

DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**IMAGE REGISTRATION BASED ON ENSEMBLE
OF EXTREME LEARNING MACHINE**



by

MOHAMED GALALELDIN ALI ELOBAID

October, 2019

İZMİR

IMAGE REGISTRATION BASED ON ENSEMBLE OF EXTREME LEARNING MACHINE

**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Electrical and Electronics Engineering**

by

MOHAMED GALALELDIN ALI ELOBAID

October, 2019

İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled “ **IMAGE REGISTRATION BASED ON ENSEMBLE OF EXTREME LEARNING MACHINE**” completed by **MOHAMED GALALELDIN ALI ELOBAID** under supervision of **Asst. Prof. Dr. Yavuz ŞENOL** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.



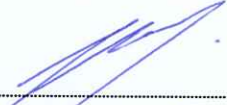
Asst. Prof. Dr. Yavuz ŞENOL

Supervisor



Prof. Dr. Mehmet Engin

(Jury Member)



Asst. Prof. Dr. Hakkı T. YALAZAN

(Jury Member)

Prof. Dr. Kadriye ERTEKİN

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to project advisor Assist. Prof. Dr. Yavuz ŞENOL for continuous guidance, generous. I am grateful for your guidance throughout my graduate education.

At last, I would like to thank my family for their trust, patience and encouragement.

Mohamed GALALELDIN ALI ELOBAID



IMAGE REGISTRATION BASED ON ENSEMBLE OF EXTREME LEARNING MACHINE

ABSTRACT

Image registration is the process that aligns a set of images obtained from various sensors, different view of points or different time in one coordinates system. Image registration with neural network using image features, is a scheme that used to estimate the geometrical transformation parameters namely (scaling, rotation and translation).

In this thesis, the image registration procedure was carried out using ensembled extreme learning machines to overcome the disturbance effect of the noise, improve the accuracy and robustness of the estimation of the geometrical transformation parameters under noisy condition. In performed experiments, three different images, which are transformed using affine transformation, were used. The features were extracted from noisy image and noise free images by discrete cosine transform. These features were given as input to neural networks and the registration parameters were calculated at the output of the network. Image registration based on neural network can be realized in two stages. In pre-registration stage, extreme learning machines, radial basis function and feed-forward type of neural networks were trained by features extracted from images, which are translated, scaled and rotated with various parameters. In registration stage, the trained networks were fed by features extracted from test images and the registration parameters were observed at the output.

The obtained results were compared in terms of different used neural networks. The results showed that extreme learning machines give better result in the present of the noise than other two neural networks.

Keywords: Image Registration, extreme learning machine, feedforward neural network, radial basis function neural network, online sequential extreme learning machine



AŞIRI ÖĞRENME MAKİNASI TOPLULUĞUNA DAYALI GÖRÜNTÜ ÇAKIŞTIRMA

ÖZ

Görüntü çakıştırma, tek bir koordinat sisteminde çeşitli sensörlerden, noktaların farklı görünümülerinden veya farklı zamanlardan elde edilen bir dizi görüntüyü hizalayan işlemdir. Görüntü özelliklerini kullanarak sinir ağı ile görüntü çakıştırma, geometrik dönüşüm parametrelerini (ölçekleme, döndürme ve öteleme) tahmin etmek için kullanılan bir işlemdir.

Bu tez çalışmasında, görüntü çakıştırma prosedürü gürültünün rahatsızlık etkisinin üstesinden gelmek, gürültülü koşullar altında geometrik dönüşüm parametrelerinin tahmininin doğruluğunu ve sağlamlığını arttırmak için birleştirilmiş aşırı öğrenme makineleri kullanılarak gerçekleştirilmiştir. Yapılan deneylerde, afin dönüşümü kullanılarak dönüştürülmüş üç farklı görüntü kullanılmıştır. Özellikler gürültülü görüntü ve gürültüsüz görüntülerden ayırık kosinüs dönüşümüyle elde edildi. Bu özellikler sinir ağına girdi olarak verildi ve çakıştırma parametreleri sinir ağına çıkışından hesaplandı. Yapay sinir ağı tabanlı görüntü çakıştırma iki aşamada gerçekleştirilebilir. Çakıştırma öncesi aşamada, aşırı öğrenme makineleri, radyal tabanlı sinir ağı ve ileri beslemeli sinir ağı, çeşitli parametrelerle ötelenmiş, ölçeklenmiş ve döndürülmüş görüntülerden elde edilen özellikler ile eğitilmiştir. Çakıştırma aşamasında, eğitilmiş ağılar test görüntülerinden çıkarılan özellikler ile beslenmiş ve kayıt parametreleri çıktıda gözlenmiştir.

Elde edilen sonuçlar kullanılan farklı sinir ağıları açısından karşılaştırıldı. Sonuçlar, aşırı öğrenme makinelerinin, gürültünün varlığında diğer iki sinir ağından daha iyi sonuç verdiğini göstermiştir.

Keywords: Görüntü çakıştırma, aşırı öğrenme makinesi, ileri beslemeli sinir ağı, radyal tabanlı sinir ağı, çevrimiçi sıralı aşırı öğrenme makinesi

CONTENTS

	Page
M.Sc THESIS EXAMINATION RESULT FORM	ii
ACKNOWLEDGEMENTS	iii
ABSTRACT	iv
ÖZ	vi
LIST OF FIGURES	x
LIST OF TABLES.....	xii
 CHAPTER ONE - INTRODUCTION	1
1.1 Introduction	1
1.2 Outlines.....	3
 CHAPTER TWO – THEORY OF IMAGE REGISTRATION	5
2.1 Introduction.....	5
2.2 Theory Definition.....	5
2.3 Transformations	5
2.3.1 Translation	6
2.3.2 Rigid Transform	7
2.3.3 Similarity Transform	8
2.3.4 Affine Transform	9
2.3.5 Projective Tranform.....	10
2.4 Image Verification.....	11
2.4.1 Corrected distortion	11
2.4.2 Uncorrected distortion	11
2.4.3 Variation of interset	12

2.5 Rectification	12
2.6 Methodology of Image Registration	12
2.7 Image Registration Steps	13
2.7.1 Detection of features	13
2.7.2 Feature Matching process	13
2.7.3 Estimation of transformation	13
2.7.4 Image transformation	14

CHAPTER THREE–IMAGE REGISTRATION USING NEURAL NETWORK **15**

3.1 Theory of Neural Network	15
3.2 Architecture of Neural Network	16
3.2.1 Single Layer Feedforward Networks	16
3.2.2 Multilayer Feedforward Networks	17
3.2.3 Recurrent Networks	18
3.3 How Do Neural Networks work?	19
3.4 Transfer Functions	20
3.4.1 Identity Function.....	20
3.4.2 Step Function.....	21
3.4.3 Sigmoid Function	21
3.4.4 Bipolar Sigmoid Function	22
3.5 Feedforward Neural Networks	22
3.5.1 Single-layer Perceptron	23
3.5.2 Feedforward Multilayer Perceptron.....	24
3.6 Backpropagation Algorithm	25
3.6.1 Standard Backpropagation For Multilayer Layer Perceptron	25

3.7 Accerlated Learning Backpropagation Algorithm	26
3.7.1 Levenberg Marquardt Backpropagation Algorithm	26
3.8 Stochastic and Batch Methods	27
3.9 Radial Basis Neural Networks	27
3.9.1 RBFNN Algorithm	28
3.10 Ensembling Extreme Learning Machine	29
3.10.1 En-ELM Algorithm.....	31
3.11 Online Sequential Extreme Learning Machine	32
3.11.1 OS-ELM Algorithm	32
3.12 Neural Networks Based Image Registration.....	32
3.13 Two Dimentional Discrete Cosine Transform	33
CHAPTER FOUR– EXPERIMENTAL WORK.....	35
4.1 Data Sets for DCT.....	35
4.2 Experiment with DCT Coefficients.....	36
CHAPTER FIVE– CONCLUSION	52
REFERENCES	53

LIST OF FIGURES

	Page
Figure 2.1 Triangle A is the original shape and triangle B is the translated triangle A	7
Figure 2.2 Triangle A is the original shape and triangle B is the translated and rotated triangle A	8
Figure 2.3 Hexagon A is the original shape and hexagon B is the translated, rotated, scaled hexagon A	9
Figure 2.4 Triangle A is the original shape and triangle B is the affine transformed triangle A	10
Figure 2.5 An example of projection transformation	11
Figure 3.1 Single-layer feedforward neural networks	17
Figure 3.2 Multi-layer feedforward neural networks	18
Figure 3.3 Recurrent network.....	19
Figure 3.4 Single input neuron model	20
Figure 3.5 Identity function.....	20
Figure 3.6 Step function.....	21
Figure 3.7 Log-sigmoid function.....	22
Figure 3.8 Bipolar sigmoid transfer function	22
Figure 3.9 Step threshold function	23
Figure 3.10 Feedforward multilayer perceptron.....	24
Figure 3.11 RBFNN Architecture	29
Figure 3.12 ELMNN Architecture	31
Figure 3.13 Neural Networks Based Image Registration	33
Figure 4.1 Reference image of girl	39
Figure 4.2 Moon image	40
Figure 4.3 Aerial image	40
Figure 4.4 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in MRCD	43
Figure 4.5 The mean absolute errors at 20 dB noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in MRCD	43

Figure 4.6 The mean absolute errors at 5 dB noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in MRCR	44
Figure 4.7 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the Moon image in MRCR	44
Figure 4.8 The mean absolute errors at 20 dB noise (ELM, OS-ELM, RBFNN and FNN) for the Moon image in MRCR	45
Figure 4.9 The mean absolute errors at 5 dB noise (ELM, OS-ELM, RBFNN and FNN) for the Moon image in MRCR	45
Figure 4.10 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in MRCR	46
Figure 4.11 The mean absolute errors at 20 dB noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in MRCR	46
Figure 4.12 The mean absolute errors at 5 dB noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in MRCR	47
Figure 4.13 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in LRFR	47
Figure 4.14 The mean absolute errors at 20 dB noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in LRFR	48
Figure 4.15 The mean absolute errors at 5 dB noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in LRFR	48
Figure 4.16 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the moon image in LRFR	49
Figure 4.17 The mean absolute errors at 20dB noise (ELM, OS-ELM, RBFNN and FNN) for the moon image in LRFR	49
Figure 4.18 The mean absolute errors at 5dB noise (ELM, OS-ELM, RBFNN and FNN) for the moon image in LRFR	50
Figure 4.19 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in LRFR	50
Figure 4.20 The mean absolute errors at 20dB noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in LRFR	51
Figure 4.21 The mean absolute errors at 5dB noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in LRFR	51

LIST OF TABLES

	Page
Table 4.1 Parameters performed in experiment for LRFR and MRCR.....	36
Table 4.2 Mean absolute error of ELM, OS-ELM, RBFNN and FNN (MRCR) for DCT features	41
Table 4.3 Mean absolute error of ELM, OS-ELM, RBFNN, FNN (LRFR) for DCT	42



CHAPTER ONE

INTRODUCTION

1.1 Introduction

Image registration is the process of aligning a number of images obtained from various sensors or from a same sensor at various time intervals. The use of image registration can be found in various areas such as multi spectral classification, change detection, environmental monitoring, medical etc (Saxena, & Singh, 2014). Two images to be registered, initially a transformation is calculated. This transformation simply maps each pixel in one image to similar pixels on the other.

