CHAPTER – VII

REMOVAL OF GAUSSIAN NOISE FROM IMAGES USING AN EFFICIENTLY TRAINED RADIAL BASIS FUNCTION NEURAL NETWORK
REMOVAL OF GAUSSIAN NOISE FROM IMAGES
USING AN EFFICIENTLY TRAINED
RADIAL BASIS FUNCTION NEURAL NETWORK

7.1 GENERAL

An image is often corrupted by noise in its acquisition and
transmission. The goal of denoising of images is to remove such noise while
retaining the important signal features as much as possible [47]. Traditionally, this is achieved by linear processing such as Wiener filtering.
Recently, several nonlinear techniques have been emerged using wavelet
thresholding or shrinkage to remove the Gaussian noise from digital images
[45, 57 - 60]. These schemes have near optimal properties in minimax
sense and perform well in simulation studies of one-dimensional curve
estimation. A nonlinear filter based on precise translation of classical
analog TV signal has been recently proposed in the literature [61]. A neural
network thresholding scheme for adaptive noise reduction has also been
suggested [48]. One of the drawbacks of these denoising schemes is high
computational complexity.

Binary images have several applications such as manuscript,
signature and credit card processing etc. in industries and banks [62]. For
example, old manuscripts are valuable and only few copies of such
documents are available. The conventional methods of preserving these
documents are to store them in libraries and museums. Problems in such
storage arise due to deterioration of paper or fading of written text due to
environmental conditions. For restoration and preservation of such
manuscripts, conventional filters used do not produce satisfactory results.
Similarly, the signature acquired though an acquisition medium may be
subjected to noise and may create problem in the verification process. In
such situation, it requires reliable denoising operation prior to verification.
Similarly, the corrupted text and the degraded manuscripts also need
substantial recovery for readability. In this chapter, this problem has been
studied in detail by treating the corrupted text or signature as degraded
binary images. A novel reconstruction filter has been designed using Radial
Basis Function Neural Network (RBFNN) trained by Extended Kalman
Gaussian noise removal using RBFNN

Filtering (EKF) algorithm for extracting the relevant information. The overall block diagram of the proposed scheme is shown in Fig. 7.1.

Fig. 7.1 Block diagram of the proposed scheme

The sectionwise development of this chapter proceeds as follows. Section II deals with the image degradation model used in the proposed scheme. An overview of RBFNN and its training schemes are discussed in Section III. Section IV presents the simulation results. Finally, the concluding remarks are given in Section V.

7.2 SIGNAL AND NOISE MODEL

In the conventional pattern recognition problem each character or numeral is represented in m x n grid, each grid representing a binary 0 or 1 depending upon black or white characteristics. The pattern recognition network employs all m x n points as inputs. The output of the network also consists of m x n points. The number of patterns to the network depends on the types of problem. The network parameters are trained based on a learning algorithm such that the output error is minimized. The desired signal is same as the corresponding input pattern. Such method of pattern recognition has two drawbacks. Firstly, the complexity of the network increases with the increase of the number of inputs. Secondly, the number of inputs being large the training time becomes more and the network may
not converge. A new pattern recognition scheme has been proposed in this chapter, which alleviates these drawbacks.

In this scheme each column of the pattern consists of $m$-bits of data and therefore there can be $2^m$ different possible combinations. A RBFNN is trained with all possible $2^m$ patterns of inputs. The trained network could then be able to identify any one of the input patterns.

In a practical situation, the bit pattern is not ideal and is often contaminated with Gaussian noise. The distorted pattern is then applied to the input of RBFNN and simultaneously the corresponding correct pattern is applied as the target value to the network. In this way, a generalized trained RBFNN is developed which yields a correct pattern from a distorted one.

A distorted manuscript obtained through a scanner may be viewed as a binary image of size $M \times N$, which contains $M.N/2^m$, distorted patterns each consisting of $m$-values. Once this segmentation is done, the previously described pattern recognition problem may be applied to recover the correct manuscript. The interesting feature of this type of pattern recognition is that once the RBFNN is trained with all possible $2^m$ noisy patterns, then the network will be capable of recovering any degraded text or manuscripts.

