Modeling of neural image compression using GA and BP: a comparative approach

G.G Rajput, Vrinda Shivashetty
Department of Computer Science
Gulbarga University
Gulbarga,India

Manoj Kumar singh
Manuro Tech Research,
Bangalore,India

***Abstract*— It is well known that the classic image compression techniques such as JPEG and MPEG have serious limitations at high compression rate; the decompressed image gets really fuzzy or indistinguishable. To overcome problems associated with conventional methods, artificial neural networks based method can be used. Genetic algorithm is a very powerful method for solving real life problems and this has been proven by applying to number of different applications. There is lots of interest to involve the GA with ANN for various reasons at various levels. Trapping in the local minima is one of the well-known problems of gradient decent based learning in ANN. The problem can be addressed using GA algorithm. But no work has been done to evaluate the performance of both learning methods from the image compression point of view. In this paper, we investigate the performance of ANN with GA in the application of image compression for obtaining optimal set of weights. Direct method of compression has been applied with neural network to get the additive advantage for security of compressed data. The experiments reveal that the standard BP with proper parameters provide good generalize capability for compression and is much faster compared to earlier work in the literature, based on cumulative distribution function. Further, the results obtained shows that general concept about GA, it performs better over gradient decent based learning, is not applicable for image compression.**

Keywords- Image compression; genetic algorithm; neural network; back propagation.

I. INTRODUCTION

Artificial neural network (ANN) technique has been used successfully for image compression with various ways [10, 11,12,13,21]. A detail survey of about how ANN can be applied for compression purpose is reported in [1,14,15,16,17]. Broadly, two different categories for improving the compression methods and performance have been suggested. Firstly, develop the existence method of compression by use of ANN technology so that improvement in the design of existing method can be achieved. Secondly, apply neural network to develop the compression scheme itself, so that new methods can be developed and further research and possibilities can be explored for future. Statistical approaches are applied in integration with neural network for enhancement of compression performance. In [2,18], principal component analysis (PCA) is applied for this purpose. PCA is one of the famous statistical methods which eliminates the correlation between different data components and consequently decrease the size of data. In classical method, covariance matrix of input data is used for extracting singular values and vectors. Neural networks are used for extracting principal value components in order to compress image data. First, different principal component analysis neural networks is presented and then a nonlinear PCA neural network is used which provides better results as shown in simulation results. Speed is one of the fundamental issues that always appear in the application of image compression. In [4,19,20,22], the problems associated with neural network for compression is discussed. Authors have given the concept of reduction of original feature space, which allows us to eliminate the image redundancy and accordingly leads to their compression. Two variants of neural network they have suggested: two layers neural network with self learning algorithm based on the weighted information criterion and auto–associative four layers feed forward network. In [5,23,24,25], a constructive One-Hidden-Layer feed forward Neural Network (OHL-FNN) architecture has been applied for image compression. The BPNN has taken as the simplest architecture of ANN that has been developed for image compression but its drawback is very slow convergence.

II. FEED FORWARD ARTIFICIAL NEURAL NETWORKS

In feed forward architecture having multilayer perceptrons, the basic computational unit, often referred to as a "neuron," consists of a set of "synaptic" weights, one for every input, plus a bias weight, a summer, and a nonlinear function referred to as the activation function . Each unit computes the weighted sum of the inputs plus the bias weight and passes this sum through the activation function to calculate the output value as $y_j = f(\sum_i w_{ji}x_i + \theta_i)$,where x_i is the *i*th input value for the neuron and w_{ji} is the corresponding synaptic weight. The activation function $f(\bullet)$ maps the potentially infinite range of the weighted sum to a limited, finite range. A common activation function is a sigmoid defined as $f(v) = \frac{1}{1+e^{-v}}$

In a multilayer configuration, the outputs of the units in one layer form the inputs to the next layer. The inputs to the first layer are considered as the network inputs, and outputs of the last layer are the network outputs. The weights of the network are usually computed by training the network.

A. Evolution of weights in neural network using GA

In recent times much research has been undertaken in the combination of two important and distinct areas: genetic algorithm and neural networks. Genetic algorithms attempt to

apply evolutionary concept to the field of problem solving, notably function optimization and have proven to be valuable in searching large, complex problem spaces. Neural networks are highly simplified models of the working of brain. These consist of a combination of neurons and synaptic connections, which are capable of passing data through multiple layers. The end result is a system which is capable of pattern and classification. In the past, algorithm such as back propagation have been developed which refine one of the principle components of the neural networks: connection weights. The system has worked well, but is prone to becoming trapped in local optima and is incapable of optimization where problems lie in a multi-model or non-differentiable problem space. Genetic algorithms and neural networks can be combined such that populations of neural networks compete with each other in a Darwinian 'survival of the fittest' setting. Networks which are deemed to fit are combined and passed onto the next generation producing an increasingly fit population, so that after a number of iterations for an optimized neural network can be obtained without resorting to a design by hand method. The primary motivation for using evolutionary technique to establish the weighting values rather than traditional gradient decent techniques such as back propagation lies in the inherent problems associated with gradient descent approaches.