In the image registration procedure there are some problems to deal with such as illumination changes, intensity variation, noise and geometric distortion.

Image registration basically consists of feature and area based methods (Zitova, & Fusser, 2003). In feature-based method, two sets of salient features represented by control points required to be taken out from the images to be registered and these features can be corners, edges, landmarks, etc. The estimation of the transformation parameters by which two images are overlapped is done by finding point wise corresponding between control points sets (control point marching). In area-based methods, there is no need for features (control points) to be extracted. Instead of features, sub images are taken out from the images to be registered and matched to estimate the transformation parameters.

A new method which is not resemble to any of the method above is image registration based on neural networks. In this technique, initially training process of the networks is performed by using global features (not control points) as input extracted from images which has been transformed by previously defined parameters, and at the output of the neural networks the registration parameters are obtained. Thereafter, the trained neural networks estimate the unknown parameters

of the questioned image, by setting the global features at the input of trained neural networks.

The initial use of neural networks in image registration is 1994, by Piraino et al (Piraino et al., 1994). The authors used a neural network to register 3-D medical images. The neural network succeeded with accuracy in the correlation between an image and its corresponding affine transformed image, while the accuracy decreases when a noise is appended or the correlation is nonlinear mapping. This problem can be solved by expanding the size of the training set. Another approach, suggested the use of Hopfield neural network to match a set of control points extracted from the images to be registered (Qian , &, Li , 1997). In work the matching process was treated with minimization of the energy function of the Hopfield neural network. A new scheme was proposed by Elhanany et al (Elhanany et al., 2000). The authors used a feedforward neural network to estimate the registration parameters (affine transform parameters). In their scheme, the network is trained with discrete cosine transform (DCT) features, these features represent the global features of reference image which has been translated, rotated and scaled. Then the neural network is tested to see the performance by simply providing the DCT features of the test image to inputs of the neural network and calculate affine transformation parameters at the output. A similar work proposed by Abche et al. (Antoine et al., 2006). In this approach, DCT features are replaced with Fourier transform (FFT) coefficients. Besides that, the registration parameters were estimated simultaneously, which helped in getting better -optimized set of registration parameters. In (Sarnel et al., 2011), the authors used radial basis function neural network with DCT features to estimate the affine transform parameters. The results showed that RBFNN increased the accuracy of the estimation in additive noise condition and overcome the long time of the training. In another research (Gadde, &, Yu, 2016), the authors used combined features with a neural network to calculate the affine transformation parameters. The combined features are from spatial domain represented by scale invariant feature transform (SIFT), and from frequency domain which represented by DCT features or discrete wavelet transform (DWT) features. Another approach in

(Xu et al., 2006), the authors used Kernel independent component analysis (KICA) in the training of the neural network.

In this thesis, the registration parameters (i.e. affine transformation parameters) were estimated by various types of neural networks. The set of data, are the features of global images which were extracted from affine transformed images by using transformation technique which is discrete cosine transform (DCT). The used network schemes were feed-forward neural network (FNN), radial basis function neural network (RBFNN), extreme learning machine (ELM), and online sequential extreme learning machine (OS-ELM). All network schemes were trained and tested using DCT. The performances of these schemes were compared in the last two chapters.

The aim of this thesis was to replace RBFNN and FNN with OS-ELM and ELMNN. This replacement is due to the drawbacks in FNN and RBFNN schemes. The emperical results show that OS- ELM in comparison with RBFNN and FNN gives accurate estimated results in availability of additive noise. Moreover, ELM is trained in short time interval.

1.2 Outline

This thesis contains five chapters and theses chapters are presented as follows:

Chapter 1: Chapter 1 presents image registration summary and describes the use of neural network in image registration.

Chapter 2: Chapter 2 presents image registration theory , types of transformations , proplems in image registration and image registration methodology.

Chapter 3: Chapter 3 presents general review of neural networks, types of neural networks, how do neural networks work, learing alogirhms FNN, RBFNN theory and ELM theory.

Chapter 4: Chapter 4 presents the steps of image registration using neural networks, data set which were generated by discrete cosine transform (DCT) , experiments carried by using different schemes which are FFNN, RBFNN, OS-ELMNN and ELMNN. The results obtained from neural network schemes are given in tables and also detailed comparison is provided between all used schemes.

Chapter 5: Chapter 5 presents the conclusion.



CHAPTER TWO

THEORY OF IMAGE REGISTRATION

2.1 Introduction

Image registration is a process that search best geometric transformation to align at least two images, that are taken from same object at various time instances or employing various sensors in one coordinate system. Image registration problem is often required by applications such as geographic information systems (GIS), medical image analysis, remote sensing computer vision and pattern recognition, weather and environment monitoring, image fusion and image mosaicking (Zitova ,& Flusser, 2003).

2.2 Theory Definition

To register image I_1 (sensed image) is to map it to image I_2 (reference image), so image registration process is to be considered as a transformation between two images according to the property of the image that needs to be registered i.e. (intensity). This mapping is spatial mapping, if the sensed image and reference image are 2D arrays, then the transformation can be defined by (R. Kokila. 2018)

$$I_2(x, y) = g(I_1(f(x, y))) \quad (2.1)$$

where f is a 2D transformation function which transform coordinates $[x, y]$, to a new coordinates $[x, y]$ and g is an intensity function.

2.3 Transformations

To align two images or more a mapping or transformation is used. This mapping can be defined as four operations named as (Scaling, Rotation, Translation and Shearing). The types of transformation are different from each other in terms of used mapping or transformation operator is used. These type of transformations are defined below:

2.3.1 Translation

Translation of the image to be registered is used corresponding to the reference image, the relation of the related points in two images can be described as

$$X = x + h \quad (2.2)$$

$$Y = y + k \quad (2.3)$$

where h and k are the horizontal and vertical pixel displacement, respectively, x and y are the origin coordinate of the image to be registered and X and Y are the coordinate of the image to be registered after translation. An example of translation is given in figure 2.1.

Equation 2.2 and equation 2.3 can also be written in matrix form as

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2.4)$$

In simple form

$$P = TB \quad (2.5)$$

where, P and B are coordinates of image to be registered and reference image respectively, and transformation matrix is represented by T . The transform matrix provides the translation in between the image to be registered and the reference image by parameters (h and k). X and Y represent the translated image coordinates.

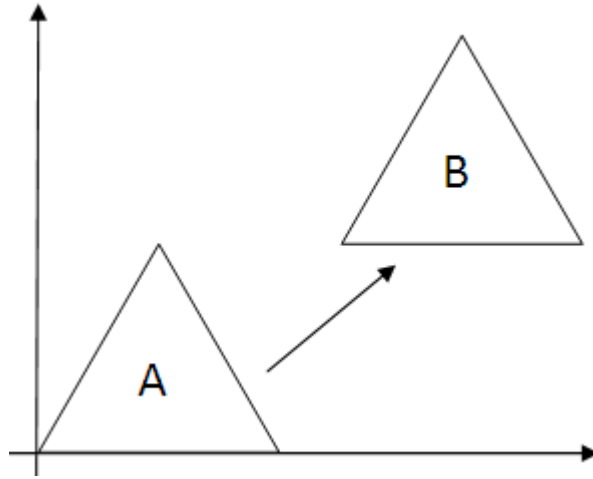


Figure 2.1 Triangle A is the original shape and triangle B is the translated triangle A

2.3.2 Rigid Transform

If an observed image is registered according to a reference image by translation and rotation, in this consideration the length between corresponding pixels and lines will be consistent from reference image to observed image. This transform is called as rigid transform and expressed as

$$X = x \cos \theta - y \sin \theta + h \quad (2.6)$$

$$Y = x \sin \theta + y \cos \theta + k \quad (2.7)$$

Equation 2.6 and equation 2.7 can also be written in matrix form as

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2.8)$$

where, θ is an angle measured in counter-clockwise direction to show orientation difference of observed image according to the reference image. The coordinates between observed and reference images determine the parameters θ , h and k . Figure 2.2 illustrates an example for image translation and rotation for rigid transformation.

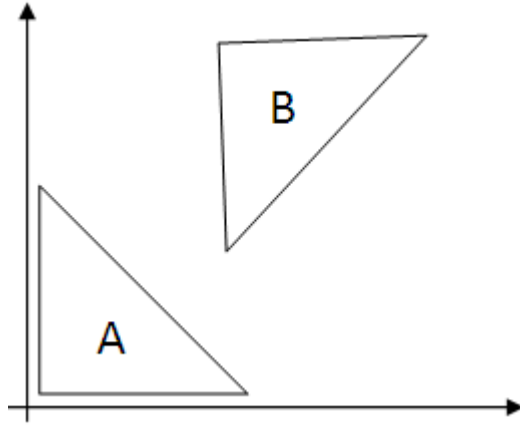


Figure 2.2 Triangle A is the original shape and triangle B is the translated and rotated triangle A

2.3.3 Similarity Transformation

In this transform the observed image is translated, rotated and scaled according to the reference image, which is illustrated in figure 2.3. The relation between corresponding points of reference image and observed image is given as

$$X = sxcos\theta - sysin\theta + h \quad (2.9)$$

$$Y = sxsin\theta + sycos\theta + k \quad (2.10)$$

where s , θ , h , and k are scaling, rotation and translation parameters, respectively.

The equations 2.9 and equation 2.10 can be written in the following matrix form

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2.11)$$

In implicit form it is represented as

$$P = TRSB$$

Where, T , R , S are translation, rotation and scaling matrices, respectively. The coordinates between observed and reference images determines the parameters θ , h and k . The scaling parameter s is determined by dividing the length of segmented

lines of two pixels in observed image by the length of segmented lines of two pixels in reference image.

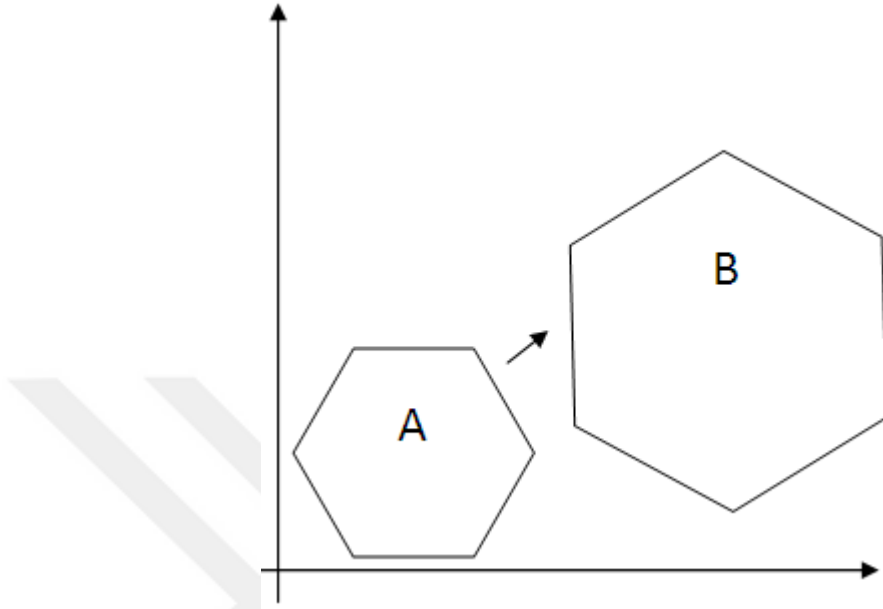


Figure 2.3 Hexagon A is the original shape and hexagon B is the translated, rotated, scaled hexagon A