In general, the noise model for the binary image considered for recovery can be mathematically modeled as:

$$ S(i, j) = X(i, j) + \sigma \cdot e(i, j) \quad (7.1) $$

where $S(i, j)$, $X(i, j)$ and $e(i, j)$ are the degraded pixel intensity, original pixel intensity and noise intensity at $(i,j)$th location respectively. The noise process $e(i, j)$ follows a random independent identically distributed (i.i.d) white Gaussian distribution with standard deviation $\sigma$. In the present case $\sigma$ is assumed to be 1. The objective of denoising is to recover the estimated image $\hat{X}(i, j)$ from $S(i, j)$. 
7.3 RADIAL BASIS FUNCTION NEURAL NETWORK (RBFNN)

Recently many efforts have been made to use RBFNN [52] for pattern classification problems due to various drawbacks inherent in Multi Layer Artificial Neural Network (MLANN). Although MLANN produces decision surfaces that effectively separate training examples of different classes, this does not necessarily result in a robust pattern identifier. To overcome this difficulty a novel RBFNN is proposed to act as an efficient pattern recognition network. In case of pattern recognition task the RBF network is trained as a pattern identifier using a noisy pattern as input and the corresponding desired pattern as the output. The schematic diagram of such a network is shown in Fig. 7.2. The number of hidden nodes, the number of input sets and the parameters of the network are decided based on the number of input and output nodes.

Fig. 7.2 Radial Basis Neural Network Architecture
The nodes within each layer are fully connected to the previous layer. The input data is first operated with the RBF function, which are assumed to be Gaussian in nature and the corresponding output are fed as the input to the neural network. Each hidden unit in the network has two parameters called a center ($\mu$), and a width ($\sigma$) associated with it. The Gaussian function of the hidden units is radially symmetric in the input space and the output of each hidden unit depends only on the radial distance between the input vector $x$ and the center parameter $\mu$ for the hidden unit. The Gaussian function gives the highest output when the incoming variables are closest to the center position and decreases monotonically as the distance from the center decreases. The response of each hidden unit is scaled by its connecting weights ($\alpha$'s) to the output units and then summed to produce the final network output. The overall network output is therefore

$$\hat{y}(n) = f(x_n) = \alpha_m 0 + \sum_{k=1}^{K} \alpha_{mk} \phi_k(x_n)$$  \hspace{1cm} (7.2)

For each input $x_n$,

- $n$ represents the time index,
- $K =$ number of hidden units,
- $\alpha_{mk} =$ connecting weight of the $kth$ hidden unit to output layer,
- $\alpha_m 0 =$ threshold,
- $m =$ number of output.

The function of $\phi_k(x_n)$ is defined as

$$\phi_k(x_n) = \exp(-\frac{1}{\sigma_k^2} \|x_n - \mu_k\|^2)$$  \hspace{1cm} (7.3)

where $\mu_k$ is the center vector for the $k$-th hidden unit and $\sigma_k$ is the width of the Gaussian function and $\|\|$ denotes the Euclidean norm. In the present problem, the network output $\hat{y}(n)$ is the recovered pattern, which is
Gaussian noise removal using RBFNN represented as $x(n)$. The RBFNN is trained using two different algorithms using conventional Back Propagation (BP) and Extended Kaman Filtering (EKF). These algorithms are discussed briefly in the following sections.

### 7.3.1 Least Mean Square (LMS) based Back Propagation Algorithm

The optimization of the cost function may be achieved by gradient descent technique by taking the partial derivative of the cost function with respect to each parameter of the RBFNN. The parameters of RBFNN include the centers, the variance of the hidden nodes and the weights of the connecting path of the neural network. The training algorithm is described as follows:

**Step 1:** Define the error vector at the output unit as $e = (d - \hat{y})$,

where, $d$ is the desired output vector and $\hat{y}$ is the estimated output vector.

**Step 2:** Define cost function $E = \frac{1}{2}e^2$ and objective of training algorithm is to minimize $E$.

**Step 3:** Take the partial derivative of $E$ with respect to different network parameters and obtain the key update equations as,

\[
\Delta \mu_k = -\eta \frac{\partial E}{\partial \mu_k} = \eta [(d - \hat{y}) \alpha_{mk}] \frac{(x_n - \mu_k)}{\sigma_k^2} (2\phi_k) 
\]

\[
\Delta \sigma_k = -\eta \frac{\partial E}{\partial \sigma_k} = \eta [(d - \hat{y}) \alpha_{mk}] \frac{(x_n - \mu_k)^2}{\sigma_k^3} (2\phi_k) 
\]

\[
\Delta \alpha_{mk} = -\eta \frac{\partial E}{\partial \alpha_{mk}} = \eta (d - \hat{y}) \phi_k 
\]

\[
\Delta \alpha_{m0} = -\eta \frac{\partial E}{\partial \alpha_{m0}} = \eta (y - \hat{y}) 
\]

where $\eta$ = convergence coefficient.

