32 bit PC with 3.40 GHz CPU and 4.00 GB RAM memory under MATLAB R2012a platform.

7.1. Experimental design

To show the usefulness of the proposed algorithm, the experimental studies are carried out for the Yale face database (available at http://cvc.yale.edu/projects/yalefaces/yalefaces.html) and the ORL database (available at http://www.uk.research.att.com/facedatabase.html). Table 1 includes a description of these two face image databases.

Each experiment consists of four steps. In the first step, we split data into 50%–30%–20% training, validation, and testing subsets, namely, 80% (50%–30% training and validation set) of the whole pattern are selected randomly for training and the remaining pattern are used for testing purposes. Therefore, in the ORL database a total of 200 images are used as the training set, 120 are the validation set, and another 80 images are used for testing set while in the Yale database a total of 90 images are used for training, 45 for validation and the rest 30 are used for testing (see Table 2). All images are in grayscale and normalized to the resolution of pixels while Histogram Equalization (HE) has been used in the preprocessing step. In the second step, $(2D)^2$LDA algorithm is generated inside the sub-images. In the third step, the classifier is designed and trained. Finally at the fourth step, the performance of the classifier is evaluated. In the assessment of the performance of the classifier, we report the % of correctly classified patterns.

Fig. 6. Samples of face images in the ORL database.

Table 3
Parameters of DE for the optimization of P-RBF NNs.

P-RBF NNs		
Number of learning iterations		100
Polynomial type		Linear, Reduced quadratic
Data split		Training : Validation : Testing = 5 : 3 : 2
Differential evolution		
Number of generations		20
Number of populations		50
F weights		0.95
Crossover		0.2
The range of Search space	Learning rate	[1e–8, 0.01]
	Momentum coefficient	[1e–8, 0.01]
	Fuzzification coefficient	[1.1, 3.0]
	Feature selection	higher than 0.5

The experiments are completed for a number of scenarios in which we considered a suite of numeric values of some essential parameters. The number of rules varied in-between 2 and 6. We used a reduced quadratic function (30). This reduced version of the quadratic function is helpful in reducing the computing burden when dealing with a large number of the pairs of the variables. The results are reported by presenting the average and standard deviation of the classification rate obtained over 5 repetitions by projection vectors d (where $d = 49$). Since the number of projection vectors (d), has a considerable impact on the performance of the algorithms, we chose the value that corresponds to the best classification result. For each combination of the parameters, the experiment was repeated 5 times. The results are reported by presenting the average and standard deviation of the classification rate obtained over these 5 repetitions of the experiment.

The numeric values of the parameters of the DE used in the experiments are shown in Table 3. To design an optimal model from this recognition algorithm, we used 4 variables such as learning rate, momentum coefficient, fuzzification coefficient, and the projection features.

7.2. *Yale face database*

In this experiment, we used $(2D)^2$LDA as a feature extraction. The accuracy produced for different feature extraction methods is given in Table 4. The entries marked in boldface indicate the corresponding best mean accuracy obtained with features extracted by the corresponding feature extraction methods.

In Table 4, the standard deviation of testing data is larger than that of the training data, and it also shown the best performance (recognition rate for testing is $98.45 \pm 1.49\%$) in L-RBFNNs when the number of rules equals to 2. Table 5 and Fig. 7 show the recognition rate of proposed method compared with the average recognition rate of LPP,

Table 4
Classification performance for Yale face dataset using $(2D)^2$LDA method.

Classifier model	Number of rules	Polynomial type	Classification rate (%)		
			Training AVG ± STD	Validation AVG ± STD	Testing AVG ± STD
DE-pRBFNNs	2	L-RBF NNs	**99.78 ± 0.50**	**100.0 ± 0.00**	**98.45 ± 1.49**
		RQ-RB FNNs	98.13 ± 2.92	100.0 ± 0.00	97.78 ± 3.14
	3	L-RBF NNs	99.24 ± 1.17	98.67 ± 2.98	98.00 ± 1.22
		RQ-RBF NNs	99.73 ± 0.59	98.56 ± 1.99	98.45 ± 2.16
	4	L-RBF NNs	98.31 ± 1.06	100.0 ± 0.00	95.78 ± 3.63
		RQ-RBF NNs	99.51 ± 0.67	98.00 ± 2.98	95.56 ± 5.21
	5	L-RBF NNs	99.24 ± 1.17	99.33 ± 1.49	98.29 ± 1.60
		RQ-RBF NNs	96.76 ± 3.30	99.11 ± 1.99	94.22 ± 5.12
	6	L-RBF NNs	98.49 ± 1.39	97.78 ± 3.14	97.11 ± 0.61
		RQ-RBF NNs	98.64 ± 1.43	98.61 ± 2.10	94.89 ± 1.49

Table 5
Comparison of the average recognition rate for several models on the Yale database.

