Impulsive Noise Removal based on Neural Network Schemes

Prof. Dr. Turki Y. Abdalla+ Asst. Prof. Dr. Abdul-Kareem Younis*
Dr. Sarah Behnam Aziz*
+ Computer Engineering Department, Engineering College, Basrah University
* Computer Science Department, Science College, Basrah University

Abstract

Interest in neural networks as an alternative to the conventional algorithmic techniques has grown rapidly in recent years. Noise removal or noise suppression is an important task in image processing. In general, the results of the noise removal have a strong influence on the quality of the following image processing techniques. In this paper, two feed forward NN schemes have been presented for impulsive noise removal. The computation is reduced by using an artificial image in training. Results of NN schemes show high performance especially when the ratio of impulsive noise in testing are the same or greater than that of training image. The presented schemes are used for grayscale and also for truecolor.

إزالة الضوضاء النبضية باستخدام الشبكات العصبية

أ. د. تركي يونس عبد الله+ أ.م.د. عبد الكريم يونس*
د. سارة بهنام عزيز*
*جامعة البصرة / كلية الهندسة / قسم هندسة الحاسبات
*جامعة البصرة / كلية العلوم / قسم علوم الحاسبات

الملخص

إن الإهتمام بالشبكات العصبية كبديل عن التقنيات الخوارزمية التقليدية قد نمى بسرعة في السنوات الأخيرة. تعتبر إزالة الضوضاء أو تقليلها عملية مهمةُ في معالجة الصورة. بشكل عام، نتائج إزالة الضوضاء لها تأثير قوي على جودة إداء تقنيات معالجة الصور اللاحقة.
هذا البحث، يُقدم شبكتين عصبيتين ذات تغذية أمامية لإزالة الضوضاء النبضية. لقد تم استخدام صورة اصطناعية في التدريب وذلك لتقليل الحسابات. تُظهر النتائج إنجازية عالية في أداء الشبكات العصبية خصوصاً عندما تكون نسبة الضوضاء النبضية في الصور أثناء مرحلة الإختبار هي نفسها أو أعلى من تلك التي في صورة التدريب. تم استعمال الشبكات العصبية المُقدّمة لكل من الصور غير الملونة grayscale وكذلك الصور الملونة truecolor على حد سواء.

1. Introduction

Sometimes, the images that we are using are corrupted by impulsive noise. This noise may be due camera imperfections (saturation) or to a noisy transmission channel [1].

Noise suppression or noise removal is an important task in image processing. In general, the results of the noise removal have a strong influence on the quality of the following image processing techniques. The ideal noise removal must preserve the edges and the detail information into the images [2].

A wide variety of filtering algorithms have been developed to detect and remove noise, leaving as much of possible of the pure image. These include both temporal filters, and spatial filters [3]. One of the most common spatial filters is the median filter which is introduced by Tukey [4] in the 1970s to be used extensively for image noise reduction and smoothing. The median filter gives good results when it is applied to low corrupted images but with highly corrupted images, it causes image blurring and edge jitter beside the low level of effective in removing the noise.

Neural networks are increasingly being employed in various fields, including image processing, pattern recognition, medicine, speech production and recognition, and business. They constitute an information processing system that share conceptual structures and characteristics with biological neural networks [5-6].

A very important feature of neural networks is their adaptive nature, where “learning by example” replaces “programming” in solving problems. This feature makes such computational models very appealing in application domains where one has little or incomplete understanding of the problem to be solved but where training data is readily available [7].

A recent work that makes use of a multi-layer neural network shows an application in the classification of multispectral remote sensing data [8]. Another approach to segment color image based on the chromaticities of the objects using a Kohonen network which was able to discriminate the main chromaticities of the image, is proposed [9]. Also, four modular networks were constructed to simulate Kuwahara filter which is used to smooth an image while preserving the edges. Their designating is transition from a fixed, i.e. it is completely hand designed with every weight set to an optimal value, to a free type which consists of only standard feed-forward modules. These standard networks having one or two hidden layers of 1, 2, 3, 4, 5, 10, 25, 50, 100, 250 units each were used. All units used the double sigmoid transfer function [10-11].

This contribution deals with noise removal and presents an effective filter to remove high ratios of impulsive noise from grayscale and truecolor images based on feed forward neural networks. The NN is used due to its capability of generalization with a reduced number of training examples. This property allows a training that can use an artificial image instead of real image and then reduce the training complexity. This new method gives good results, both in "Signal to Noise Ratio" (SNR) and edge preservation, when applied over images with high noise ratios.

The outline of this contribution is as follows: the median filter will be discussed in section 2 and a description of the architecture of the feed forward NN, the parameters that must be set in the training process and the basic steps of learning algorithm to obtain the best results will be given in section 3. Section 4 will explain the data sets will be used in training and testing the proposed NN schemes and in section 5 the experiments performed using the proposed NN schemes will be described and compared their results obtained with those obtained by the corresponding conventional median filter. Finally, section 6 contains some remarks and conclusions.

2. Median Filter

The median filter is a nonlinear operator that arranges the pixels in a local window according to the size of their intensity values and replaces the value of the pixel in the result image by the middle value in this order [4].

In this contribution, the median filter will be applied by using two types of mask (local window); cross and square (Figure 1). Both masks are of size N×N where N = 3. The first will be mentioned it by ***Cross Median Filter*** (CMF) while the second it will be mentioned by ***Square Median Filter*** (SMF).

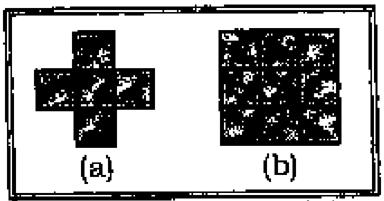


Figure 1: The mask shapes of the median filter (a. Cross b. Square).

In the truecolor image, each pixel can be described by the base colors red, green and blue (RGB). By these color triplets, all color can be created. Consequently, the color image is usually represented by triplet of matrixes RGB. Therefore, the smoothing filter will be applied on matrix of each color separately then combined the three resulted grayscale images to get the filtered truecolor image [12]. The principle is shown in figure 2.

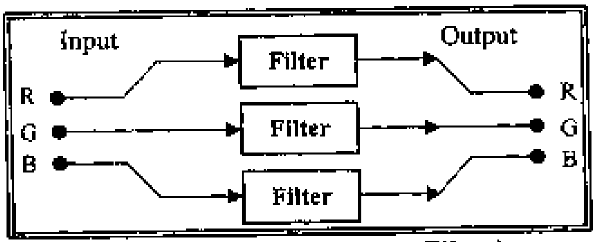


Figure 2: RGB Component Filtering

3. Neural Network

A ***Neural Network*** (NN) is a mathematical model which has a highly connected structure similar to brain cells. They consist of a number of neurons arranged in different layers (Figure 3), an input layer, an output layer and one or more hidden layers [13-18].

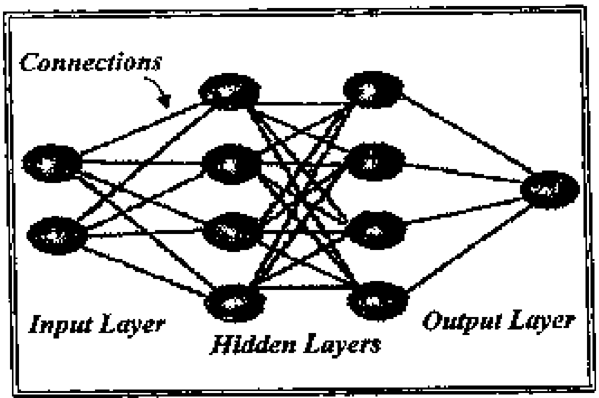


Figure 3: The Architecture of a Typical Neural Network

In this work, only three layer networks were considered since it has been shown that they have a good potential to represent linear or non linear outputs [15].

