International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

ANN Implementation for Image Compression and Decompression Using Back Propagation Technique

Syed Aseem Ahamed[1], K Chandrashekarappa[2]

[1]M. Tech Student (VLSI Design & Embedded Systems) Bangalore Institute of Technology, Bangalore, Karnataka, India

[2]Associate professor, Dept of ECE Bangalore Institute of Technology, Bangalore, Karnataka, India

Abstract: ***Here we are using one of the applications of artificial neural networks (ANN) for image compression and decompression, by making use of Back propagation algorithm. An image compressing algorithm based on back propagation (BP) network is developed after image pre-processing, which includes preparation of training pair, elimination of redundant pairs, and the compressed data with the noise fed into the receiver network for image reconstruction. The decompressed image is compared with the original image. This uses the same algorithm which is used for the compression. The algorithm is written in Verilog-HDL, for digital simulation and the hardware Implementation of this technique is done by using Vertex-2E Pro FPGA kit.***

Keywords: Artificial neural networks (ANN), Back propagation, compression, decompression, HDL, FPGA, and Simulation.

1. Introduction

More the number of Images the data requires to store them is very high and while transmission it requires more bandwidth, power, storage space, etc. so the image has to be compressed in such a way that the quality of original image should not be altered, this leads to an advantageous technique in more technical fields, Thus it has turned out to be a present day craze as well as source of competition in the race for technology and research with so much manpower, time and money involved for its development.

It is known that compression algorithms can be classified into two types – 'lossy' and 'lossless'. If the recovered image (after decompression) does not have the same quality as the original image then there has been a loss of some image data during compression. This is called a 'lossy compression algorithm'. But some algorithms have the ability to retain the quality of the image, even after the compression, and the decompression processes. Such algorithms come under the category of 'lossless compression algorithms'.

There are numerous lossy and lossless image compression techniques. Images where each bit of information is essential, a lossless compression must be used ones which have been already lossy digitalized a lossy compression is preferred, and particularly cosine transform based technique that shows excellent results. The main goal is to make use Back Propagation algorithm for the implementation of this image compression and decompression.

Among learning algorithms, back-propagation algorithm is a widely used learning algorithm in Artificial Neural Networks. The Feed-Forward Neural Network architecture is capable of approximating most problems with high accuracy and generalization ability [1][2]. This algorithm is based on the error correction learning rule. Error propagation consists of two passes through the different layers of the network, a forward pass and a backward pass. In the forward pass the input vector is applied to the sensory nodes of the network and its effect propagates through the network layer by layer. Finally a set of outputs is produced as the actual response of the network.

During the forward pass the synaptic weight of the networks are all fixed. During the back pass the synaptic weights are all adjusted in accordance with an error-correction rule.

The actual response of the network is subtracted from the desired response to produce an error signal. This error signal is then propagated backward through the network against the direction of synaptic conditions. The synaptic weights are adjusted to make the actual response of the network move closer to the desired response.

2. Back Propagation Technique

The back propagation algorithm is an involved mathematical tool; however, execution of the training equations is based on iterative processes, and thus is easily implement ableon a computer. [1] During the training session of the network, a pair of patterns is presented (Xk, Tk), where Xk in the input pattern and Tk is the target or desired pattern. The Xk pattern causes output responses at teach neurons in each layer and, hence, an output Ok at the output layer. At the output layer, the difference between the actual and target outputs yields an error signal. This error signal depends on the values of the weights of the neurons I each layer. This error is minimized, and during this process new values for the weights are obtained. The speed and accuracy of the learning process-that is, the process of updating the weights-also depends on a factor, known as the learning rate. Before starting the back propagation learning process, we need the following:

- Take the set of training patterns, input, and target
- Optimum value for the learning rate
- A criterion that terminates the algorithm and updates the weights using suitable methodology
- The sigmoid function (for nonlinearity)
- Initial weight values (typically small random values)

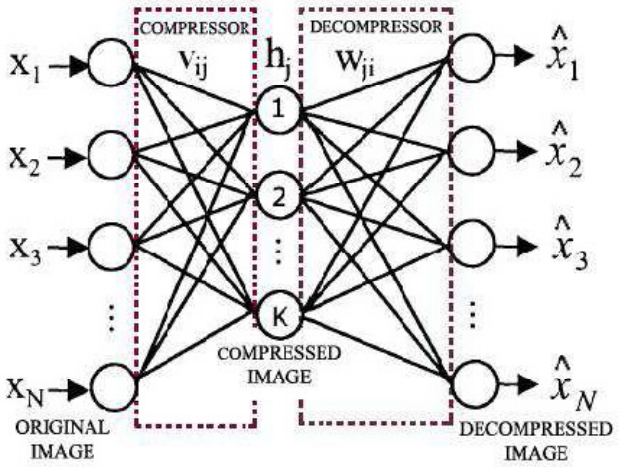

Figure 1: Basic image compression structure using neural network

The back propagation algorithm is carried out by using artificial neural network shown in figure 1, consists of input layer, hidden layer and output layer, consisting of neurons in each layer.