**Step 4:** Apply each input pattern and compute (7.2) through (7.7) for each case.
Step 5: Compute the average change of different parameters and updated once in each experiment using (7.8) - (7.11).

\[ \mu_k(n+1) = \mu_k(n) + \frac{1}{T} \sum_{t=1}^{T} \Delta \mu_k(n) \] (7.8)

\[ \sigma_k(n+1) = \sigma_k(n) + \frac{1}{T} \sum_{t=1}^{T} \Delta \sigma_k(n) \] (7.9)

\[ \alpha_{mk}(n+1) = \alpha_{mk}(n) + \frac{1}{T} \sum_{t=1}^{T} \Delta \alpha_{mk}(n) \] (7.10)

\[ \alpha_{m0}(n+1) = \alpha_{m0}(n) + \frac{1}{T} \sum_{t=1}^{T} \Delta \alpha_{m0}(n) \] (7.11)

where, \( T \) = no of patterns.

### 7.3.2 EKF Training Algorithm

Conventional BP algorithm is based on steepest descent method and therefore for correlated types of patterns the training of parameters become slow. The training time may be improved by using EKF learning algorithm instead of BP.

Using the same cost function, EKF based training algorithm of the RBFNN can similarly be derived [52]. The relevant update equations obtained as,

\[ W_n = W_{n-1} + K_n e_n \] (7.12)

where the weight vector \( W \) for single output case is given by

\[ W = [a_0, a_1, \mu_1^T, \sigma_1, \ldots, a_K, \mu_K^T, \sigma_K]^T \] (7.13)

the Kalman gain

\[ K_n = P_{n-1} a_n^T \left[ R_n + a_n^T P_{n-1} a_n \right]^{-1} \] (7.14)

\[ P_n = \left[ I - K_n a_n^T \right] P_{n-1} + Q I \]
The Jacobian $\alpha_n$ is

$$\alpha_n = \frac{\partial f(x_n)}{\partial W} \bigg|_{W=W_{n-1}}$$

$$= \left[ 1, \phi_1(x_n), \phi_2(x_n) \frac{2\alpha_1}{\sigma_1^2} (x_n - \mu_1)^T, \phi_3(x_n) \frac{2\alpha_2}{\sigma_1^2} \|x_n - \mu_1\|^2, \ldots \right]$$

$$\phi_k(x_n), \phi_k(x_n) \frac{2\alpha_k}{\sigma_k^2} (x_n - \mu_k)^T, \phi_k(x_n) \frac{2\alpha_k}{\sigma_k^2} \|x_n - \mu_k\|^2 \right]^T \quad (7.15)$$

The outputs $O_k^n (k = 1, 2, \ldots, K)$ of all hidden units are then computed. Using the largest absolute hidden unit output $|O_{max}|$, the normalized value for each hidden unit $r_k^n = |O_k^n / O_{max}^n| (k = 1, 2, \ldots, K)$ can be calculated.

### 7.4 SIMULATION RESULTS

The simulation experiments have been conducted in a Pentium III processor using MATLAB software. The complete simulation task is divided into three different experiments.

**Experiment 1:**

The aim of this experiment is to compare the convergence characteristics of different training algorithms used in MLANN and RBFNN. The training is performed using conventional BP and EKF algorithms. In the simulation, each pattern consists of 7-binary values. Gaussian noise of $-10$ dB and $-30$ dB NSR is generated and is mixed with all $2^7$ possible patterns to create distorted patterns. These are applied sequentially to both RBF and MLANN network. Both BP and EKF training algorithms are employed to train the network parameters. The convergence characteristics obtained from the simulation study are shown Figs. 7.3 and 7.4 for $-10$ dB and $-20$ dB NSR respectively. The convergence characteristics show that the RBFNN offers faster convergence compared to MLANN for both the learning algorithms. Further, the EKF trained RBFNN provides the fastest convergence. This is true for both the noise conditions. Therefore the RBFNN with EKF training being the best, is chosen for subsequent simulation study.
Gaussian noise removal using RBFNN

Fig. 7.3 Convergence Characteristics of different neural structures using different algorithms at -10 dB NSR