Data flow through the network can be briefly described in few steps:

1. ***From the input to the hidden layer:*** The input layer loads data from input vector X, and sends them as output O_i^1 to the first hidden layer. Thus,

$$O_i^1 = X, \qquad \forall i = 1..N^1 \qquad \textbf{(1)}$$

Where, N^1 is number of neurons in the input layer.

2. ***In the hidden layer:*** Neurons in the hidden layer receive the weighted input and transfer it to the next hidden or to the output layer using one of the transfer functions. The weighted input is calculated as follows:

$$I_j^l = \sum_{i=1}^{N^{l-1}} W_{ij}^l . O_i^{l-1} \qquad \forall j = 1..N^l \qquad \textbf{(2)}$$

Where, I_j^l is the weighted input to neuron j in layer l, W_{ij}^l is the connection weight from neuron i (in layer l-1) to neuron j and N^{l-1} is the number of neurons in the previous layer of layer l.

The output of neuron j in layer l as in the formula:

$$O_j = f(I_j^l) \qquad \forall j = 1..N^l \qquad \textbf{(3)}$$

3. At the output layer, the network output is compared to the desired (real) output, and the global error E is determined as:

$$E = \tfrac{1}{2}\sum_{k=1}^{N^p} (O_k - D_k)^2 \qquad \textbf{(4)}$$

Where, O_k is the output of the network, while D_k is desired (real), and N^p is number of learning patterns.

4. ***In the output layer:*** For each neuron, the scaled local error δ^L is computed as in equation 5.

$$\delta^L = \frac{\partial E}{\partial I} = \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial I} = (O - D) f'(I^L) \qquad \textbf{(5)}$$

Where f' is the derivative of the transfer function f.

5. ***Back propagation from the output back to the hidden layers:*** The scaled local error and weights changes are computed for each layer backwards.

$$\delta_j^l = f'(I_j^l) \sum_{i=1}^{N^{l+1}} \delta_i^{l+1} W_{ji}^{l+1} \qquad \forall l = L-1..2 \text{ and } j = 1..N^l \qquad \textbf{(6)}$$

$$\Delta W_{ij}^l = \eta \cdot \delta_j^l \cdot O_i^{l-1} \qquad \textbf{(7)}$$

$$W_{ij}^{l\,new} = W_{ij}^{l\,old} + \Delta W_{ij}^l \qquad \forall l = L..2, i = 1..N^{l-1} \text{ and } j = 1..N^l \qquad \textbf{(8)}$$

Where δ_j^l is the scaled local error of neuron j in layer l, ΔW_{ij}^l is the modification value of the connection weight from neuron i in layer l-1 to neuron j, and η is the learning rate. While $W_{ij}^{l\,new}$ and $W_{ij}^{l\,old}$ are the new and old value of the connection weight from neuron i in layer l-1 to neuron j in layer l, respectively.

In this work, two of the most commonly known heuristic approaches; momentum and variable learning rate are used [13, 19]. Where, the first one is used for biases while the second one is used for weights.

The biases update with momentum is calculated as:

$$\Delta b_j^{l^{new}} = \Delta b_j^{l^{old}} + (1 - mc) \cdot lr \cdot \delta_j^l \cdot O^{l-1} \qquad (9)$$

$$b_j^{l^{new}} = b_j^{l^{old}} + \Delta b_j^{l^{new}} \quad \begin{array}{l} for\ l = 1\ to\ 2, \\ for\ j = 1\ to\ N^l \end{array} \qquad (10)$$

Where, δ_j^l is the scaled local error of bias j in layer l and calculated as in equation 6, $\Delta b_j^{l^{old}}$ and $\Delta b_j^{l^{new}}$ are the previous and current modification values of the connection of the bias to neuron j in layer l. While $b_j^{l^{old}}$ and $b_j^{l^{new}}$ are the new and old values of the connection of the bias to neuron j in layer l, respectively.

On the other hand, an adaptive learning rate will attempt to keep the learning step size as large as possible while keeping learning stable. The learning rate is made responsive to the complexity of the local error surface. An adaptive learning rate requires some changes in the training. First, the initial network output and error are calculated. At each epoch new weights are calculated using the current learning rate. New outputs and errors are then calculated, if the new error exceeds the old error by more than a predefined ratio ***max_perf_inc*** (typically 1.04) the new weights are discarded. In addition, the learning rate is decreased by multiplying by ***lr_dec***. Otherwise the new weights and the learning rate are kept. If the new error is less than the old error, the learning rate is increased by multiplying by ***lr_inc*** [13, 20].

And the weights are updated using equations 6 – 8, but this is done only with the following conditions.

```
if $(newerror / old\ error) > 1.04$ then
        Weights are discarded
        $lr = lr * lr\_dec$
else                                                    (11)
        Update Weights
        if $newerror < old\ error$
                $lr = lr * lr\_inc$
```

Another modification can be made to improve the convergence speed of BPA. The ***Nguyen-Widrow*** algorithm is a specific modification for initialization the weights and biases of NN [21]

4. Data Sets

Two types of data sets are considered. The first is the data used in training phase, while the second is the data used in testing phase.

The training data types may be classified into two kinds either image dependent or image independent data. In the image independent kind, the samples of the training data is an artificial image generated randomly or formed from the representative building blocks of a natural image which are including the flat regions, varying from bright regions to dark regions and edge areas that may be sharp and blurred (Figure 4).

Although, this kind of training data is not commonly used where it is somehow difficult to find the suitable artificial image for the specific application, the training time becomes small since the size of the artificial image is not large.

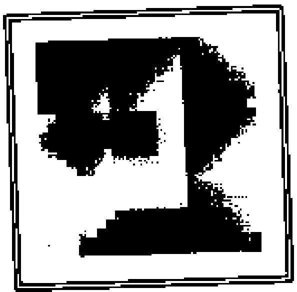


Figure 4: The Original Artificial Image

The above artificial image will be used as training data set for the proposed NN schemes in this contribution after corrupted it by impulsive noise of ratio 20% (Figure 5).

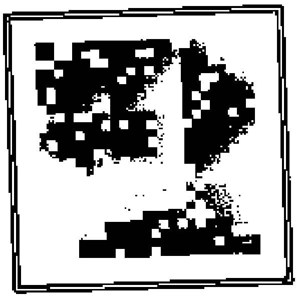


Figure 5: Corrupted Artificial Image with 20% Impulsive Noise

On the other side, the testing data set includes eight images. Four of them are grayscale images and the other four are truecolor images. The original copies of them are shown in Appendix.

5. Experiments & Results

In this section architecture of feed forward NN which consists of an input layer, a hidden layer and an output layer was formed. The number of neurons in the hidden layer is double of that in the input layer. The architecture has only one neuron in the output layer. The transfer function used in the hidden layer is tangent function while the sigmoid function is used in the output layer.

Two schemes of this architecture will be proposed. The first scheme has five neurons in the input layer that represent the noisy pixel and its four neighbors within a local window while the second scheme has nine neurons in the input layer. The neurons in the input layer of the second scheme represent the noisy pixel and its eight neighbors within a local window.

