I.J. Image, Graphics and Signal Processing, **2014, 5, 28-35**
Published Online April 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2014.05.04

High-speed Image compression based on the Combination of Modified Self-organizing Maps and Back-Propagation Neural Networks

Omid Nali
Department of Electrical Engineering, Saghez branch, Islamic Azad university, Saghez, Iran
E-mail: omid.nali@gmail.com

Abstract **—** This paper presents a high speed image compression based on the combination of modified self-organizing maps and Back-Propagation neural networks. In the self-organizing model number of the neurons are in a flat topology. These neurons in interaction formed self-organizing neural network. The task this neural network is estimated a distribute function. Finally network disperses cells in the input space until estimated probability density of inputs. Distribute of neurons in input space probability is an information compression. So in the proposed method first by Modified Self-Organizing Feature Maps (MSOFM) we achieved distributed function of the input image by a weight vector then in the next stage these information compressed are applied to back-propagation algorithm until image again compressed. The performance of the proposed method has been evaluated using some standard images. The results demonstrate that the proposed method has High-speed over other existing works.

Index Terms **—** Image compression; Back-Propagation algorithm; Modified Self-Organizing Feature Maps.

I. INTRODUCTION

The main advantage of compression is that it reduces the data storage requirements. It also offers an attractive approach to reduce the communication cost in transmitting high volumes of data over long-haul links via higher effective utilization of the available bandwidth in the data links. This significantly aids in reducing the cost of communication due to the data rate reduction. Because of the data rate reduction, data compression also increases the quality of multimedia presentations through limited-bandwidth communication channels [1]. Image compression is one of the important challenges in the information compression on the other hand artificial neural networks have become popular over the last ten years for diverse applications for image compression to machine vision and image processing.
High performance image compression algorithms may be developed and implemented in those neural networks. The neural network image compression algorithm can be summarized as follows:

1- Back-Proagation image compression
2- Hebbian learning based image compression
3- Vector Quantization (VQ) neural networks.
4- Predictive neural networks.

In this paper the proposed method is based on two major stages, the first stage is MSOFM and the second stage is back-propagation algorithm. In the first stage by MSOFM and the proposed weight update method estimated a distribute function of the input image in lower dimensions by neurons (weight matrix) this is first data compression. In second stage weight matrix is applied to back-propagation algorithm until another stage of compression is applied to the input image. Figure 1 shows a block diagram of the proposed method.

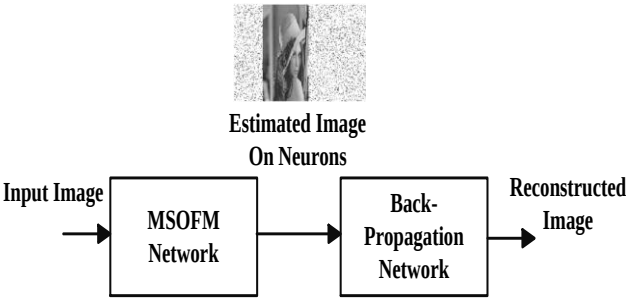

Figure 1. Block diagram of the proposed method.

As seen in figure output of the MSOFM network is estimated image on neurons based on the input image. So number of columns are reduced than the number of columns of the original image, the output of this stage is as input of back-propagation stage. In following more detail of the proposed method is presented.

The remainder of the paper is organized as follows: Section II focuses on related works Section III emphasizes on the proposed method and also comparison. Experimental results of the proposed method are presented in section IV. Finally section V provides the conclusion of this paper.

II. RELATED WORKS

In recent years many methods for image compression systems have been proposed. The Neural Network (NN) efficiency for image compression various works were