The evolution of neural networks can be classified according to the goals behind such evolution. Some schemes have proposed by introducing the evolution of weights with the fixed architecture. Other level of evolution where improvement can be expected is in the architecture is the transfer function [yao].

B. *Chromosome, Crossover & mutation operation to generate the offspring*

Initially, a population of chromosomes created contains a uniformly distributed random number. Chromosomes directly representing the weights of neural network are shown in fig.2. Hence, there is no need of any encoding mechanism in result. Crossover here can be defined as node crossover. From picked up two parents for generating off springs, any one active node from the set of hidden and output layer, pick up randomly with equal probability. This node consider as a node of crossover. Values of all incoming weights for that particular node are exchanged with available other parent. Mutation can also be considered as node mutation, where in an offspring, all incoming weights for a randomly picked up active node added with Gaussian distributed random numbers. These two processes are shown in fig.3and fig.4, respectively.

C. *Algorithm for weights evolution by GA in ANN*

The following steps are performed for determining the optimal value of weights.

(i)A population of μ parent solution X_i, i=1,…. μ, is initialized over a region M ϵ R^n.
(ii)Two parents are selected randomly with uniform distribution from population of μ parents, and two offspring will created by crossover operator as shown in Fig.2.
(iii)Mutation on newly generated offspring will be applied as shown in Fig .3.
(iv)Repeat step (ii) until population of offspring μ_o equal to μ, otherwise move to step (v).
(v)Each parent solution X_i, i=1,…. μ and offspring X_o, o=1,…. μ,is scored in light of the objective function $f(X)$.
(vi)A mixture population X_m, m = 1,…,2 μ contains both parent population and offspring population created. This mixture population randomly shuffled so that parents and offspring could mix up properly.
(vii)Each solution from X_m, m = 1,…,2 μ is evaluated against 10% of μ other randomly chosen solutions from the mix population X_m .For each comparison a 'win' is assigned if the solution's score is less than or equal to that of its opponent.
(viii)The μ solutions with the greatest number of wins are retained to be parents of the next generation.
(ix)If the difference in the best chromosome for N number of continuous generation are less than the defined threshold value k, terminate the process and the last generation best chromosome is the optimal weights, otherwise proceed to step (ii).

D. *Weight optimization with back propagation algorithm.*

Back propagation algorithm is a supervised learning algorithm which performs a gradient descent on a squared error energy surface to arrive at a minimum. The key to the use of this method on a multilayer perceptrons is the calculation of error values for the hidden units by propagating the error backwards through the network. The local gradient for the jth unit, in the output layer is calculated as (assuming a logistic function for the sigmoid nonlinearity)

$$\delta_j = y_j(1 - y_j)(d_j - y_j) \qquad (1)$$

where yj is the output of unit j and dj is the desired response for the unit. For a hidden layer, the local gradient for neuron j is calculated as

$$\delta_j = y_j(1 - y_j)\sum_k \delta_k w_{jk} \qquad (2)$$

where the summation k is taken over all the neurons in the next layer to which the neuron j serves as input. Once the local gradients are calculated, each weight wji is then modified according to the delta rule

$$w_{ji}(t+1) = w_{ji}(t) + \eta\delta_j(t)y_i(t) \qquad (3)$$

Where η a learning-rate parameter and t is time. Frequently modification is used that incorporates a momentum term that helps to accelerate the learning process

$$w_{ji}(t+1) = w_{ji}(t) + \eta\delta_j(t)y_i(t) + \alpha[w_{ji}(t) - w_{ji}(t-1)] \qquad (4)$$

Where α is a momentum term lying in the range $0 < \alpha < 1$.

III. IMAGE COMPRESSION STRUCTURE USING PERCEPTRONS NEURAL NETWORK

The structure to compress images is the three layer perceptrons, depicted in Fig.11. In order to use structure, the input image is divided into blocks with pixels equal to the same

number of neurons in input layer, say N. It means that these blocks should be of order $\sqrt{N} \times \sqrt{N}$ (in this paper this size is 8*8) so that they can be expressed in N dimensional vector and fed into the input layer. The hidden layer in this structure is the compressed image which maps N pixels to K (K<N) and finally the reconstructed image from compressed one is derived with the same number of pixels/neurons as the input. In this structure the input weights to the hidden layer are a transform matrix which scales the input vector of N-dimensional into a narrow channel of k-dimensional. Similarly, the weights of hidden to output layer are a transform matrix which scales the narrow vector of K-dimensional into a channel of N-dimensional. The input gray-level pixel values are normalized to the range [0, 1]. The reason for using normalized pixel values is due to the fact that neural networks can operate more effectively when both their inputs and outputs are limited to a range of [0, 1]. Learning is applied to train the architecture. All patterns in the input blocks of training set are also fed to output layer as the target. Once training is completed with proper performance, final weights are having the capability to map the input value of pixels into approximate same value at the output. Compression process is defined by taking the half of the trained architecture which has been utilize at the time of training ,i.e. input layer along with the hidden layer as shown in Fig.12. Remaining half of the trained architecture i.e. hidden layer along with output layer is utilized to setup the decompression, as shown in Fig.13.