From now and upward it will be referred to the first scheme by ***Cross Neural*** (CN) since the five inputs forms a cross mask. On the other side, the ***Square Neural*** (SN) will be used to refer to the second scheme since the nine inputs forms a square mask. Figure 6 shows the schematic diagram of the NN architecture.

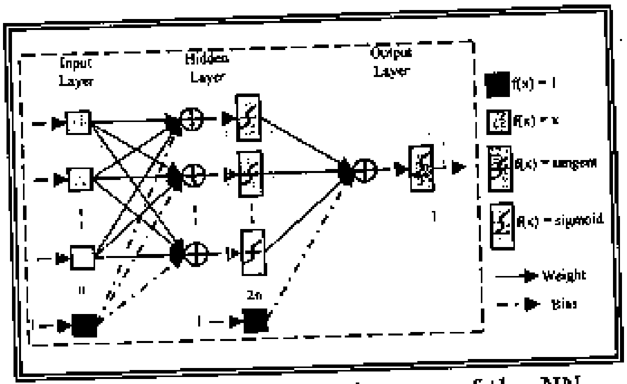


Figure 6: Schematic Diagram of the NN Architecture

5.1 Data Preparing

The patterns of training are constructed from the training artificial image. The corrupted and original 8-bit 256 gray value copies images are converted to a floating point images, with gray values in the range [0, 1]. The corrupted image will be surrounded by white pixels pad (values = 1). Then each pattern which contains the input values and the desired output value will be formed. By the same way, the patterns of testing are constructed from testing images but here the corrupted image will be surrounded by black pixels pad (values = 0). Then each pattern which contains the input values only will be formed. Then, the output of NN scheme of any test image which forms a sequence of floating point values in range [0, 1] should be converted to 8-bit 256 gray values and rearranged as matrix of size of the test image itself.

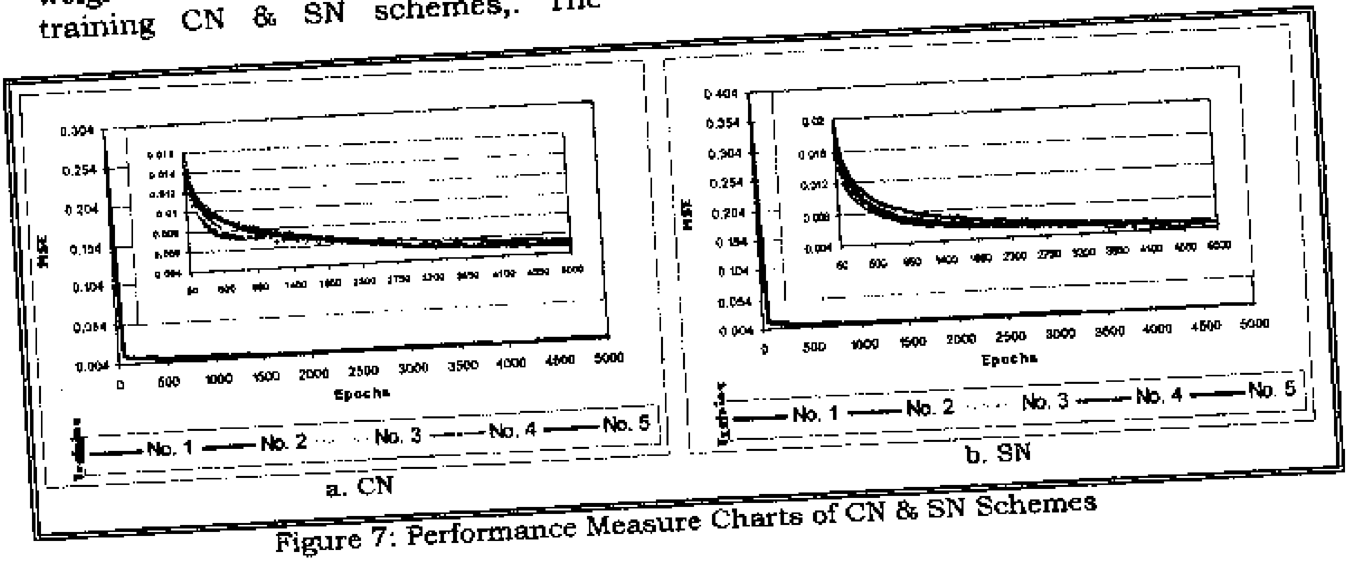
5.2 Training

An improved BPA uses momentum to biases, adaptive learning rate to weights and ***Nguyen-Widrow*** algorithm to initialize the weights and biases will be used for training CN & SN schemes,. The initial value of the learning rate is ***lr*** = 0.3 while the values of ***lr_inc***, and ***lr_dec*** were set to 1.3, and 0.1. The training is stopped after 5000 epochs.

Each of CN & SN schemes will be trained five times and the learning parameters are kept the same in all times. The final values of performance measure obtained from training CN and SN five times are listed in table 1 and figure 7 shows the performance measure charts of two NN schemes.

[illegible]	[illegible]	[illegible]
1	0.00529	0.00475
2	0.00475	0.00430
3	0.00485	0.00449
4	0.00478	0.00471
5	0.00490	0.00470

Table 1: Final Performance Measure (MSE) of CN and SN



a. CN

b. SN

Figure 7: Performance Measure Charts of CN & SN Schemes

5.3 Results

At the beginning, it is important to explain the evaluation measurement units the results relied on it.

The idea of any kind of signal processing is to achieve some kind of desired effect on the perceived information content in the signal. In image enhancement and restoration the idea is to process the data to improve illumination, or to remove defects like noise or dirt on a film. A mechanism for assessing the *damage* done to the observed picture is important to evaluate the quality of the processed output [12].

A number of simple measures can be generated to measure the ***observed error*** as follows [12].

The ***Mean Squared Error*** (MSE) is

$$MSE = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M} e(i,j)^2 \qquad (12)$$

$$e(i,j) = y(i,j) - d(i,j) \forall i = 1..N \text{ and } j = 1..M \qquad (13)$$

Where, image is of size N×M, d is the original image and y is the resulted image after corrupting or filtering. Thus the MSE is the mean of the squared error values across the entire image.

The ***Signal to Noise Ratio*** (SNR) is another popular objective measure and it has units of ***Decibels*** (dB) [9].

$$SNR = 10\log_{10}\frac{\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M} d(i,j)^2}{MSE} \qquad (14)$$

This is a ratio between the signal power, measured as the sum squared intensities in the original image *d*, and the ***noise*** power measured as the MSE of the error, *e*.

