http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 6 (8): 1252-1257, 2007
ISSN 1812-5638

Power Efficient Multilayer Neural Network for Image Compression

[1]K. Venkata Ramanaiah, [2]K. Lal Kishore and [1]P. Gopal Reddy
[1]Department of Electronics and Cimmunication Engineering, Narayana Engineering College, Nellore, India
[2]JNTU Kukatpally, Hyderabad, India

Abstract: Multi layered neural network architecture is proposed for compression of high-resolution images. The architecture considered has 64-4-64 structures, which achieves 100% compression. The proposed network is initially trained with preselected data sets for accurate training. Feeding the scaled error back into the network at specific points, reduces the training time and reduces redundancy in weight storage. Results show the performance superiority of the network as compared with JPEG compression standard. Inserting noise on the compressed sets of data tests the network performance. Reconstructed image is compared with original image using SE and number of bits per pixel. The proposed architecture models array multipliers, carry save adders, global LUTs (Lookup Tables) for neuron construction. Calculated weights are stored in memory and read whenever required for signal processing. The control unit developed manages the data processing activity saving 20% of power when compared to direct form network structures. Power efficient architecture is developed for hardware implementation. A modification in the proposed architecture enhances the compression by 100% with sum-squared error of about 47.36.

Key words: Artificial neural network, image compression, multilayer network, VHDL, power aware design

INTRODUCTION

One of the difficulties encountered in image processing is the huge amount of data used to store an image. Transmitting the raw images consumes more bandwidth. Thus, there is need to limit the resulting data volume, for storage and transmission. Image compression techniques aim to remove the redundancy present in data in a way that makes image reconstruction possible. It is necessary to find the statistical properties of the image to design an appropriate compression technique, the more correlated the image data are and the more data items can be removed. The transform-based coding techniques and in particular the block transform coding, have proved to be the most effective in obtaining large compression ratios while retaining good visual quality. Recently, neural networks have proved to be useful in image compression because of their parallel architecture and flexibility (Dony and Haykin, 1995).

Artificial Neural Networks (ANN) has been used for image compression in cases where the results are very difficult to achieve by traditional analytical methods. There have already been a number of papers published applying ANN to image compression (Cottrell and Munro, 1988; Sonehara *et al*., 1989; Namphol *et al*., 1996; Russo and Real, 1992). It is important to emphasize, although there is no sign that neural networks can take over the existing techniques, research on neural networks for image compression are still making some advances. Possibly in the future this could have a great impact to the development of new technologies and algorithms in this area.

One approach, which successfully utilizes neural networks in transform-based image compression, is what is called auto-associative transform coding. This study started by Cottrell and Munro and was restricted to two-layer networks. In this study, the technique is extended to multi-layer networks. The main goal of this study is to investigate, design and implement power aware multilayered neural network structure with modified learning algorithms that can give the best results compared with standard transform coding techniques. It is demonstrated that the proposed method gives higher compression ratios for the same quality. The paper concludes by suggesting certain modifications to the proposed method with documented comparisons.

AUTO ASSOCIATIVE TRANSFORM CODING

In transform coding, an image is subdivided into non-overlapping blocks of n×n pixels. Each block can be considered as a N-dimensional vector, N = n×n, in N-dimensional space. In transform coding the input image is reordered this set of vectors into another

Corresponding Author: K. Venkata Ramanaiah, Department of Electronics and Cimmunication Engineering, Narayana Engineering College, Nellore, India

Fig. 1: Input reordering (4×4 converter to 16×1)

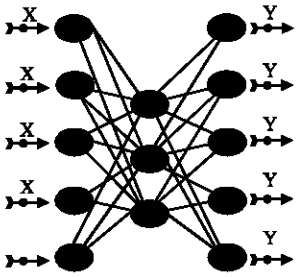

Fig. 2: Two layer feed forward neural network

M-dimensional space (M<N) such that the inverse transformation to the original N-dimensional space leads to minimum deviation from the original. For example, 64×64 image data vector is reordered to 16×256 vector by grouping 4×4 blocks of the original image as shown in Fig. 1.

