

A New Compression Technique Using an Artificial Neural Network

B. Verma, M. Blumenstein and S. Kulkarni
School of Information Technology
Faculty of Information and Communication Technology
Griffith University, Gold Coast Campus,
PMB 50, Gold Coast Mail Centre, QLD 9726, Australia
{ B.Verma, M.Blumenstein }@gu.edu.au

Abstract: In this paper, we present a direct solution method based neural network for image compression. The proposed technique includes steps to break down large images into smaller windows and eliminate redundant information. Furthermore, the technique employs a neural network trained by a non-iterative, direct solution method. An error backpropagation algorithm is also used to train the neural network, and both training algorithms are compared. The proposed technique has been implemented in C on the SP2 Supercomputer. A number of experiments have been conducted. The results obtained, such as compression ratio and transfer time of the compressed images are presented in this paper.

Keywords: Neural Networks, Image Compression, Image Processing, Image Reconstruction, Direct Solution Training Method.

1. Introduction

The transport of images across communication paths is an expensive process. Image compression provides an option for reducing the number of bits in transmission. This in turn helps increase the volume of data transferred in a space of time, along with reducing the cost required. It has become increasingly important to most computer networks, as the volume of data traffic has begun to exceed their capacity for transmission. Traditional techniques that have already been identified for data compression include: Predictive coding, Transform coding and Vector Quantization.[15],[19].

In brief, predictive coding refers to the decorrelation of similar neighbouring pixels within an image to remove redundancy. Following the removal of redundant data, a more compressed image or signal may be transmitted [15]. Transform-based compression techniques have also been commonly employed. These techniques execute transformations on images to produce a set of coefficients. A subset of coefficients is chosen that allows good data representation (minimum distortion) while maintaining an adequate amount of compression for transmission. The results achieved with a transform-based technique is highly dependent on the choice of transformation used (cosine, wavelet, Karhunen-Loeve etc.) [19]. Finally, vector quantization techniques require the development of an appropriate codebook to compress data. Usage of codebooks do not guarantee convergence and hence do not

necessarily deliver infallible decoding accuracy. Also the process may be very slow for large codebooks as the process requires extensive searches through the entire codebook [15]. Following the review of some of the traditional techniques for image compression, it is possible to discuss some of the more recent techniques that may be employed for data compression.

Artificial Neural Networks (ANNs) have been applied to many problems [3], and have demonstrated their superiority over traditional methods when dealing with noisy or incomplete data. One such application is for image compression. Neural networks seem to be well suited to this particular function, as they have the ability to preprocess input patterns to produce simpler patterns with fewer components [1]. This compressed information (stored in a hidden layer) preserves the full information obtained from the external environment. Not only can ANN based techniques provide sufficient compression rates of the data in question, but 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.

There have already been an exhaustive number of papers published applying ANNs to image compression [15-19]. Many different training algorithms and architectures have been used. Some of the more notable in the literature are: nested training algorithms used with symmetrical multilayer neural networks [19], Self organising maps, for codebook generation [15], principal component analysis networks [14], backpropagation networks [18], and the adaptive principal component extraction algorithm [17].

The purpose of this paper is to present a new technique for the compression of images, using a DSM based neural network. The method for training the ANN is non-iterative and is faster than some of the more popular algorithms such as Error Backpropagation. Results are presented for ANNs trained with both a Direct Solution Method (DSM) and the Error Backpropagation (EBP) algorithm. The results convey information about the compression ratio achieved, the quality of the image after decompression, and a comparison of the transfer time of both original and compressed images over the communication line.

The remainder of the paper is divided into 5 major sections. Section 2, discusses the direct solution based neural network for image compression. Section 3 gives an overview of the experimental method used for our research. Results are presented in Section 4, a discussion of the results is supplied in Section 5, and conclusions are drawn in Section 6.

2. Direct Solution Method based Neural Network

In this section we present a brief description of a DSM for training a neural network. This technique uses a multi-layer perceptron with a single hidden layer and has been described in detail in [13]. The technique's main objective is to first convert non-linear outputs into linear ones and establish a linear system of equations for the output layer (Figure 1). The Modified Gram-Schmidt method is used to solve the linear system of equations.

