Journal of ELECTRICAL ENGINEERING, VOL. 52, NO. 9-10, 2001, 284–288

MULTILAYER PERCEPTRON, RADIAL BASIS FUNCTION NETWORK, AND SELF–ORGANIZING MAP IN THE PROBLEM OF FACE RECOGNITION

Miloš Oravec *

In this contribution, one and two-stage neural networks methods for face recognition are presented. For two-stage systems, the Kohonen self-organizing map is used as a feature extractor and multiplayer perceptron (MLP) or radial basis function (RBF) network are used as classifiers. The results of such recognition are compared with face recognition using a one-stage multilayer perceptron and radial basis function network classifiers.

K e y w o r d s: face recognition, feature extraction, classification, multilayer perceptron, radial-basis function network, self-organizing map

1 PATTERN RECOGNITION

One-stage pattern recognition system is shown in Fig. 1. It contains direct classification of original data. Figure 2 shows a two-stage pattern recognition system consisting of a feature extractor, followed by some form of classifier. The process of feature extraction transforms original data to a feature space, where it is possible to reduce dimensionality of transformed data ($m < p$). Generally, it results in better recognition results compared to one-stage systems.

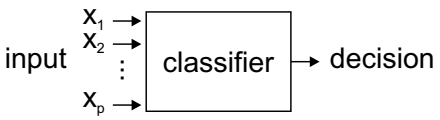

Fig. 1. One-stage pattern recognition system

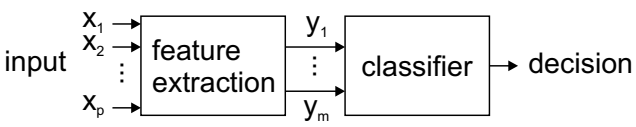

Fig. 2. Two-stage pattern recognition system including feature extraction

In this contribution, vector quantization by means of the Kohonen self-organizing map is used for feature extraction and MLP or RBF network is used for classification.

2 MULTILAYER PERCEPTRON

The basic multilayer perceptron (MLP) building unit is a model of an artificial neuron. This unit computes the weighted sum of the inputs plus the threshold weight and passes this sum through the activation function (usually sigmoid) [1]:

$$v_j = \Theta_j + \sum_{i=1}^{p} w_{ji} x_i = \sum_{i=0}^{p} w_{ji} x_i \tag{1}$$

$$y_j = \varphi_j(v_j) \tag{2}$$

where v_j is a linear combination of inputs $x_1, x_2, \ldots, x_p$ of neuron j, $w_{j0} = \Theta_j$ is threshold weight connected to special input $x_0 = -1$, y_j is the output of neuron j and $\varphi_j(\cdot)$ is its activation function. Herein we use a special form of sigmoidal (non-constant, bounded, and monotone-increasing) activation function — logistic function

$$y_j = \frac{1}{1 + \exp(-v_j)} \,. \tag{3}$$

In a multilayer perceptron, the outputs of the units in one layer form the inputs to the next layer. The weights of the network are usually computed by training the network using the backpropagation (BP) algorithm.

A multilayer perceptron represents a nested sigmoidal scheme [1], its form for single output neuron is

$$F(\boldsymbol{x}, \boldsymbol{w}) = \varphi\Big(\sum_j w_{oj} \varphi\Big(\sum_k w_{jk} \varphi\big(\ldots \varphi\big(\sum_i w_{li} x_i\big) \ldots\big)\Big)\Big) \tag{4}$$

where $\varphi(\cdot)$ is a sigmoidal activation function, w_{oj} is the synaptic weight from neuron j in the last hidden layer to the single output neuron o, and so on for the other synaptic weights, x_i is the i-th element of the input vector $\boldsymbol{x}$. The weight vector $\boldsymbol{w}$ denotes the entire set of synaptic weights ordered by layer, then neurons in a layer, and then number in a neuron.

