International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:4, 2008

Image Mapping with Cumulative Distribution Function for Quick Convergence of Counter Propagation Neural Networks in Image Compression

S. Anna Durai, and E. Anna Saro

***Abstract*—**In general the images used for compression are of different types like dark image, high intensity image etc. When these images are compressed using Counter Propagation Neural Network, it takes longer time to converge. The reason for this is that the given image may contain a number of distinct gray levels with narrow difference with their neighborhood pixels. If the gray levels of the pixels in an image and their neighbors are mapped in such a way that the difference in the gray levels of the neighbor with the pixel is minimum, then compression ratio as well as the convergence of the network can be improved. To achieve this, a Cumulative Distribution Function is estimated for the image and it is used to map the image pixels. When the mapped image pixels are used the Counter Propagation Neural Network yield high compression ratio as well as it converges quickly.

***Keywords*—**Correlation, Counter Propagation Neural Networks, Cummulative Distribution Function, Image compression.

I. Introduction

IMAGE compression research aims at reducing the number of bits needed to represent an image. In lossless compression schemes, the reconstructed image, after compression, is numerically identical to the original image. However lossless compression can only achieve a modest amount of compression. Lossy schemes are capable of achieving much higher compression. Image compression algorithms [1] take into account the psycho visual features both in space and frequency domain and exploit the spatial correlation along with the statistical redundancy. Almost all practically used algorithms adopt a Quantization stage, which makes the algorithms lossy, and thus achieve a desired compression ratio. However, usages of the algorithms are dependent mostly on the information contained in images. A practical compression algorithm for image data should preserve most of the characteristics of the data while working in a lossy manner and maximize the gain and be of lesser algorithmic complexity. . In general almost all the traditional approaches adopt a two-stage process, first, the data is transformed into some other domain and or represented by the indices of the codebook, followed by an encoding of the transformed coefficients or the codebook indices. The first stage is to minimize the spatial correlation or to make use of the correlation so as to reduce the data. Most commonly adopted approaches rely on the transformed techniques and or the use of vector Quantization. Most of the algorithms exploit spatial correlations. Discrete Cosine Transform is used practically in almost all image compression techniques. Wavelet Transform has been proven to be very effective and has achieved popularity over Discrete Cosine Transform. However, inter-pixel relationship is highly non-linear and un-predictive in the absence of a prior knowledge of the image itself. So, predictive approaches would not work well with natural images. Transform based approaches have been used by most of the researchers in some combination or the other. Among the non-transformed approaches Vector Quantization based techniques encodes a sequence of samples rather than encoding a sample and automatically exploits both linear and non-linear dependencies. It is shown that Vector Quantization is optimal among block coding techniques, and that all transform coding techniques can be taken as a special case of Vector Quantization with some constraints. In Vector Quantization, approximating a sequence to be coded by a vector belonging to a codebook performs encoding. Creation of a straight and unconstrained codebook is a computationally intensive and the complexity grows exponentially with the block size.

Artificial Neural Networks have been applied to image compression problems, due to their superiority over traditional methods when dealing with noisy or incomplete data. An Artificial Neural Network contains no domain knowledge in the beginning, but it can be trained to make decisions by mapping exemplar pairs of input data into exemplar output vectors, and adjusting its weights so that it maps each input exemplar vector into the corresponding output exemplar vector approximately. A knowledge base pertaining to the internal representations is automatically constructed from the data presented to train the network. Well-trained neural

Manuscript received September 23, 2006.

S. Anna Durai is the Principal of Government College of Engineering, Tirunelveli 627007, TamilNadu, India.

E. Anna Saro is Assistant Professor, Department of Computer Science, Sri Ramakrishna College of Arts and Science for Women, Coimbatore 641044 TamilNadu, India (e-mail: asv_srcw@yahoo.co.in).

networks represent a knowledge base in which knowledge is distributed in the form of weighted interconnections where a learning algorithm is used to modify the knowledge base from a set of given representative cases. Any functional form relating the independent variables to the dependent variables need not be imposed in the neural network model. Neural networks are thought to better capture the complex pattern of relationships among variables than statistical models because of their capability to capture non-linear relationships in data. Developers need not build the rules with logical conditions as neural networks investigate the empirical distribution among the variables and determine the weight values of a trained network.