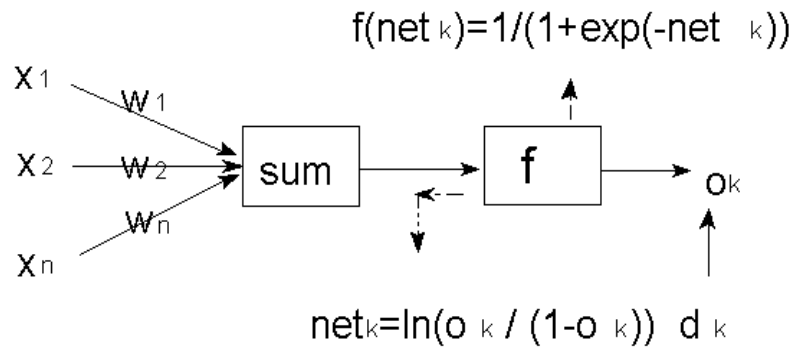
2.1 Neural Network Architecture

Let n be the number of neurons in the input layer, m the number of neurons in the output layer, N_l be the number of neurons belonging to the l th layer and $o_k^{(l)}$ be the output of the k th neuron of the l th layer, then the computation performed by each neuron can be expressed as:

$$net_k^{(l)} = \sum_{j=1}^{N_{l-1}} w_{kj}^{(l)} o_j^{(l-1)} \quad (1)$$

$$o_k^{(l)} = f(net_k^{(l)}) \quad (2)$$

where $net_k^{(l)}$ is the weighted sum of the k neurons of the l th layer, $w_{kj}^{(l)}$ is the weight by which the same neuron multiplies the output $o_j^{(l-1)}$ of the j th neuron of the previous layer and $f(\cdot)$ is a nonlinear bounded function, often the sigmoid function.



$$net_{ij} = \ln((d_{ij})/(1-d_{ij})) \quad (3)$$

where

net_{ij} = the value of net for the i th neuron and j th pair in the output layer.

d_{ij} = desired value d_{ij} of the i th neuron and j th pair in the output layer.

Using (2), Equation (3) becomes

$$w_{1i} * x_{1j}^1 + w_{2i} * x_{2j}^1 + \dots + w_{hi} * x_{hj}^1 = net_{ij}^1 \quad (4)$$

where

w_{hi} = the value of a weight from neuron h in the hidden layer to neuron i in the output layer.

x_{hj}^1 = the value of output for neuron h of pair j in the hidden layer 1.

Using equation (4) given above, for each neuron in the output layer we can write a linear system

$$Xw = net \quad (5)$$

where

net is $p \times 1$, w is $h \times 1$, $p \geq h$ and X is $p \times h$ with rank h .

The weight vector (w) is calculated using the Modified Gram-Schmidt method [13].

2.2 Training the weights of the hidden layer

The image is “compressed” in the hidden units of the ANN (Figure 1). The weights of the hidden layer are in fact set to random values. The weights used are any small, real values except zero. The reason for assigning small weights is to achieve better generalisation. The weights cannot be zero, because this will result in producing identical input vectors for the output layer, and therefore reducing the chances of finding the output layer’s weights.

2.3 Training the weights of the output layer

The weights of the output layer play an important role in ANNs because they are directly connected to the output layer. The weights (unknowns) are determined by solving a system of linear equations, which may then be used for “decompression” of the image. The equations are solved using a Modified Gram-Schmidt method. This method is stable and requires less computing power than other existing algorithms [13]. It can also comfortably solve an overdetermined system of equations (more training pairs than hidden units), which occurs regularly in image compression.

A brief description of the technique is shown below:

Step 1 : Consider a Multilayer Perceptron (MLP) with a single hidden layer.

Step 2 : Initialise the weights of the hidden layer to small random values.

Step 3 : Present the input vectors (8x8 windows) and desired output vectors (8x8 windows).

Step 4 : Develop a linear system of equations for the output layer.

- Convert the output nonlinear activation function into a linear output using Equation (1).
- Using Equation (2), Develop a system of equations as shown in Equation (5).

Step 5 : Calculate the weights of the output layer using the Modified Gram-Schmidt method (5).