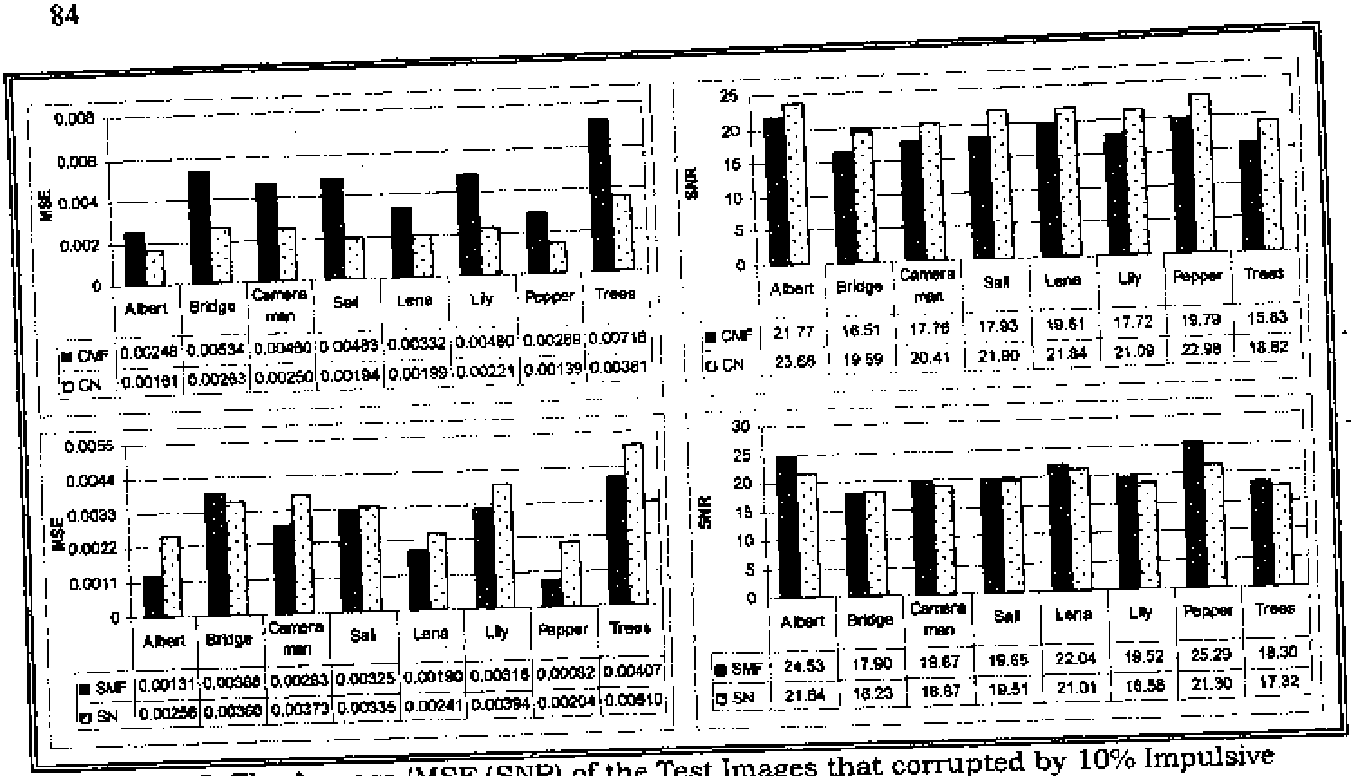
Each of equations 12 and 14 to calculate MSE and SNR are rights only for grayscale images and should be extended to be available for truecolor images.

The mean square error and signal to noise ratio for truecolor image denoted by MSE_c and SNR_c respectively are given by:

$$MSE_c = \frac{1}{3\cdot N\cdot M}\sum_{k=1}^{3}\sum_{i=1}^{N}\sum_{j=1}^{M} e(i,j,k)^2 \qquad (15)$$

$$SNR_c = 10\log_{10}\frac{\frac{1}{3\cdot N\cdot M}\sum_{k=1}^{3}\sum_{i=1}^{N}\sum_{j=1}^{M} d(i,j,k)^2}{MSE_c} \qquad (16)$$

The set of test images that are corrupted by five different ratios 10%, 15%, 20%, 25% and 30% of impulsive noise separately will be filtered by the different two NN schemes one at each time. For each noise ratio, the MSE beside SNR of each eight test images (four grayscale and four truecolor) resulted by each NN scheme from the five testing times will be calculated and their average for each image will be reported together with that of corresponding median filter and lists in figures 8, 9, 10, 11, and 12 respectively. Also, the figures of grayscale image (Sail) of 15% impulsive noise obtained from CN scheme will be shown together with that obtained by the corresponding median filter (CMF) in figure 13 and those of truecolor image (Pepper) of 25% impulsive noise obtained from SN scheme will be shown together with that obtained by the corresponding median filter (SMF) in figure 14.



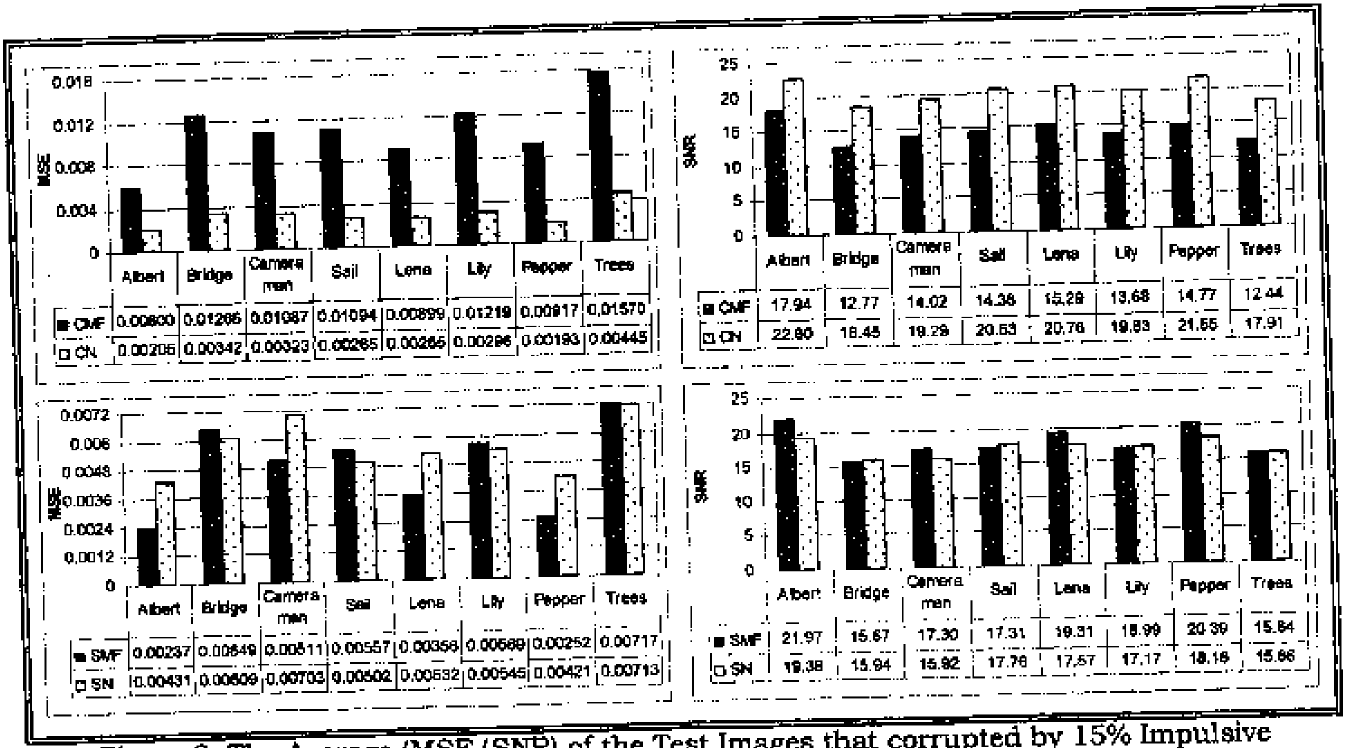
	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	0.00248	0.00534	0.00480	0.00483	0.00332	0.00480	0.00288	0.00718
□ CN	0.00161	0.00263	0.00250	0.00194	0.00199	0.00221	0.00139	0.00381

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	21 77	16.51	17.76	17.93	19.61	17.72	19.79	15.83
□ CN	23.66	19 59	20.41	21.80	21.84	21.09	22.98	18.82

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	0.00131	0.00366	0.00263	0.00325	0.00190	0.00316	0.00082	0.00407
□ SN	0.00256	0.00360	0.00373	0.00335	0.00241	0.00394	0.00204	0.00510

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	24.53	17.90	19.67	19.65	22.04	19.52	25.29	18.30
□ SN	21.84	18.23	18.67	19.51	21.01	18.58	21.30	17.32

Figure 8: The Average (MSE/SNR) of the Test Images that corrupted by 10% Impulsive Noise and filtered by different Neural Network Schemes together with the Ordinary Median Filter.