Copyright © 2014 MECS

suggested in the past. In [2] expounds the principle of back-propagation neural network with applications to image compression and the neural network models. Then an image compression algorithm based on an improved back-propagation network is developed. In [3] proposed an approach for mapping the pixels by estimating the Cumulative Distribution Function is a simple method of pre-processing any type of image. Due to the uniform frequency of occurrence of gray levels by this optimal contrast stretching, the convergence of the back-propagation neural network is augmented. In [4] presents a novel combining technique for image compression based on the Hierarchical Finite State Vector Quantization (HFSVQ). In [5] presents a compression scheme for digital still images, by using the Kohonen's neural network algorithm, not only for its vector quantization feature, but also for its topological property. In [6] a fuzzy optimal design based on neural networks is presented as a new method of image processing. The combination system adopts a new fuzzy neuron network (FNN) which can appropriately adjust input and output values, and increase robustness, stability and working speed of the network by achieving a high compression ratio. In [7] presented a method that is based on the capabilities of modified self-organizing Kohonen neural network. In [8] approach of mapping the pixels by estimating the Cumulative Distribution Function is presented this method is a simple method of pre-processing any type of image. Due to the uniform frequency of occurrence of gray levels by this optimal contrast stretching, the convergence of the back-propagation neural network is augmented. There will not be any loss of data in the pre-processing and hence the finer details in the image are preserved in the reconstructed image. In [9] used quantum neural networks and image compression using Quantum Gates as the basic unit of quantum computing neuron model, and establish a three layer Quantum back- propagation network model, then the model is used for realizing image compression and reconstruction. Since the initial weights of neural networks were slow convergence, they used a Genetic Algorithm (GA) to optimize the neural network weights, and present a mechanism called clamping to improve the genetic algorithm. Finally, they combined the Genetic algorithm with quantum neural networks to finish image compression. In [10] is an application of the back-propagation network, a new approach for reducing training time by reconstructing representative vectors has also been proposed.

III. PROPOSED METHOD FOR IMAGE COMPRESSION

The proposed method is based on two major parts, the first partition is MSOFM and the second partition is back-propagation algorithm. In the first stage by MSOFM and the proposed weight update method finally estimated a distribute function of the input image in lower dimensions by neurons (weight matrix) this is first information compression. In second stage weight matrix is applied to proposed back-propagation algorithm until another stage of compression is applied to the input image. To create a distribute function of the input image as feature map, in this method we use MSOFM, then the back-propagation has converged quickly. In following proposed method is present.

A. Modified Self-Organizing Feature Maps

In this network the second layer neurons are said to be in competition because each neuron excites itself and inhibits all the other neurons. A transfer function is defined that does the job of a recurrent competitive layer:

a= **compet**(**n**). (1)

It works by finding the index i* of the neuron with the largest net input, and setting its output to 1. All other outputs are set to 0.

$$a_i = \begin{cases} 1, & i = i^* \\ 0, & i \neq i^* \end{cases}, where \quad n_{i^*} \geq n_i, and \quad i^* \leq i, \forall n_i = n_{i^*} \tag{2}$$

A competitive layer is displayed in Figure 2.

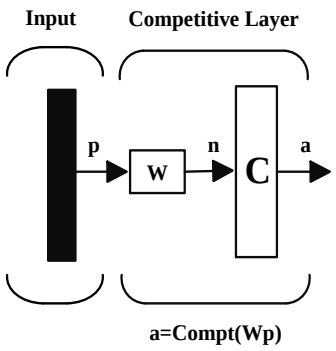

Figure 2. Competitive layer.

As with the Hamming network, the prototype vector is stored in the rows of W. The net input n calculates the distance between the input vector p and each prototype iw. The net input ni of each neuron I is proportional to the angle θ i between P and the prototype vector iw :

$$a = \mathbf{wp} = \begin{bmatrix} {}_1\mathbf{w}^T \\ {}_2\mathbf{w}^T \\ \cdot \\ \cdot \\ \cdot \\ {}_S\mathbf{w}^T \end{bmatrix} \mathbf{p} = \begin{bmatrix} {}_1\mathbf{w}^T\mathbf{p} \\ {}_2\mathbf{w}^T\mathbf{p} \\ \cdot \\ \cdot \\ \cdot \\ {}_S\mathbf{w}^T\mathbf{p} \end{bmatrix} = \begin{bmatrix} L^2\cos\theta_1 \\ L^2\cos\theta_2 \\ \cdot \\ \cdot \\ \\ L^2\cos\theta_S \end{bmatrix} \tag{3}$$

The competitive transfer function assigns one output with value 1 to the neuron whose weight vector points in the direction closest to the input vector. In biological neural networks, neurons are typically arranged in two-dimensional layers, in which they are densely

interconnected through lateral feedback. In biology, a neuron reinforces not only itself, but also those neurons close to it. Typically, the transition from reinforcement to inhibition occurs smoothly as the distance between neurons increases. Biological competitive systems, in addition to having a gradual transition between excitatory and inhibitory regions of the on-center/off-surround connection pattern, also have a weaker form of competition of a single active neuron (winner), biological networks generally have "bubbles" of activity that are centered around the most active neuron. In order to emulate the activity bubbles of biological systems, without having to implement the nonlinear on-center/off-surround feedback connections, Kohonen designed the following simplification. His self-organizing feature map (SOFM) network first determines the winning neuron i* using the same procedure as the competitive layer. Next, the weight vectors for all neurons within a certain neighbourhood of the winning neuron are updated using the Kohonen rule,

$$ {}_i\mathbf{w}(q) = {}_i\mathbf{w}(q-1) + \alpha(\mathbf{p}(q) - {}_i\mathbf{w}(q-1)) \quad i \in N_{i^*}(d) \tag{4}$$

When a vector p is presented, the weights of the winning neuron and its neighbours will move toward p. The result is that, after many presentations, neighbouring neurons will have learned vectors similar to each other.