Step 6 : Repeat step 4 through 6 for each neuron in the hidden layer.

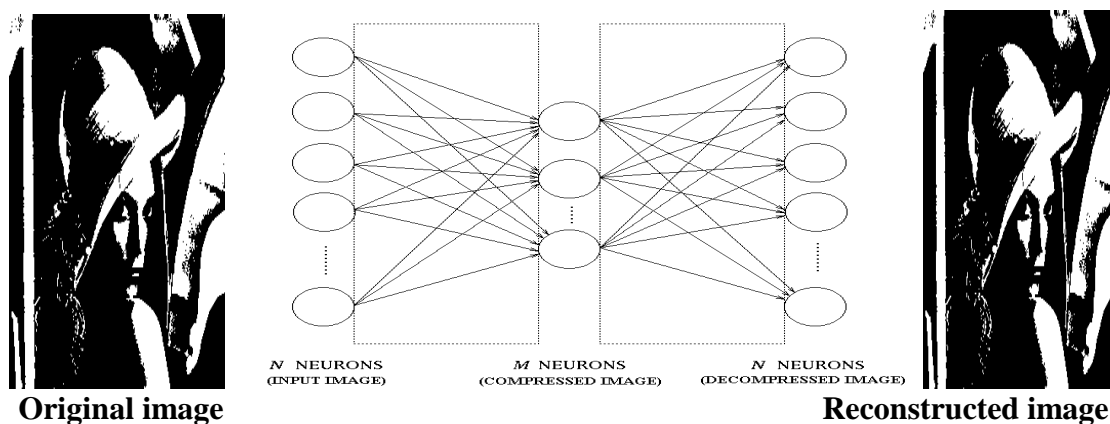


Figure 1. Compression and Decompression of Image within the ANN

3. Technique for image compression and decompression (reconstruction)

Many steps must be taken before an image can be successfully compressed using either conventional or intelligent methods. The steps proposed for image compression are as follows: 1. Image Acquisition, 2. Preprocessing, 3. Segmentation of the image, 4. Preparation of training pairs, 5. Exclusion of similar training pairs, 6. Compression of the image using an ANN trained by DSM and EBP, and 7. Reconstruction of the image.

Step 1. Image acquisition

The images were scanned using a Macintosh Flatbed scanner and a Macintosh Personal Computer running the OFOTO software package. The images were saved in Tagged Image Format (TIF). They were then easily moved across PC platforms to an IBM compatible machine running Windows 95. The Paint Shop Pro package was used to convert the TIF images into Bitmap (BMP) type images. The images were then binarised, so they could exist in a solely black and white (monochrome) format. This was achieved by performing a “nearest colour” thresholding method, which ensured that each grey level of the original image could be matched to the closest of the two binary colours. Binarisation was performed so that each image could be reduced in size, hence speeding up transfer for experimentation. Binarisation was also performed so that reduction of similar training pairs could be more easily executed in further steps.

Step 2. Preprocessing

Following binarisation each bitmap image was converted into ASCII “1s” and “0s”. This was performed so that segmentation and other preprocessing techniques could be executed more easily. This step was performed using an already implemented program written in ANSI C. A simple segmentation program was implemented in C to segment larger images into smaller blocks ready for use in conjunction with a classification method.

Step 3. Segmentation of the image

The image was then segmented into smaller images or windows using a program implemented in ANSI C. Firstly the program accepted the window size required for segmentation, it then found the total size of the image. If the image was not equally divisible into the window segments specified, the image was padded with zeroes so it could be properly divided. Segmentation was very important so as to limit the number of inputs into the ANN and to allow for the exclusion of redundant training pairs in further steps.

Step 4. Preparation of training pairs

After the larger image was broken down into smaller more useable windows, it was necessary to alter them into a form ready for use with the ANN. A file was prepared where each window was written as two identical vectors to form a training pair. In other words, the first vector would be the input vector and the second vector would be the desired output.

Step 5. Exclusion of similar training pairs

As many of the images tested were extremely large, there were many similar training pairs after segmentation. To eliminate this redundant information, a program was used to search the training file for similar training pairs (i.e. similar binary vectors) and delete them. This not only reduced the number of redundant training pairs, but reduced training time of the ANN.