Fig. 7.4 Convergence Characteristics of different neural structures using different algorithms at -30 dB NSR
Experiment 2:

The objective of this experiment is to recover characters and numerals from their degraded version using the EKF trained RBFNN reconstruction filter. The trained network obtained from Experiment 1 is used here for recovering the original characters and numerals. No further learning of the network is necessary unlike the conventional pattern recognition method. The initial network has been trained at -10 dB NSR. However during testing all possible degraded numerals and characters (-5 dB NSR) are applied and the output of the reconstruction filter is recorded. It may be noted that a degraded character or numeral has been segmented into 5 sets of consecutive patterns, each consisting of 7-gray levels. The observed results for some specific characters and numerals are shown in Fig. 7.5. The results show that even if the network has been trained at -10 dB NSR, accurate reconstruction of the characters and numerals has been obtained from the network when degradation has been set at -5 dB NSR.

In the second part of this experiment a particular character (A) is chosen and is distorted by adding Gaussian noise ranging from -10 dB to -2 dB NSR. Thus a set of distorted A values are generated and is applied to the same trained network. The outputs are shown in Fig. 7.6. The reconstructed patterns obtained show that up to -6 dB NSR the reconstruction is perfect and beyond this noise there is slight degradation in the output. Therefore, it may be concluded that the trained network at -10 dB NSR can faithfully reconstruct the original characters and numerals when the input NSR is up to -6 dB. The same is true for other characters and numerals.

Experiment 3:

In this experiment a sample of manuscripts/text is chosen. The text/manuscript is treated as binary image and bounded in a regular frame of size M x N. The complete degraded text is now is segmented in rows and columns so that there are M.N/2m number of patterns each consisting of
seven gray levels. These patterns are then contaminated with \(-10\) dB NSR. In the same way the distorted patterns are applied sequentially to the already trained network. The reconstructed results are then properly ordered to obtain complete text and shown in Fig. 7.7. These results show that the reconstruction is perfect. This is true for other degraded text/manuscript also. For comparison, a Wiener filter is simulated for reconstruction purpose. The results obtained show that the reconstruction is poor in case of Wiener filter as compared to the proposed one. The simulation results of the reconstructed text at \(-10\) dB NSR is presented in Fig. 7.8 and similar observations are also made in this case.

7.5 CONCLUSION

An investigation has been made to recover manuscripts and text from their degraded versions by using a reconstruction filter developed through a RBFNN. The degraded characters and numerals have been assumed to be made out of a set of 7-bit gray levels. Similar assumption has been made for degraded text. The degradation has been made using different strength of Gaussian noise. The reconstruction filter has been trained using \(2^7 = 128\) input patterns and 128 target patterns. For training the network both EKF and conventional BP algorithms have been used. The trained network is used for testing purpose for all situations and no further training is subsequently required for testing. It is in general observed that the EKF trained RBF network performs the best for all types of inputs and for different types of noise conditions.
Gaussian noise removal using RBFNN

Original binary Image
Noisy image
Restored image

Fig. 7.5 Restored characters and numerals at -5 dB NSR
Gaussian noise removal using RBFNN

Original binary Image
Noisy image
Restored image

Fig. 7.6 Restored characters at different noise conditions

NSR = -2 dB
NSR = -4 dB
NSR = -6 dB
NSR = -8 dB
NSR = -10 dB

Original binary image  Noisy image  Restored image

Fig. 7.6 Restored characters at different noise conditions
Gaussian noise removal using RBFNN

Fig. 7.7 Restored texts at -15 dB NSR
Gaussian noise removal using RBFNN

![Text image 1](TEST
BINARY
IMAGE 1)

Text image 1

![Text image 2 -10 dB NSR](TEST
BINARY
IMAGE 1)

-10 dB NSR

![Wiener filter](TEST
BINARY
IMAGE 1)

Wiener filter

![RBF neural network](TEST
BINARY
IMAGE 1)

RBFNN filter

![Text image 2 -10 dB NSR](TEST
BINARY
IMAGE 1)

Text image 2

![RBF neural network](TEST
BINARY
IMAGE 1)

-10 dB NSR

![Wiener filter](TEST
BINARY
IMAGE 1)

Wiener filter

![RBF neural network](TEST
BINARY
IMAGE 1)

RBFNN filter

Fig. 7.8 Restored text at -10 dB NSR