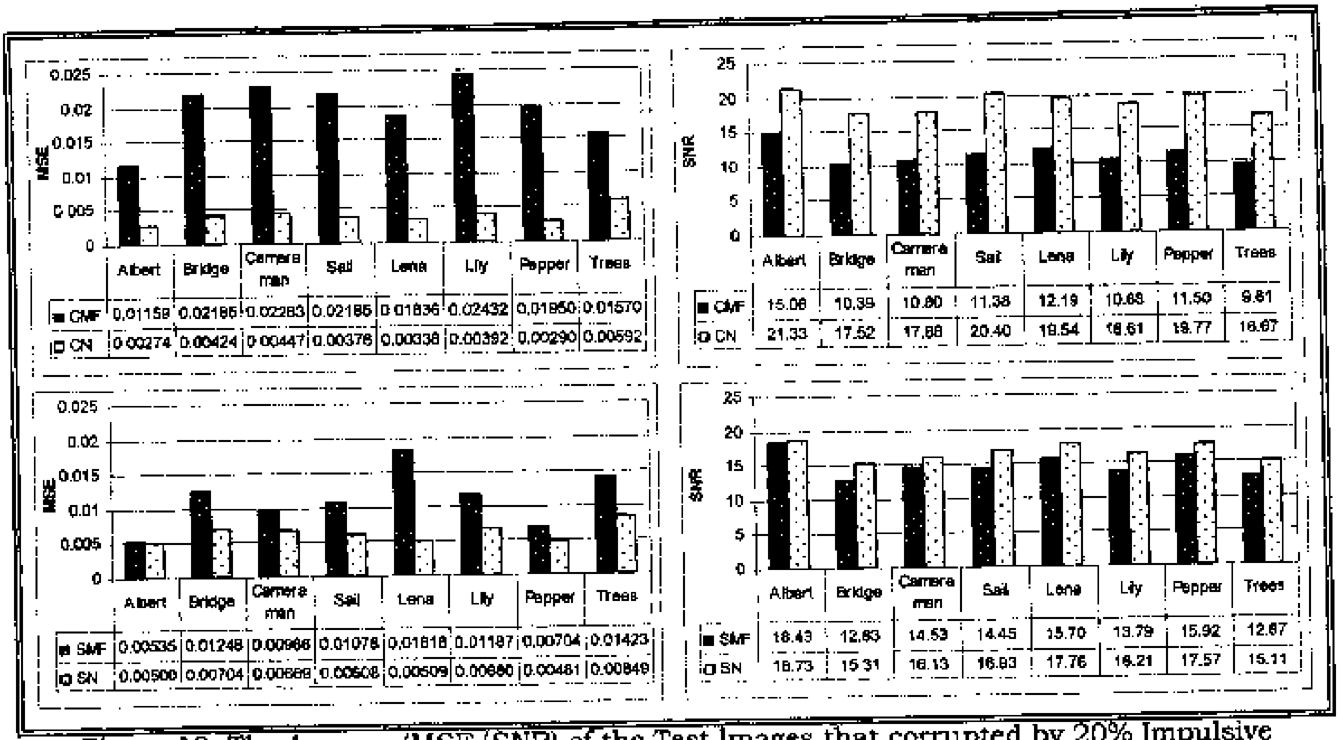
	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	0.00600	0.01266	0.01087	0.01094	0.00899	0.01219	0.00917	0.01570
□ CN	0.00206	0.00342	0.00323	0.00265	0.00265	0.00296	0.00193	0.00445

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	17.94	12.77	14.02	14.38	15.28	13.68	14.77	12 44
□ CN	22.80	18.45	19.28	20.63	20.76	19.83	21.66	17.91

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	0.00237	0.00649	0.00511	0.00557	0.00356	0.00589	0.00252	0.00717
□ SN	0.00431	0.00609	0.00703	0.00502	0.00532	0.00545	0.00421	0.00713

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	21.97	15.67	17.30	17.31	19.31	18.99	20 39	15.64
□ SN	19.38	15.94	15.92	17.76	17.57	17.17	18.16	15.66

Figure 9: The Average (MSE/SNR) of the Test Images that corrupted by 15% Impulsive Noise and filtered by different Neural Network Schemes together with the Ordinary Median Filter.



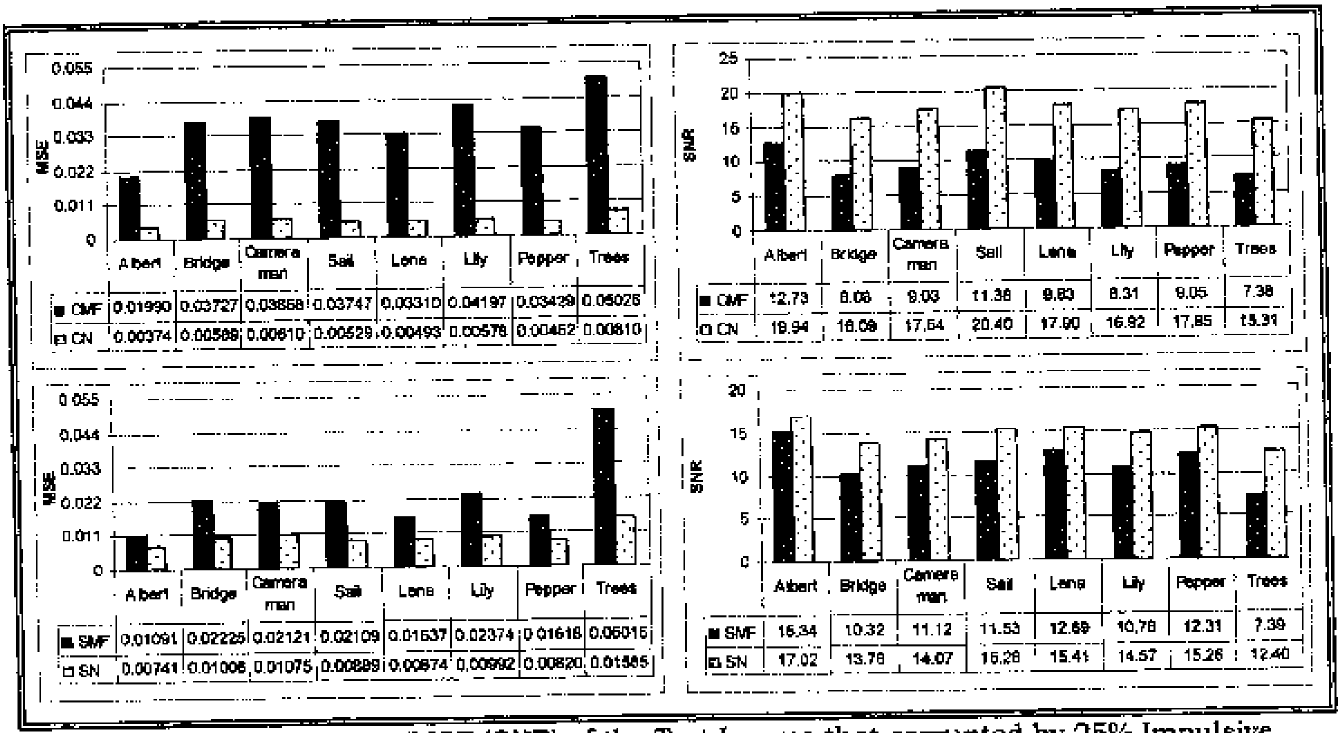
	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	0.01159	0.02185	0.02283	0.02185	0.01836	0.02432	0.01950	0.01570
□ CN	0.00274	0.00424	0.00447	0.00376	0.00338	0.00392	0.00290	0.00592