	Recognition rate (%)	Reference	Year (fcv)
LPP	87.7	[19]	2003
2D-PCA	86.7	[20]	2004
2D-LPP	88.1	[21]	2007
SR	84.9	[22]	2007
PCA-RBF NNs	94.5	[23]	2013 (5-fcv)
PCA-LDA RBF NNs	95.6	[23]	2013 (5-fcv)
$(2D)^2$PCA-RBF NNs	95.3	This study	5-fcv
$(2D)^2$LDA-RBF NNs	98.5	This study	5-fcv

Table 6
$(2D)^2$LDA based best recognition rate and, CPU running time (Yale face database).

Method	Best recognition rate	CPU running time
$(2D)^2$LDA	**98.45 ± 1.49**	3 h 27 m 12 s

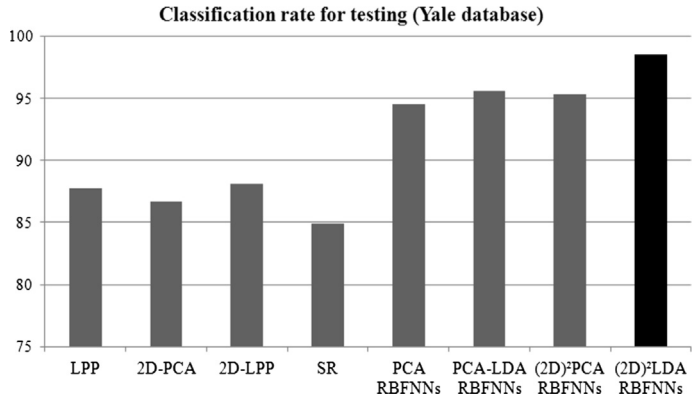

Fig. 7. Classification rate for testing comparison on Yale database.

2D-PCA, 2D-LPP, SR, PCA, $(2D)^2$PCA and fusion of PCA-LDA. And the $(2D)^2$LDA based best recognition rate and CPU running time in presented in Table 6. The results demonstrate that the proposed method yields superior results to those produced by other approaches used in the experiments.

Table 7
Classification performance on ORL face dataset using $(2D)^2$LDA method.

Classifier model	Number of rules	Polynomial type	Classification rate (%)		
			Training AVG ± STD	Validation AVG ± STD	Testing AVG ± STD
DE-pRBFNNs	2	L-RBF NNs	98.50 ± 1.45	98.50 ± 1.04	95.50 ± 1.89
		RQ-RB FNNs	99.75 ± 0.59	98.00 ± 1.42	96.50 ± 3.23
	3	L-RBF NNs	99.20 ± 0.44	97.75 ± 1.62	94.75 ± 2.71
		RQ-RBF NNs	**99.90 ± 0.22**	**99.00 ± 0.56**	**96.89 ± 1.43**
	4	L-RBF NNs	98.70 ± 1.25	98.25 ± 1.68	95.50 ± 2.74
		RQ-RBF NNs	99.90 ± 0.22	99.75 ± 0.56	95.25 ± 5.18
	5	L-RBF NNs	99.50 ± 0.35	98.75 ± 1.25	94.52 ± 2.74
		RQ-RBF NNs	99.80 ± 0.27	98.50 ± 1.04	94.50 ± 4.20
	6	L-RBF NNs	99.60 ± 0.41	98.50 ± 1.63	96.50 ± 1.85
		RQ-RBF NNs	99.70 ± 0.67	99.25 ± 0.68	96.75 ± 2.88

Table 8
Comparison of the average recognition rate for several models on the ORL database.

	Recognition rate (%)	Reference	Year (fcv)
LPP	87.4	[19]	2003
2D-PCA	92.5	[20]	2004
2D-LPP	94.2	[21]	2007
SR	91.6	[22]	2007
PCA-RBF NNs	91.0	[23]	2013(5-fcv)
PCA-LDA RBF NNs	95.0	[23]	2013(5-fcv)
$(2D)^2$PCA-RBF NNs	94.8	This study	5-fcv
$(2D)^2$LDA-RBF NNs	96.9	This study	5-fcv

Table 9
$(2D)^2$LDA based best recognition rate and, CPU running time (ORL face database).

Method	Best recognition rate	CPU running time
$(2D)^2$LDA	**96.89 ± 1.43**	4 h 23 m 28 s

7.3. ORL face database

The experiments were performed exactly in the same way as for the Yale face database.