B. Proposed Modified SOFM

In the proposed architecture the neurons have arranged in a two-dimensional pattern (pixels in the image). Each time a vector (columns of the input image) is presented, the neuron with the closest weight vector will win the competition. The winning neurons and its neighbours (Next column of weight vector) move their weight vectors closer to the input vector. For this work we are using a neighbourhood with the next column of the weight vector. Competitive layer used in the proposed method is shown in Figure 3.

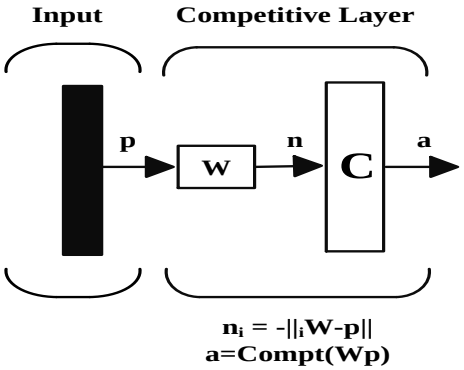

Figure 3. Competitive layer used in the proposed method.

As with the competitive network, each column of neuron in the this layer of the proposed network learns a prototype vector, which allows it to classify a region of the input space. However, instead of computing the proximity of the input and weight vectors by using the inner product, we using of the distance directly. One advantage of calculating the distance directly is that vectors need not be normalized. The net input of the proposed network will be

$$ \mathbf{n}_i = -\| {}_i\mathbf{w} - \mathbf{p} \| \tag{5}$$

Or, in vector form,

$$ \mathbf{n} = - \begin{bmatrix} \| {}_1\mathbf{w} - \mathbf{p} \| \\ \| {}_2\mathbf{w} - \mathbf{p} \| \\ \\ \| {}_{S^1}\mathbf{w} - \mathbf{p} \| \end{bmatrix} \tag{6}$$

The output of the proposed network is

$$ a = \mathbf{Compt}(\mathbf{n}) \tag{7}$$

Therefore the number of columns of the neurons whose weight vector is closest to the input vector (that is column of the input image) will output a is 1, and the other number of columns of the neurons will be 0. Any time we compute the distance directly of one column of the input image with total of columns of weight matrix. Smallest distance directly is winner in competition. Winner column of the weight vector w (q) and next column w (q+1) are updated so the probability of winning next column is increased. In the proposed method possible winning is for winner column and the next column thus allow next column neurons be win the competition so after a few iterations continuously some of the columns of the weight matrix are winner in competition at the end of the iteration input image is estimated and compressed in the number of lower columns than the input image. So input image is compressed in the weight matrix in continuous columns that are lower than the number of columns of the original image. Now in following the proposed network learning is presented. The Kohonen rule is used to improve the network. First, if p is classified correctly, then we move the column weight of the winning neurons and next column weight toward p.

$$ \begin{aligned} \mathbf{w}(q) &= \mathbf{w}(q-1) + \alpha(\mathbf{p} - \mathbf{w}(q-1)) \\ \mathbf{w}(q+1) &= \mathbf{w}(q) + \alpha(\mathbf{p} - \mathbf{w}(q)) \end{aligned} \tag{8}$$

As seen in the Figure 4 input image is estimated by the proposed Modified SOFM in compressed state. The number of winner columns in weight matrix is about 43 to 111, but input image has 200 columns based on the proposed method estimated input image is created in the weight matrix (neurons) by these winner columns i.e. by 68 columns.

