New 2D Matrix-Based Neural Network for Image Processing Applications

M. Mohamadian, H. Afarideh and F. Babapour

***Abstract*—In this study, a new nonlinear model for matrix-based classification and neural based learning has been proposed applicable to the dataset with vector/matrix output (Matrix I/O) especially in imaging analysis as a 2D data. The effect of conversion of an image into vectors was considered in order to indicate the elimination of some important information represented by the strong correlation between neighboring pixels in the matrix-based data. Furthermore, general form of the matrix based neural network was used for the purpose of segmentation of the three groups of images. This approach facilitates analysis of a new multilayer gradient descend network with matrix image inputs, multi hidden and output layer perceptron-like network. The proposed network was compared with the standard MLP network in its performance to obtain and predict desired results. Computer simulation and implementation of the model revealed that in the case of the small training set, the proposed non-linear model is more efficient than the standard linear MLP and Cai's model.**

***Index Terms*—neural networks, MLP, matrix inputs/outputs, image segmentation, matrix representation**

I. INTRODUCTION

The neural network is a general-purpose classification, clustering and function estimation tool with widespread applications. In most research areas, standard neural network (NN) is useful and effective due to the structure of their input as a one-dimensional vector. However, in some fields, e.g., image processing, when the input is in the form of a matrix, two-dimensional (2D), or when the input has higher dimensions, the standard structure and its mathematical relations are not directly applicable to the data. In these circumstances, the 2D or higher dimensional data should be converted into vectors before using in standard neural networks. However, this conversion may cause neural networks to require much more time to extract important local features in their blind training stage.

Furthermore, the neural network concept is based on learning the relation between inputs and their features, while the vectorization eliminates some of them, perhaps the important ones which affect all others; i.e. correlation between pixels. So the network wouldn't able to learn accurately and it would also hardly understand the relationship between them.

The first step in defining an artificial neural network is providing the inputs in accordance with the conventional structure of ANNs [1, 2, 3]. Thus, for a set of inputs with inherent matrix structure, such as images which are non-vector pattern, the inputs have to be vectorized into a vector pattern by different conventional or non-conventional methods such as concatenation. However, it has been proven that vectorization is not useful for recognition approach because of 2D-PCA and algebraic operations on the image.

Several researchers have recently paid full attention to vectorization problems in the other fields and applications. As such, these methods can be cited in the two-dimensional principal component analysis, which can extract features directly from the image matrices and two-dimensional Linear Discriminant Analysis (LDA) [4], also based on the image matrices, these can overcome the singularity problem implicit in the classical linear discriminant analysis. Furthermore, both mentioned methods extract features only from the row direction of image matrices. Yang *et. al.* also have developed a coined two-dimensional principal component analysis for image representation which the original image matrices are used directly, and its eigenvectors are derived for image feature extraction for constructing an image covariance matrix [5]. Moreover, It was reported in [6] another this type of approach for efficient face representation and recognition by two-directional two-dimensional PCA.

In this regard, P. Daniusis [7] has used this idea for the neural network with matrix input. He has implemented it in the linear single layer network with one scalar output. Even so, it was not appropriate for image processing approaches with at least vector output features.

In this paper, inspired by the standard MLP model [8], the matrixized versions of ANN, FMatNN (Full Matrix based Neural Network) with the ability of extended to the desired number of hidden layers and output numbers are discussed and is developed for image feature extraction. As opposed to conventional ANN, FMatNN is based on 2D matrices rather than 1D vector. That is, the image matrix does not need to be previously transformed into a vector. In various

Manuscript received January 11, 2015; revised March 11, 2015.

M. Mohamadian is with the Department of Energy Engineering and Physics, Amirkabir University of Technology, P.O. Box 4155-4494, Tehran, Iran. (Phone: +98- 21- 6454 5230; fax: +98- 21- 6454 5230; e-mail: mohamadian@aut.ac.ir).

H. Afarideh is with the Department of Energy Engineering and Physics, Amirkabir University of Technology, Tehran, Iran. (e-mail: hafarideh@aut.ac.ir).

F. Babapour is with the Faculty of Engineering, Science and Research Branch, Islamic Azad University (IAU), Tehran, Iran. (e-mail: farshid.mofrad@yahoo.com).

applications of image processing, usually the image's matrix and the effects of vicinity and correlation between matrix components, are investigated. By applying it to our imaging database, the desired results are obtained.

This approach facilitates analysis of a new multilayer gradient descent network with matrix image inputs, multi hidden and output layer perceptron-like network. Subsequently, in section 2, effects of various vectorization methods are analyzed to indicate image information loss. So next, Matrix based ANN (MatNN) is presented which could skip over the vectorization step in neural network analysis. In section 3, the mathematical relations of this new ANN are demonstrated. In addition, the data set and proposed techniques are given in details. Experimental results are mentioned in this section too. In section 4, we clarify the practical aspects of the proposed algorithms which the proposed network is compared with the standard MLP network in its performance, and predicted desired results are obtained.

II. Material and Methods

In the analysis of image processing or the other multidimensional data analysis, usually matrices are used for regression, classification and so on, while vectors are established in most procedures. Therefore, the data matrix decomposes into the vectors in different ways with the following disadvantages:

I. It may destroy neighboring information of pixel data in matrix pattern structures.
II. It usually creates very high dimensional vector and so large training data set.
III. In most cases, when the inputs are matrices, they are relatively high dimensional. If dimensionality of the input is high and cardinality of the training set is relatively low, we can face a small training sample problem [7].
IV. In our focus area, inputs are m×n image matrices which provide a vector with m.n dimensions without original inner matrix spatial information and lose correlation between pixels themselves or matrix components.

In addition to these disadvantages, various vectorization methods result different outcomes for distinctive data matrix structure. It's so related to direction of objects in the image and direction of vectorization. In the next section, we try to illustrate these effects due to implement some vectorization way.

A. Methods of Vectorization

This section attempts to illustrate the effects of different vectorization methods in 2D data processing by NNs. In such cases, we need other techniques to consider the matrix inputs.

At segmentation process which border of objects in images and curves is important to us, it will be significant how to explore the connectivity of pixels in the images. Vectorization usually performs by sorting rows of image matrix tandem (Fig.1 Method#2). Sometimes it's used another type of sorting (Fig.1) with regard to the issue. But, part of the information will be lost by re-aligning the image structure and neighborhoods commix. Especially when using this type of input data in neural networks, which learn all relations and connectivity.

