ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.3 , No.5, Pages : 461- 465 (2014)
Special Issue of ICACSSE 2014 - Held on October 10, 2014 in St.Ann's College of Engineering & Technology, Chirala, Andhra Pradesh

TRI-DCT BASED FAST BACK PROPAGATION ALGORITHM

G.Visalakshi, K.Chiranjeevi
[1]GMRIT, India, visalisowji@gmail.com
[2]GMRIT, India, chiru404@gmail.com

Abstract: In Artificial neural networks the number of neurons in hidden layers are fewer than input and output layer consequently ANN is used for image compression and decompression. Among the ANN's back propagation neural network algorithm (BP) is finest for image compression but training time is long. So the proposed DCT based fast back propagation neural network reduces the training time and improve the compression ratio by changing the input layer and hidden layer neurons. Time of conversation is further reduced by choosing a best learning rate parameter η. Finally DCT based fast back propagation neural network results such as compression ratio (CR) and peak signal to noise ratio (PSNR) are computed and compared with BP results.

Key words: Discrete Cosine Transform; image segmentation; Artificial neural networks; Back propagation neural network; Back propagation algorithm; Fast Back propagation algorithm.

INTRODUCTION

With the advance of the information age, the need for mass information storage and fast Communication links grows. Storing images in less memory leads to a direct reduction in storage cost and faster data transmissions. Image compression is to reduce irrelevance and redundancy of image data in order to be able to store or transmit data in an efficient form**.** Compression is achieved by the removal of one or more of the three basic data redundancies: Coding, Inter pixel and psycho visual redundancy. Now a day's mostly used image compression technique is DCT because of its energy compaction property. The DCT is almost similar to the K – L Transform where coefficients are approximated by Eigen values and Eigen vectors. Shape Adaptive DCT [8] compression is used for images with irregular shapes but time of conversion is high. SA-DCT is used in MPEG for encoding the image after dividing the image into shapes. In SA-DCT image is processed as a and then second, this process is continued for all the rows of the image.

SA-DCT of a 1-D sequence of length N is

$$C(u) = \alpha(u)\sum_{x=0}^{N-1} f(x)\cos\left[\frac{\pi(2x+1)u}{2N}\right]$$

(1)

For u = 0,1,2,…,N− 1.

Similarly, the inverse transformation is defined as

$$f(x) = \sum_{U=0}^{N-1} \alpha(u)C(u)\left[\frac{\pi(2x+1)u}{2N}\right] \qquad (2)$$

For x= 0,1,2,…,N-1.

In both equations (1) and (2) α*(u)* is defined as

$$\alpha(u)=\begin{cases}\sqrt{\frac{1}{N}}, & u=0\\ x\sqrt{\frac{2}{N}}, & u\neq 0\end{cases} \qquad (3)$$

Thus, the first transform coefficient is the average value of the sample sequence. This value is referred to as the "DC Coefficient". All other transform coefficients are called the "AC coefficients". The first the top-left waveform (*u* = 0) renders a constant (DC) value, whereas, all other waveforms (*u* = 1,2,…,7) give waveform sat progressively increasing frequencies.

sequence of discrete intensity levels, the first row is processed first

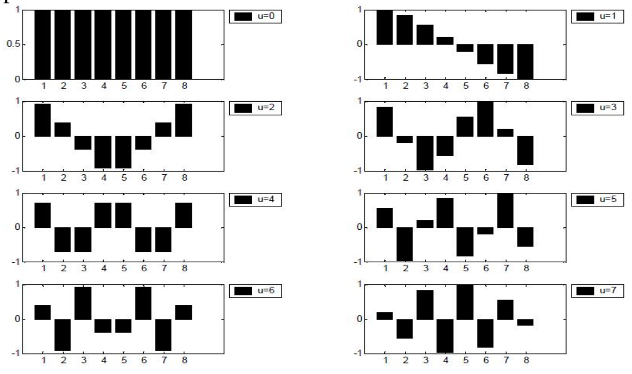


Fig 1.1 One dimensional cosine basis functions

(N=8)

These waveforms are called the cosine basis function. Note that these basis functions are orthogonal. Hence, multiplication of any waveform in Figure 1.1 with another waveform followed by a summation over all sample points yields a zero (scalar) value, whereas multiplication of any waveform in Fig 1.1 with itself followed by a summation yields a constant (scalar) value. Orthogonal waveforms are independent, that is, none of the basis functions can be represented as a combination of other basis functions.