2.3.4 Affine Transformation

Affine transform is a linear method and used for registration images by means of translate, rotate, scale, and shearing by preserving the parallelism figure 2.4. It can be expressed as

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & a & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2.12)$$

In implicit form

$$P = TRSEB$$

where, E is shearing matrix

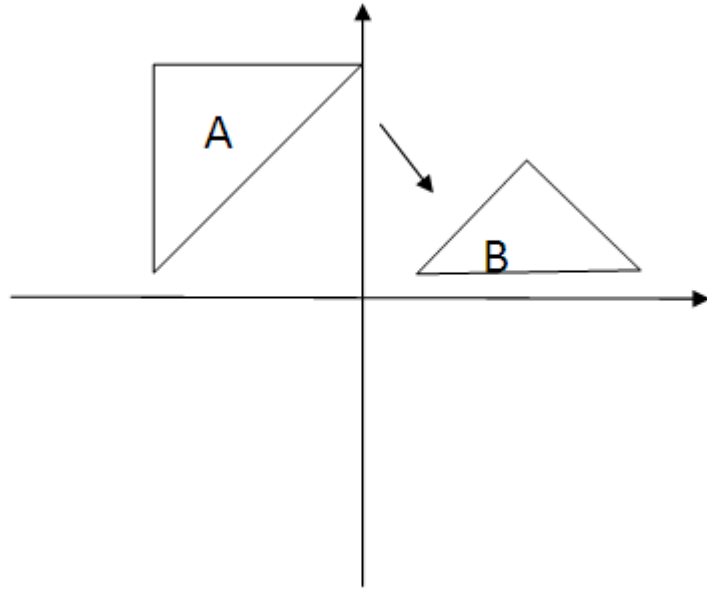


Figure 2.4 Triangle A is the original shape and triangle B is the affine transformed triangle A

2.3.5 Projective Transformation

If two images belong to a flat scene a projective transformation is used to relate the corresponding points figure 2.5. Projective transform can be described by

$$X = \frac{a_1x+a_2y+a_3}{a_7x+a_8y+1} , Y = \frac{a_4x+a_5y+6}{a_7x+a_8y+1} \quad (2.13)$$

The equation 2.13 can also be written in matrix form as

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

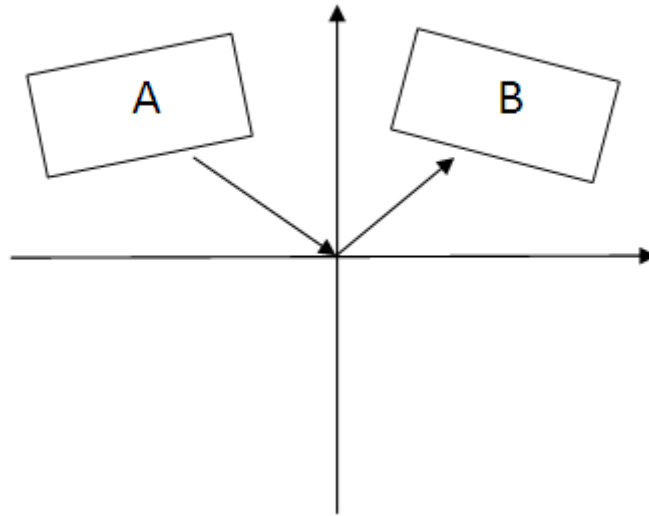


Figure 2.5 An example of projection transformation

2.4 Image Variations

To remove the distortion and detecting the changes between images, image registration process is applied. Distortions are differences between sensed and reference image and these differences are the source of misregistration (Brown., 1992). These misregistrations can be classified as;

2.4.1 Corrected Distortions

Corrected distortion is regarded to the changes in viewpoints and sensor noise. The modeling of these distortions, which are geometric, can be done by determining the type of transformation that will register the images.

2.4.2 Uncorrected Distortions

Uncorrected distortion are regarded to the scene, examples of these distortions are light, variations in the atmosphere, shadows and volumetric sensor noise.

2.4.3 Variations of Interset

The purpose of image registration is to detect the variations between two images. These variations are cause of movements of the object, growth of the object and sensitivity variations of the sensors.

2.5 Rectification

If the observed scene approximately flat and when the examined geometry is clearly known, the registration process is performed by rectification. The rectification process is used to project images to the same image plane. This process simply calculates the matching points of two images.

2.6 Methodology of Image Registration

Image registration applications are categorized into four main groups with respect to the acquisition of the images. These categorization can be defined as; different viewpoints, different times, different sensors and scene to model registration (Saxena et al., 2014).

Different viewpoints (multiview analysis): Various images are collected from various line of sight for one scene for multiview registration. Examples of applications are computer vision shape recovery, remote sensing and etc.

Different times (multitemporal analysis): In this method, images are collected from the same place at various time intervals possibly with changing conditions. The idea is to obtain the differences in the scene. The application area is monitoring of global land usage and remote sensing.

Different sensors (multimodal analysis): As clear from the name various sensors are used to obtain images from the same place. The collected images are integrated for comparison with applications. One application example is the image fusion.

Scene to model registration: This process compares a collected image with a model image and this is performed by image registration. Examples are quality control of manufactured objects, medical image analysis.

2.7 Image Registration Steps

Image registration steps can be described in four subsections (Zitova et al., 2003).

2.7.1 Detection of Features

In this step, the distinctive and salient objects (edges, line intersections, counters, corners, etc) are captured and extracted from a reference and observed images. These features can be localized by their exact positions (distinctive point, line endings, and centers of gravity) and called as control points.

2.7.2 Feature Matching Process

In feature matching, first of all the features are obtained from both reference and sensed image and then the correspondence between these features are established. For that purpose, the concentration goes to different descriptors or features of the images.

2.7.3 Estimation of Transformation

To generate transform model, the parameters of the mapping function is estimated in a way that reference image perfectly matches with the sensed image.

2.7.4 Image Transformation

In this step, the observed or sensed image is transformed using obtained transform model.



CHAPTER THREE

IMAGE REGISTRATION USING NEURAL NETWORKS

3.1 Theory of Neural Network

Neural networks are signal processing systems and combine processing units, called neurons, distributed in parallel, which has similar properties to biological neural system. The motivation of the neural networks is an attempt to build mathematical model that can realize the function of biological neuron.

A large number of processing elements called processing units or nodes are connected in various ways to set up neural networks. Connection between nodes is realized by synaptic strength called weight that represents information which is required by the network to solve a problem (Fausett., 1994).

The neural network has ability to do various tasks such as:

- Store and recall information,
- Classify patterns,
- mapping from inputs pattern to output pattern,
- finding solutions to optimization problems,
- Clustering similar patterns,

Neural network can realize the function of the biological neuron through adjusting the values of synaptic weights by learning process. The learning process is mainly divided into two methods in terms of input-output pairs: Supervised learning, which is most used technique, requires both input samples (stimulus) and corresponding output samples (targets) taken from the same system. In this method the weights are changed until the variation between the actual output (calculated network outputs) and the target dataset is minimized in an acceptable level.

In unsupervised learning technique, the target values cannot be available and the neural network training is performed by the related algorithm by acting on the input

data without any guidance. The task of the network is to cluster information with respect to similarities among input data.

In this research only three types of networks were used as extreme learning machine neural network, radial basis function neural network, and feedforward neural network. All of three schemes use supervised learning.

3.2 Neural Network Architecture

Architecture of neural network is an arrangement of the neuron units into layers and these layers are connected with strength weights. Simple neural network architecture is given in figure 3.1, in which there is an input layer to receive only inputs and an output layer. In general, the neural network is named as single layer or multilayer. In multilayer the number of the layers except the input layer determines the number of the layers which perform a computation. Input layer has no processing unit, and its function is to pass the given input to the following layer, called as hidden layer.

3.2.1 Single Layer Feedforward Neural Networks

A single layer neural network as illustrated in figure 3.1 has only one layer of connection weights from input to output neurons. Input units, which receive the signal from outside and have no processing unit. The input connection just passes the input data to output neurons through the weights.

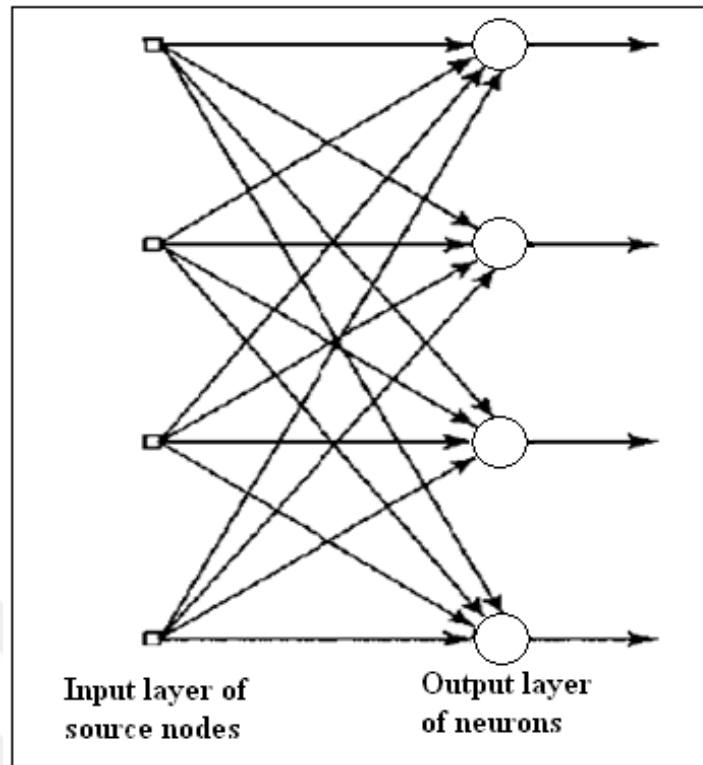


Figure 3.1 Single-layer feedforward neural networks (Haykin., 2009)

3.2.2 Multilayer Feedforward Networks

Multilayer feedforward neural network, illustrated in figure 3.2, is the more comprehensive network consists of input layer, hidden layer, output layer. There are two layers of connection weights, which are between input and hidden layer and the hidden and output layer. In this network, the computation is performed only in the hidden and output layers.