* Department of Telecommunications, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, 812 19 Bratislava, Slovakia, E-mail: oravec@ktl.elf.stuba.sk

ISSN 1335-3632

3 RADIAL BASIS FUNCTION NETWORK

The radial basis function (RBF) network [2], [3] is based on a multivariable interpolation: Given a set of N distinct vectors $\{\boldsymbol{x}_i \in R^p \mid i = 1, \ldots, N\}$ and N real numbers $\{d_i \in R \mid i = 1, \ldots, N\}$, find a function $f\colon R^p \to R$ satisfying the condition

$$f(\boldsymbol{x}_i) = d_i\,, \quad \forall i = 1, \ldots, N\,. \tag{5}$$

RBF approach works with N radial basis functions (RBF) ϕ_i, where $\phi_i\colon R^p \to R$, $i = 1, \ldots, N$ and

$$\phi_i = \phi\big(\|\boldsymbol{x} - \boldsymbol{c}_i\|\big) \tag{6}$$

where $\phi\colon R^+ \to R$, $\boldsymbol{x} \in R^p$, $\|\cdot\|$ is a norm on R^p, $\boldsymbol{c}_i \in R^p$ are centers of RBFs. Centers are set to $\boldsymbol{c}_i = \boldsymbol{x}_i \in R^p$, $i = 1 \ldots, N$. Functions ϕ_i, $i = 1, \ldots, N$ form the basis of a liner space and interpolation function f is their linear combination

$$f(\boldsymbol{x}) = \sum_{j=1}^{N} w_j \phi\big(\|\boldsymbol{x} - \boldsymbol{c}_j\|\big)\,. \tag{7}$$

The solution for calculation of unknown coefficients w_j is a simple task for interpolation problem (simple solution of a set of linear equations), we will not consider it here. (Note: Presented RBF interpolation $\phi_i\colon R^p \to R$ to a mapping $\phi_i\colon R^p \to R^q$ is straightforward.) A very often used form of RBF is the Gaussian function

$$\phi(x) = \exp\Big(-\frac{x^2}{2\sigma^2}\Big) \tag{8}$$

where σ is the width (parameter).
An interpolation problem is easy to solve, in contrast to an approximation problem (there are N given points and n_0 functions φ, where $n_0 < N$), which is more complicated. Then it is a problem to set centers $\boldsymbol{c}_i$, $i = 1, \ldots, n_0$, also parameter σ may not be the same for all RBFs.

One possible solution for RBF approximation problem is a neural network solution. RBF network is a feed-forward network consisting of an input, one hidden, and output layers.

The input layer consists of p input units for distributing the p-dimensional input vectors into the network. The connection between i-th input element and a j-th hidden neuron contains weight c_{ji} (it is i-th coordinate of the j-th neuron's center). Each hidden neuron represents RBF — it computes the distance between input $\boldsymbol{x}$ and center $\boldsymbol{c}_j$, this distance then serves as the argument of RBF ϕ. Each hidden neuron is connected to the output neuron, this connection is weighted by weight w_{jk}. Linear output neurons compute linear combinations of their inputs.

RBF network learning consists of more different steps (description of RBF network learning can be found in [2], [3])

4 SELF–ORGANIZING MAP

Self-organizing map [4] is a neural network, which we use here for design of codebook for vector quantization [5]. Let $\boldsymbol{x} = [x_1, x_2, \ldots, x_p]^\top \in R^p$ be the input vector that is connected in parallel to all the neurons i in this network. The weight vector of cell i is denoted by $\boldsymbol{w}_i = [w_{i1}, w_{i2}, \ldots, w_{ip}]^\top \in R^p$. The analytical measure for the match of $\boldsymbol{x}$ and $\boldsymbol{w}_i$ is based on Euclidean distance. The minimum distance defines the winner $\boldsymbol{w}_c$.

The cells doing learning are not affected independently of each other but as topologically related subsets on each of which similar kind of correction is imposed. The neighborhood N_c around cell c is defined. At each learning step, all the cells within N_c are updated, whereas cells outside the neighborhood are left intact. This neighborhood is centered around that cell for which the best match with input is found:

$$\|\boldsymbol{x} - \boldsymbol{w}_c\| \min_i \|\boldsymbol{x} - \boldsymbol{w}_i\|\,. \tag{9}$$

The radius of neighborhood N_c is time-dependent, it is very wide in the beginning of training and it decreases monotonically with time.