Copyright © 2014 MECS

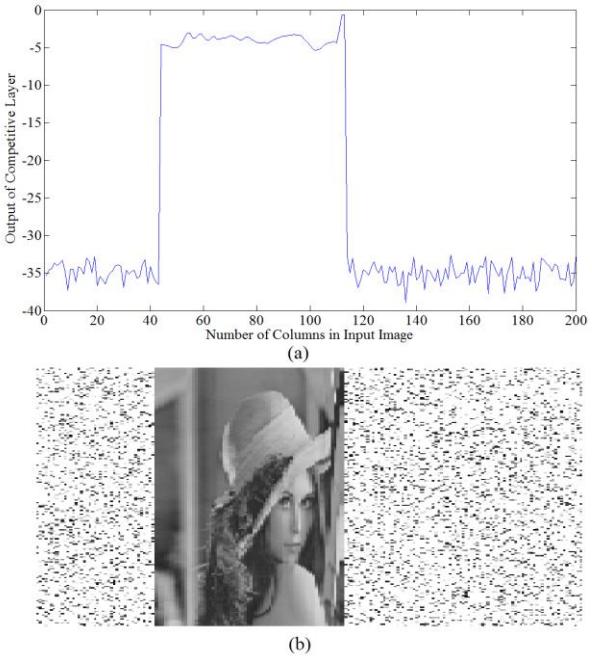

Figure 4. (a) Output of the competitive layer, (b) estimated input image on neurons.

Figure 5 shown a number of the winner columns in final iteration the number of winner columns in weight matrix is about 43 to 111. As seen in the figure order of winner columns are incrementally this is based on the proposed learning rule as explained in this section, in competitive layer competition is between columns of neurons. So closest columns of neurons (weight matrix) to column of the input image that is as input vector is winner thus winner column and next column of weight matrix are updated.

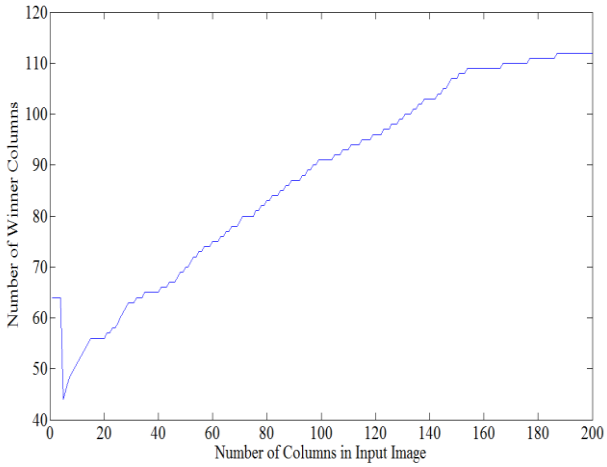

Figure 5. Number of the winner columns in final iteration.

As seen in Figure 5 by using the proposed learning rule winning probability of the next column is increased and winner columns are neighbourhood (incrementally) in the end iteration input image is formed by winner columns that a number of these columns are lower than the number of input images. Fig.6 shown histogram of the number of the winner columns in the proposed MSOM in this example number of the winner columns is between 43 to 111.

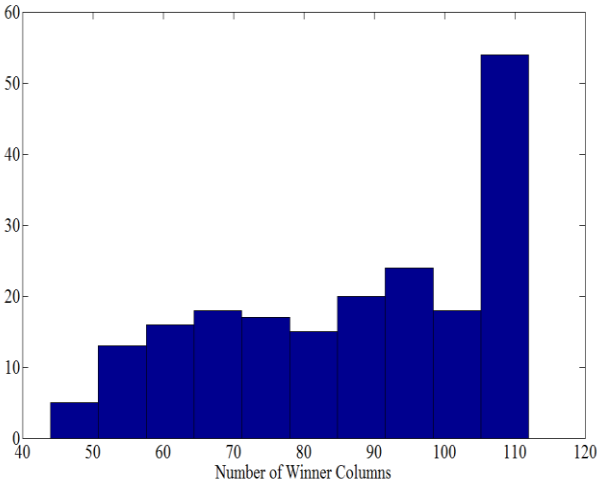

Figure 6. (a) Histogram of the number of the winner columns.

As seen in the figure approximately probability of winning of any interval of columns are equal.

C. Second stage of the proposed method based on Back-propagation

The demonstration of the limitations of single-layer neural networks were a significant factor in the decline of interest in neural networks in the 1970s The discovery and widespread dissemination of an effective general method of training a multilayer neural network. In this part of the paper, we shall discuss this training method, known as back-propagation (of errors) or the generalized delta rule. It is simply a gradient descent method to minimize the total squared error of the output computed from the net. The very general nature of the back-propagation training method means that a back-propagation net can be used to solve problems in many areas. The training of a network by back-propagation involves three stages: the feed forward of the input training pattern, the back-propagation of the associated error, and the adjustment of the weights. After training, application of the net involves only the computations of the feed forward phase. Even if training is slow, a trained net can produce its output very rapidly. Numerous variations of back-propagation have been developed to improve the speed of the training process. More than one hidden layer may be beneficial for some applications, but one hidden layer is sufficient [11]. Figure 7 shows back-propagation network.