So at first, by using a vector pattern, the obtained results of various image vectorization methods are compared and consequently, the effect of data sequence in ANN response is considered. Hence, according to the achieved results, one of the benefits of matrixazing algorithm could be observed and explained in the following way:

The above approach is examined on three different datasets; donut shape, alphabet and arrow images, to determine the effects of different vectorization and various routes for pre-processing followed by methods number 1 in 7. Four types of vectorization are illustrated in Figure 1.

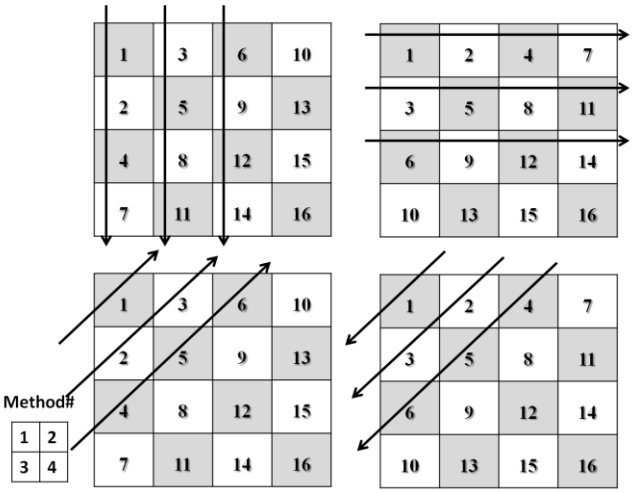
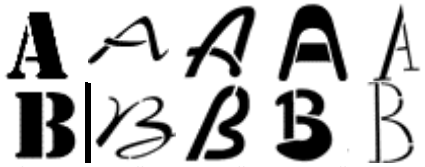


Fig. 1. Vectorization methods; Method #1 to Method #4

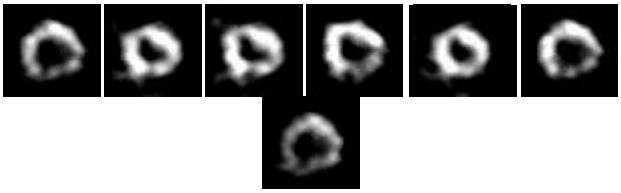
Alphabet dataset includes some of the alphabet images as process input with certain pattern images (Figure 2-a). It's used only two alphabets A and B form different datasets. The target of this network is a recognition of alphabets. The same way is repeated for donuts and arrow images in Figure 2-b and 2-c respectively. Donut shapes are short axes images of cardiac at SPECT imaging. Segmentation of the cardiac images is a serious subject of consideration in this area for diagnostic issue. Also, it's used arrow shape to highlight the effect of various ways of vectorization.

The work is started with small dataset and simple Feed Forward Neural Network (FF-NN) for all three letters, arrow and donut images. For example, in vectorization by method #4, after 2000 iterations with certain epoch, the desired result is achieved. Conversely, after 10,000 iterations under the same condition, by method #1, no result is yielded. The results are summarized in Table1.

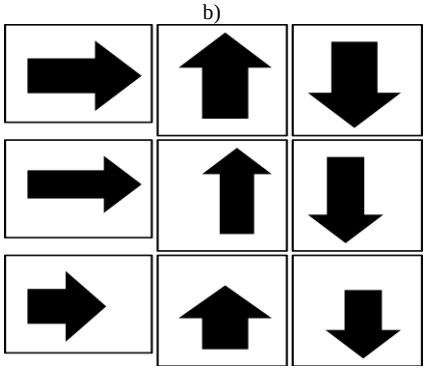
In addition, there is another parameter in Table 1 called correlation coefficient. It is due to the same image, in a different vectorization method which provides vectors with significant difference in the correlation coefficient. In various applications of image processing, the image's matrix and the effects of vicinity and correlation between matrix components are investigated. Pearson's correlation coefficient is widely used in statistical analysis, pattern recognition, and image processing [9-13].



a) Two letters from Dataset #1, Dataset #2, Dataset #3, Dataset #4, Dataset #6



b)



c)

Fig. 2. a) Letter A and B in some Alphabet dataset (Two letter from Dataset #1, Dataset #2, Dataset #3, Dataset #4, Dataset #6), b) Some donut shape image dataset sample and c) Sample of arrow dataset

Applications for the latter include comparing two images for the purposes of image registration, object recognition, and disparity measurement. The correlation coefficient has the value 1 if the two images are absolutely identical, 0 if they are completely uncorrelated, and -1 if they are completely anti-correlated, for example, if one image is the negative of the other.

Beside correlation, a novel image-based tracking algorithm using MATLAB was developed, dubbed digital differential image tracking (DDIT), which can be with similar result.

Due to the high dependence of pixel's correlation of image, the results of our simulation indicated that standard ANN scarcely learn our procedure for simple classification or segmentation of even small donut dataset with more than 50,000 iterations. Moreover, for the arrow dataset, with feed forward nonlinear standard two layers ANN, just method #1 achieved the desired classification after 3400 iterations, while the others did not spot the error (Err) above the 66%. These results are predictable due to the position and the type of the arrow dataset images. This fact indicates that the matrix based consideration is important, especially in the case of images and 2D data of unpredictable structure.

Table 1 illustrates the inappropriateness of vector pattern techniques in comparison with the matrix based networks, and also highlights the ignoring correlation effects of neighboring components. It should be noted that this test is for a small matrix dataset. Therefore, the effect would be more distinguished in the original large datasets.

TABLE I
MEAN CORRELATION COEFFICIENT AND NUMBER OF ITERATION TO REACH THIS COEFFICIENT AND ERR (ERROR)
Dataset #1: arrow dataset, Dataset #2: alphabet dataset

Vectorization method	Method #1	Method #2	Method #3	Method #4	Method #5	Method #6	Method #7	Hidden Unit
Mean Corr.Coeff.*#1	1	0.001589	0.225817	-0.045237				
Dataset #1 (Iteration No.)	3400	After 13000 Err~66%	After 12000 Err~66%	After 15000 Err~66%	-	-	-	[100, 50]
Corr.Coeff. #2	1	-0.0229	-0.0028	-0.0229	-	-	-	-
Dataset #2 (Iteration No.)	>10000	3000	>1000	2000	4000	3000	3000	100

*Mean Correlation Coefficient with method#1

B. *Matrix ANN Approach*

Besides the mentioned techniques which need the vectorization of input matrices, there are several feature extraction methods which work directly on 2D data transformation, such as 2D wavelet, 2D Fourier's transformation and so on. This type of vector-based representation is natural in most of data analyses [14]. For instance, Hong [15] has employed singular value decomposition (SVD) to extract a set of singular values as classification features directly from the image matrix and a good performance of classification was obtained on their own face database. Further, Tian *et al* [16] have developed and refined Hong's work and finally have obtained better classification performance on some public face datasets. In such cases, the vector-based representation does not consider an inner structure of the inputs, which can provide useful information.