The updating process (in discrete-time notation) is:

$$\boldsymbol{w}_i(n+1) = \boldsymbol{w}_i(n) + h_{ci}(n)[\boldsymbol{x}(n) - \boldsymbol{w}_i(n)]\,. \tag{10}$$

Denoting the coordinates of cells c and i by the vectors $\boldsymbol{r}_c$ and $\boldsymbol{r}_i$, a proper form for h_{ci} might be

$$h_{ci}(n) = h_0(n) \exp\big(-\|\boldsymbol{r}_i - \boldsymbol{r}_c\|^2/\beta^2\big) \tag{11}$$

where $h_0 = h_0(n)$ and $\beta = \beta(n)$ are suitable decreasing functions of time. More details about self-organizing map training can be found in [4].

5 FACE DATABASE

For our purposes we use the face database from MIT (Massachusetts Institute of Technology) [6] which consists of face images of 16 people (shown in Fig. 3), 27 of each person under various conditions of illumination, scale, and head orientation. It means, the total number of face images is 432. Each image is 64 (width) by 60 (height) pixels, eight-bit grayscale. An example of different face images (patterns) belonging to the same class is shown in Fig. 4.

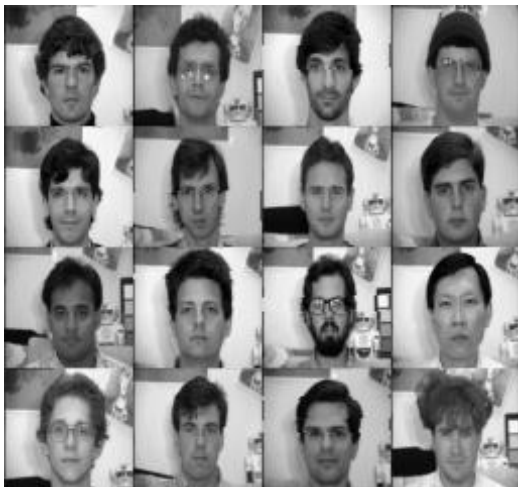

Fig. 3. Subjects in the face database

Fig. 4. Different examples of the same subject

6 FEATURE EXTRACTION METHOD

Our feature extraction method from image data is based on vector quantization (VQ) of images [5] using the Kohonen self-organizing map for codebook design. The indexes used for image transmission are used to recognize faces since they contain important features of the original image. This method is described in [7].

We perform vector quantization on 64 face images dividing original images to 4×4 blocks. Each image block (in our case 16-dimensional vector) is compared with a set of codevectors from a previously generated codebook by the self-organizing map. After a minimum distortion codevector has been selected, its index (address) can be transmitted, in our case it is used for recognition. After vector quantization, we form all the indexes corresponding to all blocks of the face image to the two-dimensional array, the so-called index image (see Section 7 for details and examples of such index images).

For image vector quantization, we used the configuration of the self-organizing map with 16×16 neurons with 16-dimensional weight vectors (neurons are arranged rectangularly), it corresponds to bit rate 0.5 bit/pixel compared to 8 bit/pixel original images. For training this map, first twelve 64×60 images from Fig. 3 divided to 4×4 blocks were used. Remaining four images from Fig. 3 were used for testing purposes.

7 SIMULATION RESULTS

First, we present the results of the one-stage recognition system, *ie* MLP or RBF network is used directly to classify the input face images. Thus, no feature extraction is performed on input data. An example of such recognition systems is briefly shown in Fig. 5.

MLP and RBF networks were trained using 48 faces (3 sets of all 16 persons with 3 different head scales) and tested on all 432 images and on images not used in the training process (*ie* $432-48 = 384$ faces). The input layer consists of $64 \times 60 = 3840$ neurons, the number of neurons in the output layer equals to the number of subjects in the face database (16), and there is one hidden layer for RBF network and there can be one or more hidden layers in the case of MLP. For example, notation $3840-32-16$ means that there are 3840 input, 32 hidden, and 16 output neurons in the neural network.