Copyright © 2014 MECS

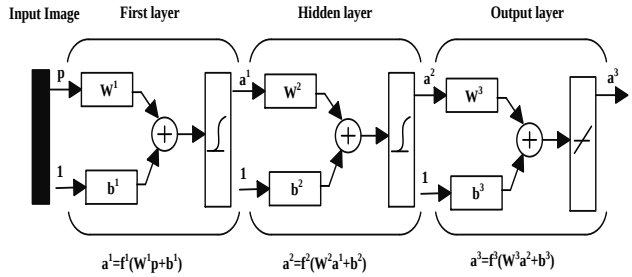

Figure 7. Back-propagation network.

The back-propagation is directly applied to image compression. The compression is achieved by estimating the value of X, the neurons in the hidden layer less than that of neurons in both the input and output layers. The "N" is the number of neurons in the input or output layer and the "X" is the number of the hidden layer. The input image is split up into a number of columns, each column has "N" pixels, which is equal to the number of input neurons.

D. Algorithm of the back-propagation network

An activation function for a back-propagation net should have several important characteristics: It should be continuous, differentiable, and monotonically nondecreasing. Furthermore, for computational efficiency, it is desirable that its derivative be easy to compute. For the most commonly used activation functions, the value of the derivative (at a particular value of the independent variable) can be expressed in terms of the value of the function. Usually, the function is expected to saturate, i.e., approach finite maximum and minimum values asymptotically. The performance index of this algorithm is

$$F(x) = E[e^T e] = E[(t-a)^T (t-a)] \tag{9}$$

Also approximate on the performance index back propagation is

$$\hat{F}(x) = e^T(k)e(k) = (t(k)-a(k))^T (t(k)-a(k)) \tag{10}$$

The sensitivity of $\hat{F}$ to change in the ^{i}th element of the net input at layer m.

$$S^m \equiv \frac{\partial \hat{F}}{\partial n^m} = \begin{bmatrix} \frac{\partial \hat{F}}{\partial n_1^{\,m}} \\ \frac{\partial \hat{F}}{\partial n_2^{\,m}} \\ . \\ . \\ . \\ \frac{\partial \hat{F}}{\partial n_{S^m}^{\,m}} \end{bmatrix} \tag{11}$$

In forward propagation

$$a^0 = \mathbf{p}$$

That **p** is input vector

$$a^{m+1} = f^{m+1}(\mathbf{w}^{m+1}a^m + \mathbf{b}^{m+1}) \qquad for \;\; m = 0,1,...,M-1$$

$$a = a^M$$

In backward propagation

$$S^{\mathbf{M}} = -2\dot{F}^{\mathbf{M}}(n^{\mathbf{M}})(t-a) \tag{12}$$

$$S^m = \dot{F}^m(n^m)(\mathbf{W}^{m+1})^T S^{m+1} \qquad for \;\; m = M-1,..,2,1$$

Where

$$\dot{F}^m(n^m) = \begin{bmatrix} \dot{f}^m(n_1^{\,m}) & 0 & ... & 0 \\ 0 & \dot{f}^m(n_2^{\,m}) & ... & 0 \\ . & . & & . \\ . & . & ... & . \\ . & . & & . \\ 0 & 0 & ... & \dot{f}^m(n_{S^m}^{\,m}) \end{bmatrix}$$

$$\dot{f}^{\,m}(n_j^{\,m}) = \frac{\partial f^{\,m}(n_j^{\,m})}{\partial n_j^{\,m}}$$

In final weight update (approximate steepest descent)

$$\mathbf{w}^m(k+1) = \mathbf{w}^m(k) - \alpha S^m (a^{m-1})^T \tag{13}$$

$$\mathbf{b}^m(k+1) = \mathbf{b}^m(k) - \alpha S^m \tag{14}$$

In proposing a method implemented algorithm is:

$$a^0 = \mathbf{p}$$

The output of the first layer is

$$a^1 = f^1(\mathbf{w}^1 a^0 + \mathbf{b}^1) =$$
$$\log sig(\left[\mathbf{w}^1\right]_{200*200}\left[a^0\right]_{200*1} + \left[\mathbf{b}^1\right]_{200*1}) = \left[a^1\right]_{200*1}$$

The output of a hidden layer is

$$a^2 = f^2(\mathbf{w}^2 a^1 + \mathbf{b}^2) =$$
$$\log sig(\left[\mathbf{w}^2\right]_{x*200}\left[a^1\right]_{200*1} + \left[\mathbf{b}^2\right]_{x*1}) = \left[a^2\right]_{x*1}$$