And also, computer experiments [17, 7] demonstrate, even when initial inputs are vectors, it can be useful to represent them as matrices (or higher order tensors) and apply matrix/tensor-based models. This approach can be especially useful in the case of small training sample [17], since matrix (or tensor) - based models usually have less parameter [7].

In this study, a new nonlinear model (which is more flexible than linear models) for matrix-based classification and any other neural based learning is proposed and applied to the three different dataset with vector/matrix output (Matrix I/O). The computer simulation and implementation of the model disclosed that in the case of the small training sample, the proposed non-linear model can be more efficient than the standard linear MLP and Cai's model [17]. Although, it seems to be more accurate for any type of images and 2D problems, because of ever-increasing the calculations, it takes more time for large size matrices; so, it needs data reduction process prior to using NN [18].

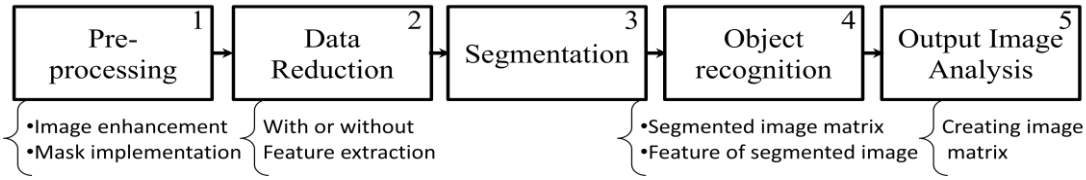


Fig. 3. Block diagram of the image processing stages

Description of MatANN

An idea of constructing MatANN is drawn from the image segmentation problems, developed especially for extracting features directly from the image matrix; moreover, stem from the fact that the number of parameters estimated by a tensor classifier can be less than which estimated by a traditional vector classifier.

Linear form of this implementation for single hidden layer networks with one neuron has been discussed by P. Daniusis *et al* [7]. But it was just for the purpose of the scalar (1D) output problem which is not proper for full matrix based operations in image processing approaches. So, this study attempts to improve the premier idea as a general form. At first, the matrix input and the vector output structure are considered to prove this idea for making better and faster configuration of the image based neural network. For several data based ones as discussed below, the desired response is elicited. Hence, improving the method for the general forms of input and output (vector and matrix forms) is done.

Different types of ANN structure, usually are used, such as: feed-forward ANNs, SOMs, Hopfield networks, probabilistic ANNs, radial basis function networks, CNNs, constraint satisfaction ANNs and RAM-networks, and a self-organizing architecture with fuzziness measures. Feed forward ANN is a versatile network, which is employed in this analysis.

Additionally, one of the challenges in image processing is segmentation. There are several ANN-based segmentation approaches adopted directly towards the pixel data or voxel data, either through the convolution operation on the image window, or from local features provided to a neural classifier. Those methods which use the pixel data require a relatively direct access to the matrix of images and correlation of neighboring pixels affected by the final results. It is expected that this network configuration will improve most of them by more accurate results and faster process. It will be considered in this future work for some known standard network and methods. Furthermore, this approach can be useful for any other types of calculation, especially estimation and prospect with converting the vectors into tensor (in our case; second order tensor), to achieve the advantages of such a model which mentioned here.

In the passage of the vector form of tensor representation, it is necessary to express the equations and the functions for the new structure. For instance, a linear function is written as $f(x)=W^Tx+b$ (similar to the usual notation of neurons in neural network's subjects) in the vector space, is converted to $f(X)=u^TXv+b$ in the matrix space (second order tensor), where x and X are inputs in the vector and matrix structures respectively, and $u \in R^n$, $v \in R^m$ are two vectors. In comparison with conventional neural network notations, W as the weight's matrix of neurons in vector space changes to recent variables; u and v which can be a vector or matrix in general form of this discussion.

Image segmentation

Segmentation is the process of partitioning a digital image into multiple segments (sets of pixels). The purpose of image segmentation is to simplify and/or change the representation of an image into meaningful regions with respect to a particular application. When it is considered as a classification task, the purpose of segmentation will be to assign labels to individual pixels or voxels [19].

According to the previous discussion, the survey area in segmentation is the object recognition. The feature-based segmentation fields are ignored here. Figure 3 shows the block diagram of this approach to achieve the aim. Some models were attempted to implement in each part at pre-processing stage (block#1) entered after image enhancement and some noise reduction processes. In this way, the special mask for each dataset is founded.

Moreover, two paths are prospected at block #2; with or without data reductions (feature extractions) which refer to the above discussion, but at first this part is ignored.
The main block is segmentation, block#3 where the model is examined.

At block #4, two ways of object recognition are defined according to the desired result displaying for next parameter calculation aims: No.1 training the FF neural network for a segmented image as an output. No.2 estimating the origin and radius of the boundary circle of the network. The first one is the general form of any type of object segmentation, although, the second one can be used for some other application too.

III. Results and Discussion

In comparison with standard MLP, from the explained definition in this model for m×n dimensional matrix data, it is needed to estimate only m+n parameters. However, in the standard linear back propagation, the m×n parameters must be estimated.

This study attempts to generalize MLP standard linear model in the multilayer perceptron framework [20,21], analyze mathematical relations of this model, and apply it to the image segmentation or classification problems.