IV. Performance Parameters

Evaluation criteria used for comparison in this paper, is compression ratio (CR) and the peak signal-to-noise ratio (PSNR). For an Image with R rows and C columns, PSNR is defined as follows:

$$PSNR = 10\ log_{10}\left(\frac{255^2}{\frac{1}{RC}\sum_{I=1}^{R}\sum_{J=1}^{C}(X_{IJ}-\bar{X}_{IJ})^2}\right)$$

Compression ratio (CR) which is a criterion in compression problems is defined as the number of bits in original image to number of bits in the compressed image. This criterion in the sense of using neural net structure is defined as follow:

$$CR = \frac{N \cdot B_I}{K \cdot B_H}$$

In this equation N and K are the neurons/pixels available in the input and hidden layer respectively and B_I and B_H are the number of bits needed to encode outputs of input and hidden layer. If the number of bits needed to encode the input layer and the number of bits needed to encode the hidden layer be the same, the compression ratio will be the number of neurons in the input layer to hidden layer. As an example for the gray level images which are 8 bits long if we encode the compressed image with the same number of bits in a block of 8×8 and the network of with 16 neurons at the hidden layer, the compression ratio will be 4:1. And for the same network with floating point used to encode the hidden layer, the compression ratio will be 1:1 which means no compression.

To verify the developed design for evolving the weights of neural network two different experiments are considered as explained in the following section. This will give the confidence to apply the developed method for image compression.

V. Pattern Recognition and the XOR Problem

The pattern recognition problem consists of designing algorithms that automatically classify feature vectors associated with specific patterns as belonging to one of a finite number of classes. A benchmark problem in the design of pattern recognition systems is the Boolean exclusive OR (XOR) problem. The standard XOR problem is shown in figure below. Here, the diagonally opposite corner-pairs of the unit square form two classes, A and B (or NOT A). From the figure, it is clear that it is not possible to draw a single straight line which will separate the two classes. This observation is crucial in explaining the inability of a single-layer perceptrons to solve this problem .This problem can be solved using multi-layer perceptrons (MLPs), or by using more elaborate single-layer ANNs.

A. *Performance of GA with ANN for weight evolution over XOR problem.*

A feed forward architecture of 2-2-1 designed and weights has evolved by above defined method of GA. Population size taken as 20 and condition of terminating criteria is, if the best chromosome error in 50 continuous generation is less than 0.00001.

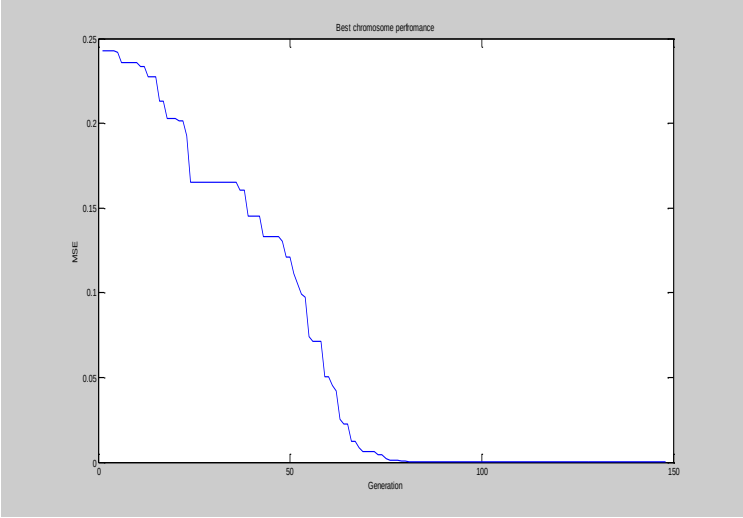


Figure.1. Error performance with generation for best chromosome.

Expected output	0	1	1	0
Output GA+ANN	0.000	1.000	1.000	0.000
Total generation	148			
Mean square error at last generation	6.613782079731114e-009			