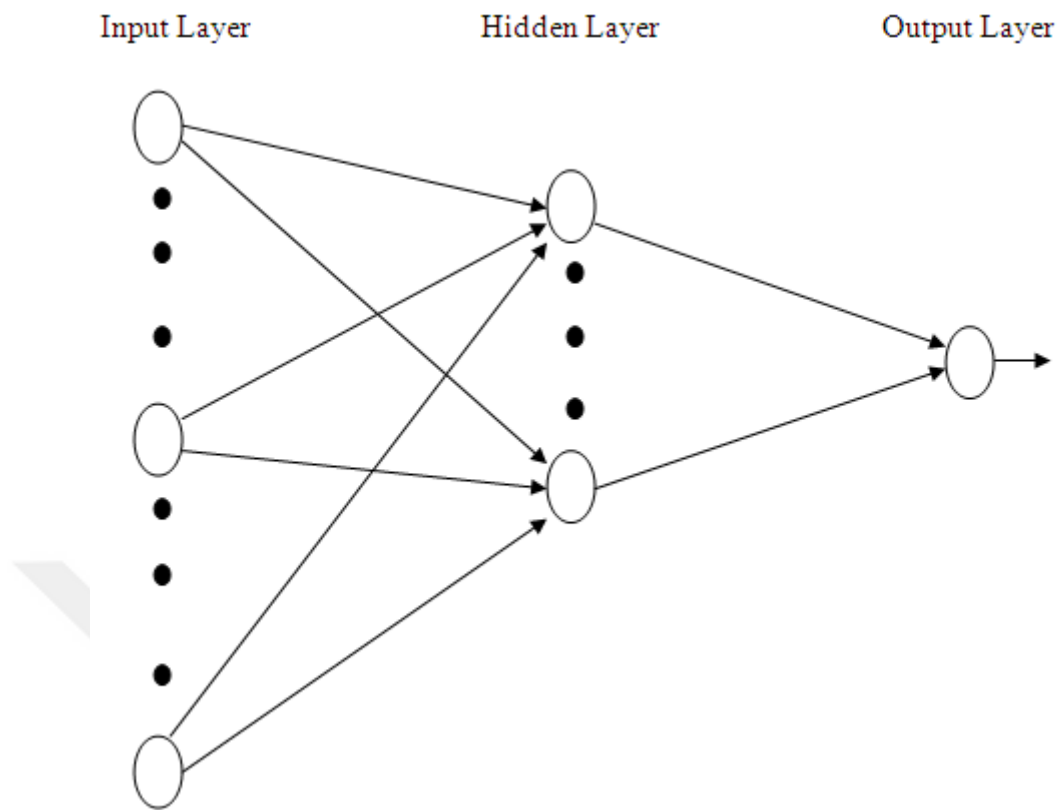


Figure 3.2 Multi-layer feedforward neural networks

3.2.3 Recurrent Networks

Recurrent neural networks also called dynamic neural networks have feedback loops from input to output. Figure 3.3 shows the recurrent neural network. The presence of the feedback loop enables the network having a memory for past states.

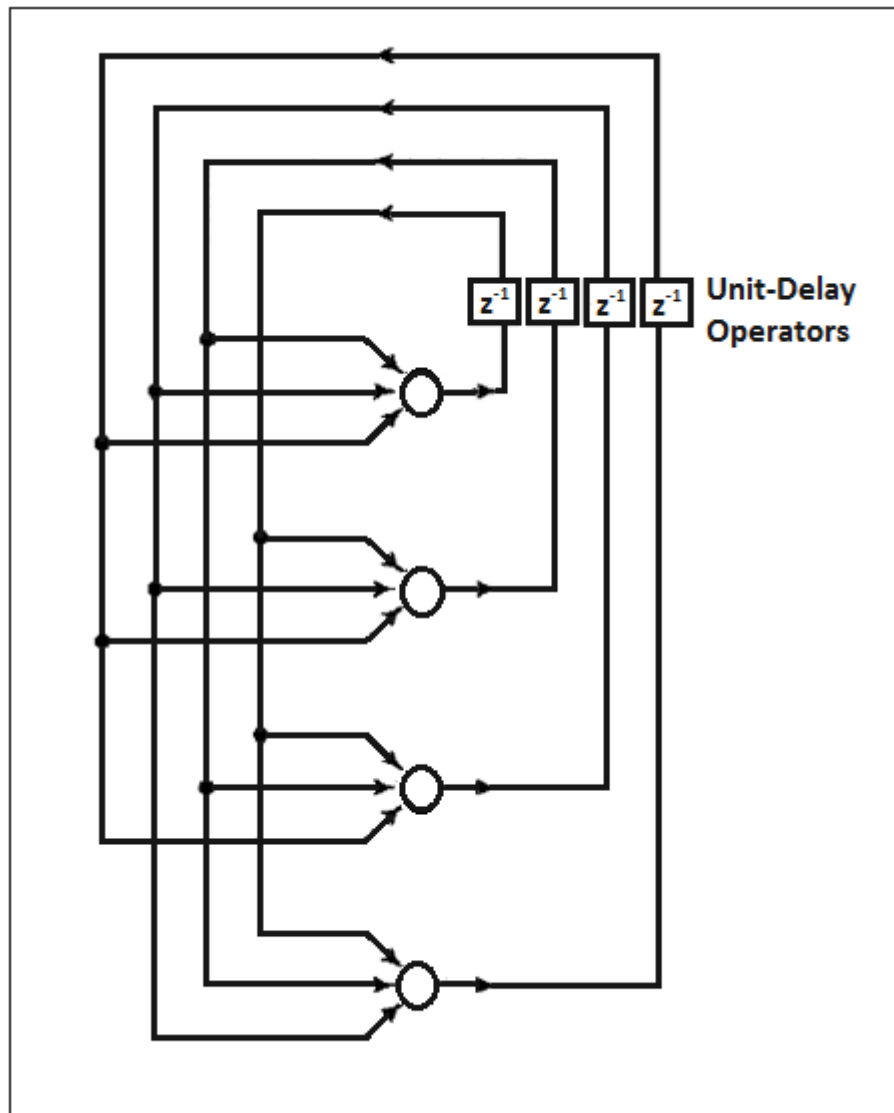


Figure 3.3 Recurrent network

3.3 How Do Neural Networks Work?

Figure 3.4 shows a single input neuron model. In this figure X represent the input of the model, w corresponds to the weight that connect the input to the summation unit, b represent threshold value of the model, f represents activation function (or transfer function) which is linear or non-linear and O is the output of the neuron. In order to calculate the output O , firstly the input X is multiplied by corresponding weight value, then calculated value is summed with the threshold value b , and N represent input of activation function. Taking the transfer function of summation unit the output O is calculated, which is mathematically defined by $O = f(X.W + b)$. In

order for the neuron to give the desired output the values of W and b should be adjusted by a proper learning algorithm.

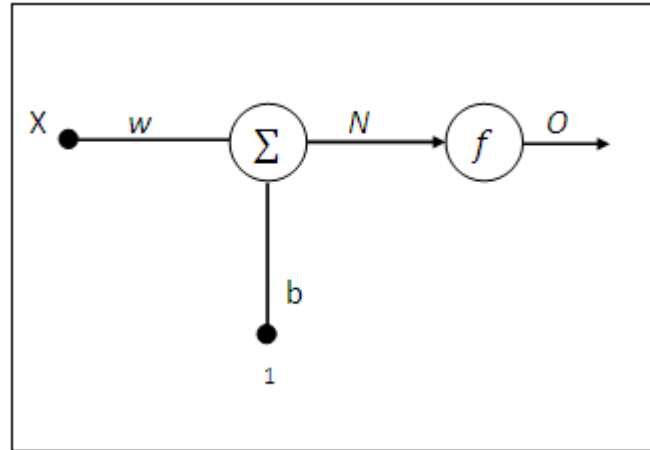


Figure 3.4 Single input neuron model

3.4 Transfer Function Types

There are various transfer functions that are described below.

3.4.1 Identity Function

Figure 3.5 Shows the graph of the identity function sometimes is called linear function. The input and the output of this function are equal to each other or output follows the input, and mathematically can be expressed as $f(x) = x$.

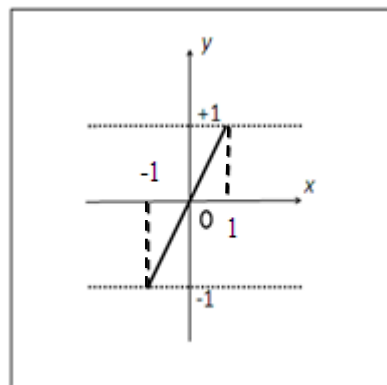


Figure 3.5 Identity function

3.4.2 Step Function

The step function is illustrated in figure 3.6 and it is mathematically expressed by

$$f(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (3.1)$$

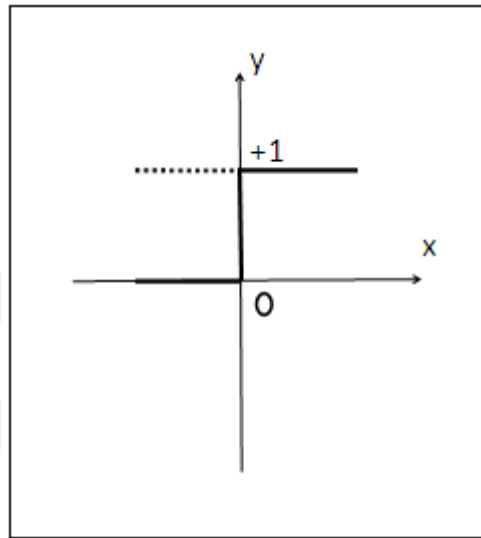


Figure 3.6 Step function

3.4.3 Sigmoid Function

Figure 3.7 illustrates the sigmoid type function. The biggest advantage of this function is that its derivative can be taken and it is a nonlinear function and used in solving nonlinear tasks. The sigmoid type function ranges in $[0, 1]$ for all $x \in \mathbb{R}$ and mathematically has the following expression

$$y = \text{logsig}(x) = \frac{1}{(1+10^{(-x)})} \quad (3.2)$$

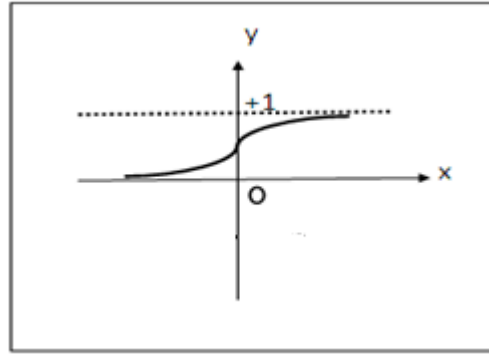


Figure 3.7 Log-Sigmoid function

3.4.4 Bipolar Sigmoid Function

Figure 3.8 shows the graph of the Bipolar sigmoid transfer function, the output of this function lies between $[-1, 1]$. The mathematical expression of this Bipolar sigmoid function is written as

$$f(x) = \frac{2}{1+e^{(-\sigma x)}} - 1 \quad (3.3)$$

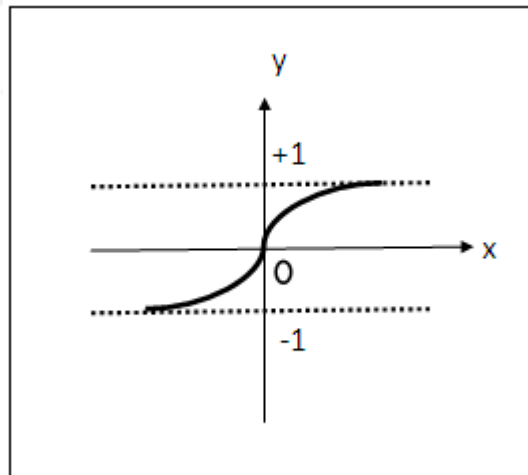


Figure 3.8 Bipolar sigmoid transfer function

3.5 Feedforward Neural Networks

In feedforward neural network structure connections between the neurons are done in between two consecutive layers and do not generate a recurrent loop. It may be a single layer or multilayer neural network, in both the information move from the input layer up to the output layer in one direction.

3.5.1 Single-layer Perceptron

Single-layer perceptron performs a transformation from multidimensional input space to a single output. Single-layer perceptron structure consists of input neurons, output neurons and connections between input-output neurons. To calculate the output of the neuron, initially the sum of the multiplication of the inputs and the corresponding connection weights is calculated, then the output of each neuron is compared with the threshold value. If the calculated value of the summation is bigger than the threshold value then the neuron becomes active, otherwise it becomes inactive. Figure 3.9 shows a step threshold function, with threshold value T .