For linear transformation coding, 16×256 reordered input image is transformed to 4×256 using input weight matrix of dimension 4×16. Each column is transformed using the input layer. The compressed vector y can be calculated as Y = W×P (4×16*16×1 = 4×1). Y compressed vector is transformed into P′ = W×Y (16×4 * 4×256 = 16×256) using weight vector compressed vector is transformed into P′ =W×Y (16×4 * 4×256 = 16×256) using weight vector of 16×4. The problem here is to find the appropriate W that minimizes the deviation between P and P′. Cottrel and Munro (1988) and Sonehara *et al.* (1989) used a two-layered neural network model with the number of neurons in the first layer smaller than that in the second layer (Fig. 2). The model is trained with vector set p as an input and as an output at the same time using any standard supervised neural network training techniques such as back propagation method. After training, (i.e., the error between the input p and the output P′ is minimized in the mean square sense) the resulting first layer weight matrix can be taken as W and the resulting second layer weight matrix will be W′. Exploiting the generalization capability of the neural network, which allows the same network to be used for untrained data, the same W and W′ can be used for untrained images.

Namphol *et al.* (1996) used cascaded compressors of the previous type (i.e., the output image of the first compressor is used as the input to a second compressor which is trained independently from the first compressor) and a high compression ratio with good quality were reported. Russo and Real (1992) developed a fast training algorithm for this model and they showed that the network converges to a subspace spanned by the M principle components of the training data. Carrato *et al.* (1991) used non-linear elements for these two-layered networks and showed that it performs better than the linear one. Kono *et al.* (1990) developed a modification to this model. They allowed the slope of the non-linear function to be adjusted during the training phase. Carrato *et al.* (1991) method is extended for multi-layered networks, rather than just for 2-layer networks (Vilvovic, 2006).

USING MULTI-LAYERED NETWORKS

It is well known in neural network science that when classes cannot be separated by a hyper plane, the single layer perceptron is not appropriate (Vilvovic, 2006). Multi-layer perceptrons overcome many of the imitations of single-layer perceptrons. Multi-layer perceptrons are feed-forward nets with one or more layers of nodes between the input and output nodes. With multiple data sets considered to train the network to obtain optimum weights, two layered network may reach a local minima and may get saturated and give erroneous results, instead if the network is extended, the chances of reaching local minima gets reduced and the performance of the network improves. The discussion on the classification and approximation capabilities of the multi-layer neural network is based on the fact that all the layers are trained simultaneously. As the complexity of the network increases with introduction of one more layer to the two-layer network, the training time increases. In order to reduce the training time, the property adopted in back propagation is modified. The mean square error between p and P′ is :

$$E = \frac{1}{M}\sum(p - p')^2 \quad (1)$$

Modified error propagation: The error obtained is suitably scaled and propagated backwards, the weights are adjusted based on the scaled weights rather than using them as it is as discussed into avoid saturation. The algorithm is tested on multiple data sets.

Proposed architecture: The operation of the 2-layer neural network of Fig. 2 can be viewed as a coder and decoder, each of which is a single layer perceptron. In order to make sure that the network on training does not

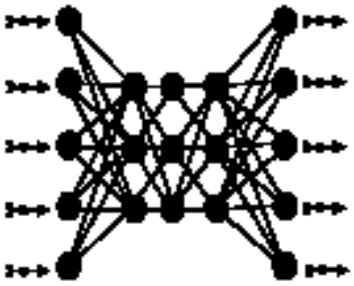

Fig. 3: Proposed multi-layered network

get saturated and reaches local minima, two more additional layers are introduced as shown in Fig. 3.

Initially three standard images were used to train the proposed 4-layers network where the ratio between the most inner layer and the output layer is 4:64. The results are:

- Compression ratio for 2-layers is 64:16 and for 4-layers is 64:4.
- Mean-squared error for 2-layers is 86.49 and for 4-layers is 47.36.