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	15.06	10.39	10.80	11.38	12.19	10.68	11.50	9.61
□ CN	21.33	17.52	17.88	20.40	19.54	18.61	19.77	18.67

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	0.00535	0.01248	0.00966	0.01078	0.01818	0.01187	0.00704	0.01423
□ SN	0.00500	0.00704	0.00669	0.00608	0.00509	0.00680	0.00461	0.00949

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	18.43	12.83	14.53	14.45	15.70	13.79	15.92	12.67
□ SN	18.73	15.31	16.13	16.83	17.76	16.21	17.57	15.11

Figure 10: The Average (MSE/SNR) of the Test Images that corrupted by 20% Impulsive Noise and filtered by different Neural Network Schemes together with the Ordinary Median Filter.



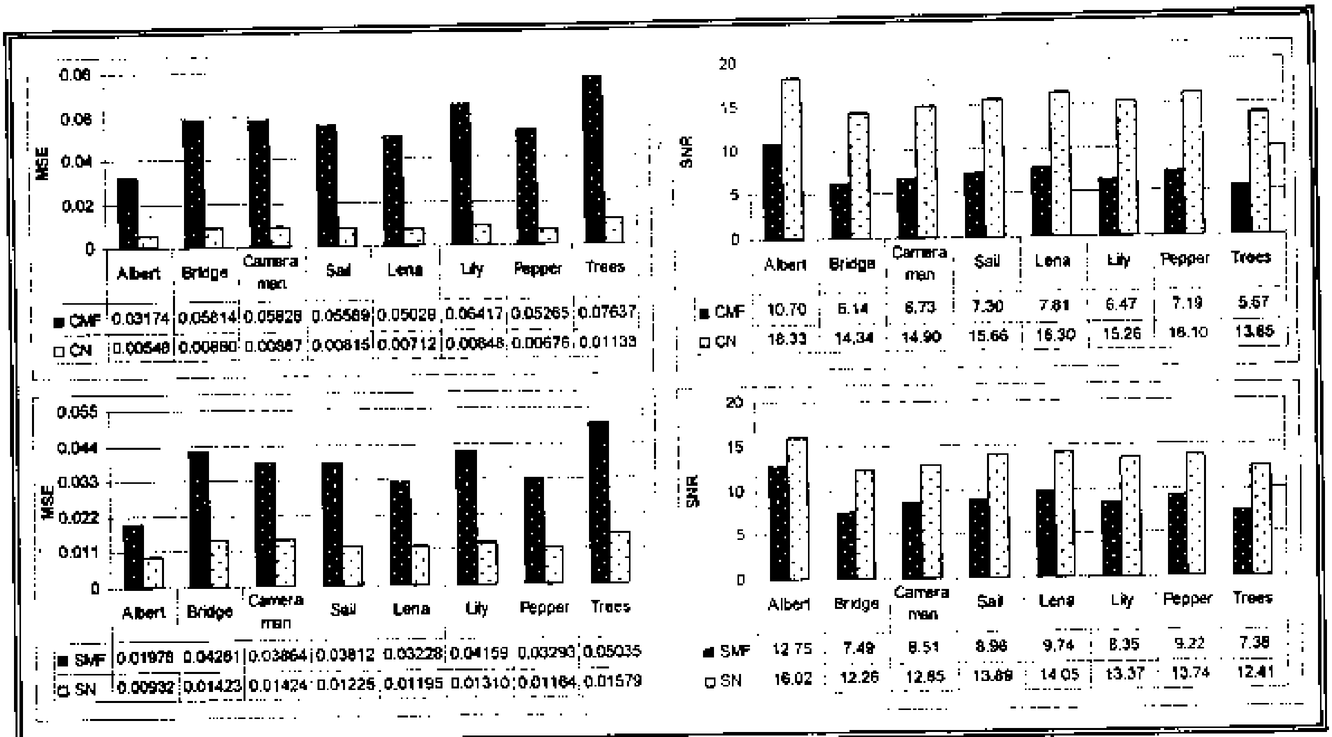
	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	0.01990	0.03727	0.03858	0.03747	0.03310	0.04197	0.03429	0.05026
□ CN	0.00374	0.00589	0.00610	0.00529	0.00493	0.00576	0.00462	0.00810

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	12.73	8.08	9.03	11.38	9.83	8.31	9.05	7.38
□ CN	19.94	16.09	17.64	20.40	17.90	16.92	17.85	15.31

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	0.01091	0.02225	0.02121	0.02109	0.01537	0.02374	0.01618	0.05015
□ SN	0.00741	0.01006	0.01075	0.00889	0.00874	0.00992	0.00820	0.01585

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	15.34	10.32	11.12	11.53	12.69	10.78	12.31	7.39
□ SN	17.02	13.78	14.07	15.28	15.41	14.57	15.26	12.40

Figure 11: The Average (MSE/SNR) of the Test Images that corrupted by 25% Impulsive Noise and filtered by different Neural Network Schemes together with the Ordinary Median Filter.



	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	0.03174	0.05814	0.05828	0.05589	0.05028	0.06417	0.05265	0.07637
□ CN	0.00548	0.00880	0.00887	0.00815	0.00712	0.00848	0.00676	0.01133

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ CMF	10.70	6.14	6.73	7.30	7.81	6.47	7.19	5.67
□ CN	18.33	14.34	14.90	15.66	16.30	15.26	16.10	13.65

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	0.01978	0.04261	0.03864	0.03812	0.03228	0.04159	0.03293	0.05035
□ SN	0.00932	0.01423	0.01424	0.01225	0.01195	0.01310	0.01164	0.01579

	Albert	Bridge	Camera man	Sail	Lena	Lily	Pepper	Trees
■ SMF	12.75	7.49	8.51	8.96	9.74	8.35	9.22	7.38
□ SN	16.02	12.26	12.85	13.89	14.05	13.37	13.74	12.41

Figure 12: The Average (MSE/SNR) of the Test Images that corrupted by 30% Impulsive Noise and filtered by different Neural Network Schemes together with the Ordinary Median Filter.