A. Matrix Based ANN Model (MatANN)

To introduce our proposed model, it is proper to have an overview of perceptron neural networks and its associated mathematical relations. The basic form of a multilayer perceptron (MLP) which is defined for single hidden layer is

expressed by the following formula:

$$\hat{y}(x) = \varphi(\sum_{i=1}^{N} w_i x_i + b) = \varphi(\mathbf{w}^T\mathbf{x}+b)$$

Where **w** denotes the vector of weights, **x** is the vector of inputs, b is the bias and φ is the activation function.
Most of the time, MLPs with two or even more hidden layers are used. For example, an MLP with two hidden layers is written as follows:

$$\hat{y}(x) = \varphi^2\left(\sum_{i=1}^{N} w_i^2 \varphi^1\left(\sum_{j=1}^{M} w_j^1 x + b^1\right) + b^2\right)$$

In analogy with MLP model, the following information comprises some phrases of the first proposed model algorithm which introduced and reviewed as Matrix input/Vector output Neural Network (MVNN) model with two hidden layers.

$$y(X) = f^2(W^2 * f^1(u_i X v_{i\ i\epsilon 1:N} + b^1) + b^2)$$

Our first model is written as follows for single (ability to extend to multi) non-linear hidden layer and linear/non-linear output layer network with the desired neuron in each layer.

$$y(X) = f^2(W^2 * a^1 + b^2)$$
$$a^1 = f^1(u_i X v_{i\ i\epsilon 1:N} + b^1)$$

Where $\boldsymbol{y}$ is the output vector, $\boldsymbol{X}$ is the input matrix, $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ are the weight vectors of the hidden layer, and $\boldsymbol{W}$ is the weight matrix of the output layer**,** b^i are bias of layers, f^i are activation functions of each layer, a^1 are the output hidden layer and N is the number of hidden layer neurons. In this notation, it is expressed the new form of weight matrix/vector with matrix input using two matrix/vector, $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$, instead of just a single conventionally weight matrix.

B. *Parameter Calculation and Phrases*

Different methods can be employed to estimate the parameters of the model. In this section, our proposed MVNN (Matrix Input/vector output NN) and FMNN (general matrix Input/Output NN) models are paraphrased by the known training algorithms: the gradient descents [22,2]. Gradient descent is a first-order optimization algorithm and works in spaces of any number of dimensions, even in infinite-dimensional ones. The gradient descent can take much iteration to compute a local minimum with a required accuracy. Beside the training algorithm, it should define proper performance function which indicates our error function to aim desired neural network results.

The designated set of training data is made up of M input-output by $T = \{(X_i, Y_i)\}_{i=1}^{M}$, where each X_i *is* $m \times n$ matrix (input) and *Yi* is the desired (target) vector/matrix response of *i*th input (output). The transfer function in tensor least square analysis is as follows:

$$\min_{u,v,w,b} \sum_{i=1}^{m} [f^2(W^2 * f^1(u_i X v_{i\ i\epsilon 1:N} + b^1) + b^2) - y_i]^2$$

Gradient Descent

As stated before, primarily we exerted the gradient descent algorithm to train our network. In this regard, the mean squared error is minimized using the gradient descent algorithm [20]. It means that the performance function in the matrix based gradient descent algorithm, as mentioned above, generally is as follows:

$$\mathrm{Err}(..) = \frac{1}{M} \sum_{(X,y)\in T} (\hat{y}(X) - y)^2$$

So, we compute the partial derivatives of performance function in vector output space by the following formulae:

$$\frac{\delta \mathrm{Err}(..)}{\delta b_k^1} = \frac{1}{M} \sum_{(X,y)\in T} (\hat{y}(X) - y).\, f^{2'}(W_{:,k} \times a_k^1 + b^2).W_{:,k}.\, f^{1'}(u_k X v_k + b_k^1)$$

$$\frac{\delta \mathrm{Err}(..)}{\delta b_k^2} = \frac{1}{M} \sum_{(X,y)\in T} (\hat{y}(X) - y).\, f^{2'}(W_{k,:} \times a^1 + b_k^2)$$

$$\frac{\delta \mathrm{Err}(..)}{\delta u_{k,l}} = \frac{1}{M} \sum_{(X,y)\in T} (\hat{y}(X) - y).\, f^{2'}(W_{:,k} \times a_k^1 + b^2).W_{:,k}\, .\, f^{1'}(u_k X v_k + b_k^1) \sum_{i=1}^{n} v_{k,i} X_{l,i}$$

$$\frac{\delta \mathrm{Err}(..)}{\delta v_{k,l}} = \frac{1}{M} \sum_{(X,y)\in T} (\hat{y}(X) - y).\, f^{2'}(W_{:,k} \times a_k^1 + b^2).W_{:,k}\, .\, f^{1'}(u_k X v_k + b_k^1) \sum_{j=1}^{m} u_{k,j} X_{j,l}$$

$$\frac{\delta \mathrm{Err}(..)}{\delta W_{k,l}} = \frac{1}{M} \sum_{(X,y)\in T} (\hat{y}(X) - y).\, f^{2'}(W_{k,l} \times a^1 + b_k^2).a_l$$

Since,

$$u_k^T X v_k = \sum_{i=1}^{n} v_{k,i} \sum_{j=1}^{m} u_{k,j} X_{j,i} = \sum_{j=1}^{m} u_{k,j} \sum_{i=1}^{n} v_{k,i} X_{j,i}$$

In this research, we applied this modified phrases using MATLAB software to analysis the 2D I/O (Input/Output) neural network. It is computed all partial derivatives of performance function versus all five parameters. Then in the following, it is implemented on some various datasets to evaluate our proposed method.

I/O matrix ANN (General Full Matrix Neural Network– FMatNN)

Extended equations of the general form are written as follows, although it is similar to the vector form by some difference in detail, which cause some complexity in software implementation.

$$Y_{k1,k2}(X) = f^2(W_{k1,k2} * f^1(u_i X v_{i\ i\epsilon 1:N} + b^1) + B_{k1,k2}^2)$$

Where, $W_{k1,k2}^k$ is the weight matrix of layer two. It is an N1×N2 matrix whose each component is a vector in hidden

layer size. The superscript k, points to this hidden neuron number. $B^2_{k1,k2}$ is the bias of layer 2 which is also a matrix in this new structure. Other variables are similar to the previous phrase. N1 and N2 are the sizes of neurons in layer 2 which is also the size of target matrix. Figure 4 shows this structure.