As shown in Table 7, when the number of rules increases, standard deviation on the testing data is mostly larger than that of training data. And it also shown the best performance (recognition rate for testing is 96.89 ± 1.43%) in RQ-RBFNNs when the number of rules equals to 3. The experimental results show that the $(2D)^2$LDA method achieves higher face recognition rate than other methods (see Tables 8, 9, and Fig. 8).

7.4. An extended face recognition experiments – IC&CI face database

IC&CI (Intelligent Control & Computational Intelligence) face database contains a set of face images taken by students in the University of Suwon. There are 10 different images for each of 35 subjects. Each image was digitized and presented by a 200 × 200 pixel array with gray levels. Images feature frontal view faces with different facial expressions, and occlusions. The meaning of each image is described in Table 10 and samples coming from this database are shown in Fig. 9.

In this experiment, we use PCA-LDA fusion method (*Case 1*), and $(2D)^2$LDA method (*Case 2*) as a feature extraction. As shown in Table 3, the experiments were realized in the same way as in case of Yale and ORL face databases except for the number of rules ranging now from 2 to 5. And Table 11 compares the two methods in term if their top recognition rate, and it shows the running times for learning of the optimized DE-pRBFNNs models.

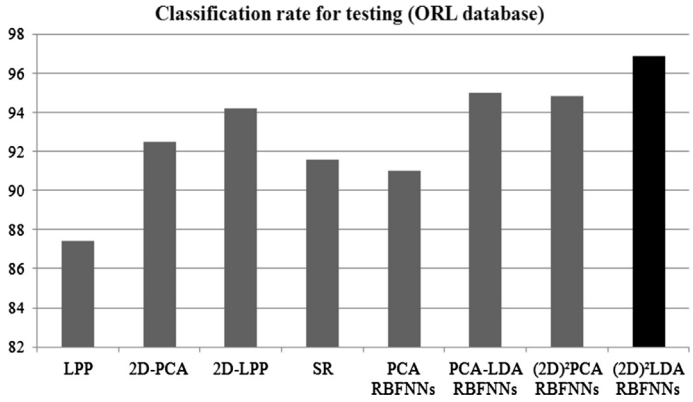

Fig. 8. Classification rate for testing data – a comparative overview.

Table 10
Meaning of each images.

Number of image	Meaning of image
1	30° turn to the left
2	Neutral expression
3	30° turn to the right
4	Smile
5	Scream
6	Anger
7	Wearing glasses
8	Wearing cap
9	Using a cell phone
10	Wearing scarf

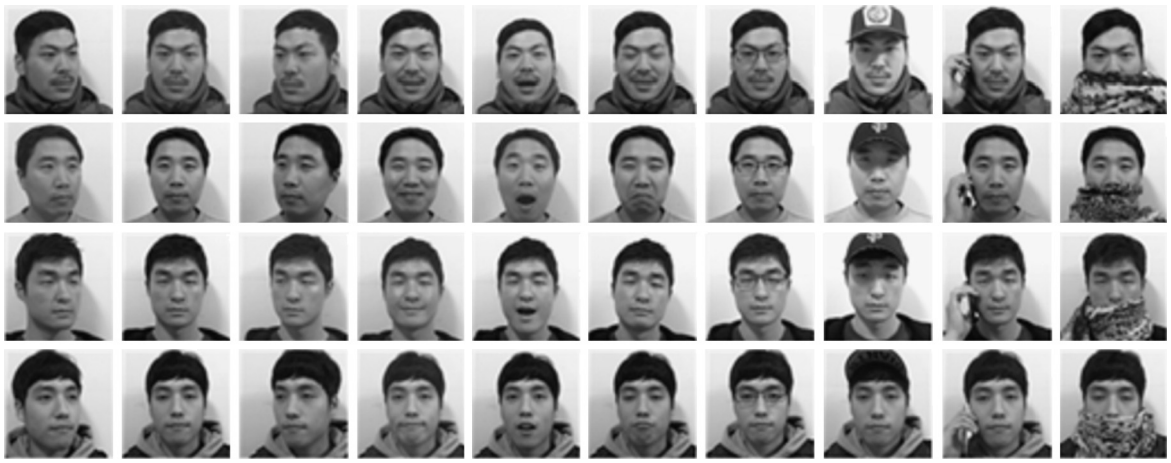

Fig. 9. Samples of face images in the IC&CI database.

