NeuroFuzzy Network Schemes for Impulsive Noise Reduction in Digital Images

Turki Y. Abdalla
Computer Engineering Department
Engineering College
University of Basrah, Iraq

Abdul-Kareem Younis
Computer Science Department
Science College
University of Basrah, Iraq

Sarah Behnam Aziz
Computer Science Department
Science College
University of Basrah, Iraq

ABSTRACT
Noise reduction or noise removal is an important task in image processing. In general, Results of the noise removal have a strong influence on the quality of the following image processing techniques. On the other side, the integrated system of neurofuzzy networks are more interesting and applied for different applications. In this contribution, two neurofuzzy network schemes have been presented for impulsive noise removal. The computation is reduced by using an artificial image in training. High performances are obtained. Results of neurofuzzy schemes show that the performance is increased as the ratio of the noise is increased. The presented schemes are used for grayscale and also for true color images.

Keywords: Fuzzy Neural Network, Impulsive noise reduction, Image processing, Noise Removal.

1. INTRODUCTION
Unfortunately, noise and other impairments associated with the measurement or the transmission apparatus can significantly degrade the value of the information carried by the digital images. This usually declines their perceptual fidelity and also decreases the performance of the task for which the image was created [1]. Noise reduction is an essential part of any image processing based system, whether the final information is used for human perception or for an automatic inspection and analysis [1-2].

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 [1]. One of the most common spatial filters is the median filter which is introduced by Tukey [3] 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. Several researchers introduced Neural Network (NN) and Fuzzy Logic (FL) systems for different tasks of image processing [4-9]. Both NN and FL systems are aimed at exploiting human-like knowledge processing capability. Where, NN concentrate on the structure of human brain, i.e., on the hardware whereas FL system concentrate on software [10, 11-12].

A very interesting combination is the NeuroFuzzy (NF) architecture, in which the good properties of both methods are attempted to bring together and in many cases, good results have been achieved. The strength features of NF networks makes them well suited to a wide range of applications.

Integrated NF system combines the advantages of NN and FL systems. While the learning capability is an advantage from the viewpoint of FL system, the formation of linguistic rule base is an advantage from the viewpoint of NN. In a fused architecture, the NN learning algorithms are used to determine the parameters of FL system.

A common way to apply a learning algorithm to fuzzy system is to represent it in a special NN like architecture, usually the NF network has a number of layers, and each layer represents one or more steps of FL system [13]. The most common NF systems are based on two types of fuzzy inference systems; Mamdani and Sugeno combined with NN learning algorithm. Several types of NF structures have been proposed like Hall NeuroFuzzy structure [14] and FuNN structure [15]. The varying very often depends on the application. Image processing is an application in which the NF networks are implemented for different problems. For example, A. Al-Amre proposed image matching techniques based on fuzzy and neurofuzzy systems [16] .

This contribution deals with noise reduction and presents an effective filter to remove high ratios of impulsive noise from grayscale and truecolor images based on neurofuzzy networks and in order to reduce time of training; an artificial image will be used as training data set instead of real image. The quality in both terms of "Mean Square Error" (MSE) and "Signal to Noise Ratio" (SNR) are better than those obtained by the conventional image filtering techniques.

The outline of this paper is as follows: the median filter will be discussed in section 2 and a brief explain of the NF network with the basic steps of its learning algorithm will be given in section 3. Section 4 will describe the data sets used in training and testing the proposed NF schemes and in section 5 the experiments performed using the proposed NF schemes will be described and compared their results 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 [3].

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 by ***Cross Median Filter*** (CMF) while the second will be mentioned by ***Square Median Filter*** (SMF).

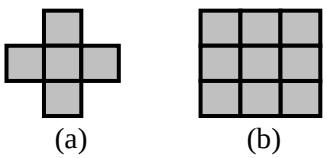


Figure 1: The mask shapes (a: Cross, b: Square).

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 [17]. The principle is shown in figure 2.

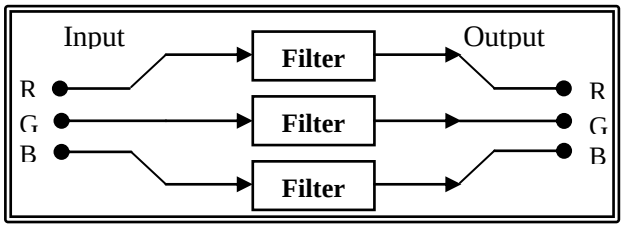


Figure 2: RGB Component Filtering

3. NEURO FUZZY NETWORK

The architecture of the NF network based on Mamdani fuzzy inference system consists of five layers; they represent an input layer, fuzzification layer, rule antecedent layer, rule consequent layer, and combination and defuzzification layer respectively. [11-13].