If the input sequence has more than *N* sample points then it can be divided into subsequences of length *N* and DCT can be applied to these chunks independently. Here, a very important point to note is that in each such computation the values of the basis function points will not change. Only the values of *f* (*x*) will change in each sub-sequence.

This is a very important property, since it shows that the basis functions can be pre-computed offline and then multiplied with the sub-sequences. This reduces the number of mathematical operations (i.e., multiplications and additions) thereby rendering computation Efficiency.

The 2-D DCT is a direct extension of the 1-D case and is given by

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{\pi(2x+1)u}{2M}\right]\cos\left[\frac{\pi(2y+1)v}{2N}\right] \quad (4)$$

α(*u*)& α(*v*) is defined as

$$\alpha(u)\&\ \alpha(v) = \begin{cases} \sqrt{\frac{1}{N}}, & u,v = 0 \\ \sqrt{\frac{2}{N}}, & u,v \neq 0 \end{cases} \quad (5)$$

For *u* ,*v* =0,1,2,…, *N* −1

The inverse transform is defined as

$$f(x,y) = (u)\alpha(v)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} C(u,v)\cos\left[\frac{\pi(2x+1)u}{2M}\right]\cos\left[\frac{\pi(2y+1)v}{2N}\right]$$

— (6) For *x*, *y*=0, 1, 2,…, *N* −1

Artificial neural networks are examined as possible solutions for solving complex problems such as image and signal processing[1]. ANNs are composed of many interconnected neurons that operate in parallel and connected together via weights [2]. The Back propagation Neural Network (BPNN) is a hierarchical feed-forward ANN composed of three or more fully interconnected layers of neurons. The most frequently used training algorithm in BPNN is a back propagation algorithm (BP). The BPNN is the simplest ANN architecture for image compression but its disadvantage is very slow convergence and simple problems may take hundreds of iterations to converge[3]. Various literature researches focused on improving the speed of convergence. Simultaneously these researches applied only to limited patterns and take longer time to converge.

Several BP learning speed- up algorithms that utilize the steepness of activation function. Simulations showed that increasing the gain in fact will increase the speed of convergence[4]. These algorithms can converge quicker than the standard BP on some problems. But, these algorithms may suffer from increased instability and they frequently fail to converge within a limited time. [5]The cause for the instability is an unfortunate choice for the first weights. To defeat the instability, it is planned that automatic weight re-initialization be used whenever the convergence speed becomes very slow due to a local minimum. The simulations perform mixed up a set of problems including exclusive-or (XOR) and encoder.

The proposed modified Back propagation algorithm (MBP) approach for the learning process of BPNN with best possible initialization. [6]The MBP minimizing the sum of the squares of linear and non-linear errors for all output units for an proficient method in ANN. Appropriate initialization plays a impertinent role in the ANN. consequently, they used optimum initialization method for weight initialization which ensures that the outputs neurons are in the active region and the range of activation function is fully utilized. The planned method is implemented on 2 bit parity problem, 4 bit parity checker and encoder problem and produced good results.

The BPNN for image compression and developed algorithm based on BP. The blocks of novel image are classified into three classes: background blocks, object blocks and edge blocks, considering the features of intensity change and visual discrimination[7].

Finally to improve the convergence time for learning the BPNN image compression system. This is done by modify the BPNN architecture, modify the BP learning parameters such as learning rate and momentum variable, adding Bias variable, controlling the weights between the layers and so on.