Recognition results are summarized in Tables 1 and 2 for MLP and RBF networks, respectively. Face set named "all" contains all faces included in the database (432 faces); face set named "all\train" contains all the faces except the training set (384 faces). More neural network configurations are considered, also the number of weights is shown (in italics) in order to get insight into computational demands during neural network training; biases of neurons are not counted here.

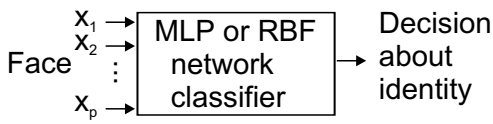

Fig. 5. One-stage recognition system using MLP or RBF network

Table 1. Recognition results for MLP

Net/faces	all (432 faces)	all\train (384 faces)
3840-16 *(61440)*	80.56 % (348)	78.12 % (300)
3840-16-16 *(61696)*	66.20 % (286)	61.98 % (238)
3840-32-16 *(123392)*	68.75 % (297)	64.84 % (249)
3840-48-16 *(185088)*	67.59 % (292)	63.54 % (244)
3840-96-16 *(370176)*	70.60 % (305)	66.93 % (257)
3840-144-16 *(555264)*	78.24 % (338)	75.52 % (290)
3840-192-16 *(740352)*	69.68 % (301)	65.89 % (253)

Table 2. Recognition results for RBF network

Net/faces	all (432 faces)	all\train (384 faces)
3840-16-16 *(61696)*	47.45 % (205)	40.89 % (157)
3840-32-16 *(123392)*	76.62 % (331)	73.70 % (283)
3840-48-16 *(185088)*	80.56 % (348)	78.12 % (300)
64 × 60-96-16 *(370176)*	80.32 % (347)	77.86 % (299)
64 × 60-144-16 *(555264)*	80.32 % (347)	77.86 % (299)
64 × 60-192-16 *(740352)*	79.63 % (344)	77.08 % (296)

Table 3. Recognition results for proposed method, classifier in the form of MLP

Net/faces	all (432 faces)	all\train (384 faces)
240-16 *(3840)*	82.18 % (355)	79.95 % (307)
240-16-16 *(4096)*	73.15 % (316)	69.79 % (268)
16 × 15-32-16 *(8192)*	75.93 % (328)	72.92 % (280)
240-48-16 *(12288)*	78.94 % (341)	76.30 % (293)
240-96-16 *(24576)*	79.17 % (342)	76.56 % (294)
240-144-16 *(36864)*	79.17 % (342)	76.56 % (294)
240-96-48-16 *(28416)*	65.51 % (283)	61.20 % (235)

Now let us concentrate on the behaviour of two-stage recognition systems based on the feature extraction method proposed in Section 6. Such a face recognition system is briefly shown in Fig. 6.

Table 4. Recognition results for proposed method with RBF network classifier

Net/faces	all (432 faces)	all\train (384 faces)
240-16-16 *(4096)*	76.39 % (330)	73.44 % (282)
240-32-16 *(8192)*	80.79 % (349)	78.39 % (301)
240-48-16 *(12288)*	82.87 % (358)	80.73 % (310)
240-96-16 *(24576)*	77.55 % (335)	74.74 % (287)
240-144-16 *(36864)*	78.94 % (341)	76.30 % (293)

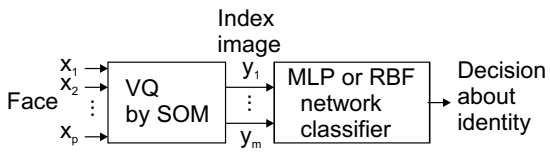

Fig. 6. Proposed face recognition system

Fig. 7. Index images (each is 16 × 15 pixels) corresponding to Fig. 3, 400 % zoom

After vector quantization each face image is represented by 240 eight-bit indexes, we form them to a 16×15 8 bit/pixel image (examples of such index images corresponding to original faces from Fig. 3 are shown in Fig. 7) which then serves as the input to MLP or RBF network classifier. For 16 × 15 pixel images the MLP or RBF network must contain 240 units in the input layer, 16 units in the output layer, and it can contain also one or more hidden layers for MLP or one hidden layer for RBF network. MLP and RBF networks were again trained using 48 faces (3 sets of all 16 persons with 3 different head scales) and tested on all 432 images and on images not used in the training process (*ie* 384 faces).