In order to derive a learning algorithm for a NF network with a gradient descent technique, the inference rule must use differentiable membership function type, for example in this work the Gaussian membership function will be used.The adjusted parameters in the NF network can be divided into two categories based on ***if*** (antecedent) part and ***then*** (consequent) part of the fuzzy rules. For example in the antecedent part, the mean and variance are fine-tuned, whereas in the consequent part, the adjusted parameters are the consequence weights.

The gradient descent based on BP algorithm is employed to adjust the parameters in NF network by using training patterns. Moreover, the algorithm which is used for NF architecture is explained, both feed forward phase and the BP of errors.

a) Forward Phase

This phase computes the activation values of all the nodes in the network from the first to fifth layers.

1. ***Input layer*****:** The nodes in this layer only transmit input values (crisp values) to the next layer directly without modification. Thus,

$$Net_i = x_i \qquad \forall i = 1..N \qquad \textbf{(1)}$$

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

2. ***Fuzzification layer*****:** The output function of this node is the degree that the input belongs to the given membership function. Hence, this layer acts as the fuzzifier. Each membership function is Gaussian and an input signal activates only M neighboring membership functions simultaneously. For a Gaussian-shaped membership function, the activation function for each node is:

$$\mu_{(i,j)} = \exp\left(-\left(Net_i - C_{(i,j)}\right)^2 / 2 * \sigma^2_{(i,j)}\right) \quad \forall i = 1..N \text{ and } j = 1..M \qquad \textbf{(2)}$$

3. ***Rule Antecedent layer*****:** The implication method is performed by this layer and applied using the products. Where the output of each node is calculated as:

$$u_{M^{(i-1)}+j} = \prod_{j=1}^{M}\prod_{i=1}^{N} \mu_{(i,j)} \quad \forall i = 1..N \text{ and } j = 1..M \qquad \textbf{(3)}$$

4. ***Rule Consequent layer*****:** The normalization process is applied in this layer.

$$\bar{u}_k = \frac{u_k}{\sum_{k=1}^{M^N} u_k} \qquad \forall k = 1..M^N \qquad \textbf{(4)}$$

5. ***Combination and Defuzzification layer*****:** This layer performs defuzzification to produce a crisp output value. Among the commonly used defuzzification strategies, the COG method yielded the best result. In this layer, linear output activation function is used:

$$y = \sum_{k=1}^{M^N} \bar{u}_k \bullet w_k \qquad \textbf{(5)}$$

b) Backward Phase

The goal of this phase is to minimize the error function:

$$E = \tfrac{1}{2}(y - d)^2 \qquad \textbf{(6)}$$

Where, d is the desired output.

The learning algorithm in NF is realized by adjusting connection weights of the neurons beside the centers and widths of membership functions.

$$w_k^{new} = w_k^{old} + \Delta w_k \qquad \forall k = 1..M^N \qquad \textbf{(7)}$$

Where,

$$\Delta w_k = -\eta \cdot \delta_k^{w} \qquad \textbf{(8)}$$

$$\delta_k^{w} = \frac{\partial E}{\partial w} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial w} = (y - d) \cdot \frac{u_k}{\sum_{k=1}^{M^N} u_k} \qquad \textbf{(10)}$$

While, adaptation of centers and widths of membership functions is as follows:

$$c_{(i,j)}^{new} = c_{(i,j)}^{old} + \Delta c_{(i,j)} \qquad \forall i = 1..N \text{ and } j = 1..M \qquad \textbf{(11)}$$

$$\sigma_{(i,j)}^{new} = \sigma_{(i,j)}^{old} + \Delta \sigma_{(i,j)} \qquad \forall i = 1..N \text{ and } j = 1..M \qquad \textbf{(12)}$$

$$\Delta c_{(i,j)} = -\eta \cdot \delta^{c}_{(i,j)} \quad \textbf{(13)}$$

$$\delta^{c}_{(i,j)} = \frac{\partial E}{\partial y}\cdot\frac{\partial y}{\partial u}\cdot\frac{\partial u}{\partial \mu}\cdot\frac{\partial \mu}{\partial c} \quad \textbf{(14)}$$

$$= \frac{\left(2\cdot(y-d)\cdot(w_k - y)\cdot u_k \cdot (Net_i - c_{(i,j)})\right)}{\left(\sigma^2_{(i,j)}\cdot\sum_{k=1}^{M^N} u_k\right)}$$