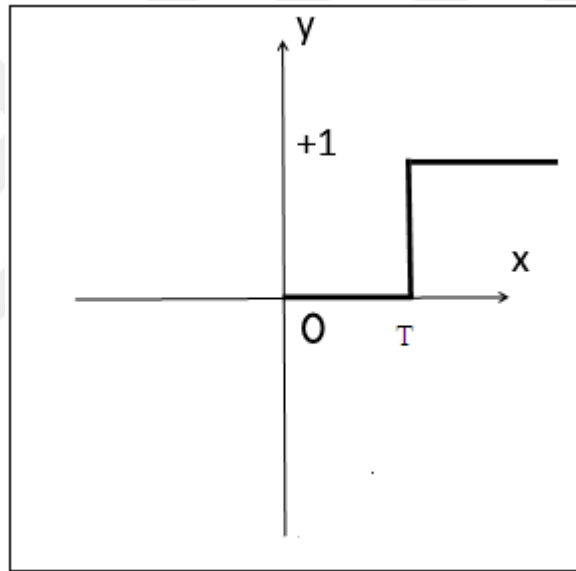


Figure 3.9 Step threshold function

Widrow-Hoff delta rule is algorithm used to train Single-layer perceptron. In this algorithm, the weights are updated as

$$w_i(t+1) = w_i(t) + \mu e(t) x_i(t) \quad (3.4)$$

where, $w(t)$ is the weights, $e(t)$ represents the error between desired and calculated output and μ is the learning parameter that takes values as $0 \leq \mu \leq 1$. The disadvantage of the single-layer perceptron is that it can only solve the linearly

seperable problems. The more complicated problems are solved by multi-layer perceptrons.

3.5.2 Feedforward Multilayer Perceptron

Feedforward multilayer perceptron is a neural network, in which neurons are arranged in cascaded neuron layers. This network structure has three layers as input layer, hidden layer and output layer. Figure 3.10 shows the connection of related layers. All these layers are connected in forward direction via synaptic weights. The signal from outside world goes to the input layer and these inputs are distributed to the neurons of first hidden layer. The following hidden layer neurons act in the same forward direction if any available, while the output layer gets the input from the last hidden layer and transfer to outside as output of the network. Multilayer perceptron can have any number of hidden layer depending on the complication of input feature structure. The mission of the hidden layer neurons is to extract the information from input data.

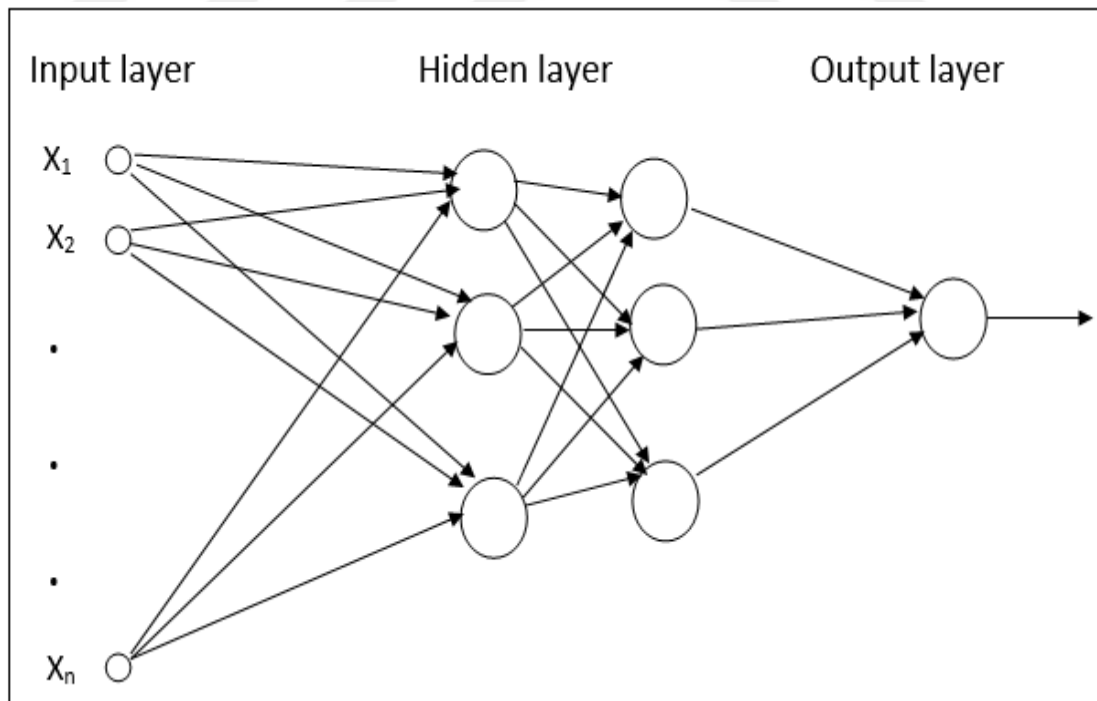


Figure 3.10 Feedforward multilayer perceptron

3.6 Backpropagation Algorithm

In neural network training generally backpropagation algorithm is used. By the backpropagation algorithm, MLP can perform non-linear mapping and association task. The training by backpropagation algorithm is based on the steepest gradient descent approach, which is used to minimize the energy function representing the error. The training of MLP using backpropagation algorithm is carried in two phases.

Feedforward phase, during this phase the input nodes receive the signal and send it to the hidden nodes. Then the hidden nodes compute its activation and send to the output node. Finally, the output nodes compute its activation and offer it as actual output of the neural network.

In training phase, during this phase the difference between actual output and the target is calculated, representing the error. Then the error is fed to the network in the backward direction. To minimize the error the values of the weights are adjusted. (Ham & Kostanic., 2001).

3.6.1 Standard Backpropagation Algorithm for Feedforward Multilayer perceptron

Step 1. Initialize the network weights randomly.

Step 2. From the training data set, present an input data pattern and calculate the network response.

Step 3. Calculate the local error for the hidden layer and for the output layer using the following equation.

The local error for output layer:

$$\delta_j^{(s)} = (d_{qh} - y_{out,j})g(v_j^{(s)}) \quad (3.5)$$

Where δ is the local error, d_{qh} is the desired output, $y_{out,j}$ is the actual output, $v_j^{(s)}$ is the activation function of the neuron and $g(\cdot)$ is the derivative of the nonlinear activation function.

The local error for hidden layer:

$$\delta_j^{(s)} = (\sum_{h=1}^{n_{s+1}} \delta_h^{(s+1)} w_{hj}^{(s+1)}) g(v_j^{(s)}) \quad (3.6)$$

Where w_{hj} is the connection weight.

Step 4. Update the weights according to

$$w_{ji}^{(s)}(k+1) = w_{ji}^{(s)}(k) + \mu^{(s)} \delta_j^{(s)} y_{out,i}^{(s)} \quad (3.7)$$

Step 5. If the network has not converged, continue steps 2 through (Ham & Kostanic., 2001).

3.7 Accelerated Learning Backpropagation Algorithm

Learning MLPNN with backpropagation algorithm is actually the minimization of square of the error, which is called as mean square error (MSE), between the desired and actual outputs. The MSE problem is studied using numerical analysis. Acceleration of the backpropagation Learning algorithm requires the use of advanced numerical analysis techniques.

3.7.1 Levenberg-Marquardt Version of Backpropagation Algorithm

Step 1. Initialize the connection weights.

Step 2. Calculate the network output for the given input pattern.

Step 3. Calculate Jacobian matrix associated with input-output pair using

$$J_{i,j} \approx \frac{\Delta e_i}{\Delta w_i} \quad (3.8)$$

Where Δe_i present the change in output error with respect to the perturbation of the weight Δw_i

Step 4. When all input-output are presented update the weights using

$$w(k+1) = w(k) + [J_k^T J_k + \mu_k I]^{-1} J_k^T e_k \quad (3.9)$$

Step 5. Jump to step 2 if convergence of the network is not satisfied, otherwise stop (Ham et al., 2001).

3.8 Stochastic and Batch Methods

For any given input-output data pair, the back-propagation algorithm updates the network weights in one of two methods. In sequential updating method the weights are updated in a sequential manner, one pattern after another. This method is suitable for classification problems. In batch updating method, the updating of the weights and the biases, are done on an epoch-by-epoch basis. The batch method gives a more accurate estimation of the gradient vector, beside that it's suitable for nonlinear regression analysis and its applications. For large data set size batch method generates disadvantage because of long calculation time duration.

3.9 Radial Basis Function Neural Network

The radial basis function neural network (RBFNN) has a capability of universal approximation and regularization. Beside that, for nonlinear mapping of input data to output data RBFNN is a good candidate. If a RBFNN is well-chosen and trained, it becomes good approximator for various type of functions.

The architecture of RBFNN consists of three layers, which are named as input layer, hidden layer, and output layer, respectively. Figure 3.10 illustrates a RBFNN architecture.

- The RBFNN is connected to the environment through sensory nodes of input layer. There is no calculation at this layer.
- Calculation is performed at the hidden layer that contains nonlinear processing nodes. Within various functions the Gaussian function is mostly used to calculate hidden layer outputs. The Gaussian function for hidden layer is

$$z_j = e^{-\|x - c_j\|^2 / \sigma^2} \quad (3.10)$$

Where z_j represents j th hidden neuron output value, $x \in \mathbb{R}^{n \times 1}$ is the input, $c_j \in \mathbb{R}^{n \times 1}$ represent RBFNN centre which is chosen from the input set, σ is the spread parameter of the gaussian function, and $\|\cdot\|^2$ is the Euclidean norm.

- Output layer neurons get input from hidden layer through trainable connection weights and responsible to generate the output using the following formulation

$$y_i = \sum_{j=1}^K w_{ij} z_j \quad (3.11)$$

Where, y_i represent the neuron i at the output of RBFNN and w_{ij} represents the connection weights between the hidden and output layers and K represent hidden layer neuron numbers. The weights between hidden and output layer are found by the training process.

3.9.1 RBFNN Algorithm

1. Choose the centres of RBF function from the set of input vectors.
2. Calculate the spread parameters σ using

$$\sigma = \frac{d_{\max}}{\sqrt{K}} \quad (3.12)$$

where d_{\max} is the maximum Euclidean distance between selected centres, and K represent the number of centres.

3. Determine the hidden layer matrix Φ .

$$\Phi = \begin{bmatrix} Z(c_1, x_1, \sigma) & \cdots & Z(c_M, x_1, \sigma) \\ \vdots & \ddots & \vdots \\ Z(c_1, x_N, \sigma) & \cdots & Z(c_M, x_N, \sigma) \end{bmatrix}_{N \times M}$$

4. Find the weight vector of the output layer using.

$$w = \Phi^+ y_d \quad (3.13)$$

where Φ^+ is the pseudoinverse of the nonlinear mapping matrix Φ and y_d represent the desired output (target value).