That x is the number of neurons in the hidden layer. The third output layer is,

$$a^3 = f^3(\mathbf{w}^3 a^2 + \mathbf{b}^3) =$$
$$purelin(\left[\mathbf{w}^3\right]_{200*x}\left[a^2\right]_{x*1} + \left[\mathbf{b}^3\right]_{200*1}) = \left[a^3\right]_{200*1}$$

Copyright © 2014 MECS

The error would then be,

$$e = t - a^3 = [t]_{200*1} - [a^3]_{200*1} = [e]_{200*1}$$

The next stage of the algorithm is to back propagation the sensitivities. Before we begin the back propagation, we will need the derivatives of the transfer functions. For the first layer,

$$\dot{f}^1(n) = \frac{d}{dn}\left(\frac{1}{1+e^{-n}}\right) = \frac{e^{-n}}{1+e^{-n}} = \left(1-\frac{1}{1+e^{-n}}\right)\left(\frac{1}{1+e^{-n}}\right) = (1-a^1)(a^1)$$

For the hidden layer we have

$$\dot{f}^2(n) = (1-a^2)(a^2)$$

For the third layer we have

$$\dot{f}^3(n) = \frac{d}{dn}(n) = 1$$

We can now perform the back-propagation. The starting point is found in the third layer, using Eq (12):

$$S^3 = -2\dot{F}^3(n^3)(t-a) = -2\left[\dot{f}^3(n^3)\right](t-a)$$

$$S^2 = \dot{F}^2(n^2)(\mathbf{w}^3)^T S^3 = \begin{bmatrix} (1-a_1^2)(a_1^2) & 0 & \dots & 0 \\ 0 & (1-a_2^2)(a_2^2) & \dots & 0 \\ . & . & & . \\ . & . & \dots & . \\ . & . & & . \\ 0 & 0 & \dots & (1-a_{200}^2)(a_{200}^2) \end{bmatrix}_{200*200} \left[S^3\right]_{200*1}$$

And for S^1,

$$S^1 = \dot{F}^1(n^1)(\mathbf{w}^2)^T S^2 = \begin{bmatrix} (1-a_1^1)(a_1^1) & 0 & \dots & 0 \\ 0 & (1-a_2^1)(a_2^1) & \dots & 0 \\ . & . & & . \\ . & . & \dots & . \\ . & . & & . \\ 0 & 0 & \dots & (1-a_{200}^1)(a_{200}^1) \end{bmatrix}_{200*200} \left[S^2\right]_{200*1}$$

The final stage of the algorithm is to update the weights. For simplicity, we will use a learning rate α=0. 66. From (13) and (14) we have:

$$\mathbf{w}^3(1) = \mathbf{w}^3(0) - \alpha S^3 (a^2)^T$$

$$\mathbf{w}^2(1) = \mathbf{w}^2(0) - \alpha S^2 (a^1)^T$$

$$\mathbf{w}^1(1) = \mathbf{w}^1(0) - \alpha S^1 (a^0)^T$$

And for bias

$$\mathbf{b}^3(1) = \mathbf{b}^3(0) - \alpha S^3$$

$$\mathbf{b}^2(1) = \mathbf{b}^2(0) - \alpha S^2$$

$$\mathbf{b}^1(1) = \mathbf{b}^1(0) - \alpha S^1$$

We continue to iterate until the difference between the network response and the target function reaches some acceptable level. After training of this network reconstructed image is achieved.

IV. COMPARISON AND EXPERIMENTAL RESULTS

The quality of an image is measured using the parameters like Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR). The MSE and the PSNR are the two error metrics used to compare image compression quality. These two parameters define the quality of an image reconstructed at the output layer of the neural network. The MSE between the reconstructed image and original image should be as small as possible so that the quality of reconstructed image should be near to the original image. The MSE metric is most widely used for in simple to calculate, MSE is computed by averaging the squared intensity difference of reconstructed image RI and the original image, OI.

$$MSE = \frac{1}{NM}\sum_{i=1}^{M}\sum_{j=1}^{N}\left[O_I(i,j) - R_I(i,j)\right]^2 \quad (15)$$

That N, M are the dimensions of the image.

Then based on MSE the PSNR is calculated, so the PSNR is defined as follows:

$$PSNR = 10\log_{10}\left[255^2 / MSE\right](db) \quad (16)$$

PSNR represents a measure of the peak signal to noise ratio the higher the PSNR, the better the quality of the compressed or reconstructed image. The computer system used for the investigation had the specifications of 2.20GHz CPU, 2.00GB RAM and the MATLAB version 7.4 was applied for the study. The proposed algorithm is simple and high speed. The performance of the network has been evaluated using some standard images. Total images in this work are 200*200 pixel In the follow achieved result of different image are demonstrated. Figure 8 shows the Lena Image original image and two reconstructed images with α=0.66 and α=0.1 that α is learning rate in back-propagation algorithm. Table I shows achieved result of Lena image.