$$\Delta \sigma_{(i,j)} = -\eta \cdot \delta^{\sigma}_{(i,j)} \quad \textbf{(15)}$$

$$\delta^{\sigma}_{(i,j)} = \frac{\partial E}{\partial y}\cdot\frac{\partial y}{\partial u}\cdot\frac{\partial u}{\partial \mu}\cdot\frac{\partial \mu}{\partial \sigma} \quad \textbf{(16)}$$

$$= \frac{\left(2\cdot(y-d)\cdot(w_k - y)\cdot u_k \cdot (Net_i - c_{(i,j)})^2\right)}{\left(\sigma^3_{(i,j)}\cdot\sum_{k=1}^{M^N} u_k\right)}$$

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 3).

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.

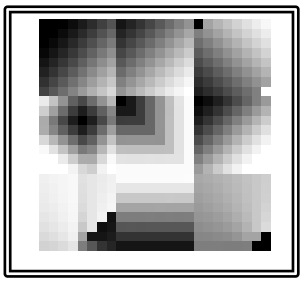


Figure 3: The Original Artificial Image

The above artificial image will be used as training data set for the proposed NN schemes after corrupting it by impulsive noise of ratio 20% (Figure 4).

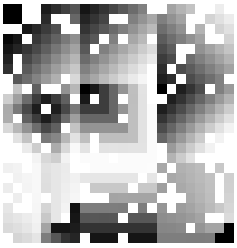


Figure 4: Corrupted Artificial Image with 20% Impulsive Noise

On the other side, the testing data set includes eight images. Four of them are grayscale images and the other four are truecolor images.

5. EXPERIMENTS & RESULTS

In this section the proposed architecture is based on Mamdani NF. It has five layers. They represent an input layer, fuzzification layer, rule antecedent layer, rule consequent layer, and combination and defuzzification layer respectively. Two schemes of these NF will be presented. The first scheme has five inputs that represent the noisy pixel and its four neighbors within a local window while the second scheme has nine neurons in the input layer. Both schemes have one output and the number of membership functions for each input is seven. From now and upward it will be referred to the first scheme by Cross NeuroFuzzy (CNF) and the second one by Square NeuroFuzzy (SNF).

In common Mamdani NF architecture, full connections are used to connect the neurons of fuzzification layer (membership functions). In another words, the number of connections are equal to number of membership functions powered to number of inputs. i.e. number of connections is 16807 in CNF and 40353607 in SNF. As look, the number of connections is huge and that will cause impossible learning in training. To solve this problem, only the neurons that facing each other were connected and additional one that connects all the neurons of fuzzification layer together. That's, to have only 8 connections. The schematic diagram of the first NF architecture with our suggested modification is shown in Figure 5

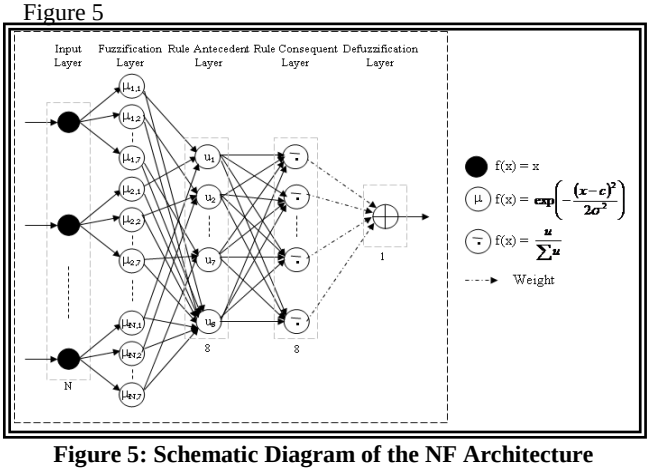


Figure 5: Schematic Diagram of the NF Architecture

5.1 Data Preparing

The patterns of training are constructed from the 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

There are several parameters to be adjusted by the gradient descent based on BP algorithms. These parameters are mean and variance of Gaussian memberships and the consequence weights. The BP algorithm that its forward and backward phases illustrated in section three will be used in training each of CNF and SNF networks taking into consideration the following points.