Neural networks seem to be well suited to image compression, as they have the ability to preprocess input patterns to produce simpler patterns with fewer components. This compressed information preserves the full information obtained from the external environment. Not only can Artificial Neural Networks based techniques provide sufficient compression rates of the data in question, but also security is easily maintained. This occurs because the compressed data that is sent along a communication line is encoded and does not resemble its original form. . Many different training algorithms and architectures have been used. The two different types of image compression approaches adopting Artificial Neural Networks are direct pixel-based encoding and decoding and pixel-based encoding and decoding based on a modular approach. Different types of Artificial Neural Networks [2] have been trained to perform image compression. Feed-forward networks, Self-Organizing Feature Maps, Learning Vector Quantization network, Counter Propagation Networks, Auto-associated networks have been applied to image compression. These networks contain at least one hidden layer, with fewer units than the input and output layers. The network is then trained to recreate the input data. Its bottleneck architecture forces the network to project the original data onto a lower dimensional manifold from which the original data should be predicted. The most advanced approaches are based on specialized compression modules. These approaches either combine different Artificial Neural Networks to obtain the best possible image compression rate or they combine more traditional statistical methods with one or more Artificial Neural Networks. Artificial Neural Networks approaches compete with well-established compression techniques such as JPEG .The major advantage of Artificial Neural Networks is that their parameters are adaptable, which may give better compression rates when trained on specific image. However in multi layer feed forward Neural Networks, the training process is slow and its ability to generalize a pattern-mapping task depends on the learning rate and the number of units in the hidden layer.

Counter Propagation Neural Networks are very popular because of its simplicity in calculations and strong power of generalization. It is based on clustering of data, a method by which large sets of data are grouped into clusters of smaller sets of similar data. A pixel may be viewed as a vector in a three-dimensional space and an image file is a collection of such vectors. So image compression problem may be viewed as clustering of those vectors into groups based on their similarities. Counter Propagation Neural Networks accepts a large amount of image data compresses it for storage or transmission and subsequently restores it when desired. When the network is presented a new input vector, the cluster in which the vector lies is first determined and is then represented by the reproduction vector of that cluster. Thus by using an encoded version of this reproduction vector for storage or transmission in place of the original input vector, considerable savings in storage is achieved. Counter Propagation Neural Networks are based on unsupervised learning. Further Self-organization in Counter propagation networks is a natural-clustering process in which the network performs competitive learning to perceive pattern classes based on data similarity. Hence learning is fast in the Counter propagation networks.

Images are of various types, like natural images, medical images, satellite images etc. As the size of these images increases, more number of samples is needed for training the Counter Propagation Neural Networks and consequently the Network takes longer time to converge. The compression ratio achieved is also not high. In addition the image samples will be of redundant in nature. To overcome these difficulties a new approach has been proposed. The image is mapped by estimating the Cumulative Distribution Function. Mapping results in reduction of the inter-pixel redundancies. This will augment the formation of clusters in the Network, which helps in quick convergence and to achieve high compression ratios.

This paper is divided into six sections. The background theory of Counter Propagation Neural Networks is discussed in Section II. The Algorithm for compression of Gray scale images using Counter Propagation Neural Networks is detailed in section III. Mapping of image pixels by estimating the Cumulative Distribution Function of the image to improve the compression ratio and convergence time of the network is explained in Section IV.The experimental results are discussed in section V and followed by conclusion in section VI.

II. Counter Propagation Neural Networks

Robert Hecht-Nielsen developed the Counter Propagation Neural Networks [3], [4] as a means to combine an unsupervised Kohonen layer with a Grossberg layer. This network synthesizes complex classification problems, while trying to minimize the number of processing elements and training time. The network got its name from this counter-posing flow of information through its structure. There is only one Feedforward path from input layer to output layer. The general form of the Counter Propagation Neural Networks can be seen in Fig. 1.