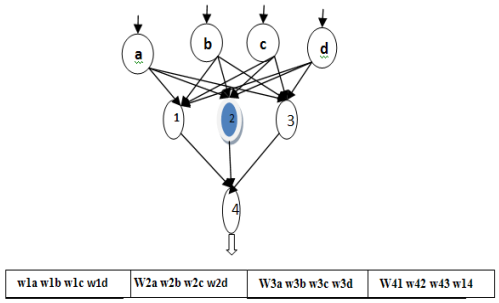


Figure 2. Chromosome development

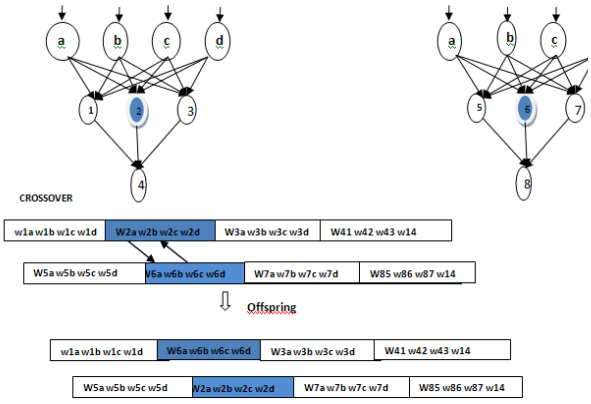


Figure 3. Crossover operation

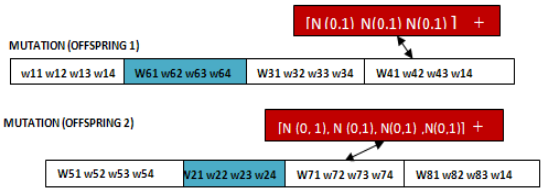


Figure 4. Mutation operation

B. *Performance of GA with ANN for weight evolution over one block of image*

To see the performance of GA based weights optimization design over real image, a block of image containing 8*8 pixels has been taken. Population of 50 chromosomes is taken with 16 hidden nodes in ANN architecture. To see the proper convergence a large value of generation 2000 has defined for termination. Performance of experiment has shown in table (2).Mapping result is shown below. Convergence of learning for best chromosome is shown in Fig. 5.

(a) Input block:

162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
164	164	158	155	161	159	159	160
160	160	163	158	160	162	159	156
159	159	155	157	158	159	156	157

(b) mapped output:

161	162	162	161	162	157	164	162
162	163	162	162	161	157	164	162
164	162	163	160	163	158	162	160
163	161	163	163	161	157	163	161
163	162	163	161	161	158	162	160
164	163	160	155	161	159	160	160
161	160	163	157	161	162	160	155
160	158	155	158	158	159	157	159

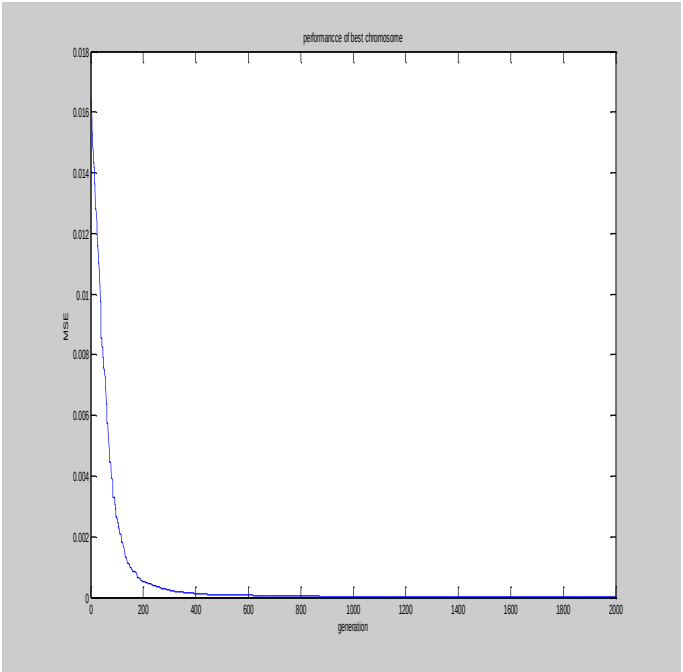


Figure 5. Error performance

With the above two different experiments, performance given by GA based weights optimization seems very impressive and results show the design of GA for neural learning is working better. This has given enough confidence to deploy the GA for image compression.

VI. IMAGE COMPRESSION WITH GA AND ANN

A population of 50 chromosomes is applied for evolving the weights up to 200 generations. Compression ratio for this experiment defined as 4:1. Performance plot for compression is shown in Fig.6 and in Fig.7 .Decompression result of Lena image is shown in Fig.8. Table (3) shows the parameter values. From the result it is very clear that the process is taking very long time for completing the cycle of generation. Even convergence is not proper and the result of compression performance is very poor and cannot be consider for practical purpose.