The improvement in the performance of the proposed 4-layer network over the 2-layer network increases with the increase in the compression ratio. The error observed in each of the iterations is scaled. The error is fed back only if it is more than a certain threshold h, which is continuously decreased (halved in our experiment) in the following training stage. Initially, by choosing h quite large, we will easily be able to find a starting solution satisfying the conditions. We then reset the parameter h to h/2 and retrain the network. This method has drastically reduced the training time. In this algorithm, the mean-squared error between the desired and the actual outputs with respect to the inputs to the non-linearities is minimized. This is in contrast to the standard algorithm which minimizes the mean-squared error between the desired and the actual outputs with respect to weights.

With the above sets of reading observed several sets of images having multiple properties were selected to train the network. The compressed data sets were introduced with noise before given to the reconstruction network. The performance of the network with noise introduction is found to be very good and even outperforms JPEG compression techniques. The results are shown in Fig. 4.

The pictures clearly show that the input image can be considered for high compression. The decompressed image has a lost few information when compared to the original image due to noise, which is undesirable. Therefore this decompressed image can be enhanced to

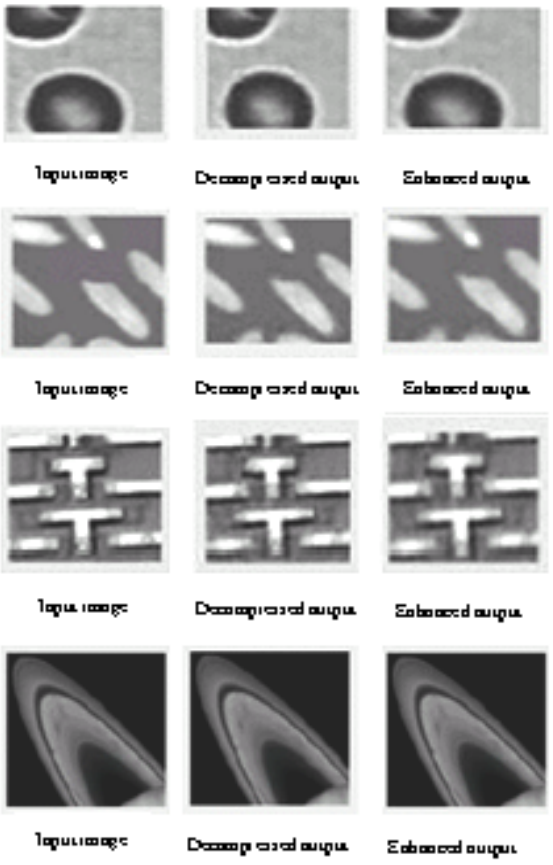

Fig. 4: Reconstructed images with noise introduced during reconstruction

nullify the above. So, more compression as well as good image quality is achieved.

The results obtained using the proposed network is compared with the JPEG compression techniques with the same set of image patterns. The results are shown in Table 1.

Table 1 describe about the type of the image and the kind of edge details i (image) has got. The mean square error comparisons between the JPEG and the proposed neural network based image compression. It shows that the mean square error is almost equal to that of JPEG. But with persistent training and implementing multi layered architecture this difference can still be reduced. The proposed Neural Network based compression takes very less memory when compared to JPEG, therefore efficient bandwidth utilization.

Neural network (N-N) over JPEG

- N-N based image compression works well for all types of images.

Table 1: Comparisons with JPEG technique

	Description		Mean square		Memory	
Image name	Type	Edges	Proposed N-N based	JPEG	Proposed N-N based	JPEG
Test 1	Pattern	Vertical	$2.25\times e^{-4}$	$1.46\times e^{-4}$	512 bytes	1.13 K bytes
Test 2	Pattern	Oblique	$4.66\times e^{-4}$	$1.66\times e^{-4}$	512 bytes	1.25 K bytes
Test 3	Pattern	Horizontal	$11.64\times e^{-4}$	$2.41\times e^{-4}$	512 bytes	1.32 K bytes
Rice	Real world	Mixed	$1.50\times e^{-4}$	$4.70\times e^{-5}$	512 bytes	806 bytes
Blood	Medical	Curved	$6.06\times e^{-4}$	$1.70\times e^{-4}$	512 bytes	970 bytes
Saturn	Satellite	Curved	$2.60\times e^{-4}$	$6.84\times e^{-5}$	512 bytes	853 bytes

- Compression ratio is fixed. No matter what the input image is, the compressed output memory is fixed.
- Therefore the bandwidth can be reserved and it rest assured that the reserved bandwidth is completely utilized and hence efficient.
- Any number of images can be fed for compression at once. Therefore parallelism is achieved.
- The architecture is reconfigurable as images of any size can be fed as input.
- The N-N based image compression technique takes very less time.
- The whole N-N based architecture can be mapped on hardware very easily.