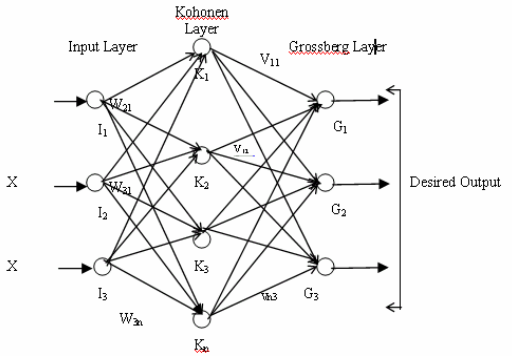

Fig. 1 Counter Propagation Neural Network Architecture

It is a three-layer architecture. An input vector is applied to the nodes on the first layer. The input nodes of the Kohonen layer are connected to the Kohonen neurons by weights w_{ij} while the Kohonen outputs are connected to the Grossberg layer by the connecting weights v_{kg} .The output of the Kohonen layer is 'winner take all 'and is determined by the sum of the x_i w_{ij} connections to each node. Each node on the second layer calculates its net input value and a competition is held to determine which unit has the largest net input value. That node is the only node that sends a value to the nodes in the third layer. The output of the Grossberg layer is found by scaling the output of the Kohonen layer with the connecting weights to the Grossberg layer. Therefore, for any given input, the output will be generated through two layers of transformation, where the input layer has a fixed number of nodes depending on the input vector size and the output is defined by the application requirements. The number of neurons in the Kohonen layer can be optimized to obtain the desired output accuracy.

The Kohonen layer is trained by a geometrical fitting procedure, where the weight vector w_{new} is found by comparing the input vector x to the original weight vector w_{old} and applying a training rate coefficient α. Equation (1) is iteratively applied through the training process until the trained weights are achieved.

$$w_{new} = w_{old} + \alpha (x - w_{old}) \quad (1)$$

Once the Kohonen layer has been trained, the Grossberg layer can be trained by applying a training vector with a known classification. The previously trained Kohonen winning node will enable the appropriate weight vector v_{kg} and Grossberg output can be forced to a desired binary output response. Then the vector weights can be trained by

$$v_{new} = v_{old} + \beta (y_1 - v_{old}) k_i \quad (2)$$

In Equation (2), β is gradually decremented from 0.1 to 0.0. k is the Kohonen output vector and y is the desired output vector. The previous connecting weight is v_{old} and the new weight is v_{new} .

III. Algorithm for Compression of Gray Scale Images using Counter Propagation Neural Network

The Counter Propagation Neural Networks can be adopted for compression of images [5], [6] by dynamically organizing the middle layer. For the network to operate properly, the input vector is normalized. The Counter Propagation Neural Networks use a standard Kohonen paradigm, which self-organizes the input sets into classification zones. It follows the classical Kohonen learning law. This layer acts as a nearest neighbor classifier in that the processing elements in the competitive layer autonomously adjust their connection weights to divide up the input vector space in approximate correspondence to the frequency with which the inputs occur. The output layer for counter-propagation is basically made up of processing elements, which learn to produce an output when a particular input is applied. Since the Kohonen layer includes competition, only a single output is produced for a given input vector. This layer provides a way of decoding that input to a meaningful output class. Since only one output from the competitive Kohonen layer is active at a time and all other elements are zero, the only weight adjusted for the output processing elements are the ones connected to the winning element in the competitive layer. In this way the output layer learns to reproduce a certain pattern for each active processing element in the competitive layer. If several competitive elements belong to the same class, that output processing element will evolve weights in response to those competitive processing elements and zero for all others.

A. Methodology

The information of a gray scale image is defined by its gray scale value and is stored pixel by pixel in a predetermined order. In a bitmap file, 8 bits are allocated for storing each pixel's information. During Compression lesser number of bits as much as possible are allowed to make compression process effective. One way to achieve this objective is to group those pixels together, which are exactly same or very close to each other with respect to gray scale values. Counter Propagation Neural Networks are best suited for clustering purpose. The information in the image, which is to be compressed, has to be retrieved pixel by pixel. The number of neurons in the input layer will be one for each pixel of the image say x_i. The number of neurons in the Kohonen layer k_j will vary based on the total number of clusters that are allowed. In standard counter propagation networks the middle layer has a fixed number of neurons depending on the compression ratio requirements. The compression ratio achieved is proportional to the number of neurons in the middle layer. In order to compromise the compression ratio and quality of Image an optimal number of neurons are to be selected. Existing techniques use trial and error procedure to select the number of neurons in the middle layer. These approaches take considerable time and the network does not converge on many occasions. To over come these difficulties the optimum number of neurons can also be selected by dynamically organizing the neurons in the middle layer. Each

of the neuron in the middle layer has connecting weights w_{ij} for the neurons of the input layer. The weights will be trained till they can distinguish the similar pixels from the dissimilar pixels. The neurons in the final layer are responsible for reproduction of the gray scale information of pixels.