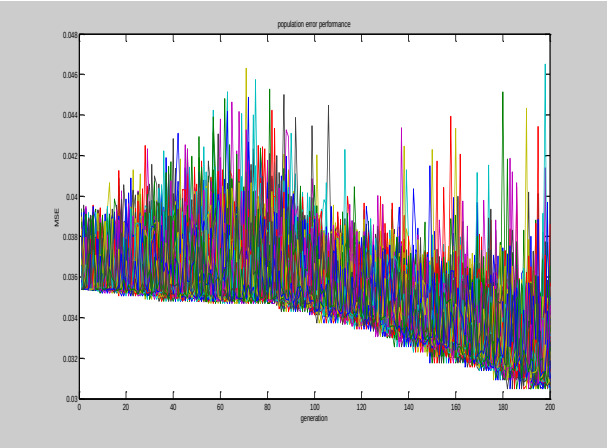


Figure 6. Population Error plot

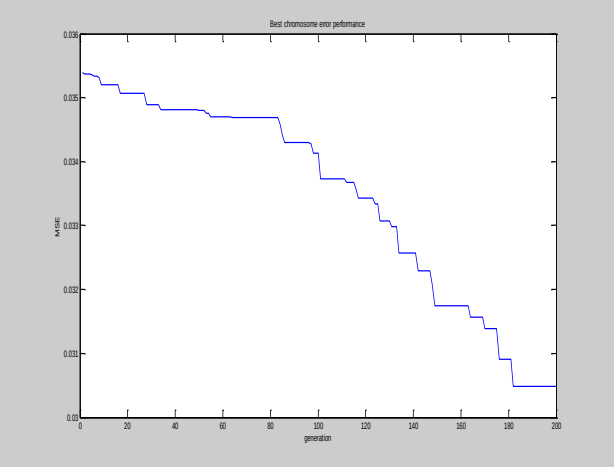


Figure 7. Best chromosome Error plot

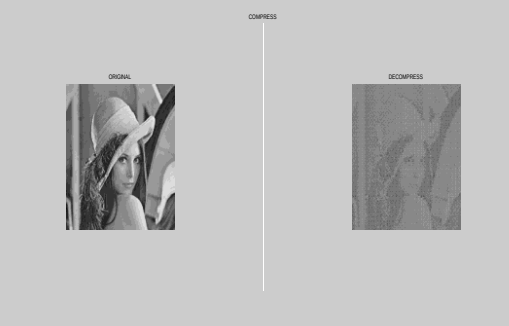
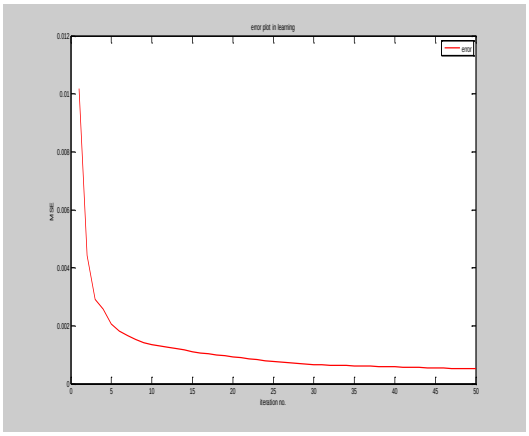


Figure8: Decompression by GA with ANN

VII. IMAGE COMPRESSION USING ANN AND STANDARD BACK PROPAGATION ALGORITHM

Standard back propagation is one of the most widely used learning algorithms for finding the optimal set of weights in learning. A single image "Lena" is taken as training data. The error curve of learning is shown in Fig.9 for below defined set of parameters. Further, different images are tested to generalize the capability of compression. The processes repeated for two different compression ratio by changing the number of hidden nodes in neural architecture. The performance observed during the time of training and testing is shown in table 4, table 5 for compression ratio 4:1 and in table 6, table 7 or 8:1, respectively. Table 8 given the comparison with [9]



Parameter setting for back propagation learning: Initial random weights value taken from uniform distribution in range of [-0.0005 +0.0005]. Learning rate: 0.1 ; Momentum constant: 0.5;Bias applied to hidden and output layer nodes with fixed input as (+1). Allowed number of iteration : 50