Tables 11 and 12 shows the recognition rates of the PCA-LDA fusion method (*Case 1*), and $(2D)^2$LDA method (*Case 2*) based on the p-RBFNNs classifier. As shown Table 12, the best performance (recognition rate for testing is $95.43 \pm 1.19\%$) in L-RBFNNs when the number of rules equals to 3. Finally, these tables reveal that the top recognition accuracy of the proposed $(2D)^2$LDA method (*Case 2*) is significantly higher than that of the existing the PCA-LDA fusion method (*Case 1*). Conversely, the PCA-LDA fusion method consume a little more running time than the $(2D)^2$LDA.

Table 11
Classification performance for the IC&CI face dataset using PCA-LDA fusion method.

Classifier model	Number of rules	Polynomial type	Classification rate (%)		
			Training AVG ± STD	Validation AVG ± STD	Testing AVG ± STD
DE-pRBFNNs	2	L-RBF NNs	96.80 ± 1.04	85.71 ± 2.26	80.86 ± 3.72
(*Case 1*)		RQ-RB FNNs	**99.89 ± 0.25**	**86.57 ± 2.39**	**82.86 ± 3.62**
	3	L-RBF NNs	95.99 ± 1.56	84.00 ± 4.09	80.00 ± 4.04
		RQ-RBF NNs	99.89 ± 0.25	85.71 ± 2.67	79.14 ± 5.21
	4	L-RBF NNs	96.11 ± 0.74	85.43 ± 2.75	82.29 ± 4.24
		RQ-RBF NNs	99.66 ± 0.51	84.86 ± 2.78	78.29 ± 3.26
	5	L-RBF NNs	95.89 ± 1.48	87.43 ± 3.10	81.63 ± 5.19
		RQ-RBF NNs	98.86 ± 1.14	84.29 ± 3.50	80.29 ± 4.21

Table 12
Classification performance on the IC&CI face dataset using $(2D)^2$LDA method.

Classifier model	Number of rules	Polynomial type	Classification rate (%)		
			Training AVG ± STD	Validation AVG ± STD	Testing AVG ± STD
DE-pRBFNNs	2	L-RBF NNs	98.74 ± 0.75	95.43 ± 1.86	93.43 ± 1.92
(*Case 2*)		RQ-RB FNNs	99.89 ± 0.25	97.14 ± 1.01	94.23 ± 2.35
	3	L-RBF NNs	**98.89 ± 0.58**	**97.71 ± 0.78**	**95.4 ± 1.19**
		RQ-RBF NNs	99.77 ± 0.581	96.85 ± 1.20	93.14 ± 3.70
	4	L-RBF NNs	98.40 ± 1.24	97.43 ± 1.19	93.72 ± 2.16
		RQ-RBF NNs	99.77 ± 0.51	96.57 ± 1.27	93.60 ± 1.99
	5	L-RBF NNs	98.06 ± 1.19	97.43 ± 1.20	94.28 ± 2.86
		RQ-RBF NNs	99.89 ± 0.25	97.14 ± 1.01	94.86 ± 2.96

Table 13
Compares the two methods of recognition rate and, CPU running time.

Method	Best recognition rate	CPU running time
PCA-LDA fusion	**82.86 ± 3.62**	3 h 53 m 32 s
$(2D)^2$LDA	**95.43 ± 1.19**	3 h 47 m 17 s

8. Conclusions

In this study, we proposed the face recognition algorithm realized with feature extraction from $(2D)^2$LDA and optimized polynomial-based RBF NNs. In the preprocessing part, face features are first extracted by the $(2D)^2$LDA. Since that method is based on the image matrix, it is simpler and more straightforward to use for feature extraction and better than several methods (such as PCA, fusion of PCA-LDA, and $(2D)^2$PCA in terms of classification rate in experimental study. In recognition part, the P-RBF NNs involve a partition function formed by the FCM clustering and used here as an activation function of the neurons located in the hidden layer. The proposed model has polynomials weights. Given this, it is capable of generating more complex nonlinear discriminant functions. The experimental results have shown that $(2D)^2$LDA-pRBFNNs can be used under various conditions and can achieve higher recognition rates than the existing methods (see Table 13).

Acknowledgements

This work was supported by the GRRC program of Gyeonggi province [GRRC Suwon 2015-B2, Center for U-city Security & Surveillance Technology] and also supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning [grant number NRF-2015R1A2A1A15055365].