Random weights are assigned between input Layers to Kohonen layer and they are kept constant. These weights are responsible to distinguish the similar pixels and the dissimilar pixels. After assigning the weights the net value of each neuron in Kohonen layer will be computed using the formula $Net_{(j)} = x_1 w_{1j}$. The neuron having high net value is declared as the winning neuron for the current pixel. When the pixel having the similar type is applied to the input layer, the same neuron in the middle layer is going to fire. In this way the clustering process will be fully automated. After processing all the pixels to form clusters, the next step is to select a representative value for each cluster or group. The representative for a particular cluster is the average of all the values within the cluster. The weight v_{ij} between the winning neuron of the current pixel and the final layer is adjusted using equation (2). All the pixels of the image are applied several times until the weights between these layers converge. The value of β will be slowly increased. The algorithm to train the Counter Propagation Neural Network is given below.

B. Algorithm

Step1: Initialize the input layer weights randomly. Assign a value to the error limit e_L.
Input the next pixel p = (xi) do:
Step2: Apply the pixel value to the neuron in the Input layer
Step3: for j = 1 to n do
Step4: Multiply input component with weight vector component and calculate the output NET_j using the equation $NETj = \Sigma x_i w_{ij}$
Step5: Assign pixel on that group j for which the value of NET_j is the maximum
Step6: Adjust the weight between output layer and intermediate layer using the formula
$v_{ij}(new) = vij\ (old) + \beta (y_j - v_{ij}\ (old))$
Step7: Compute the error = y_i – desired output.
Check if e_L ‹ error, go to step1
If error › e_L then introduce one more neuron in the middle layer and go to step6
Step8: Repeat steps 1-7 until v_{ij} values are converged
Step 9: Write the compression file.

C. Decompression Process

First the header part is retrieved form the compressed file from which the height and width of the original file is retrieved. The total number of pixels will be calculated by multiplying the height and width. Next the winning neuron for each pixel is retrieved from the compressed file. The corresponding representative pixel value is also retrieved and it is written in the retrieval file. This procedure is repeated for all the pixels of the image.

IV. Mapping of Pixels by Estimating the Cumulative Distribution Function of the Image to Improve the Performance of the Counter Propagation Neural Network

An image may contain a number of distinct gray levels with narrow difference with their neighborhood pixels. If the gray levels of the pixels in an image and their neighbors are mapped in such a way that the difference in the gray levels of the neighbor with the pixel is minimum, then compression ratio as well as the convergence of the Counter Propagation Neural Network can be improved. To achieve this, the Cumulative distribution function [1], [7] is estimated for the image and it is used to map the image pixels. When the mapped image pixels are used, the Counter Propagation Neural Network yields high compression ratio as well as it converges quickly.

Consider an image as a collection of random variables, each with cumulative distribution and density of

$$F_x(x) = \text{Prob}\ \{X \le x\} \tag{3}$$

$$p_x(x) = \frac{d}{dx} F_x(x) \tag{4}$$

Now consider forming the following random variable.

$$Y = F_x(x) \tag{5}$$

Here Y is the result of mapping the random variable x through its own cumulative distribution function. The cumulative distribution of Y can be easily computed.

$$\begin{aligned} F_y(y) &= \text{Prob}\ \{Fx(x) \le y\} \\ &= \text{Prob}\ \{X \le F_x^{-1}(y)\} \\ &= F_x(F_x^{-1}(y)) \\ &\quad 0 \text{ for } y < 0 \\ &= y \text{ for } 0 \le y \le 1 \\ &\quad 1 \text{ for } y > 1 \end{aligned} \tag{6}$$

This shows that y has a uniform distribution on the interval (0,1) Therefore; the histogram of an image can be equalized by mapping the pixels through their cumulative distribution function $F_x(x)$. In order to implement histogram equalization, the cumulative distribution function for the image is estimated. It is done using the image histogram. Let h(i) be the histogram of the image formed by computing the number of pixels at gray level i. Typically, the pixels take on the values i=0, …,L-1 where L = 256 is the number of discrete levels that a pixel can take on. The cumulative distribution function can then be approximated by

$$F_x(i) = \frac{1}{h(L-1)} \sum_{j=0}^{j=1} h(j) \tag{7}$$

Here the normalization term assures that $F_x(L-1)=1$. By applying the concept of equation (3), a pixel of $X_{s\ is}$ equalized at the position s Є S where S is the set of position in the image.

$$Y_s = F_x(X_s) \tag{8}$$

However, Y_s has a range from 0 to 1 and may not extend over the maximum number of gray levels. To correct these problems, we first compute the minimum and maximum

values of Y_s.

$$Y_{max} = \max_{s \in S} Y_s \quad (9)$$

$$Y_{min} = \min s \in SY_s \quad (10)$$

And then we use these values to from Z_s, a renormalized version of Y_s

$$Z_s = (L-1)\frac{F_x(X_s) - Y_{min}}{Ymax - Ymin} \quad (11)$$

The transformation form X_s to Z_s. is Histogram Equalization.