Figure 9. Error plot in back propagation

Compression ratio: 4:1

Figure 10. Performance by gradient decent

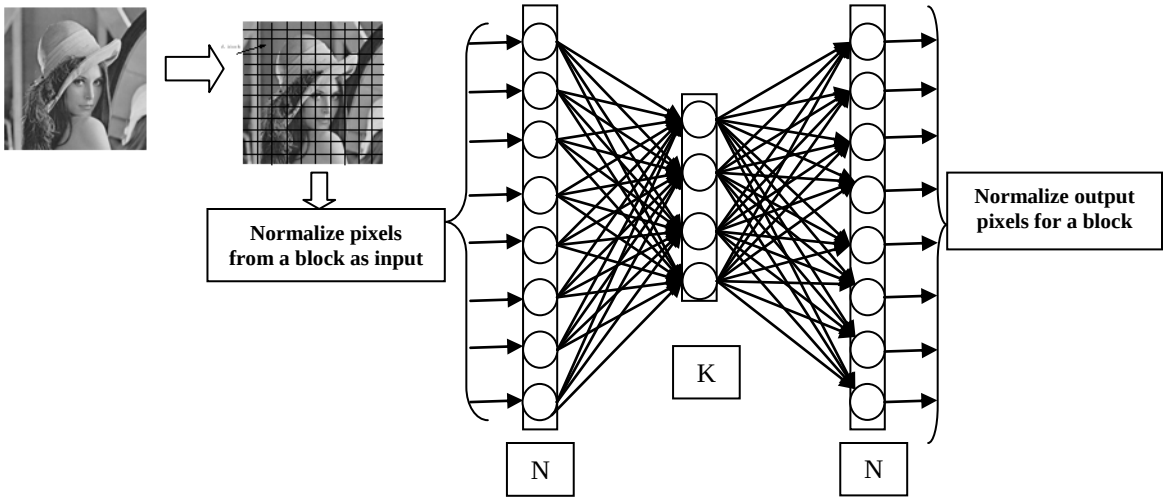


Figure 11. Architecture of neural network at the time of training

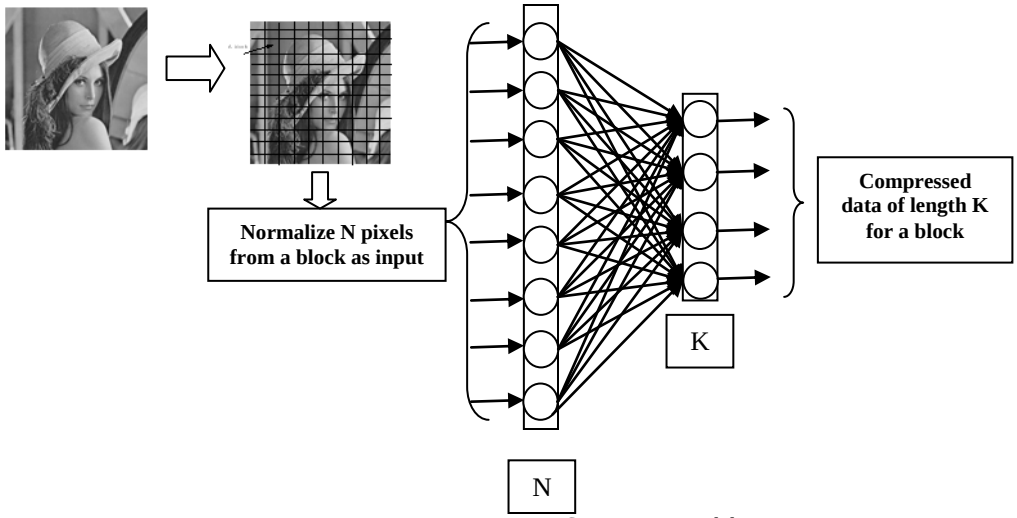


Figure 12. Compression module

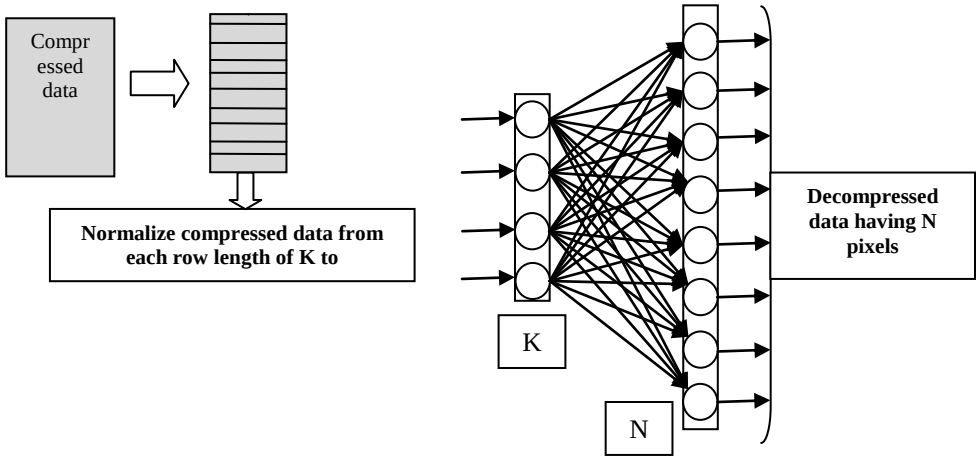


Figure 13. Decompress module

Table 3

Training image	Image format	Learning Time	No. of epoch.	Error	CR	PSNR(db)	Comp. Time(sec.)	Decom. Time(sec.)
Lena	BMP	1.771	200	0.035	4:1	15.15	0.0692	0.1665

Table 4

Training image	Image format	Learning Time	No. of epoch.	Error	CR	PSNR(db)	Comp. Time(sec.)	Decom. Time(sec.)
Lena	BMP	66.69	50	0.005	4:1	32.6017	0.0686	0.1881

Test image	Image format	CR	PSNR(db)	Comp. Time	Decom.Time
Elaine	TIF	4:1	31.0652	0.0691	0.1754
Boat	TIF	4:1	28.5801	0.0700	0.1844
Pepper	TIF	4:1	29.5953	0.0696	0.1875

Table 5

Training image	Image format	Learning Time	No. of epoch.	Error	CR	PSNR(db)	Comp. Time(sec.)	Decom. Time(sec.)
Pepper	TIF	61.32	50	0.005	4:1	31.4338	0.0692	0.1845