Step 6. Compression of the image using an ANN trained by DSM and EBP

The DSM based neural network from Section 2 along with the EBP [13] algorithm were compared for the task of image compression. For each experiment, many different images were used for training the ANN and for compression/decompression. We transmitted the outputs of the hidden layer along with the weights of the output layer to the remote site for image decompression. Our system was specifically designed to deal with similar images which needed to be uploaded/downloaded on a regular basis. We have therefore made no provision for compression or decompression of images not used in training. However, this may be the focus of an extension of our work in the future.

Step 7. Implementation of a program for the reconstruction of the image

Finally, the output produced by the ANN in the form of decompressed vectors or windows, was reconstructed to produce the full original image. The ASCII "1s" and "0s" were converted into their proper graphical representation, and were compared with the original image.

4. Experimental results

The segmentation, preprocessing and classification programs were implemented and executed using the SP2 Supercomputer. The operating system was AIX and the chosen language for implementation was C. The experiments were conducted using diverse images, varying in size.

At first, preliminary experiments were conducted to test our system. Following these experiments, two neural-based methods were tested and compared. A Direct Solution Method was used for training, following this the Error Backpropagation algorithm was used.

In preliminary experiments as shown in Section 4.1, the binarised mammogram image was reduced to windows of size 16x16. However, for all other experiments detailed in Tables 1 and 2, window sizes of 8x8 were used for experimentation.

The times taken for transferring original and compressed images were found following transfer of files via FTP from our local site to a remote site. These times were manually noted for each image.

4.1 ANN Architecture

The architecture of the ANN was a simple single layer feedforward network. For both DSM and EBP experiments image segments (or windows) were fed into the input layer of the ANN to train it. The ANN received many different images for training and testing.

4.2 Preliminary experiments

For some preliminary experiments, we used binarised mammograms to test our system. Figures 2(a) and (b), show original and reconstructed mammogram images. The dimensions of the above mentioned images were 336x416 (reduced to smaller windows with 16x16 dimensions). The number of hidden units used in experimentation was 100.

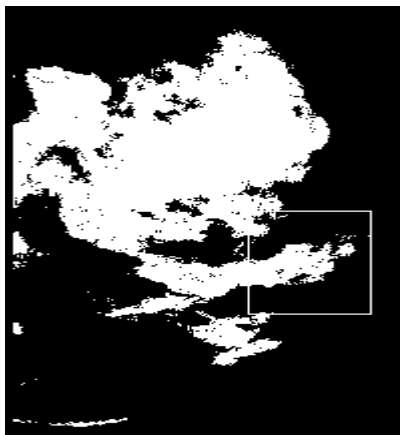


Figure 2(a) Original Image

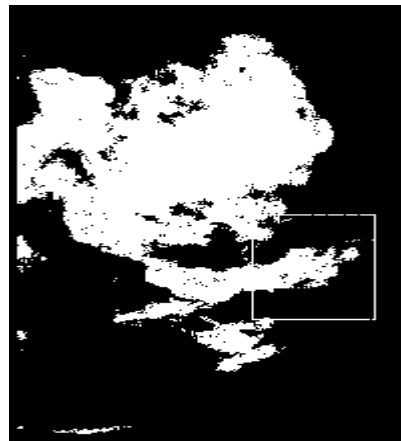


Figure 2(b) Reconstructed Image

4.3 Experiments using DSM

To test the compression performance of the DSM, many images were used. Table 1 shows the performance of the DSM for compression of various images. Some of the images used for testing the DSM's compression and decompression capabilities are shown in Figures 3-5. As can be seen, all images suffered little or no deterioration in quality. A compression ratio of 64:20 or approximately 3:1, was achieved for all images tested. This was possible when the ANN contained only 20 units in the hidden layer.