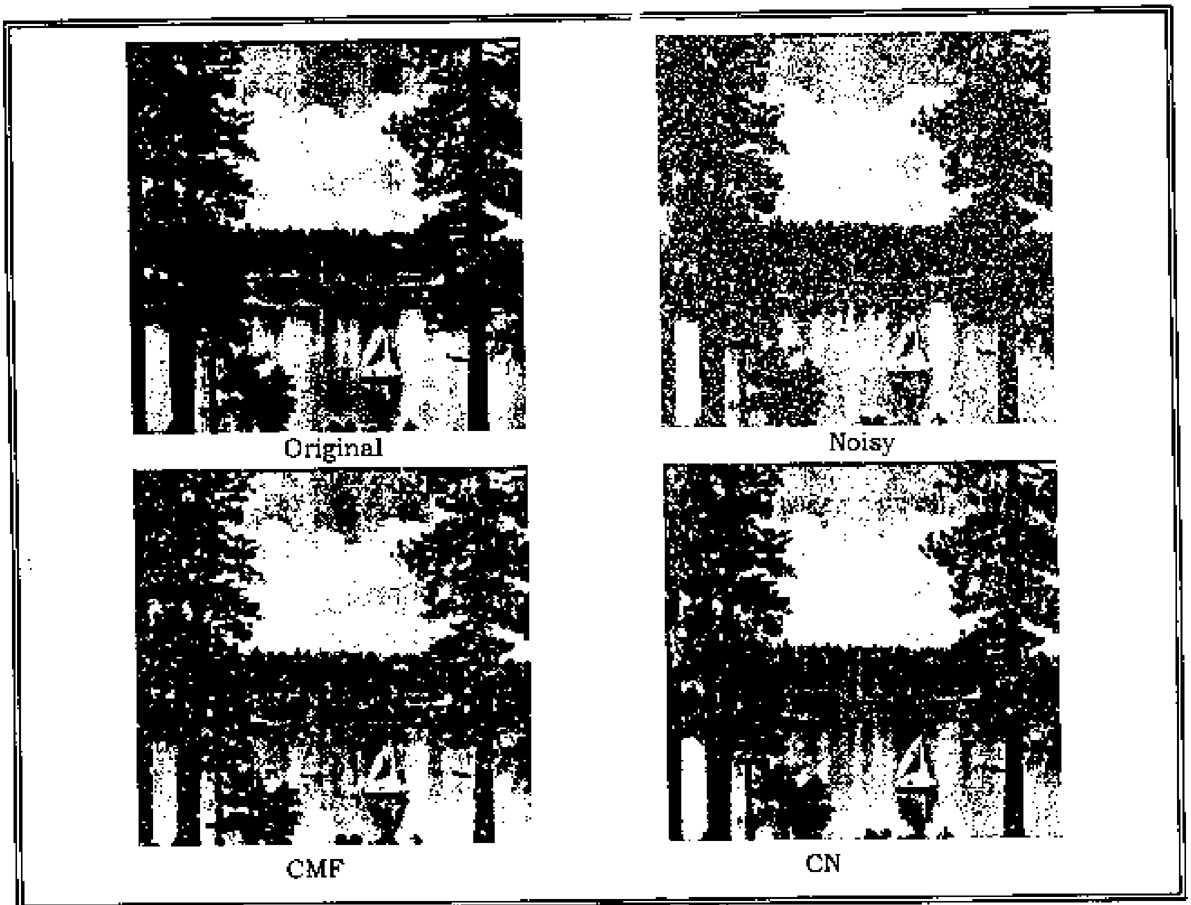


Figure 13: The Output of the Sail Image corrupted by 15% Impulsive Noise that obtained by CN Scheme together with the Output of CMF

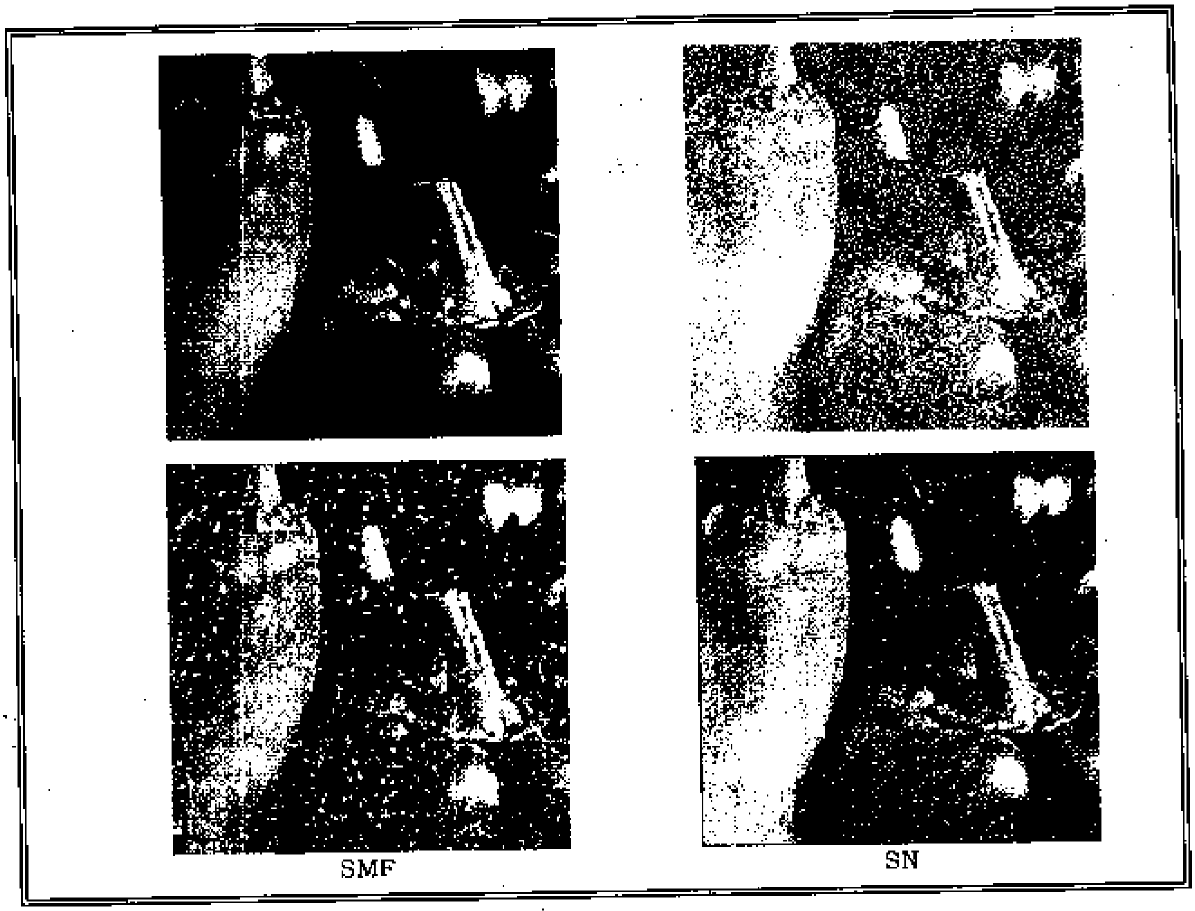


Figure 14: The Output of the Pepper Image corrupted by 25% Impulsive Noise that obtained by SN Scheme together with the Output of SMF

6. Conclusions

function approximation which can be trained based on a set of examples. Given their general nature, NN would seem useful tools for image processing.

In this paper, two feed forward NN schemes have been proposed for impulsive noise removal. The NN schemes (CN and SN) are used to simulate the work of median filter. A training strategy for NN is presented that is based on using artificial image which reduce the time of computation. The artificial image is formed from significant parts chosen to be processed; these significant parts will exhibit the highest edge-like and flat region-like characteristics. As to the noise ratio, the performance of the two proposed NN schemes are best when the noise ratio in testing is the same or greater than that in training since they give smallest values for MSE when compared with that of conventional median filter . For example in the case of 30% impulsive noise ratio , the values of MSE is 0.005 for the first proposed neural network scheme CN and 0.009 for the second scheme SN while in the conventional filter the values are 0.031 for CMF and 0.019 for SMF . .

References

1. E. Abreu, M. Lightstone, S.K. Mitra, K. Arakawa, ***A new Efficient Approach for the Removal of Impulse Noise from Highly Corrupted Images***, *IEEE Trans. on Image Processing*, Vol. 5, No. 6, 1996, pp. 1012-1025.