Top level diagram of the proposed architecture: The network trained with the modified schemes modeled using VHDL and the verification is carried out with different tests of data vectors and compared with the results obtained with Matlab.

The weight matrix obtained from Matlab is scaled to integer values, 2's complement sign representation is adopted in order to preserve the sign of the weights. Suitable scaling is done on the weights to reduce the noise due to rounding and scaling. The weights are stored in memory blocks; shared by multiple MAC units this reduces area complexity. The MAC unit is constructed using array multipliers, carry save adders. This produces faster results with more space requirement. This architecture also consumes more power as at any given point of time more logic blocks are being used. In order to reduce power an FSM based control unit is developed which send sequence of control bits to enable the required MAC unit when required for processing. This control unit is synchronized with the input sample (Fig. 5). This reduces the power consumption by 20%.

Multiplier: Multipliers form a major part of a neural network. Various multiplier algorithms available are:

- Successive Addition method.
- Shift Add Method.
- Booths Multiplier.
- Array Multiplier.

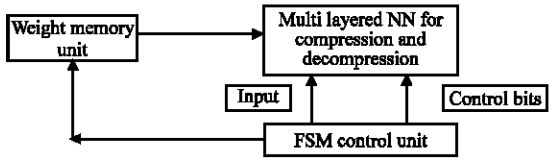

Fig. 5: Top level module of HDL model

The first three algorithms consume way too much hardware and execution time. Therefore these are not advisable for high-speed architectures. Instead we have used array multipliers with intermediate partial fraction additions being added with carry save adders.

Adders: Adders have to be used in order to add bias matrix to the multiplied output. Implementation of an adder in hardware is pretty straightforward. Having a full adder can do it.

Note: Adders are a sub-entity in the implementation of multipliers. Minimizing the number of adders in the multipliers is the key towards achieving high-speed multiplication.

Transfer functions: The transfer function in the compression layer is TANSIG PURELIN. Literally any function can be realized using these two transfer functions. The transfer function is realized using LUT approach. Only one memory unit is used per transfer function for the entire neuron sets. The data is read out in time multiplexed clock sequences.

RESULTS

Simulation results: Simulation waveforms obtained from for the CSD Multiplier done using Xilinx ISE and ModelSim.

The example shown (Fig. 6) is 40×0.6, where 40 can represent a pixel value and 0.6 can correspond to an element in the weight matrix wherein the two have to be multiplied to obtain the result. 40×0.6 gives 24 as its answer and that's precisely we obtain, when we implement our algorithm on hardware. The whole process takes only 6 Clock pulses and therefore very fast when compared to 10 to 12 clock pulses of other multipliers.