Test image	Image format	CR	PSNR(db)	Comp. Time	Decom.Time
Elaine	TIF	4:1	30.3149	0.0689	0.1834
Boat	TIF	4:1	28.4153	0.0693	0.1864
Pepper	BMP	4:1	31.4238	0.0696	0.1928

Table 6

Training image	Image format	Learning Time	No. of epoch.	Error	CR	PSNR(db)	Comp. Time(sec.)	Decom. Time(sec.)
Lena	BMP	43.03	50	0.0010	8:1	29.7739	0.0665	0.1799

Test image	Image format	CR	PSNR(db)	Comp. Time	Decom.Time
Elaine	TIF	8:1	29.4417	0.0336	0.1833
Boat	TIF	8:1	26.0333	0.0578	0.1790
Pepper	TIF	8:1	27.3020	0.0576	0.1821

Table 7

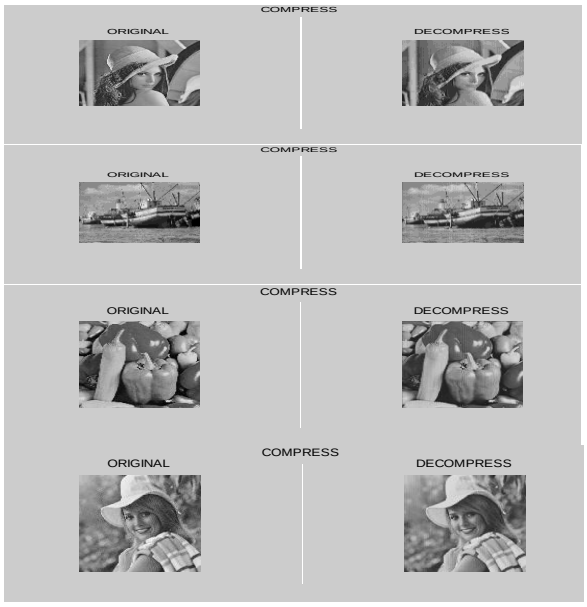
Training image	Image format	Learning Time	No. of epoch.	Error	CR	PSNR(db)	Comp. Time(sec.)	Decom. Time(sec.)
Pepper	TIF	47.4454	50	0.0012	8:1	28.6450	0.0675	0.1857

Test image	Image format	CR	PSNR(db)	Comp. Time	Decom.Time
Elaine	TIF	8:1	28.2311	0.0581	0.1795
Boat	TIF	8:1	26.2294	0.0583	0.1855
Lena	BMP	8:1	28.5439	0.0570	0.1871

Table 8

[Durai & Sarao]				Proposed method	
Image	*CR*	*PSNR(db)*	*Time(sec)*	*PSNR(db*	*)* *Time(sec)*
Lena	4:1	28.91	182	**32.60**	66.69
Pepper	4:1	29.04	188	**31.43**	61.32
Boat	4:1	29.12	178	**29.68**	64.75

Compression ratio: 8:1



Figure(14). Performance by gradient decent

VIII. CONCLUSION

There is increasing demand of image compression based processing in various applications. Numbers of methods are available and up to some extent they are generating satisfactory results. However, with changing technology there is still a wider scope to work in this area. New techniques may be proposed which could either replace or provide the support of existing methods. Compression techniques based on neural network have good scope in both ways. The general perception about GA is that, it performs better over back propagation based learning has proven wrong in the present work. Even though same algorithm of GA performs very well for XOR classification and mapping of small data, for image compression.GA based learning for neural network is suffering from curse of very slow convergence and poor quality of compression. Whereas, back propagation has shown high converging speed with good quality of compression. The method is also applicable to a wide range of different image file types. Security of compressed data is another inherent advantage available if compression happen by neural network in direct approach (i.e. until weights are not available it is nearly impossible to find the contents of the image from compressed data).

ACKNOWLEDGMENT

I would like to convey my thanks to my guide Dr. G.G. Rajput for providing helpful guidance and encouragement. It is my pleasure to express deep sense of gratitude and profound to Mr. Manoj Kumar Singh for guiding & giving me a strength of a solution.

I would like to thank my family and friends for the splendid support and encouragement, without whom this paper would have been just a dream.

Last but not the least I extend my sincere thanks to all the teaching and non-teaching staff of the computer science dept. Gulbarga University, Gulbarga.

REFERENCES

[1] J.Jiang, “Image compression with neural networks :A survey ”.Signal Processing: Image Communication 14 (1999) ,737-760

[2] Moghadam, R.A. Eslamifar, M. “Image compression using linear and nonlinear principal component neural networks” .Digital Information and Web Technologies, 2009. ICADIWT '09 . Second International Conference on The London,Issue Date: 4-6 Aug. 2009 ,pp: 855 – 860.