All of these literature studies need large convergence time for BPNN training on image compression and decompression system[9]. Therefore, we require ANN learning algorithm to speed up the convergence time for BPNN image compression/decompression system. Simultaneously, we need to get good results from this BPNN image compression system which results in fewer storage requirements (high CR) and suitable PSNR. As a result, in this research, we suggested to use the fast back propagation algorithm (FBP) to train the BPNN image compression system to speed up the learning process and reduce the convergence time.

The research is organized as follows: section II describes the BPNN image compression system. Section III includes details about the FBP. Section IV includes the implementation of FBP in BPNN image compression system. Section V includes the results of using FBP in BPNN image compression learning. Finally, Section VI concludes this work.

BPNN IMAGE COMPRESSION SYSTEM

The three layered BPNN was designed for image compression based under the consideration of encoder problem as shown in Fig.1. This BPNN is fed by analog gray level of image as an input (the range between 0 and 255) and produces a proper compressed code at outputs of hidden layer units. In the reconstruction procedure, this BPNN produce an analog gray level image with the outputs of output layer units. The input and output layers have Ni units each one, and a middle layer with N_h units (N_h <

N_i) that is the channel involving the input layer and output layer.

Aim of BPNN for image compression system involves determining the number of network's layers. BPNN with three layers and a suitable number of hidden layer units is a good option as one layer of sigmoid hidden units is enough to estimated any continuous function[10]. It is renowned practically that at what time BPNN architecture is designed with more

than one hidden layer, the possibility of occurring BP problems such as trapping in local minimum, error oscillation and slow convergence would increase very much than when we design BPNN with three layers only. This would result in BPNN capability of compressing and decompressing very well.

The next step in designing the BPNN involves determining the number of neurons in each layer. The number of neurons in input layer (N_i) is equal to the number of neurons in output layer (N_o). This number is governed by the dimension of image sub block (P P).

The input to the network is a vector of real numbers consequently to the structure of BPNN. The input vectors must have a set dimension. The number of hidden layer units (N_h) is less than the number of input layer units. The number of hidden layer units affects on the compression performance of BPNN.

The next step in the BPNN image compression system is to initialize the network learning parameters such as learning rate and momentum variable, weight connections to use them during the learning process.

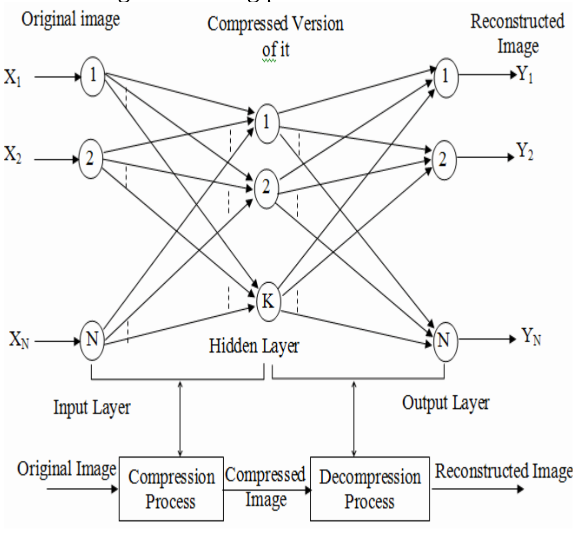


Fig 1.2 BPNN Image Compression System

Requirements of BPNN Image Compression system

BPNN architecture was recommended in this research to obtain optimum results in CR and PSNR and reducing the convergence rate. The BPNN architecture used in this research is alike to the BPNN architecture which was recommended in our earlier research.Also in this research, we recommended to use the same processes which were used in our earlier research such as: image normalization and segmentation; initialization BPNN learning parameters; and preparation of BPNN training/testing set. The BPNN learning process of image compression covers all layers of BPNN (input, hidden and output layer). The learning process is accomplished by using the BP and using a set of images

as training patterns. When learning process is finished then the BPNN image compression process requires the input and hidden layers. But the image decompression process requires the hidden and output layers.