1- Only the facing neurons in the Fuzzification layer will be connected together instead of the full connections. Therefore, the output of the nodes in the Rule Consequent Layer (3rd layer) will be calculated as follows:

The output of the first seven nodes is computed as:

$$u_j = \prod_{i=1}^{N} \mu_{(i,j)} \qquad \forall j = 1..\ M \qquad \textbf{(17)}$$

Where, N is number of inputs (N = 5 | 9) and M is number of membership functions (M = 7).
While, output of the last node (8th node) will be calculated as in equation 3. Therefore, the carrying out of the last two layers of NF networks will be achieved as:

$$\bar{u}_j = \frac{u_j}{\sum_{j=1}^{M+1} u_j} \qquad \forall j = 1.,2,\quad .M+1 \qquad \textbf{(18)}$$

$$y = \sum_{j=1}^{M+1} \bar{u}_j \bullet w_j \qquad \textbf{(19)}$$

2- BP algorithm will be improved by using adaptive learning rate. Therefore, the parameters are updated under the following conditions.

$$\begin{aligned} &if\ (new\ error / old\ error) > 1.04\ then \\ &\quad New\ values\ of\ parameters\ are\ discarded \\ &\quad lr = lr * lr\_dec \\ &else \\ &\quad Update\ Parameters \\ &\quad if\ new\ error < old\ error \\ &\quad\quad lr = lr * lr\_inc \end{aligned} \qquad \textbf{(20)}$$

3- Weights will be updated by using equation 7. Where, $\partial E / \partial y$ is calculated as follows:

$$\frac{\partial E}{\partial y} = (y - d) \qquad \textbf{(21)}$$

4- The adaptation of the other parameters (Centers and Widths of membership functions) are affected by the type of connections. The actual effect gets from the term of centers and widths errors (δ^C and δ^σ). Thus, centers and widths of membership functions are adapted as in equations 11 and 12 respectively. Where, δ^c and δ^σ are calculated as follows:

$$\begin{aligned} \delta^c_{(i,j)} &= \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial u} \cdot \frac{\partial u}{\partial \mu} \cdot \frac{\partial \mu}{\partial c} \\ &= \left(2 \cdot (y-d) \cdot (w_j - y) \cdot \bar{u}_j \cdot (Net_i - c_{(i,j)}) \right) \Big/ \sigma^2_{(i,j)} \\ &+ \left(2 \cdot (y-d) \cdot (w_{M+1} - y) \cdot \bar{u}_{M+1} \cdot (Net_i - c_{(i,j)}) \right) \Big/ \sigma^2_{(i,j)} \end{aligned} \qquad \textbf{(22)}$$

$$\forall i = 1..N \text{ and } j = 1..M$$

$$\begin{aligned} \delta^\sigma_{(i,j)} &= \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial u} \cdot \frac{\partial u}{\partial \mu} \cdot \frac{\partial \mu}{\partial \sigma} \\ &= \left(2 \cdot (y-d) \cdot (w_k - y) \cdot \bar{u}_j \cdot (Net_i - c_{(i,j)})^2 \right) \Big/ \sigma^3_{(i,j)} \\ &+ \left(2 \cdot (y-d) \cdot (w_{M+1} - y) \cdot \bar{u}_{M+1} \cdot (Net_i - c_{(i,j)})^2 \right) \Big/ \sigma^3_{(i,j)} \end{aligned} \qquad \textbf{(23)}$$

$$\forall i = 1..N \text{ and } j = 1..M$$

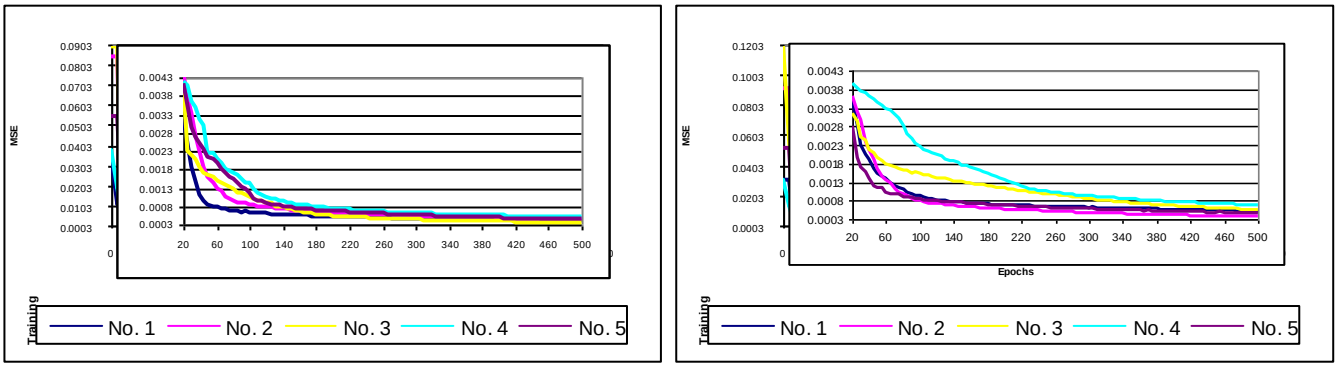
The initial values of each of the learning rate (η), weights, and widths are generated randomly in range between 0 and 1; where, the width of each membership functions of all the inputs are the same while the values of ***lr_inc***, and ***lr_dec*** were set to 1.3, and 0.1 and the initial values of centers of membership functions of each input are et as follows:

$$c_{(i,j)} = (j-1)/(M-1) \qquad \forall i = 1..N \text{ and } j = 1..M \qquad \textbf{(24)}$$