[3] Palocio,Crespo,Novelle,”image /video compression with artificial neural network”.Springer-Verlag,IWANN 2009,part ii,LNCS 5518, pp:330-337.2009

[4]Bodyanskiy,grimm,mashtalir,vinarski,”fast training of neural network for image compression”.Springer-Verlag ,ICDM 2010,LNAI 6171, PP;165-173,2010.

[5] Liying Ma, Khashayar Khorasani,” Adaptive Constructive Neural Networks Using Hermite Polynomials for Image Compression “,Advances in Neural Networks, springer,Volume 3497/2005, 713-722, DOI: 10.1007/11427445_115

[6] Dipta Pratim Dutta, Samrat Deb Choudhury, Md Anwar Hussain, Swanirbhar Majumder, "Digital Image Compression Using Neural Networks," act, pp.116-120, 2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies, 2009

[7] B. Karlik, “Medical Image Compression by Using Vector Quantization Neural Network”, ACAD Sciences press in Computer Science,vol. 16, no. 4, 2006 pp., 341-348.

[8] Y. Zhou., C. Zhang, and Z. Zhang, “Improved Variance-Based Fractal Image Compression Using Neural Networks”, Lecture Notes in Computer Science, Springer-Verlag, vol. 3972, 2006, pp. 575-580

[9] Durai S.A. and E.A. Saro,. Image compression with back-propagation neural network usingcumulative distribution function. World Acad. Sci. eng. Technol:60-64, 2006.

[10] Rao, P.V. Madhusudana, S. Nachiketh, S.S. Keerthi, K. “image compression using artificial neural network”.EEE,ICMLC 2010, PP:121-124.

[11] Dutta, D.P.; Choudhury, S.D.; Hussain, M.A.; Majumder, S.; ”Digital image compression using neural network”.IEEE,international Conference on Advances in Computing, Control,Telecommunication …….Technologies, 2009. ACT '09.

[12] N.M.Rahim, T.Yahagi, “Image Compression by new sub-image bloc Classification techniques using Neural Networks”, IEICE Trans. On Fundammentals, Vol. E83-A, No.10, pp 2040-2043, 2000

[13] M. S. Rahim, "Image compression by new sub- image block Classification techniques using neural network. IEICE Trans. on Fundamentals of Electronics, Communications, and Computer Sciences, E83-A (10), (2000),pp. 2040- 2043.

[14] D. Anthony, E. Hines, D. Taylor, and J. Barham, “A study of data compression using neural networks and principal component analysis,”in *Colloquium on Biomedical Applications of Digital Signal Processing*,1989, pp. 1–5.

[15] G. L. Sicuranzi, G. Ramponi, and S. Marsi, “Artificial neural networkfor image compression,” *Electronics Letters*, vol. 26, no. 7, pp. 477–479,March 29 1990.

[16]M.Egmont-Petersen, D.de.Ridder, Handels, “Image Processing withNeural Networks – a review”, Pattern Recognition 35(2002) 2279-2301, www.elsevier.com/locate/patcog

[17] M. H. Hassoun, *Fundamentals of Artificial Neural Networks*,MIT Press, Cambridge, MA, 1995.

[18] K.I. Diamantaras, S.Y. Kung, Principal Component Neural Networks: theory and applications , John Wiley &Sons,1996

[19] S. Rizvi, L. Wang," Nonlinear vector prediction using feedforward Neural network", IEEE Trans. Image Process., 6(10), (1997), pp. 1431-1436.

[20] A. K. Ashikur, C. Mofizur, "A new approach for compressing images Using neural network", CIMCA Proceeding, (February 2003), Vienna – Austria.

[21] M. Egmont-Ptersen, D. de Ridder, H. Handels, "Image processing withNeural networks- a review", Pattern Recognition , 35 (2002) pp. 2279- 2301.

[22] R. D. Dony, S. Haykin, "Neural network approaches to image compression",Proc. IEEE 83 (2) (February 1995) pp. 288- 303.

[23] B.Verma, B.Blumenstin and S. Kulkarni, Griggith University, Australia,"A new Compression technique using an artificial neural network".

[24] M.A.Otair, W.A.Salameh (Jordan), "An Improved Back-Propagation Neural Networks using a Modified Non-linear function", The IASTED Conference on Artificial Intelligence and Applictions, Innsbruck, Austria, February 2006.

[25] T. M. Nabhan and A. Y. Zomaya, "Toward generating neural network structures for function approximation," *Neural Networks*, vol. 7, no. 1, pp. 89–99, 1994.

AUTHORS' PROFILE

Vrinda Shivashetty pursuing Ph.D in image compression from the University of Gulbarga University,Gulbarga, Karnataka.Field of interest includes the intelligent image processing, evolutionary computation.

G.G Rajput presently working under the Dept. of studies & Research in computer science, Gulbarga University, Gulbarga.

Manoj kr. Singh is having background of R&D in Nanotechnology, Evolutionary computation,Neural network etc.Currently he is holding the Post of director in Manuro Tech. Research. He is alsoactively associated with Industry as a consultant & guiding number of research scholars.