The process then starts by applying the first input pattern Xk and the corresponding target output Tk. The input causes a response to the neurons of the first layer, which inturn cause a response to the neurons of the next layer, and so on, until a response is obtained at the output layer. That response is then compared with the target response; and the difference (the error signal) is calculated. From the error difference at the output neurons, the algorithm computes the rate at which the error changes as the activity level of the neuron changes. So far, the calculations were computed forward (i.e., from the input layer to the output layer). Now, the algorithm steps back one layer before that output layer and recalculate the weights of the output layer (the weights between the last hidden layer and the neurons of the output layer) so that theoutput error is minimized. The algorithm next calculates the error output at the last hidden layer and computes new values for its weights (the weights between the last and next-to-last hidden layers). The above process will repeat for other layers.

3. Digital Implementation of Neuron

Digital model of Neural Architecture comprises of several components like multiplier, adders along with the nonlinear activation function circuit as shown in Figure 2. The nonlinear activation function is designed by Look UP Table (LUT). The proposed Digital architecture of Neuron is trained using Back propagation algorithm. The neural architecture as shown in Figure 1 is thus a complete digital structure. The functionality of the designed Neural architecture is verified for digital operation like AND, OR, NOT and XOR.

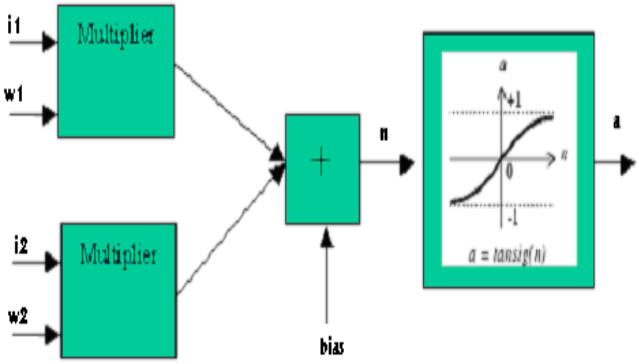

Figure 2: Digital Model of 2 input 1 output Non Linear Artificial Neuron

Every digital image is specified by the number of pixels associated with the image. Each pixel in a gray-level image is described by an intensity of the image at that point. A gray-scale image has no color information. Therefore, very pixel in a gray-scale image has different shade of gray which is commonly represented by 8(Unsigned integers of 8 bits). With limited memory space, it becomes useful to compress the digital image so that it occupies less memory and also becomes easier to share over a medium such as the internet. tif format grayscale image (256 x 256) has been used to demonstrate the technique.

The weight are the random numbers which will be in the range of -1 to +1, after multiplying with the input will get the weighted input. These weighted inputs are then added together along with the bias value which inhibits or exhibits the output value depending on the pre-set threshold value in the function which is a sigmoid function.

4. Implementation

The implementation of image compression and decompression will follows the bellow methods.

The input image that has been used for compression purposes is a 256 x 256 image, taken in a tif format. This image can be broken into blocks of size 8 x 8. There will then be 64 pixels per block. Totally, there will be [32 x 32] = 1024 blocks. The 64 pixels in each block then becomes the input vector to the neural net. The main idea in using a neural network to compress images is to code these 64 coefficients using a reduced number of coefficients (reduce the dimension of the data) as shown in the figure 3.

An input layer with 64 dimensions and a hidden layer with 8 (any number less than 64) dimensions, for example, means that the 64 pixels of the image (8 x 8) block which is applied to the input layer has been transformed into 8 coefficients in the hidden layer, by adding the weighted matrix along with the bias matrix.

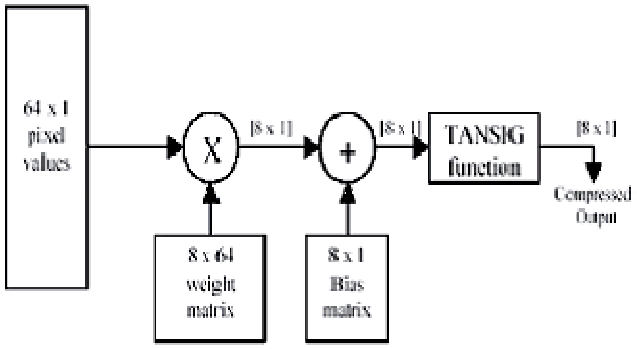

Figure 3: Image Compression Block Diagram

Then, one could again use an output layer which has 64 dimensions to recover the original 64 pixels. The basic idea here is to learn the identity mapping or rather associative mapping which means the output of the neural net is the same as its input. Thus, with a 64 dimensional input layer, an 8 dimensional hidden layer, and a 64 dimensional output layer – the neural network has been used for image compression. The decompression method is exactly opposite to that of the compression shown in the figure 4.