Copyright © 2014 MECS

Figure 8. Lena Image and two reconstructed images with α=0. 66 and α=0.1 by the proposed method.

TABLE I: RESULTS OF LENA IMAGE.

Image 200*200	PSNR (db)	MSE	Time (Sec)	Bits per pixel
Lena	67.7513	26.6388	18.2768	2

Figure 9 shows the Barbara Image original image and two reconstructed images with α=0.66 and α=0.1 that α is learning rate in back-propagation algorithm.

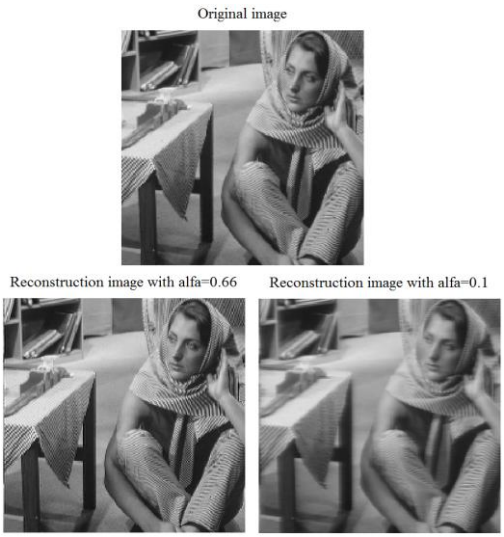

Figure 9. Barbara Image and two reconstructed images with α=0. 66 and α=0.1 by the proposed method.

Table II shows achieved result of Barbara image.

TABLE II: RESULTS OF BARBARA IMAGE.

Image 200*200	PSNR(db)	MSE	Time (Sec)	Bits per pixel
Barbara	68.1886	25.3309	15.9491	2

Figure 10 shows Peppers image original image and two reconstructed images with α=0.66 and α=0.1 that α is learning rate in back-propagation algorithm. Table III shows achieved result of the peppers image.

TABLE III: RESULTS OF PEPPERS IMAGE.

Image 200*200	PSNR(db)	MSE	Time (Sec)	Bits per pixel
Peppers	69.2541	22.4065	18.3992	2

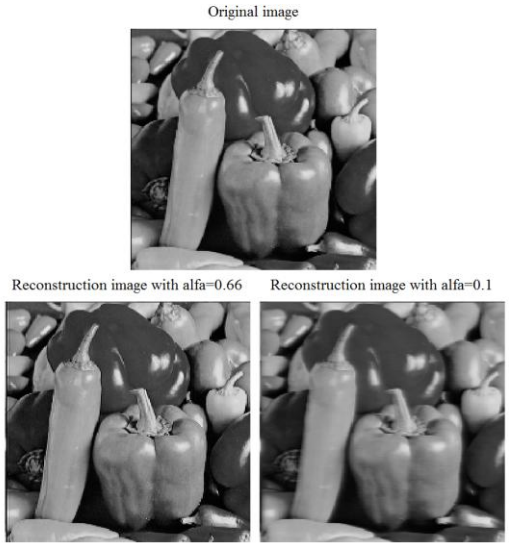

Figure 10. Peppers image and two reconstructed images with α=0. 66 and α=0.1 by the proposed method.

Figure 11 shows the Baboon image original image and two reconstructed images with α=0.66 and α=0.1 that α is learning rate in back-propagation algorithm. Table IV shows achieved result of the Baboon image.

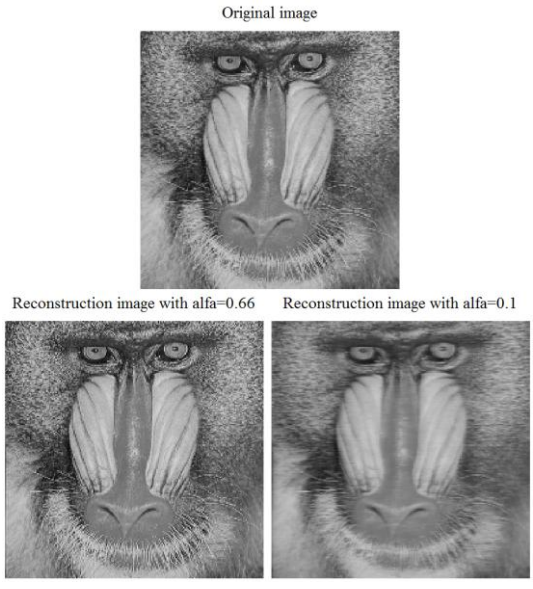

Figure 11. Baboon image and two reconstructed images with α=0. 66 and α=0.1 by the proposed method.