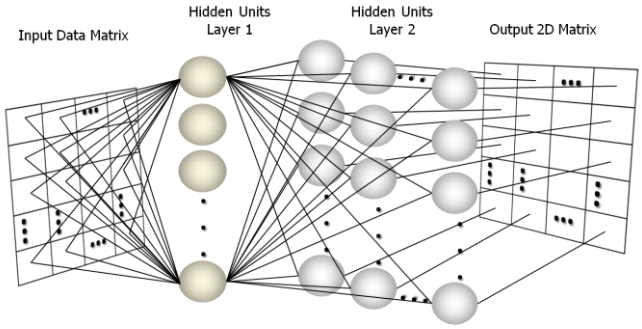


Fig. 4. FMNN neuron structure

Also, according to this modification, the partial derivatives of the performance function in the matrix output space are converted to the expressions as follows:

$$e^{i}_{N1\times N2} = (\hat{Y}(X) - Y) \equiv (a^{2}_{N1\times N2} - \text{Target}^{i}_{N1\times N2})$$

Where $a^{2}_{N1\times N2}$ is output matrix of the second layer, $\text{Target}^{i}_{N1\times N2}$ is desired output and $e^{i}_{N1\times N2}$ is error function. So partial derivatives of the performance function which is error function will be:

$$\frac{\delta \text{Err}(..)}{\delta b^1_k} = \frac{1}{M\times N1\times N2}\sum_{(X,y)\in T}\sum_{j=1}^{N2}\sum_{i=1}^{N1} e_{i,j}\,.\,f^{2'}(W\times a^1 + b^2)_{i,j} \,.\, W^k_{i,j}\,.\,f^{1'}(u_k X v_k + b^1_k)$$

$$\frac{\delta \text{Err}(..)}{\delta b^2_{k1,k2}} = \frac{1}{M}\sum_{(X,y)\in T} e_{k1,k2}\,.\,f^{2'}(W^{all}_{k1,k2}\times a^1 + b^2_{k1,k2})$$

$$\frac{\delta \text{Err}(..)}{\delta u_{k,l}} = \frac{1}{M\times N1\times N2}\sum_{(X,y)\in T}\sum_{j=1}^{N2}\sum_{i=1}^{N1} e_{i,j}\,.\,f^{2'}(W\times a^1 + b^2)_{i,j} \,.\, W^k_{i,j}\,.\,f^{1'}(u_k X v_k + b^1_k)\sum_{p=1}^{n2} v_{k,p}\,X_{l,p}$$

$$\frac{\delta \text{Err}(..)}{\delta v_{k,l}} = \frac{1}{M\times N1\times N2}\sum_{(X,y)\in T}\sum_{j=1}^{N2}\sum_{i=1}^{N1} e_{i,j}\,.\,f^{2'}(W\times a^1 + b^2)_{i,j} \,.\, W^k_{i,j}\,.\,f^{1'}(u_k X v_k + b^1_k)\sum_{p=1}^{m} u_{k,p}\,X_{p,l}$$

$$\frac{\delta \text{Err}(..)}{\delta W^k_{i,j}} = \frac{1}{M}\sum_{(X,y)\in T} e_{i,j}\,.\,f^{2'}(W\times a^1 + b^2)_{i,j}\,.\,a^1_k$$

Where, e^i is the difference of network output and the desired target of i*th* training data which is a matrix−N1×N2 size.

Review the training algorithm

In our proposed MVNN/FMNN training process, the weight changes are calculated according to the above computed equations by the following expressions in several times. The number of iterations depends on the problem and the data population. The summarized algorithm steps are as follows:

1. Set initial weights *bi*, **u***i*, and **v***i* , number >0 and the learning rate η>0, sufficiently large natural number *N* and *i*=0;
2. Calculate the weight corrections:

$$\Delta b^{1,2}_k = -\eta\,\frac{\delta \text{Err}(..)}{\delta b^{1,2}_k}$$

$$\Delta u_{k,l} = -\eta\,\frac{\delta \text{Err}(..)}{\delta u_{k,l}}$$

$$\Delta v_{k,l} = -\eta\,\frac{\delta \text{Err}(..)}{\delta v_{k,l}}$$

$$\Delta W_{k,l} = -\eta\,\frac{\delta \text{Err}(..)}{\delta W_{k,l}}$$

3. Check the performance function for minimization of the cost with the network goal.

4. Repeat Step 2 until the cost (4) is greater than and *i*<*N*.

The same calculation is performed in the general form of the weight corrections of Δb^1_k , $\Delta b^2_{k1,k2}$, $\Delta u_{k,l}$, $\Delta v_{k,l}$, $\Delta W^k_{k1,k2}$.

C. Results of simulation

Many different techniques are used for the image segmentation and classification [23]. In this section, our proposed MVNN/FMNN model is used to SPECT images of myocardial (short axis slices) segmentation, as a donut shape image, although the method is examined for other types of dataset too which are introduced in the following. The aim is to compare the new proposed model and the standard MLP with some training algorithms and parameter settings.

The Data set

Three groups of data are used; first one included five alphabets for the purpose of recognition of them. It is a well known application of neural networks in optical character recognition (OCR) which can be improved in terms of accuracy and speed. The second one included some multi-directional donut shape images, which in this case, the purpose is segmentation of them. We employed short axis images of cardiac SPECT that are similar to donut shape. And the third one included three groups of arrow shape images with different direction for direction identification. Each of our datasets has been selected for especial aim.

Beside the particular application of them, this selection illustrates the performance of our method in a distinctive issue of segmentation and classification. For the first group, a number is allocated to each alphabet as an output. However, for the next one, the manual segmentation is used as the target and for the arrows, target is direction Number. The input matrix in all three groups is a matrix of intensity levels of alphabets, donut shapes and arrow images which all are 2D matrix.

Alphabets Input Data

The input matrix is a matrix of intensity levels of the images; $X_t \equiv$ Alphabets Pattern Images, 2D matrix. In this study, 5 alphabets from 9 datasets are considered. The target value for this matrix is defined by $y_t \equiv$ number of it from alphabets lists – scalar number from 1 to 5 for each database. The set of training data is defined by $S_{training}=\{(X_t, y_t)\}_{t=1}^{5*9}$. It should be noted that these are handwriting alphabets, which have not any defined structure to learn to the system. In this field of research, many approaches have been tried to solve character recognition problem using different methods [24].