simulation program of BP

The number of links between each two layers is calculated by multiplying the total number of neurons of the two layers, then adding the number of bias neurons connections of the second layer (bias connections of a layer is equal to the number of layer neurons). The total number of connections (network size) is given by the Eq.1

Nw = [(Ni Nh) + Nh] + [(Nh No) + No] ………… (7)

The BP simulation program contains the following steps:

1) Initialization of network weights, learning rate (η) and Threshold error. Set iterations to zero.
2) Open the file which contains the image training set.
3) Total error = zero; iterations$\rightarrow$ iterations**+1**
4) Get one vector from the file and feed it to the input layer units.
5) Initialize the target output of that vector.
6) Calculate the outputs of hidden layer units.
7) Calculate the outputs of output layer units.
8) Calculate the error, where error = desired output – actual output
   Total error$\rightarrow$ Total-error + error
9) Calculate the delta sigma of output layer units, and then adjust the weights between the output and hidden layer units.
10) Calculate the delta sigma of the hidden layer units, and then adjust the weights between the hidden and input layer units.
11) While there are more vectors in the file, go to step 4 12) if Threshold error >= Total- error then stop, else go to step 3.

FAST BACKPROPAGATION ALGORITHM (FBP)

The reputation of multi-layered ANNs motivated several researchers to center of attention on heuristic technique for accelerating the BP by adapting the learning rate (η) for the period of training or by using various other heuristics to improve the convergence time of the algorithm.

The progress of fast and efficient learning algorithms for ANNs has been a subject of interest over the past few years. Accordingly, some new algorithms have been proposed for training neural networks with various architectures that converge faster than BP. The growth of fast learning algorithm (FBP) for feed-forward ANNs is based on minimization of objective function after the initial adaption cycles. This minimization can be achieved by reducing lemda (λ) from unity to zero during the training of the network.

The BPNN training by FBP follows the same steps of BPNN training mentioned before. The FBP algorithm differs from the original BP algorithm in the development of the alternative training criterion. This criterion indicates that λ must change from 1 to 0 in training phase (i.e. as the total error decreases, should approach zero). This recommended that the value of should be determined in each adaption cycle from the total error at that point, consequently to some suitable rule (λ = (E)) where E is the error of the network. The above discussion indicates that λ =1 when E>>1. Here, for any positive integer n, $1/E^n$ approach zero and, therefore, $\exp(-1/E^n)$ 1. Alternatively, when E<<1, $1/E^n$ becomes very large so $\exp(-1/E^n)$ 0. As a result, a proper rule for the reduction of from one to zero is the following Eq.8:

λ= λ (E) = exp (- μ/ En) …………. (8)

Here a positive real number is μ and positive integer is n. The lesser the integer n, the quicker the reduction of λ when E>>1. It has been practically confirmed that if λ is much lesser than unity in the initial adaption cycles the algorithm may be trapped in a local minimum. This is an indication that n should be greater than 1. Thus, λ is determined in the training of any network according to Eq. 9:

$$\lambda = \lambda(E) = \exp(-\mu / E) \quad \text{……………......} \quad (9)$$

In BPNN guidance by using FBP, all the hidden layer neurons and all the output layer neurons use the hyperbolic tangent function

instead of sigmoid function used by all the BPNN neurons when using BP. Therefore Eq. 9 is modified for hyperbolic tangent function as follows:

$$F'(NETj) - \frac{e^{NET_j} - e^{-NET_j}}{e^{NET_j} + e^{-NET_j}} \quad \text{……….} \quad (10)$$

So that F (NETj) lies between –1 and +1.

IMPLEMENTATION OF FBP

In the present study, a simulation program to implement FBP was built using C# programming language. This program contains the implementation of the complete steps of BPNN compression and decompression system using FBP. In the initial stage of compressed file (using BPNN), there is a header block that includes information about the source image, and the compression process parameters.