Table 1. Performance of DSM with a variety of images

Image	Dimensions	Hidden Units	Total Training Pairs	Reduced Training Pairs	Time Required to Send Original Image (secs)	Time Required to send Compressed Image (secs)
Lena	256 x 256	20	1024	404	0.32730	0.15570
Bell	128 x 144	20	288	115	0.03953	0.01938
Camera	136 x 216	20	459	210	0.02232	0.01070
Church	184 x 152	20	437	121	0.02219	0.01091
Cards	88 x 240	20	330	90	0.01747	0.00840
Tree	144 x 216	20	486	109	0.03869	0.01084

4.4 Experiments using the EBP algorithm

The EBP algorithm was then compared with the DSM training algorithm. After decompression, it was found that the images were almost identical to the original images, with little or no deterioration in quality. The recognition rates are listed in Table 2. Also listed, are transmission times of full and compressed images over a network. Some of the images tested (before and after compression) are displayed in Figures 3-5. As with the DSM, the best compression ratio (64:20 or 3:1) was achieved with an ANN containing 20 hidden units.

Table 2. Performance of EBP with a variety of images

Image	Hidden Units	Learning Rate	Momentum	No. of Iterations	Total Training Pairs	Reduced Training Pairs	Time Required to Send Original Image (secs)	Time Required to send Compressed Image (secs)
Lena	20	0.2	0.2	200	1024	404	0.3273	0.1557
Bell	20	0.2	0.2	200	288	115	0.03953	0.01938
Camera	20	0.2	0.2	200	459	210	0.02232	0.01070
Church	20	0.2	0.2	200	437	121	0.02219	0.01091
Cards	20	0.2	0.2	200	330	90	0.01747	0.00840
Tree	20	0.2	0.2	200	486	109	0.03869	0.01084



Figure 3(a) Original Image



Figure 3(b) Using DSM



Figure 3(c) Using EBP

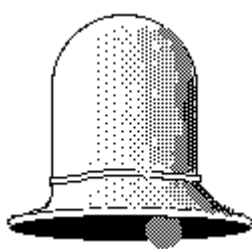


Fig 4(a) Original figure

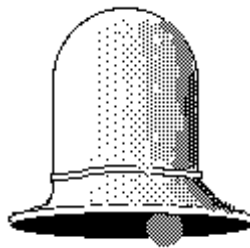


Fig 4(b) Using DSM

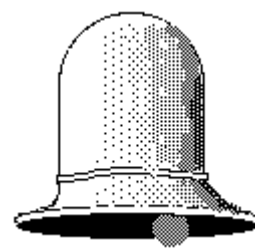


Fig 4(c) Using EBP

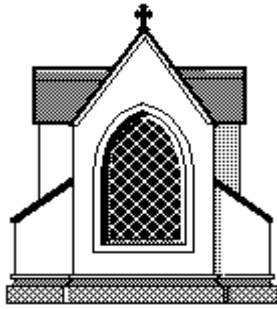


Fig 5(a) Original figure

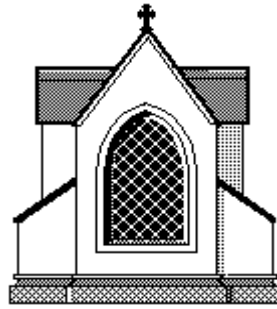


Fig 5(b) Using DSM

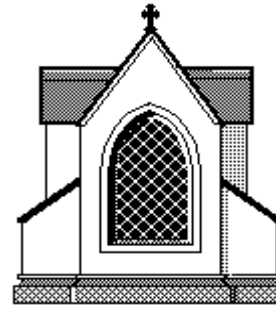


Fig 5(c) Using EBP

5. Discussion of Results

The results in Section 4 demonstrate the compression capabilities of a DSM based neural network technique and a comparison with the EBP algorithm. As can be seen from Tables 1 and 2, both methods performed quite well. The recognition rates in both tables reflected how accurately the images were decompressed. Both techniques showed very little or no deterioration of the image quality when using an appropriate number of neurons in the hidden layer of the ANN. As the number of hidden units was lowered, the greater the deterioration in the decompressed images (Deteriorated images are not shown in this paper). However, the lower the number of hidden units, the better the compression ratio of the image. Overall, it was found that the DSM trained the neural network with greater speed. This could be attributed to the fact that it is not an iterative process like EBP. Although training time is not recorded in this paper, the reader is referred to an earlier paper [13] detailing a comparison between the EBP and DSM algorithms. The paper contains many experimental results showing the DSM's superior training time for a variety of problems.