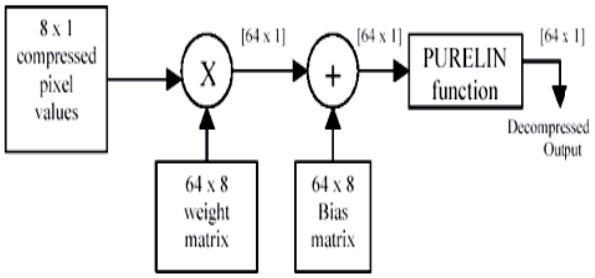

Figure 4: Image decompression Block Diagram

5. Experimental Results and Discussion

The RTL structure of the neuron network consist of both compression and decompression is as shown in the figure 5.

Figure 5: RTL structure of neural network

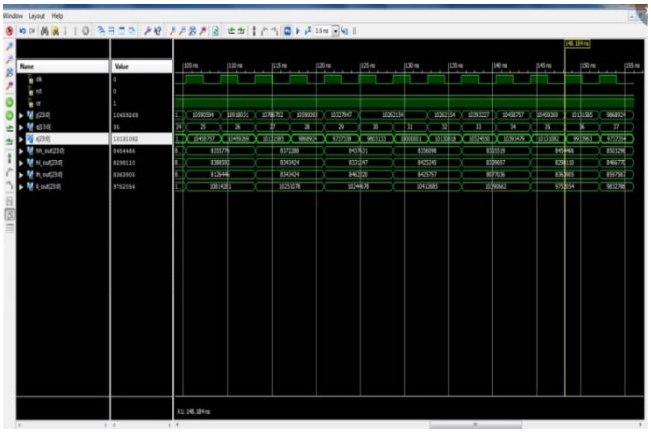

Figure 6: simulation result

Simulation result is as shown in figure 5 which is obtained by writing the code in verilog –HDL, synthesizing the code in Xilinx 12.0 & simulating its test bench using ISim .here this code is synthesized at the clock maximum frequency of 57.783Mhz (minimum period- 17.306ns) andwith a delay of 17.306ns.the image which we obtained in simulation is not exactly same as the compression is a lossy method, up to 70.34% of compression is achieved.

6. Hardware Implementation

For the hardware implementation here we have used vertex 2E pro FPGA kit, & VGA interface for viewing of the image. This is done with the use of Xilinx 10.0for theplacement and implementation process. Hardware implementation report is as shown below in figure 6 & figure 7 shows the compressed image.

Device Utilization Summary (estimated values) [-]			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1981	4656	42%
Number of Slice Flip Flops	1175	9312	12%
Number of 4 input LUTs	3546	9312	38%
Number of bonded IOBs	254	232	109%
Number of MULT18X18SIOs	20	20	100%
Number of GCLKs	1	24	4%

Figure 6: hardware report

Figure 7: (A) image used for compression,
(B): Image after compression.

7. Conclusion

The digital implementation of image compression and decompression has been done using back propagation algorithm, keeping the quality of the image considerably same even after transmitting via noisy channel. Thus the data transmission can be done with less bandwidth, power, and storage space etc. As the future work we can implement digitally fast training algorithms for the artificial neural networks and also we can implement other training algorithms to train the ANN digitally.

8. Acknowledgement

I would like to express my sincere gratitude to Mr. K Chandrashekarappa for his guidance in the course of my research and also for his generous discussion and help.

References

[1] Simon Haykin, Neural Networks: A Comprehensive Foundation (2nd edition), Prentice-Hall Inc., 1999.
[2] Pachara V Rao," Image compression using Artificial Neural Networks", 2010 second conference on Machine Learning and Computing.
[3] DP Dutta, "Digital Image Compression using Neural Networks", 2009 International Conference on Advances in computing, Control, and Telecommunication Technologies.
[4] S. Anna Durai, E. Anna Saro, "Image Compression with Back-Propagation Neural Network using Cumulative Distribution Function",pp185-189 International Journal for Applied Science and Engg. Technology
[5] S. Majumder, Md. A. Hussain, "A comparative study of image
[6] compression techniques based on SVD, DWT-SVD and DWTDCT"
[7] CI2.2 pg 500-504 at International Conference on Systemics, Cybernetics, Informatics (ICSCI-2008) under Pentagram Research, Hyderabad held 2-5 January 2008.
[8] J. Jiang, "Image Compression with Neural Networks-A survey", Signal processing: Image commn, vol.14, no.9,pp. 737-760,1999.

Author Profile

Mr. Syed Aseem Ahmed: currently pursuing M. Tech in the stream of VLSI Design & Embedded Systems from Bangalore Institute of Technology under VTU, Completed my B.E graduation in ECE from Sri Bhagawan Mahaveer Jain college of Engineering, Bangalore under VTU, India

Mr. K Chandrashekarappa is currently working as an Associate professor in Electronics & Communication dept. Bangalore Institute of Technology, Bangalore.