Simulation Program of FBP

The implementation of the FBP simulation program is similar to the implementation of BP simulation program but it differs in the following points:

1) The FBP algorithm requires, at the initial stage of the training, setting the value of λ to 1.

2) Prior to the beginning of the BPNN training, the FBP algorithm requires the determination of value that is used in the training for modifying the value of λ. usually μ lies in the range [0.1-10].

3) Throughout the training of the BPNN using FBP algorithm, all the hidden layer neurons and output layer neurons use the hyperbolic tangent function as an alternative of sigmoid function that is used by all neurons in BPNN training using BP algorithm. The hyperbolic tangent function has a range of [-1- +1], thus, image normalization is done by transforming the coordinates [0-255] into the coordinates [-1 - +1].

4) The FBP algorithm requires setting to very small value, usually between [0.01-0.1].

5) The FBP algorithm need setting the value of ß greater than 1 (i.e. The range is [2-6]), because when ß=1, then tanh (error) behaves like a linear function and the convergence of the resulting algorithm is expected to be equal to that of the BP algorithm. If the value of ß is sufficiently larger than unity, the convergence of the algorithm is expected to be improved.

6) The rule used for computing the δ of output layer neurons in BP algorithm is modified (Eq.6). δ in FBP is calculated as follows :

$$\delta_j^2 = \lambda(O_j^2 - t_j^P) + (1\lambda)\tan(\beta(O_j^2 - t_j^P)) - (11)$$

7) In the FBP algorithm, the Eq.3 is calculated in each iteration for the purpose of λ adaptation.

BPNN Compression Process:

BPNN compression process can be summarized by the following steps:

1) Read the image pixels from the file, and then normalize it by converting it from range [0-255] into range [0-1].
2) Divide the image into non-overlapping blocks (image segmentation).
3) Rasterize the image by converting each two-dimensional block into one-dimensional vector.
4) Apply the rasterized vector into the input layer units.
5) Compute the outputs of hidden layer units by multiplying the input vector by the weight matrix between the input and hidden layer.
6) Store the outputs of hidden layer units after denormalizing them in a compressed file.
7) While there are more rasterized image vectors go to step 4.

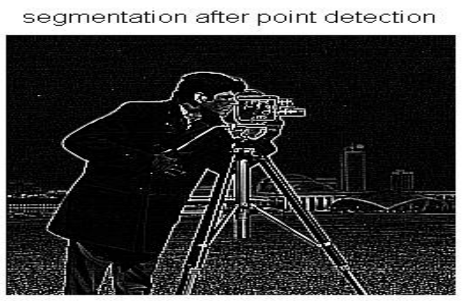
EXPERIMENTAL RESULTS

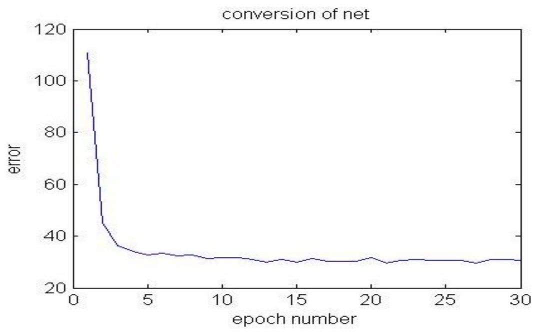
After the implementation of decoding process, the SNR (Signal to Noise Ratio) and PSNR and NMSE (Normalized Mean Squared Error) must be calculated between the original and reconstructed images to prove the quality of the decoded image by the original one.