TABLE IV: RESULTS OF BABOON IMAGE.

Image 200*200	PSNR(db)	MSE	Time (Sec)	Bits per pixel
Baboon	67.3420	27.9241	15.4096	2

Table V shows the results of different methods.

TABLE V: RESULTS OF DIFFERENT METHODS.

Method	Image	Size	PSNR(db)	Time(Sec)	MSE
[8]	Lena	256*256	28.91	182	---
[8]	Pepper	256*256	29.04	188	---
[12]	Lena	256*256	26.33	---	---
[5]	Lena	256*256	24.7	---	---
[9]	Lena	256*256	45.13	---	---
[10]	Lena	256*256	38.6035	3270.101	20.127
[10]	Pepper	256*256	32.2945	3983.112	24.29

As seen in the above tables the proposed method has High-speed over other existing works in the image compression.

V. CONCLUSION

In this paper we construct a high-speed image compression based on the combination of modified self-organizing map and back-propagation neural networks. The proposed method is based on two major parts, the first partition is the MSOFM and the second partition is back-propagation algorithm. In the first stage by MSOFM and the proposed weight update method estimated a distribute function of the input image in lower dimensions by neurons (weight matrix) this is first information compression and back-propagation is second stage of the image compression. The results show design works properly and compare to previous implementation it is able to work at higher speeds. Speed and PSNR are two features of the proposed method over other existing works.

REFERENCES

[1] Tinku Acharya, Ping-Sing Tsai, "JPEG2000 Standard for Image Compression Concepts, Algorithms and VLSI Architectures", John Wiley & Sons, INC., Publication, 2004.

[2] Tang Xianghong Liu Yang "An Image Compressing Algorithm Based on Classified Blocks with BP Neural Networks" International Conference on Computer Science and Software Engineering, Date: 12-14 Dec. 2008 Volume: 4, pp. 819-822.

[3] S. Anna Durai, and E. Anna Saro "Image Compression with Back-Propagation Neural Network using Cumulative Distribution Function" World Academy of Science, Engineering and Technology 2006, pp. 185-189.

[4] Karlik, Bekir "Medical Image Compression by using Vector Quantization Neural Network (VQNN) " Neural Network World, January 1, 2006.

[5] Christophe Amerijckx, Michel Verleysen, "Image Compression by Self-Organized Kohonen Map", IEEE Transactions on Neural Networks, Vol. 9, No. 3, May 1998, pp. 503-507.

[6] Abdul Khader Jilani, Abdul Sattar, "A Fuzzy Neural Networks based EZW Image Compression System", International Journal of Computer Applications (0975 – 8887), Volume 2 – No.9, June 2010.

[7] Tomasz Praczyk, "Better Kohonen Neural Network in Radar Images Compression", Computational Methods in Science and Technology 12(2), 2006, pp. 157-164.

[8] S. Anna Durai, and E. Anna Saro, "Image Compression with Back-Propagation Neural Network using Cumulative Distribution Function", International Journal of Engineering and Applied Sciences 3:4 2007.

[9] LI Huifang, LI Mo, "A New Method of Image Compression Based on Quantum Neural Network", 2010 International Conference of Information Science and Management Engineering, pp. 567-570.

[10] K.Siva Nagi Reddy, B.R.Vikram, L. Koteswara Rao, B.Sudheer Reddy, "Image Compression and Reconstruction Using a New Approach by Artificial Neural Network", International Journal of Image Processing (IJIP), Volume (6) : Issue (2), pp. 68-86, 2012.

[11] Laurene Fausett, "Fundamentals of Neural Networks Architecture, algorithms, and applications", Prentice Hall, Inc, 1993.

[12] Venkata Rama Prasad Vaddella, Kurupati Rama, "Artificial Neural Networks For Compression Of Digital Images: A Review", International Journal of Reviews in Computing, 2010, pp.75-82.

Omid Nali, was born in 1986 in Sagez, Iran. He received his B.SC. Degree in Electrical Engineering from the Lorestan University, Iran, in 2009 and he received his M.SC. In the Azad Islamic university of Tabriz, Iran in 2013. His research interests include digital image processing, computer vision, and mechateronic.