Donut (or spiral) Input Data

The input matrix is also a matrix of intensity levels of donut images; $X_t \equiv$ Donut Images, 2D matrix. The target value for this matrix is defined by $y_t \equiv$ segmentation vector of Input Image (in MVNN) – this vector is the reshaped manual segmented image matrix. The set of training data is defined by $S_{training}=\{(X_t, y_t)\}_{t=1}^{11}$, and the set of testing data is defined by $S_{testing}=\{(X_t, y_t)\}_{t=1}^{n}$. In general form (FMNN), it will be examined to process the matrix target as an output. As mentioned earlier, these donut shape images are short axes images of myocardial.

Arrows Input Data

The input matrix is a matrix of intensity levels of three different direction arrow images; $X_t \equiv$ Arrow Images, 2D matrix. In this study, 3 directions from 3 datasets are considered. The target value for this matrix is defined by $y_t \equiv$ number of directions – scalar number from 1 to 3 for each database which 1 is toward up, 2 is left and 3 right. The set of training data is defined by $S_{training}=\{(X_t, y_t)\}_{t=1}^{3*3}$.

D. *Prediction Algorithms*

For the first step in this paper, it is only checked with standard MLP and compared with the two following algorithms:

- MVNN/FMNN (GD) – the MVNN/FMNN non-linear two layer model, trained with the gradient descent algorithm.
- NN (GD) – the standard 2 layer MLP non-linear model, trained with the gradient descent algorithm.

Standard MLP is suitable for the neural network with an inherently scalar or vector output. However, for the purposes of image processing, vectorization for converting matrices of image to proper input requires much more time to keep the effects of neighboring pixel correlation in the results. I mean, because of displacement the location of pixels compared to each other and their neighbors, neural network need more time to learn the true relation of pixels.

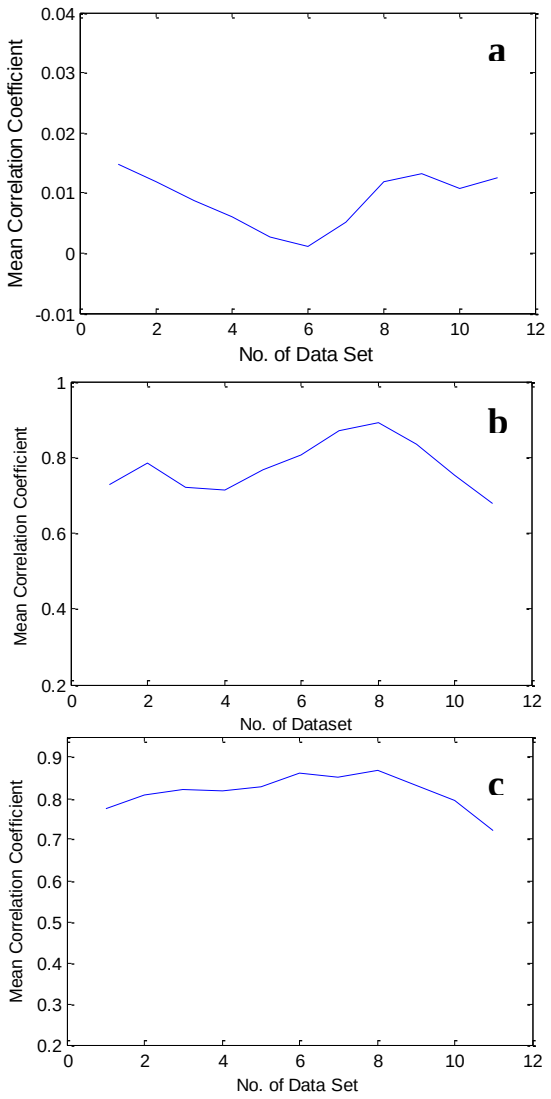


Fig. 5. Mean correlation coefficient of a) standard MLP b) MVNN c) FMNN for the donut shape image dataset

The network, which is trained with matrix I/O will learn properly the correlation between pixels. So, the appropriate parameter for comparison can be the correlation coefficient, which is mentioned in Figure 5. As can be seen in Figure 5, the mean correlation coefficient of the standard MLP is less than 0.02 while it is higher than 0.7 and 0.72 for MVNN and FMNN respectively (see Table 2).

IV. Conclusion

Conventionally, the mean squared error (MSE), sum squared error (SSE), etc. are used for performance of linear regression. In statistics, the mean squared error (MSE) of an estimator measures the average of the squares of the "errors", that is, the difference between the estimator and