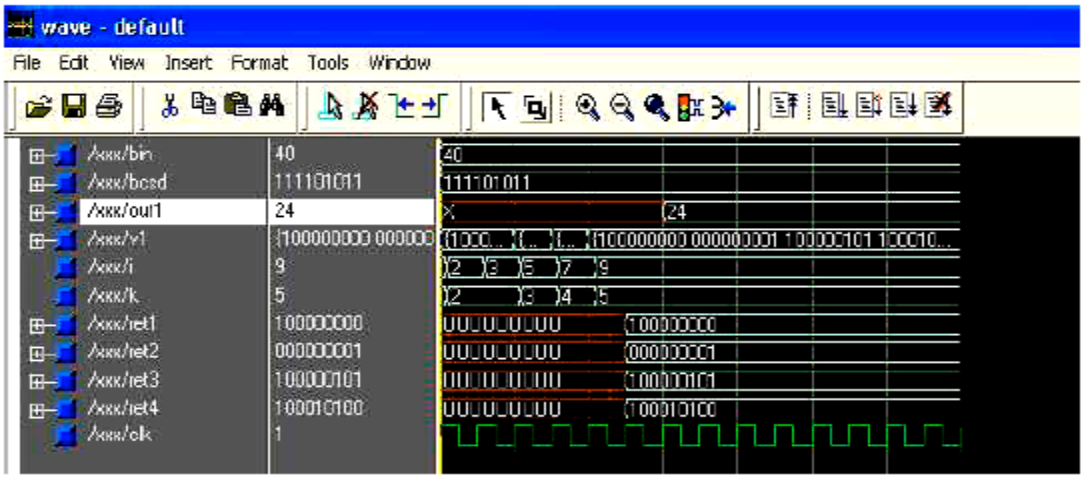

Fig. 6: Multiplication results

Hardware analysis and results: The VHDL code developed to model the MNN is tested using test vectors fed through test bench along with control bits to time share the MAC block to save power. The RTL Code is synthesized using Xilinx ISE for a single neuron; the total hardware resources required is 32% of the total resources available on Spartan IIE FPGA. The network saves power of 20% when compared with direct implementation when results are observed on Power analyzer tool from Xilinx. Further power reduction can be achieved with concepts such as clock gating and power gating techniques. The individual block required for the MAC unit are realized individually and the results are produced.

The results of multiplier implementation on FPGA are as follows:

- Time taken by one multiplier to produce output: 13 ns
- Total power consumed by one multiplier: 25 mW.
- Total hardware utilization of one multiplier on FPGA (Spartan 2E) : 4%

Note: So the total time taken for the compression of a 4×4 image block will still be 13 ns with 16 parallel multipliers, which is faster when compared to JPEG. JPEG becomes slower because the whole architecture contains many processes. One of the processes namely the DWT that can be implemented on hardware itself takes about 6000 clock pulses for processing 8×8 image block. And apart from this, there are many other processes involved such as zigzag scanning, DWT, IDWT and Huffmans coding to name a few.

CONCLUSIONS

- The major disadvantages of JPEG have overcome in Neural Network based image compression.
- Only four output values for 64 input values at the end of the compression stage.
- Greater parallelism achieved because of the implementation on FPGA.
- Modified training algorithm to reduce the training time.
- Control unit to reduce power of multilayered architecture.

Reconfigurable hardware because it works for images of any size like 64×64, 128×128, 256×256, 512×512, 1024×1024, 2048×2048 etc. (This is important as satellite images are very large).

REFERENCES

Carrato, S., A. Premoli and G.L. Sicuranza, 1991. Linear and nonlinear neural networks for image compression. Proceeding International Conf. on Digital Signal Processing.

Cottrell, G.W. and P. Munro, 1988. Principle components analysis of images via back propagation. In: SPIE. Vol. 1001. Visual Commun. Image Process., 88: 1070-1077.

Dony, R.D. and S. Haykin, 1995. Neural network approaches to image compression. Proc. IEEE., 83: 288-303.

Kono, R., M. Arai and H. Imai, 1990. Image compression using neural network with learning capability of variable function of a neural unit. SPIE Vol. 1360. Visual Commun. Image Proc., 90: 69-75.

Namphol, A., S.H. Chin and M. Arozullah, 1996. Image compression with hierarchical neural network. IEEE. Trans. Aerospace Electron. Syst., 32: (1).

Russo, L.E. and C.E. Real, 1992. Image compression using outer product neural network. Proceeding IEEE. Int. Conf. Acoust. Speech and Signal Process. San Francisco, CA., 92: 377-380.

Sonehara, N., M. Kawato, S. Miyake and K. Nakane, 1989. Image data compression using neural network model. Proceeding IJCNN, Washington DC., pp: 11-35, 11-41.

Vilvovic, I., 2006. An experience in image compression using neural networks. 48th International Symposium ELMAR, Zadar, Croatia.