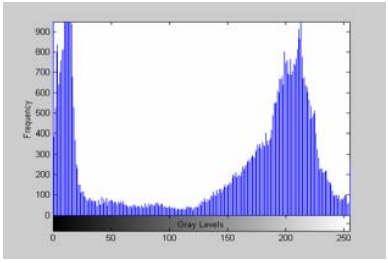

Fig. 2 Histogram of sample image

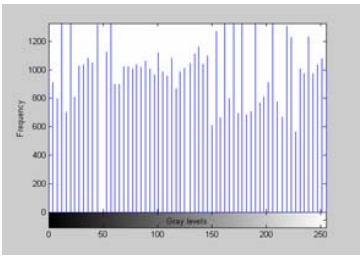

Fig. 3 Histogram of Equalized image

Histogram equalization does not introduce new intensities in the image. Existing values will be mapped to new values resulting image with less number of the original number of intensities.

The main problem of the clustering algorithm is that, two pixels of different values after normalization may form two similar vectors. Hence they will fall on the same clusters although actually these two pixels are having very dissimilar pixel values. The Representative value of the cluster, which is the average value of the pixels in the cluster, will not be the true representation. While retrieving such information during the decompressing process the original image will get distorted. Secondly, some of the neurons in the Kohonen layer may not get fired at all if the pixel values are concentrated to only a limited range. This leads to a situation where the Network is either idle or execute with less efficiency. The overall time taken for simulation will be unnecessarily long. To remove these problems, the existing approaches preprocess the raw pixel values by mathematical calculations which make the simple network more complex. Hence the new approach of preprocessing the pixels by mapping the pixels by estimating the cumulative Distribution function of the image is proposed.

Mapping of the pixels by estimating the Cumulative Distribution Function of the image results in correlation of the pixels and due to the presence of similar pixel values the formation of clusters will be very fast and thus the training time is drastically reduced resulting in quick convergence of the network. Due to similarity in pixels the net winning neurons or clusters in the middle layer will be less and thus high compression ratio could be achieved. Since the pixel values after mapping by Cumulative Distribution Function represent the entire range of the gray scale, the value of each representative of the clusters will be unique. Hence the number of bits required for storing the indices in the compressed file will be less, which further enhances the compression ratio. Similarly in the decompression phase since indices are less, the time taken for generation of weights and consequently the time for decompression also gets reduced. As such the total simulation time is very much reduced. In this neural network for image compression, since the indices retrieved during decompression represent the whole range of gray levels, the topology of the input pattern is maintained without much reorganization during this process and hence the reconstruction errors are less and the finer details of the image are preserved. The Quantization Error is traditionally related to all forms of clustering algorithms. However since mapping distributes the data sparsely, the Quantization Error is reduced.

The quality of the decompressed image and the convergence time has been experimentally proved to be better than achieved by conventional methods and by the same algorithm without mapping the image by Cumulative Distribution Function. The correlation of pixel values plays a vital role in augmenting the convergence of the neural network for image compression and this is a simple mechanism compared to other existing transformation methods, which are computationally complex and lengthy process.

V. Experimental Results

The performance of the Counter Propagation Neural Network for image compression with the concept of mapping the image by Cumulative Distribution Function has been tested in various types of standard images as well as medical MRI scan images and satellite images. More than 15 experiments were conducted by varying the values of the parameters namely training rate coefficients α and β and the number of neurons in the Kohonen layer of the Network. Those parameters that produced the optimum values in respect of time of convergence, compression ratio and quality of image have been adopted to test the new approach in various types of images. The results achieved for some samples of images are illustrated in Table I. The corresponding values obtained without adopting the new approach are illustrated in Table II. The proposed approach resulted in quick convergence of the Network. The quality of the decompressed images is also high and PSNR values up to 29.98dB have been obtained by the proposed approach for compression ratio of 9.8. To present an idea of the performance of the proposed