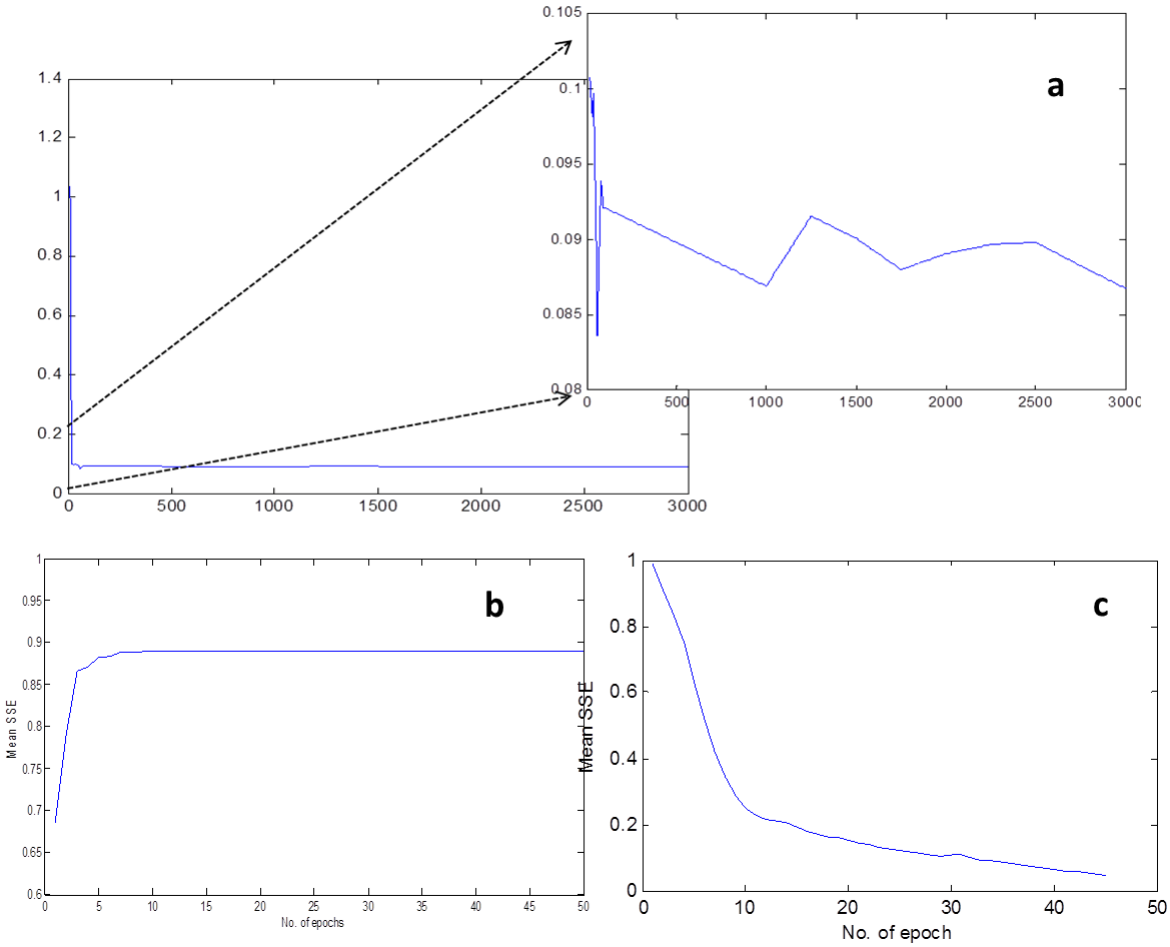


Fig. 6. SSE plot of a) MVNN b) FMNN and c) MLP for the donut shape image dataset

what is estimated. Also, SSE is the sum of the squared differences between each observation and its group's mean. Although in this paper SSE is calculated, as mentioned earlier, another parameter called 'correlation coefficient' is discussed such as performance function too, which is more meaningful for images or other naturally matrix based data. This function calculates the correlation coefficient of the network output and desired output (called 'target' in neural network notations).

It was considered similar to SSE as a checking parameter of the iteration loop. However, all algorithms are trained to minimize the mean squared error. Furthermore, overall SSE is calculated for the same scenario using the standard neural network for donut shape dataset to compare with MVNN/FMNN as illustrated in Figure 6. Neural network seems hardly to achieve the desired output with very high number of iterations, although its SSE value decreases with the increasing iteration number.

As noted, the sum square error (SSE) is one way of performance measurement, by which the simulation shows the decline SSE for both methods, while the correlation coefficient of standard MLP is very low, even for the high iteration number (>10000) is less than 0.1. Therefore, this means that SSE is an inappropriate parameter for diagnosing network performance in some situations and it should be better to have both performance functions (SSE and average correlation) at the same time for image and any other natural 2D datasets.

A. Comparison of the Prediction Models

The Tables 2 and 3 show the average SSE and the average correlation coefficient over the test set for the NN and MVNN/FMNN proposed algorithms with various settings, respectively. These parameters were compared as a performance function. Since the distinguished difference was observed at lower iteration, training was not repeated for higher epochs. Even so, the standard NN did not achieve MVNN/FMNN values at higher iteration.

TABLE II
Comparison of the MVNN/FMNN (GD) and NN (GD) Mean SSE

	MVNN (GD)	FMNN(GD)	NN(GD)	No. of iteration
Hidden Units	100	80	10	
-		0.8889	-	100
0.0907		0.8889	0.8243	1500
0.0926		0.8889	0.4222	3000
-		-	0.2831	9000
-		-	0.2032	15000

Mean SSE is noted in the above table as a comparison of our proposed MVNN/FMNN (Gradient Descent) and conventional neural network (Gradient Descent).

Mean correlation coefficient (and min-max value) is indicated in table 3 as a comparison of our proposed MVNN/FMNN (Gradient Descent) and conventional neural network (Gradient Descent).

TABLE III
COMPARISON OF THE MVNN /FMNN (GD) AND NN (GD)
Mean correlation coefficient (and min-max value)

	MVNN (GD)	FMNN(GD)	NN(GD)	No. of iteration
Hidden Units	100	80	10	
-		0.8162 (0. 8668 – 0.7223)	-	100
0.815 (0.646 – 0.961)		0.8162 (0. 8668 – 0.7223)	25.6056×10^{-3} $(16.55 - 33.90) \times 10^{-3}$	1500
0.767 (0.677-0.891)		-	14.8357×10^{-3} $(10.09 - 19.25) \times 10^{-3}$	3000
-		-	11.0552×10^{-3} $(5.35 - 21.49)\times 10^{-3}$	9000
-		-	8.97×10^{-3} $(1.016 - 14.77) \times 10^{-3}$	15000

Figure 7 indicates a sample result of the proposed method for the donut image segmentation process. As was noted, donut shapes are cardiac SPECT short axis image which is illustrated manual segmentation, our proposed FMNN segmentation and error images of typical sample respectively.

a b c

Fig. 7. Donut sample image data; a) Manual segmentation, b) FMNN segmentation, c) error image

At the first step, we followed the other well-known methods. So, our method applied just for finding the outer boundary and again for inner one. Figure 8 points out our proposed FMNN segmentation of the outside and inside region respectively.

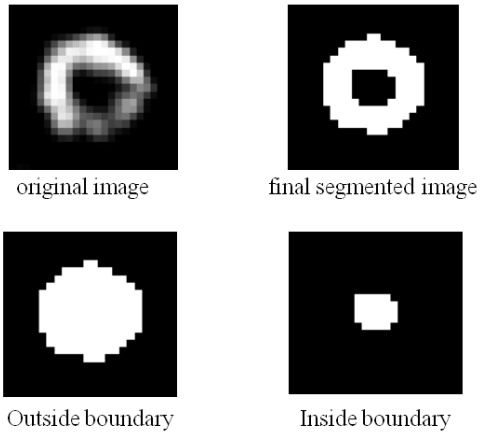


Fig. 8. Proposed FMNN segmentation for inside and outside independently

In our main goal, we used this method to achieve the segmentation of the LV region of cardiac at SPECT images. And also, it could be applied as an initial segmentation step which can create a mask for other techniques. Therefore, the FMNN based technique was employed for segmentation of LV. And so, the results of proposed method used to extract the region of interest at LV in the image. Final regions of LV after applying the FMNN are given in figure 9.