5. Finally, calculate the RBFNN output. (Ham et al., 2001).

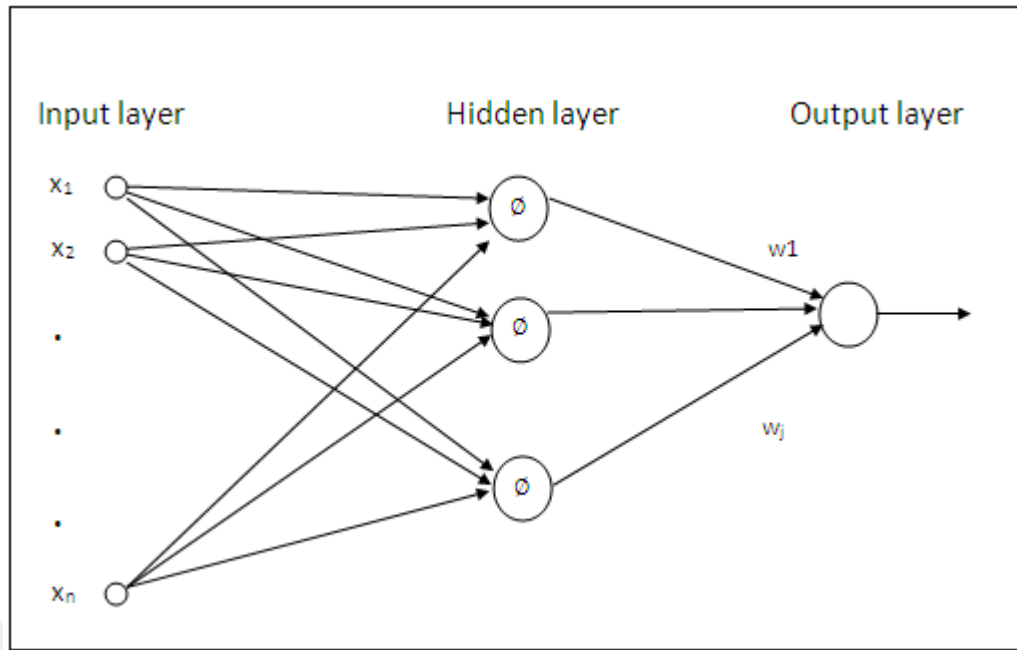


Figure 3.11 RBFNN architecture

3.10 Ensemble of Extreme Learning Machines

Extreme learning machine (ELM) is introduced by (Huang et al. 2004) for obtaining better learning capability for single-hidden layer feedforward neural networks (SLFNs). The advantage of ELM comes from the fast training time with respect to SLFNs. In addition, ELM has universal approximation ability. Operation area of ELMNN is wide and covers compression, clustering, forecasting, classification and regression analysis and so on. The performance of ELM can be increased by various derivatives of ELM. Grouping of several learning machines help to have better performance for generalization and lower the possibility of over fitting problem. In this study, several ELMs were trained with different initial weight and bias values, then the ensemble of independently trained ELMs were averaged.

SLFNs for ELM algorithm consists of fully connected three layers. The connection goes from input to output and the hidden layer is between input layer and output layer. The input layer is responsible of passing the information to hidden layer neurons. The input neurons are connected to hidden nodes through weights which are initially randomly assigned. Hidden layer neurons includes activation functions which includes sigmoid function, hyperbolic tangent function, Gaussian function,

cosine function and not limited to these given functions. There are total of N observed input-output data set to be used to generate the required ELM. Any output of output layer neurons from $i=1, 2, \dots, m$ of the ELM is given as

$$y_i = f_i(x) = \sum_{j=1}^M \beta_{ij} G_j(\mathbf{w}_j, b_i, \mathbf{x}_j) = \mathbf{H}(\mathbf{x})\boldsymbol{\beta}, \quad i = 1, 2, \dots, m \quad (3.14)$$

Where $\mathbf{x}_i \in R^{n \times 1}$ and $\mathbf{y}_i \in R^{m \times 1}$ are input and output vector respectively, $G_j(\cdot)$ is any one of the function of hidden layer node, β_{ij} are the weight values between hidden layer and output layer nodes, \mathbf{w}_j, b_i are the hidden nodes parameters and represent the input weight vector and bias of each hidden neuron respectively, finally M represents the neuron numbers in the hidden layer.

Figure 3.11 gives the schematic illustration of the ELM. The important part of the ELM here is to have randomly selected hidden layer parameters, which are weights and the bias values. These parameters are not changed in an iterative way. Therefore, it brings efficiency to the ELM by speeding up the training process. There are various defined continuous functions for hidden layer nodes and it is possible to select different functions for each node. However, generally for simplicity all nodes have the same function. Functions are selected empirically. For cosine function $G(\mathbf{w}, b, \mathbf{x}) = \cos(\mathbf{w} \cdot \mathbf{x} + b)$, where $\mathbf{w} \cdot \mathbf{x}$ is the inner product of input parameter weight vector and input vector.

The actual output of ELM is $N \times M$ and can be represented as $\mathbf{H}\boldsymbol{\beta}$. The hidden layer mapping \mathbf{H} matrix and output weight vector $\boldsymbol{\beta}$ can be expressed as

$$\mathbf{H} = \begin{bmatrix} G(\mathbf{w}_1, \mathbf{x}_1, b_1) & \cdots & G(\mathbf{w}_1, \mathbf{x}_N, b_1) \\ \vdots & \ddots & \vdots \\ G(\mathbf{w}_M, \mathbf{x}_1, b_M) & \cdots & G(\mathbf{w}_M, \mathbf{x}_N, b_M) \end{bmatrix}_{N \times M}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_M^T \end{bmatrix}_{M \times m}$$

The optimal set of output weights of the ELM is directly calculated by means of mean squared error between the target output and obtained network outputs as $\boldsymbol{\beta} =$

H^+B , where B is the target value of the ELM. H^+ represent the Moore-Penrose generalised inverse of hidden layer mapping matrix H .

3.10.1 ELM Algorithm

1. Assign hidden layer parameters weight and bias values randomly
2. Determine the hidden layer mapping matrix H
3. Determine the output vector weight as $\beta = H^+B$
4. Calculate the output as $y_i = f_i(x)$

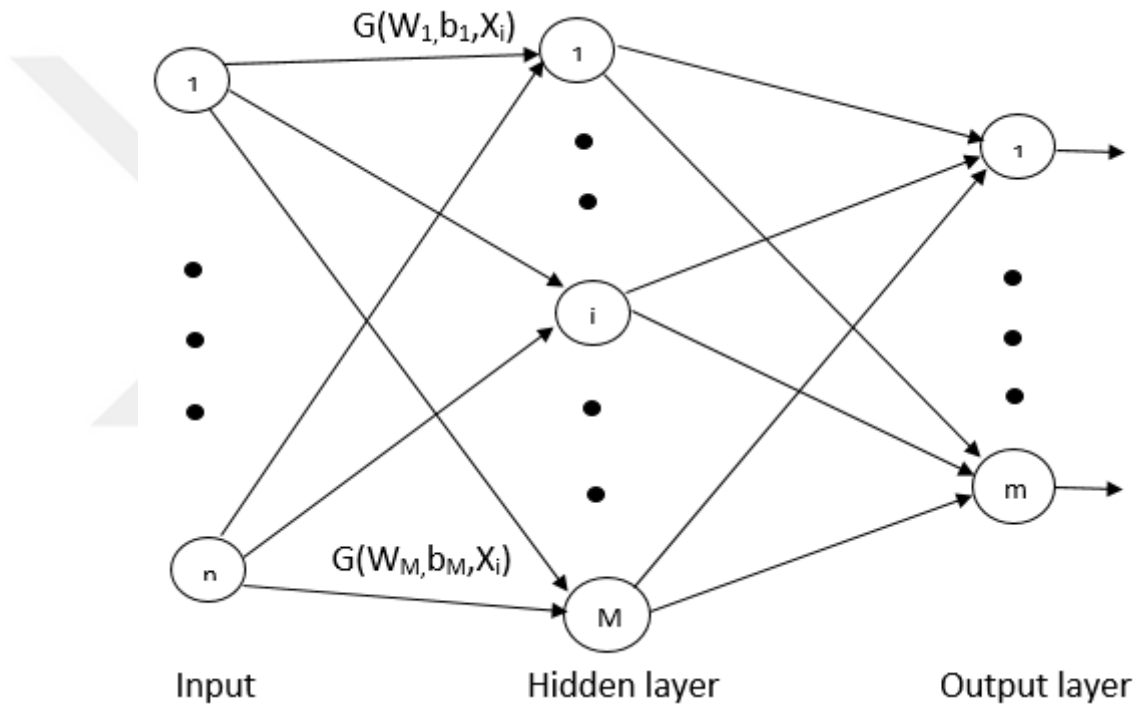


Figure 3.12 ELMNN architecture

3.11 Online Sequential Extreme Learning Machine

In the training of ELMNN all the training set need to be presented at the learning stage. But sometimes all training set may not be available at once (i.e. may come as chunk or one by one). In this case the learning of ELM is done by using online sequential learning algorithm.

3.11.1 Online Sequential Algorithm for ELM

1. Given an initial training set $S_0 = \{x_i, t_i\}_{i=0}^{L_0}$, $L_0 \times N$.
2. Random initialization of input weights and bias.
3. Determine the initial hidden layer matrix H_0 .
4. Calculate the initial output weights $B_0 = M_0 H^T T_0$, where $M_0 = (H^T H)^{-1}$ and $T_0 = [t_1, \dots, t_{L_0}]$.
5. Introduce the new set of data to the neural network $S_{k+1} = \{x_i, t_i\}_{i=L_0+1}^{L_1}$, where S_{k+1} is the new data samples.
6. Calculate the hidden matrix for the new data (k+1) samples H_{k+1}
7. Calculate the output weights B_{k+1} as

$$M_{k+1} = M_k - M_k H_{k+1}^T (I + H_{k+1} M_k H_{k+1}^T)^{-1} H_{k+1} M_k \quad (3.15)$$

$$B_{k+1} = B_0 + M_{k+1} H_{k+1}^T (T_{k+1} - H_{k+1} B_k) \quad (3.16)$$

8. Finally Set $k=k+1$ and start from step 5.

3.12 Neural Network-Based Image Registration

As seen from figure 3.13 image registration based on neural network processing is performed in two stages, which are pre-registration and registration. In pre-registration stage, initially training set is generated by applying affine transformation, which cover translation, rotation and scaling, to given reference images. Then, various amount of noise is added to these images to evaluate system performance. The second step in this stage is to obtain the global features (e.g. FFT and DCT coefficients). This does not help only to reduce the amount of useful data but also speed up the training processing time and also provides better generalisation

performance. The last step is the training of the neural network by use of extracted global features as input to network and the corresponding affine transformed parameters.

In the registration stage, some images called as test images, which are not used in the training stage, are used to obtain global features. Initially, test images are transformed by some parameters. Finally, the trained neural network is inputted with the global features of the transformed test images and expected to estimate the affine transformed parameters.

The use of the neural network in image registration make the procedure of the registration simple, and fast which made it favourable in the application that need to be done in short time. The results show that the neural network gives correct results within the presence of various amount of noise.

The network performance is increased by adding some noise values to training image set. This simply increase the generalisation capability of the neural network and enhance the immunity of the network.

Using affine-transform images which covers complete parameter space, helps the neural network to make better estimation of parameters.

