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 to 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. 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. Typical algorithms that have been developed are: Huffman coding, Arithmetic coding, Shannon-Fano method, Statistical modeling and their variations [4].

Artificial Neural Networks have been applied to many problems [3], and have demonstrated their superiority over classical 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.

The purpose of this paper is to present a new technique for the compression of images, which uses 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 a 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 Neural Network that uses DSM for training of the hidden layer.
Step 1: Consider a two layer 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 function.
   - Develop a system of equations.
Step 5: Calculate the weights of the output layer.
Step 6: Repeat the step 4 through 6 for each neuron in the hidden layer.

2.1 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 generalization. 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.

2.2 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 equation (more training pairs than hidden units), which occurs regularly in image compression.

![Original image and Reconstructed image](image.png)

**Figure 1. Compression and Decompression of Image within the ANN**

3. Algorithm 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. It
was necessary to export the images yet again to another software package: Paintbrush. The images were then converted to a solely black and white (monochrome) format. This was necessary for binarisation to be completed in further steps.

**Step 2. Preprocessing**
Binarisation was one of the first techniques employed. It is a technique that converts the black and white pixels of a monochrome BMP-type image into “1s” and “0s” respectively. In this form, segmentation and other preprocessing techniques can be executed much more easily. Binarisation was performed using an already implemented program written in ANSI C. Binarisation was very important, for further preprocessing steps, such as segmentation. 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 is then segmented into smaller images or windows. This step is 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 is broken down into smaller more useable windows, it is necessary to alter them into a form ready for use with the ANN. A file is prepared where each window is 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 is used to search the training file for similar training pairs and delete them. This not only reduces the number of redundant training pairs, but reduces 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 are compared for the task of image compression.

**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, is reconstructed to produce the full original image. The binarised pixels are converted into their proper graphical representation, and are compared with the original image.

**4. Experimental Results**
The segmentation, preprocessing and classification programs were implemented and run using the SP2 Supercomputer. The operating system was UNIX 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 Network methods were tested and compared. A Direct Solution Method was used for training, following this the Error Backpropagation algorithm was used.

**4.1 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.
4.2 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 3:1, was achieved for all images tested. This was possible when the ANN contained only 20 units in the hidden layer.

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

4.3 Experiments using the EBP Algorithm
The EBP algorithm was then used to compare 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>
<thead>
<tr>
<th>Image</th>
<th>Dimensions</th>
<th>Hidden Units</th>
<th>Total Training Pairs</th>
<th>Reduced Training Pairs</th>
<th>Time Required to Send Original Image (secs)</th>
<th>Time Required to Send Compressed Image (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>256 x 256</td>
<td>20</td>
<td>1024</td>
<td>404</td>
<td>0.32730</td>
<td>0.15570</td>
</tr>
<tr>
<td>Bell</td>
<td>128 x 144</td>
<td>20</td>
<td>288</td>
<td>115</td>
<td>0.03953</td>
<td>0.01938</td>
</tr>
<tr>
<td>Camera</td>
<td>136 x 216</td>
<td>20</td>
<td>459</td>
<td>210</td>
<td>0.02232</td>
<td>0.01070</td>
</tr>
<tr>
<td>Church</td>
<td>184 x 152</td>
<td>20</td>
<td>437</td>
<td>121</td>
<td>0.02219</td>
<td>0.01091</td>
</tr>
<tr>
<td>Cards</td>
<td>88 x 240</td>
<td>20</td>
<td>330</td>
<td>90</td>
<td>0.01747</td>
<td>0.00840</td>
</tr>
<tr>
<td>Tree</td>
<td>144 x 216</td>
<td>20</td>
<td>486</td>
<td>109</td>
<td>0.03869</td>
<td>0.01084</td>
</tr>
<tr>
<td>Image</td>
<td>Hidden Units</td>
<td>No. of Iterations</td>
<td>Total Training Pairs</td>
<td>Reduced Training Pairs</td>
<td>Time Required to Send Original Image (secs)</td>
<td>Time Required to Send Compressed Image (secs)</td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
<td>-------------------</td>
<td>----------------------</td>
<td>------------------------</td>
<td>---------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>Lena</td>
<td>20</td>
<td>200</td>
<td>1024</td>
<td>404</td>
<td>0.3273</td>
<td>0.1557</td>
</tr>
<tr>
<td>Bell</td>
<td>20</td>
<td>200</td>
<td>288</td>
<td>115</td>
<td>0.03953</td>
<td>0.01938</td>
</tr>
<tr>
<td>Camera</td>
<td>20</td>
<td>200</td>
<td>459</td>
<td>210</td>
<td>0.02232</td>
<td>0.01070</td>
</tr>
<tr>
<td>Church</td>
<td>20</td>
<td>200</td>
<td>437</td>
<td>121</td>
<td>0.02219</td>
<td>0.01091</td>
</tr>
<tr>
<td>Cards</td>
<td>20</td>
<td>200</td>
<td>330</td>
<td>90</td>
<td>0.01747</td>
<td>0.00840</td>
</tr>
<tr>
<td>Tree</td>
<td>20</td>
<td>200</td>
<td>486</td>
<td>109</td>
<td>0.03869</td>
<td>0.01084</td>
</tr>
</tbody>
</table>

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 a large number of neurons in the hidden layer of the ANN. However, the lower the number of hidden units, the greater the deterioration in the decompressed images. Conversely, 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 it is not an iterative process like EBP.
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. Further research is currently ongoing, and more experiments will be presented in the final version of the paper.

References