The results validate our prediction about the importance of the inner structure of the data and the correlation effect between neighboring pixels. The nature of our data seems to be non-linear, since non-linear models concluded better than linear ones. In addition, our proposed nonlinear full matrix based neural network learned features of images properly, accurately and quickly.

We indicated the results of this new approach for some well known applications of neural networks using at image processing issue, such as segmentation, classification and recognition. Although our datasets were small and limited, confidently it will illuminate the better evaluation for a enough large number of datasets.

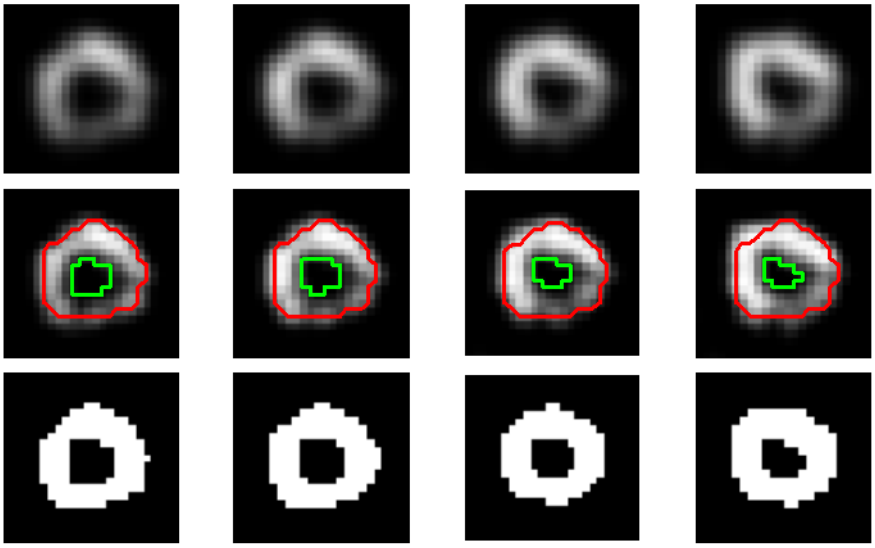


Fig. 9. Several mid-slice cardiac SPECT images from one patient, top: original filtered images, from top to bottom: proposed results (LV areas)

REFERENCES

[1] Hornik, K., M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators", Neural Networks 1989; 2: 359–366.

[2] Matheus, C. J. and Hohensee, W. E., "Learning in artificial neural systems", Computational Intelligence 1987; 3: 283–294.

[3] Chih-Hung Wu, Xian-Ren Lo, and Chen-Sen Ouyang, "A Comparative Study on CT Image Segmentation Using FCM-based Clustering Methods", Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2015, IMECS 2015, 18-20 March, 2015, Hong Kong, pp14-18.

[4] Zuo, W.-M., Wang, K.-Q., Zhang, D., "Assembled Matrix Distance Metric for 2DPCA-Based Face and Palm print Recognition", Proc. Of ICMLS 2005; 4870-4875.

[5] Yang, J., Zhang, D., Frangi, A. F. and Yang, J., "Two-Dimensional PCA:A New Approach to Appearance-Based Face Representation and Recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence 2004; 26(1): 131-137.

[6] Daoqiang Zhanga,b, Zhi-Hua Zhoua, "2D2PCA: Two-directional two-dimensional PCA for efficient face representation and recognition", Neurocomputing 2005; 69(1): 224–231.

[7] Daniusis, P., Vaitkus, "Neural network with matrix inputs", Informatica 2008; 19(4): 477-486.

[8] Pinkus, A., "Approximation theory of the MLP model in neural networks", Acta Numerica 1999; 8: 143–195.

[9] Rodgers J.L., Nicewander J. and W.A., "Thirteen ways to look at the correlation coefficient", American Statistician 1995; 42(1): 59-66.

[10] James M., "Pattern Recognition", John Wiley and Sons, New York 1988; 36-40.

[11] Barnea D.I., Silverman H.F., "A class of algorithms for fast digital image registration", IEEE Transaction on computers 1972; C-21:179:186.

[12] Hall E.H., "computer image processing and recognition", Academic, New York 1979; 480-485.

[13] Press W.H., Flannery B.P., Teukolsky S.A. and Vetterling W.T., "Numerical recipes in Pascal, Cambridge University Press", New York 1989; 532-534.

[14] Beymer, D. and Poggio, T., "Image representations for visual learning", Science 272.5270; 1996: 1905-1909.

[15] Hong, Zi-Quan., "Algebraic feature extraction of image for recognition", Pattern recognition 1991; 24(3): 211-219.

[16] Tian, Y., Tan, T-N., Wang, Y-H and Fang, Y-C., "Do singular values contain adequate information for face recognition", Pattern Recognition 2003; 36(3): 649-655.

[17] Cai, D., X. He and J. Han. "Learning with Tensor Representation", 2006

[18] Zhang, Z., Chow, T. W.S. and Ye, N., "Semisupervised Multimodal Dimentionality Reduction", Computational Intelligence 2013; 29: 70–110.

[19] M. E-Petersen et. al. "Image processing with neural networks—a review", Pattern Recognition 2002; 35: 2279–2301.

[20] Haykin, S., "Neural Networks: A Comprehensive Foundation", 2nd Edition. Prentice Hall 1998.

[21] Kazuhiko Takahashi, Sakie Sakae, and Masafumi Hashimoto, "Remarks on Solving Algebraic Riccati Matrix Equations using a Hopfield Neural Network and Application to Optimal Control Problems," Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2015, 18-20 March, 2015, Hong Kong, pp217-220

[22] Hai-Jun Rong, Guang-Bin Huang, Sundararajan, N., Saratchandran, P., "Online Sequential Fuzzy Extreme Learning Machine for Function Approximation and Classification Problems. Systems", Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions 2009; vol. 39(4): 1067-1072.

[23] Amit Mishra and Zaheeruddiny, "Design of Fuzzy Neural Network for Function Approximation and Classification", IAENG International Journal of Computer Science, vol. 37, no. 4, pp326-340, 2010

[24] Phokharatkul, P. and Kimpan, C., "Handwritten Thai Character Recognition Using Fourier Descriptors and Genetic Neural Networks", Computational Intelligence 2002; 18: 270–293.