To verify the compression performance, the values: CR, PSNR, bit per pixel rate (bpp) and convergence time are measured. The amount compression (degree of data reduction obtained as a result of the compression process) is CR. The image compression system in the BPNN and the CR is
to the data out from the hidden layer neurons. The CR can be expressed as Eq.7

$$CR = \frac{Ni \text{ (pixel size in bits)}}{Nh \text{ (pixel size in bits)}} \quad \text{….} \quad (7)$$

Where bpp rate is the number of bits necessary to represent each pixel value of compressed image. The improved compression performance is with the highest CR, the smallest amount bpp rate, and highest PSNR. Here conducted Simulations to calculate the compression performances of BPNN image compression system which learned with FBP. The effectiveness of BPNN was experienced by several experiments using real world images as shown in Fig.2





CONCULSION

The proposed back propagation neural network algorithm outperforms the exited algorithms in peak signal to noise ratio, compression ratio and time of learning for different epoch numbers. The results appear to be promising in image compression/decompression system. Also, from results, we note that the FBP can be successfully reduced the learning time (BPNN convergence time) in comparison with original BP. At the same time with maintaining the same performance of BP results (i.e. CR and PSNR).

References

[1] N. K. Ibrahim, R.S.A. Raja Abdullah and M.I. Saripan. "Artificial Neural Network Approach in Radar Target Classification," *Journal of Computer Science*, vol. 5, no.1, 2009, pp.23-32, ISSN: 1549-3636, Science Publications.

[2] R. P. Lippmann. "An Introduction to Computing with Neural Nets," *IEEE ASSP Magazine*, vol.4, no.2, April 1987, pp.4-22.

[3] N. B. Karayiannis and A. N. Venetsanopoulos. Artificial Neural Networks: Learning Algorithms, Performance Evaluation, and Applications, Kluwer Academic Publishers, London, 1993.

[4] R. C. Gonzales and P. Wintz. *Digital Image Processing*, second edition, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA,1987, ISBN: 0-201-11026-1.

[5] C. Tai-Hoon, R. W. Conners and P. A. Araman. "Fast Back-Propagation Learning Using Steep Activation Functions and Automatic Weight Reinitialization," Conference Proceedings, 1991 *IEEE International Conference on Systems, Man, and Cybernetics "Decision Aiding for Complex Systems"*, Omni Charlottesville Hotel and the University of Virginia, Charlottesville, Virginia, Vol. 3, 91CH3067 - 6 lSSN# 0-7803-0233-8/91, 13-16October1991, pp.1587-1592.

[6] V.V. Joseph Rajapandian and N. Gunaseeli. "Modified Standard Backpropagation Algorithm with Optimum Initialization for Feedforward Neural Networks," *International Journal of Imaging Science and Engineering (IJISE)*, vol.1, no.3, July 2007, GA, USA, ISSN: 1934-9955.

[7] T. Xianghong and L. Yang. "An Image Compressing Algorithm Based on Classified Blocks with BP Neural Networks," *Proc. of the international conference on computer science and software engineering*, IEEE Computer Society, Wuhan, Hubei, vol. 4, Dec 2008, pp.819-822, ISBN: 978-0-7695-3336-0, DOI: 10.1109/CSSE.2008.1357.

[8] Fatima B. Ibrahim. "Image Compression using Multilayer Feed Forward Artificial Neural Network and DCT," *Journal of Applied Sciences Research*, vol.6, no.10, 2010, pp. 1554-1560, INSInet Publication.

[9] O. N. A.AL-Allaf. "Improving the Performance of Backpropagation Neural Network Algorithm for Image Compression/Decompression System

[10] V. S. Jagesh and P. Chi-Sang. "Linear Independence of Internal Representations in Multi Layer Perceptrons," *IEEE Transactions on Neural Networks*, vol.10, no.1, 1999, pp.10-18, ISSN: 1045-9227, DOI:10.1109/72.737489.

Gurugubelli Visalakshi received B.Tech Electronics and Communication Engineering from J.N.T.U. Kakinada, and doing M.Tech in GMR Institute of Technology with DECS specilization. His research interests are in Signal Processing and Image Processing.

K.Chiranjeevi received B.E Electronics and Communication Engineering from J.N.T.U. Anantapur, and M.TechInstrumentation and Control from JNTU Kakinada, He is working as Asst. Professor in GMR Institute of Technology. His research interests are in Signal Processing and Image Processing.