The training is stopped after 500 epochs and it is fail if the error is increased for more than five sequence epochs.
Each of CNF & SNF 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 CNF and SNF five times are listed in table 1 and figure 6 shows the performance measure charts of two NF schemes.

Table 1: Final Performance Measure (MSE) of CNF and SNF

Training No.	MSE	
	CNF	SNF
1	0.00429	0.00367
2	0.00417	0.00368
3	0.00442	0.00344
4	0.00437	0.00370
5	0.00433	0.00375



a. CNF b. SNF

Figure 6: Performance Measure Charts of CNF & SNF Schemes

5.3 Results

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 [17].

A number of simple measures can be generated to measure the ***observed error*** as follows [17].

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

$$MSE = \frac{1}{N \cdot M} \sum_{i=1}^{N} \sum_{j=1}^{M} e(i,j)^2 \qquad \textbf{(25)}$$

$$e(i,j) = y(i,j) - d(i,j) \quad \forall i = 1..N \text{ and } j = 1..M \qquad \textbf{(26)}$$

Where, image is of size N×M, d is the original image and y is the resulted image after corrupting or filtering.

The ***Signal to Noise Ratio*** (SNR) is another popular objective measure [17].

$$SNR = 10 \log_{10} \frac{\frac{1}{N \cdot M} \sum_{i=1}^{N} \sum_{j=1}^{M} d(i,j)^2}{MSE} \qquad \textbf{(27)}$$

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 25 and 27 to calculate MSE and SNR are rights only for grayscale images and should be extended to be available for true color images. The mean square error and signal to noise ratio for truecolor image denoted by MSE_c and SNR_c respectively are given by:

$$MSE_c = \frac{1}{3 \cdot N \cdot M} \sum_{k=1}^{3} \sum_{i=1}^{N} \sum_{j=1}^{M} e(i,j,k)^2 \qquad \textbf{(28)}$$

$$SNR_c = 10 \log_{10} \frac{\frac{1}{3 \cdot N \cdot M} \sum_{k=1}^{3} \sum_{i=1}^{N} \sum_{j=1}^{M} d(i,j,k)^2}{MSE_c} \qquad \textbf{(29)}$$

The set of test images that are corrupted by five different ratios 10%, 20%, and 30% of impulsive noise separately will be filtered by the different two NF 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 NF 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 in figures 7,8, , and 9 respectively. Also, the figures of grayscale images (Albert) of 20% and (Sail) of 30% impulsive noise obtained from CNF scheme will be shown together with that obtained by the corresponding median filter (CMF) in figures 10 and 11 respectively and those of truecolor images (Pepper) of 20% and (Lena) of 30% impulsive noise obtained from SNF scheme will be shown together with that obtained by the corresponding median filter (SMF) in figures 12 and 13 respectively.