2. R. Gonzales, and R. Woods, ***Digital Image Processing***, 1st Edition, Addison Wesley PCI, 1992.

3. G. Baker, ***Image Noise***. Website: www.cs.nu.oz.au/~gavinb/download.php?file=noise.pdf

4. J. Tukey, ***Nonlinear (nonsuperposable) methods for smoothing data***, Cong. Rec. EASCOM, pp. 673, 1974.

5. Anderson, J. A. *An introduction to neural networks*. MIT Press, 1995.

6. Hertz, J.; A. Krogh and R. G. Palmer. *Introduction to the theory of neural computation*. Addison-Wesley Publ. Co., 1991.

7. K. Hornik, M. Stinchombe, and H. White, ***Multilayer Feed Forward Networks are Universal Approximators***, Neural Networks, vol. 2, pp. 359-366, 1989.

8. E. Schlünzen, Classificação de dados multiespectrais utilizando redes neurais: a influência da amostragem no processo de treinamento. *Anais do Workshop sobre Visão Cibernética*. São Carlos, Agosto 1994.

9. J. Moreira, and L. Costa, ***Neural-based color image segmentation and classification using self-organizing maps***, GAPIS-Architecture and Image & Signal Processing Research Group, 1996.

10. D. Ridder, R. Duin, L. Vliet, and P. Verbeek, ***Nonlinear image processing using artificial neural networks***, Pattern Recognition Group, Dept. of Applied Physics, Delft University of Technology, Netherlands, 2003.

11. D. Ridder, R. Duin, P. Verbeek, and L. Vliet, ***The Applicability of Neural Networks to Non-linear Image Processing***, Pattern Recognition Group, Applied Physics Dept., Delft University of Technology, Delft, Netherlands, 1999.

12. A. Kokaram, ***Introduction to Electrical Engineering: Digital Image and Video Processing***, Dept. of Electronic and Electrical Engineering, Trinity College, Dublin University, 2004.

13. L. Fausett, ***Fundamentals of Neural Network***, 1st Edition, Prince Hall, New Jersey, 1994.

14. J. Hertz, A. Krogh, and R. Palmer, ***Introduction to the Theory of Neural Computation***, Addison-Wesley, New York, 1991.

15. A. Minns, ***Artificial Neural Network as Sub Symbolic Process Descriptors***, PhD thesis, Deflt University of Technology, (Holland), Abbot M. B., 1998.

16. H. Paramahamsan, ***Fundamental Properties of Synthetic O-D Generation Formulations and Solutions***, M. thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1999.

17. T. Samad, ***Back-propagation improvements based on heuristic arguments***, Proceedings of International Joint Conference on Neural Networks, Washington, vol. 1, pp. 565-568, 1990.

18. V. Ooten, and B. Nienhuis, ***Improving the convergence of the back-propagation algorithm***, Neural Networks, vol. 5, pp. 465-471, 1992.

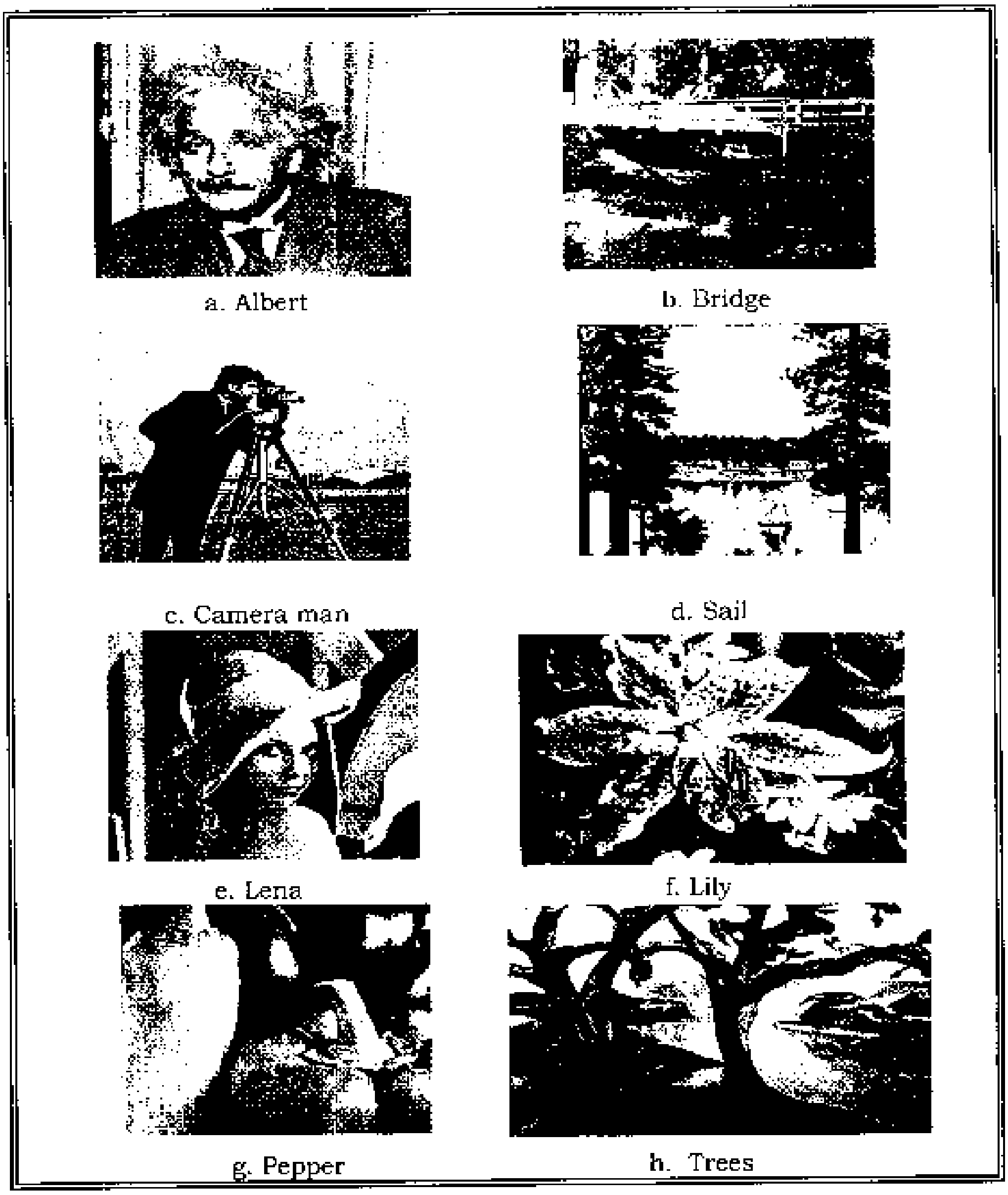
19. A. Minai, and R. Williams, ***Acceleration of Back-Propagation Through Learning Rate and Momentum Adaptation***, in International Joint Conference on Neural Networks, IEEE, pp. 676-679, 1990.

20. R. Jacobs, ***Increased rates of convergence through learning rate***

adaptation, Neural Networks, vol. 1, pp. 295-308, 1988.

21. D. Nguyen and B. Widrow, ***Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights***, Proceeding of the International Joint Confrence on Neural Networks,vol.3,pp.21 26,1990

Appendix



a. Albert

b. Bridge

c. Camera man

d. Sail

e. Lena

f. Lily

g. Pepper

h. Trees

Figure 15: Original copies of the Testing Images Set
(a-d): The Grayscale Images.
(e-h): The Truecolor Images