As could be seen in Tables 1 & 2, the time to transfer the compressed images was clearly less than sending a whole image. In most cases it took half the time and in some cases a third of the time, to send the entire image.

6. Conclusion

In this paper, we have presented a Direct Solution Method based technique and we have compared it with an existing intelligent technique (EBP) for image compression. Our approach, using a fast training algorithm (DSM), has been implemented in the C programming language on the SP2 Supercomputer. We segmented, compressed, decompressed and reconstructed various images using our proposed method. Results showed that a superior training time and compression could be achieved with our method.

References

- [1] Jiang, J., (1996), A Neural Network Design for Image Compression and Indexing. International Conference on Artificial Intelligent Expert Systems and Neural Networks, Hawaii, USA, pp 296-299.

- [2] Joutsensalo, J., (1994), Non Linear Data Compression and Representation by Combining Self Organising Map and Subspace Rule, IEEE International Conference on Neural Networks Vol-2, pp 637-642.
- [3] Blumenstein, M., (1996), The Recognition of Printed and Handwritten Postal Address using Artificial Neural Networks, Dissertation, Griffith University, Australia.
- [4] Nelson, M., (1991), The Data Compression Book, M & T Publishing Inc.
- [5] Gelene R., and Sungar, M., (1994), Random Network Learning and Image Compression using Artificial Neural Networks, Vol-6, IEEE, pp 3996-4001.
- [6] Sicuranza, G.L, Ramponi G, and Marsi S., (1990), Artificial Neural Network for Image Compression, Electronic Letters, pp 477-479.
- [7] Ash, T., (1989), Dynamic Node Creation in Backpropagation Networks, Neural Networks, pp 365-375.
- [8] Kenue S. K., (1992), Modified Back-Propagation Neural Network with Applications to Image Compression, SPIE Vol.1709, Applications of Artificial Neural Networks, pp 394-401.
- [9] Carbonara, M., Fowler J., and Ahalt, S., (1992), Compression of Digital Video Data using Artificial Neural Network Differential Vector Quantization, SPIE Vol. 1709, Applications of Artificial Neural Networks, pp 422-433.
- [10] Min, K., and Min, H. (1992) Neural Network Based Image Compression using AMT DAP 610, SPIE Vol. 1709, Application of Artificial Neural Networks, pp 386-393.
- [11] Panchanathan, S., Yeap, T., and Pilache, B., (1992), A Neural Network for Image Compression, SPIE Vol. 1709, Application of Artificial Neural Networks, pp 376-385.
- [12] Hambaba, M., Coffey B., and Khemlani, N., (1992), Image Coding using a Knowledge based Recognition System, SPIE Vol. 1709., Application of Artificial Neural Networks.
- [13] Verma, B., (1997), Fast Training of Multilayer Perceptrons, IEEE Trans. on Neural Networks, Vol. 8, No. 6, pp 1314-1321.
- [14] Oja, E., (1991), Data Compression, Feature Extraction, and Autoassociation in Feedforward Neural Networks in Artificial Neural Networks, (Eds) Kohonen et al., Elsevier Science Publishers, pp 737-745.
- [15] Dony, R. D., and Haykin, S., (1995), Neural Network Approaches to Image Compression, Proceedings of the IEEE, Vol. 23, No. 2, pp 289-303.

- [16] Carato, S., (1992), Neural Networks for Image Compression, Neural Networks Advances and Applications 2, (Ed) Gelenbe E., Elsevier Science Publishers, pp 177-198.
- [17] Kung, S. Y. et al., (1994), Adaptive Principal Component Extraction (APEX) and Applications, IEEE Transactions on Signal Processing, Vol. 42, No. 5, pp 1202-1217.
- [18] Mougeot, M. et al., (1991), Image Compression with Backpropagation: Improvement of the Visual Restoration using Different Cost Functions, Neural Networks, Vol 4., pp. 467-476.
- [19] Namphol, A. et al., (1996), Image Compression with a Hierarchical Neural Network, IEEE Transactions on Aerospace and Electronic Systems, Vol. 32, No. 1, pp. 327-337.