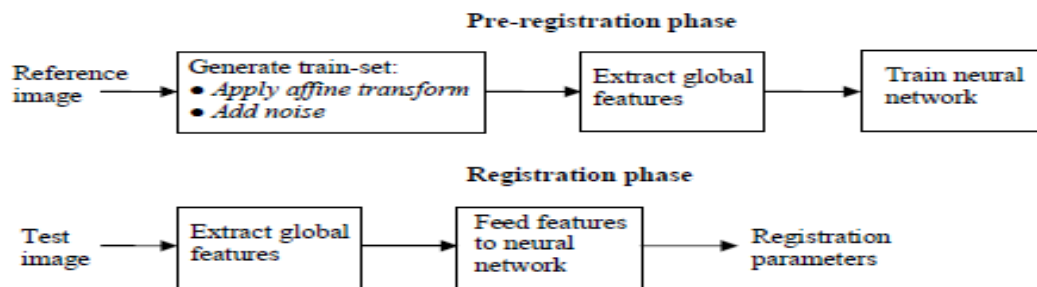


Figure 3.13 Neural network based image registration (Elhanany et al., 2000)

3.13 Two Dimensional Discrete Cosine Transform

Discrete cosine transform (DCT) is a unitary transform invented in 1974 by (N.Ahmed et al.). The DCT has applications in signal and image processing

particularity in compression and decompression. Applying DCT on a set of data, the output is a set of numbers called the DCT coefficients. The first number of the DCT coefficients is the largest number called the DC coefficient, while the rest are small numbers called the AC coefficients. The types of DCT are divided to four types, according to the used cosine basis. The most used DCT is the two-dimensional discrete cosine transform 2D-DCT, and mathematically can be written as

$$G_{ij} = \sqrt{\frac{2}{m}} \sqrt{\frac{2}{n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{m-1} P_{xy} \cos \left[\frac{(2y+1)j\pi}{2m} \right] \cos \left[\frac{(2x+1)i\pi}{2n} \right] \quad (3.17)$$

$$\text{where , } C_i = \begin{cases} \frac{1}{\sqrt{2}} & i = 0 \\ 1 & i > 0 \end{cases} \text{ for } i = 0, 1, \dots, N-1 \quad C_j = \begin{cases} \frac{1}{\sqrt{2}} & j = 0 \\ 1 & j > 0 \end{cases} \text{ for } j = 0, 1, \dots, N-1$$

The P_{xy} is the input data and G_{ij} is the DCT coefficients matrix. The first coefficient G_{00} is called the DC coefficient and the rest of the coefficients are AC. The extraction of DCT features from an image is carried in several steps. At the first, the image is broken into 8×8 blocks, then the DCT eq (3.17) is applied to the image blocks, which results in 8×8 DCT coefficients. After that, the DCT coefficients are quantized and then some of significant DCT coefficients are chosen (David Salomon and Giovanni Motta. Saeed Dabbaghch et al).

CHAPTER FOUR

EXPERIMENTAL WORK

In this research, the image registration process is carried out by means of various neural networks instead of classical methods. Register of an image using neural networks can solve some drawbacks, encountered in the image registration, these drawbacks such as deformation and the noise.

Using the FNN scheme and RBFNN scheme in the image registration leads to some drawbacks in both schemes. Case one, FNN scheme is a time consumer, because of well-generalized FNN scheme depends on a number of training trails. Beside that FNN scheme does not give accurate and robust results under the noisy condition. Case two, using the RBFNN scheme can overcome the long period of the training which is encountered in the FNN scheme.

In this thesis, The purpose is to replace the FNN scheme and RBFNN scheme with ensembled ELM schemes (i.e. ELM and OS-ELM) . ELM schemes have the ability to boost the accuracy and robustness of the results. Besides that, it is not a time-consuming neural network.

4.1 Data Sets for DCT

Table 4.1 exposes affine transformation parameter values for carrying out the experiments in the medium range for coarse registration (MRCR), local range for fine registration (LRFR). In the same table training and test image counts are also given. To examine the accuracy and robustness of the used networks, the values of affine transform parameters of the test data were chosen at a point farthest from the values used in the training data. A reference image and two sample images generated from the reference image through translation, rotation and scaling is given in Figure 4.1. Figure 4.2 and Figure 4.3 are two other reference images, which are named as moon and aerial, respectively. All used reference images have 400×400 pixels size and 256 gray level. All generated images have 128×128 pixels size. After the images

were generated, white Gaussian noise were appended to them. Then, to reduce data dimension frequency coefficient of these images were calculated by discrete cosine transform. From DCT plane, a 6×6 matrix area in the lowest frequency band was cut and used as global features vector. This was done to M images, where M corresponds to all generated image numbers. After cutting out a zero frequency coefficients from obtained 6×6 region of coefficients, $35 \times N$ matrix of coefficient was obtained and used in the training of the FNN, RBFNN and ELM based schemes. For test data, a matrix with size $35 \times M$ of coefficient was obtained in the same way that used in generating the training data.

Table 4.1 Parameters performed in experiment for LRFR and MRCR

Transformation parameters	Parameters (LRFR)		Parameters (MRCR)	
	Training data	Test data	Training data	Test data
(S) Scaling	0.900, 0.965, 1.035, 1.100	0.93, 1.00, 1.07	0.70, 0.85, 1.00, 1.15, 1.30	0.77, 0.92, 1.07, 1.24
(R) Rotation (degree)	-5, -2, 2, 5	-3, 1, 4	-30, -15, 0, 15, 30	-25, -12, 5, 22
(VT) Vertical Translation	-5, -2, 2, 5	-4, 0, 3	-20, -10, 0, 10, 20	-18, -5, 7, 16
(HT) Horizontal Translation	-5, -2, 2, 5	-3, 1, 4	-20, -10, 0, 10, 20	-16, -4, 4, 17
(M) Generated image numbers	256	81	625	256

4.2 Experiment with DCT Coefficients

In the experiment, affine transformation registration parameters were estimated by means of various neural networks, which are two types of ELMs, RBFNN and FNN. The training data and test data for both ranges (MRCR and LRFR), were extracted by DCT from reference images which have been transformed and appended by additive Gaussian noise. Each network was trained and tested using noise-free data and two noisy data which are having 5 dB and 20 dB signal-to-noise ratio (SNR), respectively. Hidden layer neurons of the feedforward neural network was assigned with a tan-sigmoid and at the output layer a linear transfer function was chosen. The backpropagation training algorithm of the FNN was selected as the Levenberg-Marquardt. For FNN to converge in a shorter time, the affine transformation

parameters given to FNN outputs were normalized in the training stage. In the end of network output calculation, the transformation parameter values were denormalized back. Early stopping of training was also used for obtaining better generalisation performance. The FNN's registration error results are shown in Table 4.2 and Table 4.3. Each result was found after about 25 training trials.

In the second experiment, the transformation parameters were estimated by using the RBFNN scheme. The RBFNN scheme contains three layers, the input layer, a hidden layer which is a non-linear processing unit (Gaussian type transfer function), and an output layer which is a linear calculation unit. Normally, the number of the hidden neuron in RBFNN is equal to the number of the training set in the input vector. In this experiment, the RBFNN has a moderate size of hidden neurons. Getting a good non-linear mapping from input space to output space using RBFNN scheme depends on the centre spread parameters of the Gaussian function which control the spread of the width of the RBF. In this experiment, the centre spread parameters were chosen empirically according to the minimum error.

In the third experiment, ELM and OS-ELM schemes were used to estimate the affine transformation parameters. The used ELM schemes have 500 hidden neurons (processing units). In order for ELM to have a good performance, two facts must be considered, firstly the choice of the non-linear function and the continuous distribution probability by which the random weights and biases are generated. In the MRCR experiment, the ELM had a cosine function and OS-ELM had sigmoid function in the hidden layer nodes. The weights and biases were generated by uniform distribution probability with a range; this range was assigned empirically according to the minimum error. The test results showed that all ELM gives better performance with 500 neurons in the hidden layer. In the LRFR experiment, the ELM and OS-ELM had a sigmoid transfer function. The weights and biases were generated in the same way used in MRCR experiment. In the LRFR experiment and MRCR experiment, nine models of ELM were generated and their average is taken (Ensemble). It is known that the ensemble of ELM can boost the accuracy and reduce the overfitting.

Table 4.2 shows the results of all used schemes for MRCR. For the Aerial image which has high frequency, OS-ELM scheme gives slightly better performance in the estimation of the scaling parameters for all SNR values. In the estimation of the rotation parameters, which are the largest values as shown in table 4.1, OS-ELM scheme gives better performance than the other schemes. For the estimation of the translation parameters, OS-ELM and RBFNN have same performance results.

In the moon image which has low frequency, the results show that OS-ELM and ELM give more accurate and robust results than RBFNN and FNN in the presence of the noise.

For girl image in MRCR, the results point out that the performance of OS-ELM and RBFNN exceeds the performance of other systems and the results obtained by OS-ELM and RBFNN are relatively same.

Table 4.3 shows the results of all used schemes for LRFR. For The Aerial image in noise free condition, RBFNN scheme gives more accurate results than other schemes. In 20 db noise condition OS-ELM, RBFNN and ELM give relatively same results and their performance exceed the performance of FNN. In 5db noise condition OS-ELM and ELM schemes give accurate results comparing with other schemes.

In moon image for LRFR experiment with noise free condition, RBFNN scheme accuracy exceeds the accuracy of the other schemes and even FNN give better results than OS-ELM and ELM in some estimation results. While OS-ELM and ELM schemes give better accuracy in 20 db noise condition experiment. In 5db experiment the performance of OS-ELM and ELM exceeds the other schemes.

In girl image for LRFR experiment, RBFNN exceeds the other schemes in term of the performance in noise free condition. Meanwhile, ELM schemes give accurate results than other schemes when the noise increases.

Generally we can say that the main competitive for ELM schemes is RBFNN scheme. RBFNN gives more accurate results in noise free condition as we have seen

in LRFR experiment. However, its performance decrease when the noise increases. On the other hand ELM schemes give low performance in noise free condition, and high performance when the noise increases. Beside that its performance is more robust than RBFNN scheme. The reason that the performance of ELM exceeds the other scheme's performance in high noise condition is that ELM has the capability to reduce the effect of the noise . This can be done by selecting the weights properly (i.e. the input weights are assigned randomly with a range) (Dudek., 2016).

For better understanding, the results are illustrated in graphics in figure 4.4 to figure 4.21.