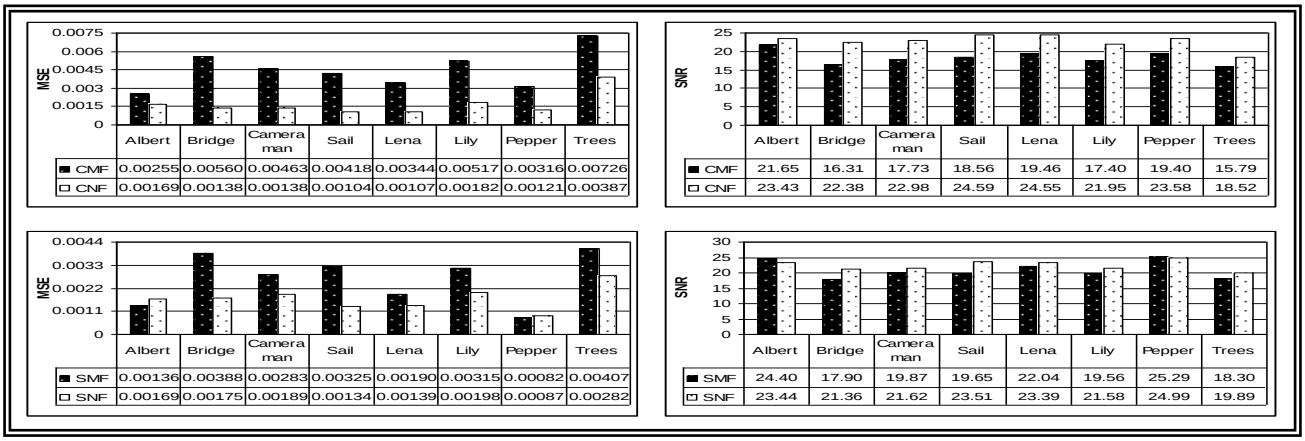


Figure 7: The Average (MSE/SNR) of the Test Images that corrupted by 10% Impulsive Noise and filtered by different NeuroFuzzy Network Schemes together with the Ordinary Median Filter.

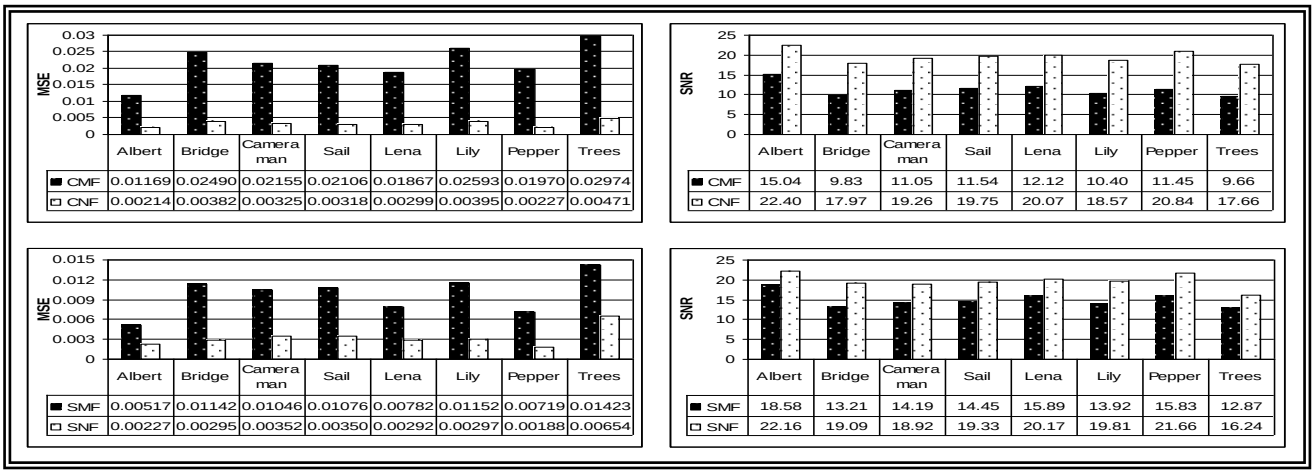


Figure 8: The Average (MSE/SNR) of the Test Images that corrupted by 20% Impulsive Noise and filtered by different NeuroFuzzy Network Schemes together with the Ordinary Median Filter.