approach the results are compared with those obtained from a similar existing compression scheme [5]. The simulation has been carried out on the Lena, Mandrill and Boat images. PSNR values of 27.9dB, 28.34dB and 28.48 dB have been obtained against compression ratios of 4.97, 4.55 and 4.90 for the respective images. The new approach using Cumulative Distribution Function for mapping the image has achieved PSNR value of 29.01dB for Lena, 29.76dB for Mandrill and 29.45dB for Boat image against compression ratios of 9.7, 9.8 and 9.1 respectively. Further the proposed approach is computationally simple when compared to the hybrid method referred above for comparison. The original image, image mapped by Cumulative Distribution Function and the decompressed image for the samples of images tested are illustrated. (Fig. 4,5&6) The visual quality of all the decompressed images irrespective of the type and nature of the images are better than achieved by other conventional methods. The Experimental Results and the test images are appended.

VI. Conclusion

The Counter Propagation Neural Network for Image Compression has been used to compress various types of Gray scale images. By adopting the proposed approach the PSNR achieved is 29.98 dB for compression ratio of 9.80. The time taken for simulation has been reduced to nearly 50%. The performance of the Counter Propagation Neural Network has been substantially improved by the proposed approach. Mapping the image by Cummulative Distribution Function has helped the Counter Propagation Neural Network to converge easily and to achieve high compression ratios compared to previous Techniques. The method can be extended for progressive compression of images by dynamically estimating the Cumulative distribution function of the image in the regions of interest and thereafter-compressed using Counter Propagation Neural Network.

Appendix

TABLE I
Experimental Results of the Proposed Approach

S.No	Image	CR	PSNR (dB)	TIME (Sec)
1	Cameraman	9.4.1	29.84	79
2	Lena	9.7:1	29.01	84
3	Pepper	9.8:1	29.14	81
4	Fruits	9.8:1	29.98	79
5	Boat	9.1:1	29.45	68
6	Mandrill	9.8:1	29.76	63
7	Abdomen(Mri)	9.4:1	29.60	57
8	Thorax (Mri)	9.8:1	29.72	52
9	Satellite	9.0:1	29.30	65

TABLE II
Experimental Results without Mapping

S.No	Image	CR	PSNR (dB)	TIME (Sec)
1	Cameraman	4.1:1	26.92	156
2	Lena	4.6:1	26.15	162
3	Pepper	4.2:1	26.65	158
4	Fruits	4.3:1	27.16	156
5	Boat	5.8:1	26.85	133
6	Mandrill	5.1:1	27.94	121
7	Abdomen(Mri)	5.2:1	26.11	115
8	Thorax (Mri)	4.9:1	26.14	103
9	Satellite	3.9:1	26.41	120

Fig. (4a) Fig. (4b) Fig. (4c)
(4a) Original image (4b) mapped by CDF (4c) Decompressed
PSNR = 29.01dB, Convergence Time = 84 Sec, C.R = 9.70

Fig. (5a) Fig. (5b) Fig. (5c)
(5a) Original image (5b) Mapped by CDF (5c) Decompressed
PSNR = 29.76dB, Convergence Time = 63 Sec, C.R = 9.80

Fig. (6a) Fig. (6b) Fig. (6c)
(6a) Original image (6b) Mapped by CDF (6c) Decompressed
PSNR = 29.45dB, Convergence Time = 68 Sec, C.R = 9.10

References

[1] Rafael C. Gonzalez, Richard E.Woods, *Digital Image Processing*, 2nd Ed., (PHI, 2005).
[2] Simon Haykin, *Neural Networks A Comprehensive Foundation.* Second Edition, (Pearson Education 2001).
[3] James A. Freeman, David M. Skapura, Neural Networks Algorithms, Applications and Programming Techniques, Pearson Education, 2004, 213 – 262.
[4] Marjan Vracko, Kohonen Artificial Neural Network and Counter Propagation Neural Network in Molecular Structure-Toxicity Studies, *Current Computer-Aided Drug Design,* 2005,1,73-78.
[5] K. M. Ashikur Rahman and Chowdhury Mofizur Rahman, New Approach for Compressing Color Images using Neural Network: CIMCA *2003 Proceeding/ISBN 1740880684; M.Mohammadian (Ed.) 12-14 February 2003, Vienna-Austria.*
[6] Hamdy S. Soliman Ahmed Abdelali, Toward Lossless Image Compression, *ISCA* 8, 2003.
[7] Rafael C. Gonzalez, Richard E. Woods, Steven L.Eddins, *Digital Image Processing Using Matlab*, (Pearson).