References

[1] S.-H. Lin, S.-Y. Kung, L.-J. Lin, Face recognition/detection by probabilistic decision-based neural network, IEEE Trans. Neural Netw. 8 (1997) 114–132.
[2] H.H. Song, S.W. Lee, A self-organizing neural tree for large-set pattern classification, IEEE Trans. Neural Netw. 9 (1998) 369–380.
[3] W. Zhou, Verification of the nonparametric characteristics of back propagation neural networks for image classification, IEEE Trans. Geosci. Remote Sens. 32 (2) (1999) 771–779.
[4] G.E. Tsekouras, J. Tsimikas, On training RBF neural networks using input–output fuzzy clustering and particle swarm optimization, Fuzzy Sets Syst. 221 (16) (2013) 65–89.
[5] V. Chinnadural, G.D. Chandrashekhar, Neuro-levelset system based segmentation in dunamic susceptibility contrast enhanced and diffusion weighted magnetic resonance images, Pattern Recognit. 45 (9) (2012) 3501–3511.
[6] J.C. Wojdel, L.J.M. Rothkrantz, Mixed fuzzy-system and artificial neural network approach to automated recognition of mouth expression, in: ICANN 98, Springer, London, 1998.
[7] R.O. Duda, P.E. Hart, D. Stork, Pattern Classification, Wiley, 2000.
[8] X. Luan, B. Fang, L. Liu, W. Yang, J. Qian, Extracting sparse error of robust PCA for face recognition in the presence of varying illumination and occlusion, Pattern Recognit. 47 (2014) 495–508.
[9] D. Zhang, Z. Zhou, $(2D)^2$PCA: two-directional two-dimensional PCA for efficient face representation and recognition, Neurocomputing 69 (2005) 224–231.
[10] P. Nagabhushan, D.S. Guru, B.H. Shekar, $(2D)^2$FLD: an efficient approach for appearance based object recognition, Neurocomputing 69 (2006) 934–940.
[11] L. Chen, et al., A new LDA-based face recognition system which can solve the small sample size problem, Pattern Recognit. 33 (10) (2000) 1713–1726.
[12] H. Yu, J. Yang, A direct LDA algorithm for high-dimensional data with application to face recognition, Pattern Recognit. 34 (10) (2001) 2067–2070.
[13] X.-Y. Jing, D. Zhang, Y.F. Yao, Improvements on the linear discriminant technique with application to face recognition, Pattern Recognit. Lett. 24 (15) (2003) 2695–2701.
[14] A. Staiano, R. Tarliaferri, W. Pedrycz, Improving RBF networks performance in regression tasks by means of supervised fuzzy clustering, Neurocomputing 69 (2006) 1570–1581.
[15] A. Aiver, K. Pyun, Y.Z. Huang, D.B. O'Brien, R.M. Gray, Lloyd clustering of Gauss mixture models for image compression and classification, Signal Process. Image Commun. 20 (5) (2005) 459–485.
[16] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, 1981.
[17] R. Storn, Differential evolution, a simple and efficient heuristic strategy for global optimization over continuous spaces, J. Glob. Optim. 11 (1997) 341–359.
[18] K. Dervis, O. Selcuk, A simple and global optimization algorithm for engineering problems: differential evolution algorithm, Turk. J. Electr. Eng. 12 (2004) 53–60.
[19] Z. He, P. Niyogi, Locality preserving projections, in: NIPS 16, MIT Press, 2003, pp. 153–160.
[20] D. Cai, X. He, J. Han, Spectral regression for efficient regularized subspace learning, in: IEEE International Conference on Computer Vision, ICCV, Rio de Janeiro, Brazil, 2007, 2007, pp. 1–7.
[21] J. Yang, D. Zhang, A.F. Frangi, J. Yang, Two-dimensional PCA: a new approach to appearance-based face representation and recognition, IEEE Trans. Pattern Anal. Mach. Intell. 26 (1) (2004) 131–137.
[22] D. Hu, G. Feng, Z. Zhou, Two-dimensional locality preserving projections (2DLPP) with its application to palmprint recognition, Pattern Recognit. 40 (1) (2007) 339–342.
[23] S.-K. Oh, S.-H. Yoo, W. Pedrycz, Design of face recognition algorithm using PCA-LDA combined for hybrid data pre-processing and polynomial-based RBF neural networks: design and its application, Expert Syst. Appl. 40 (2013) 1451–1466.
[24] S.-K. Oh, W.-D. Kim, W. Pedrycz, B.-J. Park, Polynomial-based radial basis function neural networks (p_RBF NNs) realized with the aid of particle swarm optimization, Fuzzy Sets Syst. 163 (2011) 54–77.
[25] A. Sharma, K.K. Paliwal, A new perspective to null linear discriminant analysis method and its fast implementation using random matrix multiplication with scatter matrices, Pattern Recognit. 45 (2012) 2205–2213.
[26] S.-K. Oh, W.-D. Kim, W. Pedrycz, H.-S. Park, Fuzzy radial basis function neural networks with information granulation and its parallel genetic optimization, Fuzzy Sets Syst. 237 (2014) 96–117.