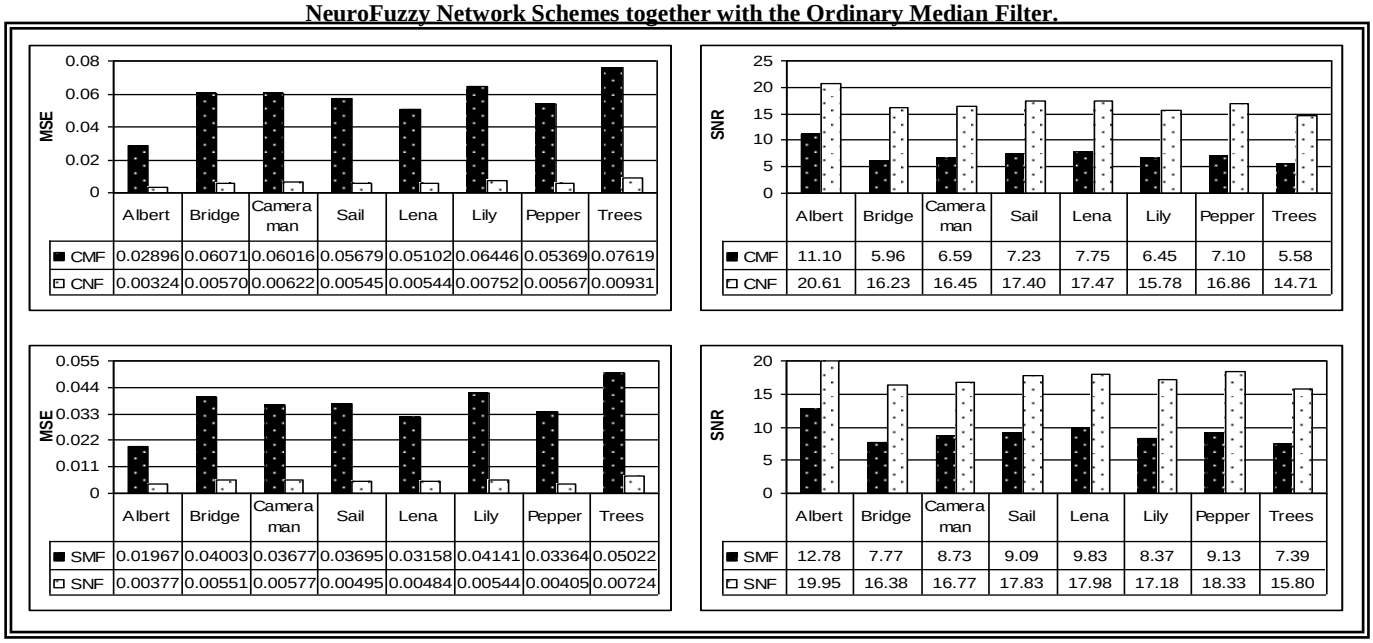


Figure 9: The Average (MSE/SNR) of the Test Images that corrupted by 30% Impulsive Noise and filtered by different NeuroFuzzy Network Schemes together with the Ordinary Median Filter.

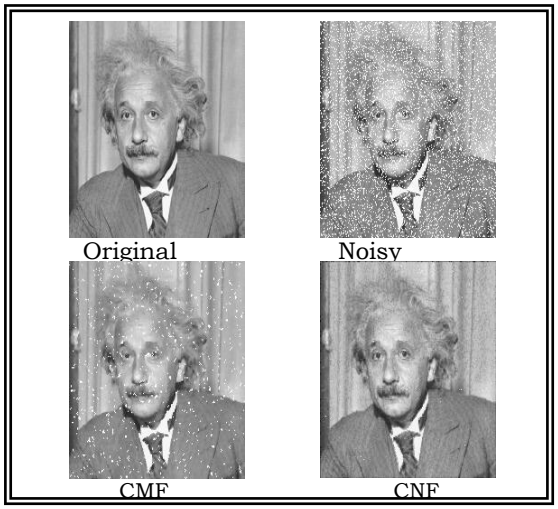


Figure 10: The Output of the Albert Image corrupted by 20% Impulsive Noise that obtained by CNF Scheme together with the Output of CMF

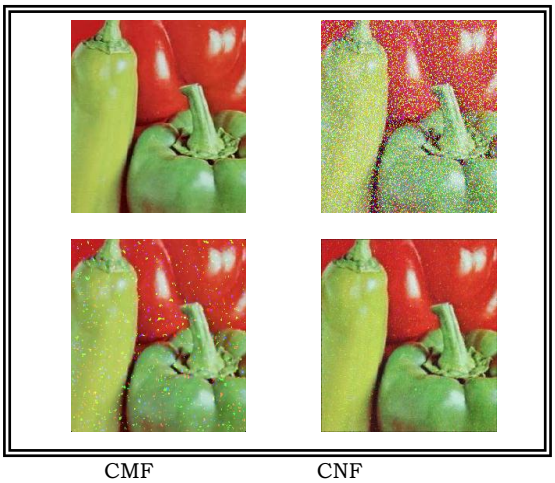


Figure 11: The Output of the sail Image corrupted by 30% Impulsive Noise that obtained by CNF Scheme together with the Output of CMF