Recognition results are presented in Tables 3 and 4 for classifiers in the form of MLP or RBF network, respectively. Again, also the number of weights is shown in order

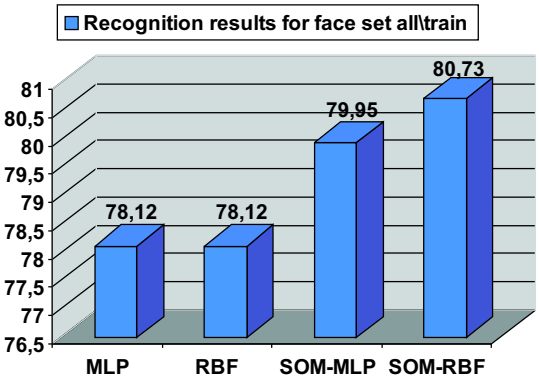

Fig. 8. Comparison of all presented methods

to get insight into computational demands during neural network training; biases of neurons are not counted here.

Comparison of best results (% of successfully recognized face images) for each of four presented methods is presented in Fig. 8. Only the neural network configurations with best recognition results are presented here. There is no difference between one-stage recognition systems. Two-stage systems working with feature extraction can reach better results. The best result (80.73 %) was reached by the system using the self-organizing map as a feature extractor followed by RBF network classifier.

8 CONCLUSION

We presented face-recognition results for one and two-stage recognition systems based on neural networks.

One-stage classifiers represented by MLP and RBF networks have strong computational demands and it is not a trivial task to get such a large network to convergence (comparison of the number of weights can be seen in tables presented above).

Two-stage systems using the Kohonen self-organizing map as a feature extractor followed by MLP and RBF network as a classifier were also presented. For feature extraction, this method uses the block approach. In order to perform vector quantization, the face images are divided to small blocks of pixels. This results in low computational demands both for self-organizing map and for classifier in the form of MLP or RBF networks. Convergence of such networks during training is without problems.

References

[1] HAYKIN, S.: Neural Networks — A Comprehensive Foundation, Macmillan College Publishing Company, New York, 1994.

[2] HLAVÁČKOVÁ, K.—NERUDA, R.: Radial Basis Function Networks, Neural Network World (1993), 93–102.

[3] POGGIO, T.—GIROSI, F.: Networks for Approximation and Learning, Proc. of the IEEE **78** No. 9 (1990), 1481–1497.

[4] KOHONEN, T.: The Self-Organizing Map, Proc. of the IEEE **78** No. 9 (1990), 1464–1480.

[5] NASRABADI, N. M.—KING, R. A.: Image Coding Using Vector Quantization: A Review, IEEE Trans. on Communications **36** No. 8 (1988), 957–971.

[6] TURK, M.—PENTLAND, A.: Eigenfaces for Recognition, Journal of Cognitive Neuroscience **3** No. 1 (1991), 71–86.

[7] ORAVEC, M.: A Method for Feature Extraction from Image Data by Neural Network Vector Quantization, Proc. of the 6th International Workshop on Systems, Signals and Image Processing IWSSIP'99, Bratislava, Slovakia, June 2-4, 1999, pp. 73–76.

Received 31 May 2001

Miloš Oravec (Ing, PhD) received the Ing and Dr degrees from the Faculty of Electrical Engineering, Slovak University of Technology in 1990 and 1996, respectively. Now, he is a teacher at the Department of Telecommunications, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology. His current research interests include neural networks and signal processing.