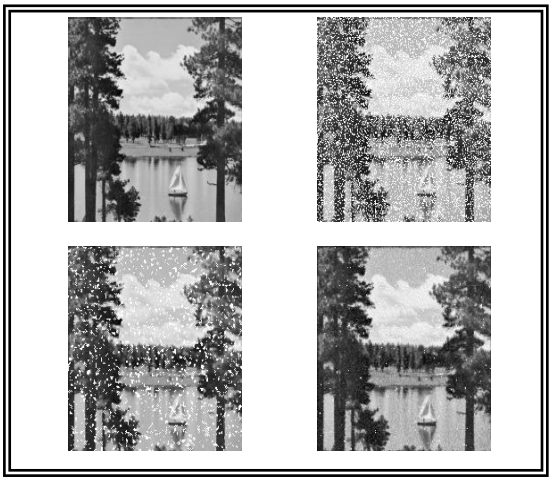
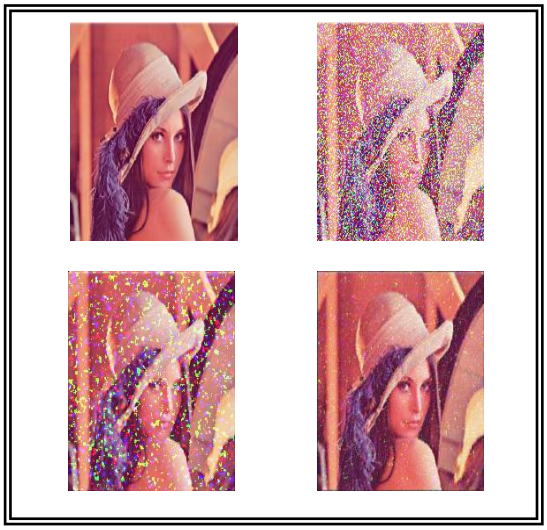


Figure 12: The Output of the Pepper Image corrupted by 20% Impulsive Noise that obtained by CNF Scheme together with the Output of CMF



CMF CNF

Figure 13: The Output of the Lina Image corrupted by 30% Impulsive Noise that obtained by CNF Scheme together with the Output of CMF

6. CONCLUSION

Since both of neural network and fuzzy logic have their limitations and advantages, therefore the neural network and fuzzy logic are combined together to pass these limitations and use their advantages and the neurofuzzy architecture has been proposed and applied for noise reduction. The neurofuzzy architecture is based on Mamdani fuzzy logic system and neural network algorithms. In common Mamdani NF architecture, full connections are used to connect the neurons of fuzzification layer (membership functions). In our problem this architecture of connections will cause to have a huge number of connections therefore a new architecture is presented where only the neurons that facing each other will be connected and additional one that connects all the neurons of fuzzification layer together to cover all the possibilities.

Two NF schemes have been presented for impulsive noise removal. The NF schemes (CNF and SNF) are used to simulate the work of median filter. A training strategy for NF is presented that is based on artificial image which deal to 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. The performance of the testing results obtained by both NF schemes are increased as the ratio of noise is increased. The two presented schemes are used for grayscale and also for true color images without any modifications or additions just each scheme will be applied on the image of each color plane in sequence.

7. REFERENCES

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

[2]. J. Gomes, L. Velho, ***Image Processing for Computer Graphics***, New York, Springer-Verlag Inc., 1997.

[3]. J. Tukey, ***Nonlinear (non superposable) methods for smoothing data***, Cong. Rec. EASCOM, pp. 673, 1974.

[4]. 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.

[5]. 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.

[6]. K. Arakawa, ***Median filter based on fuzzy rules and its application to image restoration***, Computer Science Dept., Meiji University, Japan, 1996.

[7]. 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.

[8]. C. Vertan, N. Boujemaa, ***A Fuzzy Credibility Approach To Color Image Filtering***, Image Processing and Analysis Laboratory (IPAL), Bucharest Polytechnic University, Romania, 2000.

[9]. 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.

[10]. J. Zhang and A. Morris, ***Fuzzy neural networks for nonlinear systems modeling***, IEE Proc.-Control Theory and Application, vol. 142, pp. 551-556, 1995.

[11]. S. Horikawa, T. Furuhashi, and Y. Uchikawa, ***On fuzzy modeling using fuzzy neural networks with the back propagation algorithm***, IEEE Trans. Neural Networks, vol. 3, pp. 801-806, 1992.

[12]. L.-X. Wang, ***Adaptive Fuzzy systems and control Design and stability analysis***, Prentice Hall, New Jersey, 1994.

[13]. H. Koivo, ***Soft Computing In Dynamical Systems***, PhD thesis, Helsinki University of Technology, 2001.

[14]. C. Isik, ***1997 Annual Meeting of the North American Fuzzy Information Processing Society-NAFIPS***, Institute of Electrical and Electronic Engineers, 1997.

[15]. N. kasabov, J. Kim, A. Gray, and M. Watts, ***FuNN - A Fuzzy Neural Network Architecture for Adaptive Learning and Knowledge Acquisition***, Information Science Dept., Otago University, Dunedin, New Zealand, 2001.

[16]. A. Al-Amre, ***Image Matching***, M. Sc. Thesis, Computer Science Dept., College of Science, Basrah University, 2005.

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