Figure 4.1 Reference image of girl (a), (b) and (c) generated images from reference image (a) by scaling, rotation, and translation (Park et al., 2001)



Figure 4.2 Moon image (Bakshi et al., 2012)



Figure 4.3 Aerial image (Doré et al., 2011)

Table 4.2 mean absolute error of ELM, OS-ELM, RBFNN and FNN (MRCR) for DCT features

Images	Signal-to-Noise ratios (dB)	Networks	S	R	HT	VT
Aerial Image	Noise free	OS-ELM	0.013	1.0	1.0	1.3
		RBFNN	0.014	1.3	1.0	1.3
		ELM	0.015	1.3	1.1	1.5
		FNN	0.022	1.7	1.2	1.6
	20	OS-ELM	0.013	1.0	0.9	1.3
		RBFNN	0.014	1.3	1.0	1.3
		ELM	0.016	1.3	1.1	1.5
		FNN	0.022	1.7	1.1	1.4
	5	OS-ELM	0.016	1.2	1.0	1.4
		RBFNN	0.018	1.5	1.1	1.5
		ELM	0.019	1.6	1.2	1.5
		FNN	0.032	2.2	1.7	2.2
Moon Image	Noise free	OS-ELM	0.012	1.0	0.4	0.5
		RBFNN	0.014	1.0	0.5	0.5
		ELM	0.014	1.0	0.4	0.5
		FNN	0.021	1.7	0.9	0.8
	20	OS-ELM	0.013	1.0	0.4	0.5
		RBFNN	0.017	1.4	0.6	0.6
		ELM	0.014	1.1	0.4	0.5
		FNN	0.025	1.8	1.0	0.9
	5	OS-ELM	0.016	1.2	0.5	0.5
		RBFNN	0.023	1.3	1.1	0.9
		ELM	0.017	1.3	0.6	0.6
		FNN	0.031	2.4	1.5	1.3
Girl Image	Noise free	OS-ELM	0.012	0.7	0.7	0.7
		RBFNN	0.011	0.8	0.9	0.7
		ELM	0.013	0.8	0.9	0.8
		FNN	0.018	1.2	1.2	1.1
	20	OS-ELM	0.012	0.7	0.8	0.7
		RBFNN	0.011	0.8	0.8	0.9
		ELM	0.013	0.9	1.0	0.8
		FNN	0.020	1.4	1.3	1.0
	5	OS-ELM	0.014	0.9	0.8	0.8
		RBFNN	0.013	1.0	0.9	0.8
		ELM	0.015	1.1	1.0	1.0
		FNN	0.021	1.6	1.4	1.5

Table 4.3 Mean absolute error of ELM, OS-ELM, RBFNN, FNN (LRFR) for DCT features

Images	Signal-to Noise ratios (dB)	Networks	S	R	HT	VT
Aerial Image	Noise free	OS-ELM	0.0003	0.07	0.018	0.08
		RBFNN	0.0002	0.01	0.007	0.02
		ELM	0.0004	0.09	0.02	0.09
		FNN	0.0004	0.04	0.02	0.03
	20	OS-ELM	0.0004	0.06	0.017	0.07
		RBFNN	0.0006	0.06	0.03	0.05
		ELMNN	0.0005	0.06	0.03	0.07
		FNN	0.001	0.1	0.04	0.07
	5	OS-ELM	0.0005	0.05	0.02	0.07
		RBFNN	0.002	0.18	0.1	0.15
		ELM	0.0004	0.08	0.03	0.07
		FNN	0.003	0.22	0.17	0.25
Moon Image	Noise free	OS-ELM	0.001	0.05	0.05	0.02
		RBFNN	0.0003	0.03	0.004	0.01
		ELM	0.001	0.06	0.05	0.03
		FNN	0.0006	0.06	0.01	0.02
	20	OS-ELM	0.001	0.05	0.06	0.03
		RBFNN	0.001	0.09	0.06	0.04
		ELM	0.001	0.06	0.06	0.03
		FNN	0.001	0.1	0.05	0.06
	5	OS-ELM	0.001	0.06	0.05	0.04
		RBFNN	0.003	0.2	0.18	0.18
		ELM	0.001	0.07	0.03	0.06
		FNN	0.004	0.3	0.24	0.19
Girl Image	Noise free	OS-ELM	0.0004	0.04	0.05	0.04
		RBFNN	0.0001	0.005	0.004	0.004
		ELM	0.0004	0.03	0.05	0.03
		FNN	0.0003	0.01	0.01	0.02
	20	OS-ELM	0.0004	0.05	0.05	0.04
		RBFNN	0.0007	0.05	0.05	0.03
		ELM	0.0005	0.04	0.03	0.03
		FNN	0.0009	0.05	0.04	0.05
	5	OS-ELM	0.0006	0.04	0.05	0.04
		RBFNN	0.003	0.2	0.17	0.15
		ELM	0.0005	0.03	0.05	0.04
		FNN	0.003	0.2	0.15	0.19

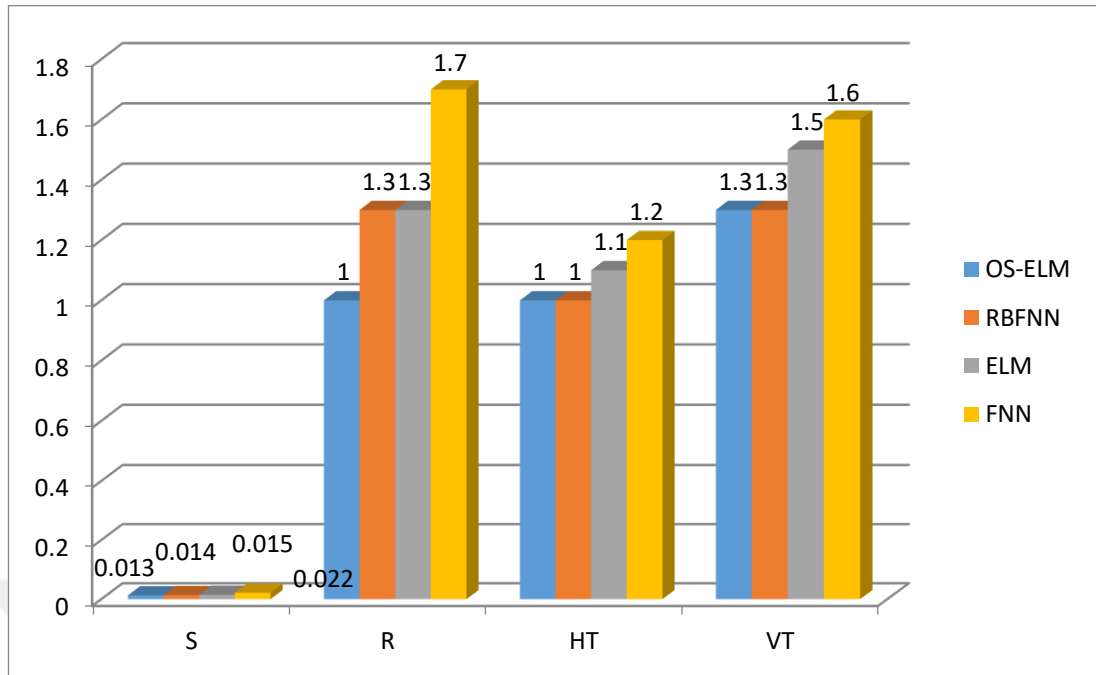


Figure 4.4 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in MRCR

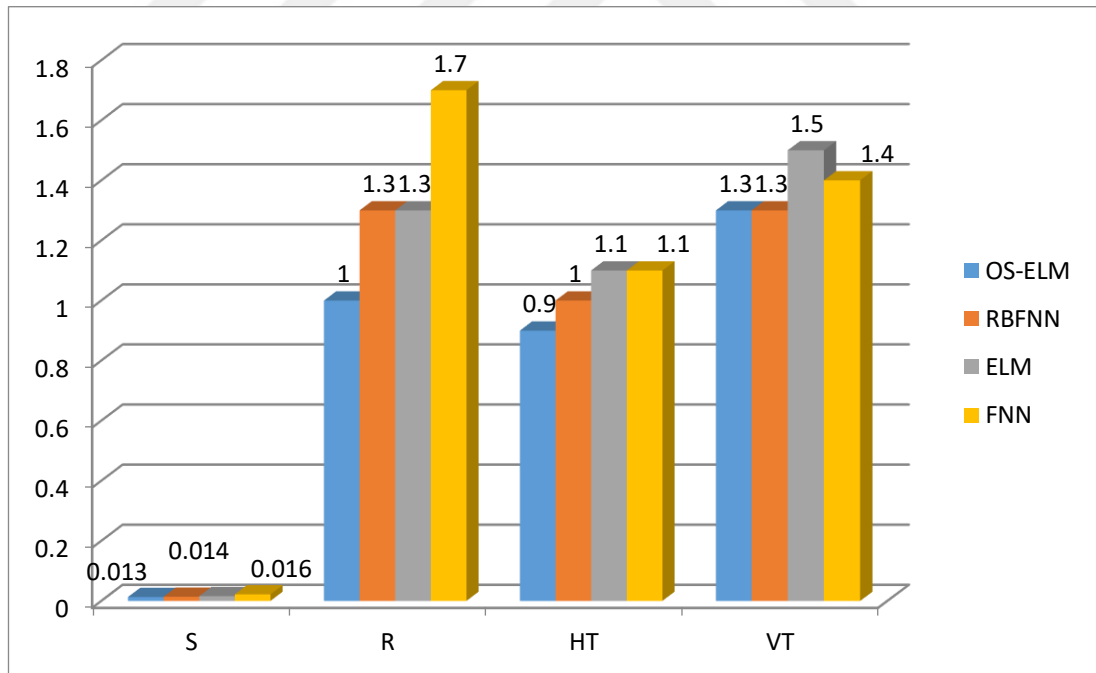


Figure 4.5 The mean absolute errors at 20 dB noise (ELM, OS-ELM, RBFNN and FNN) for aerial image in MRCR

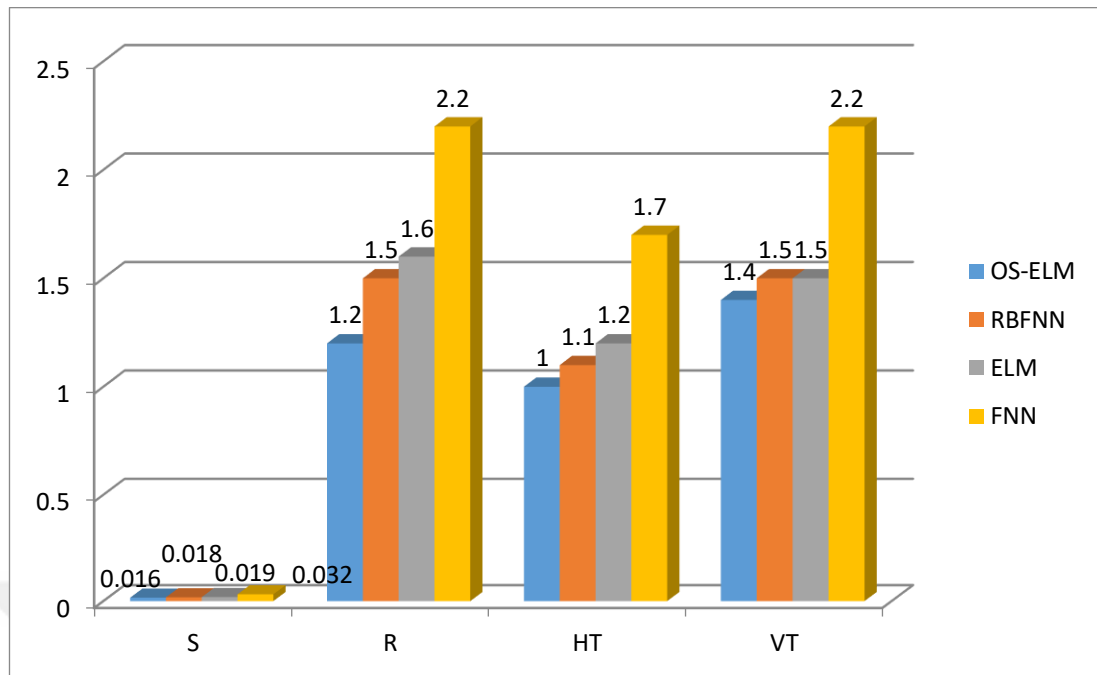


Figure 4.6 The mean absolute errors at 5 dB (ELM, OS-ELM, RBFNN and FNN) for the aerial image in MRCR

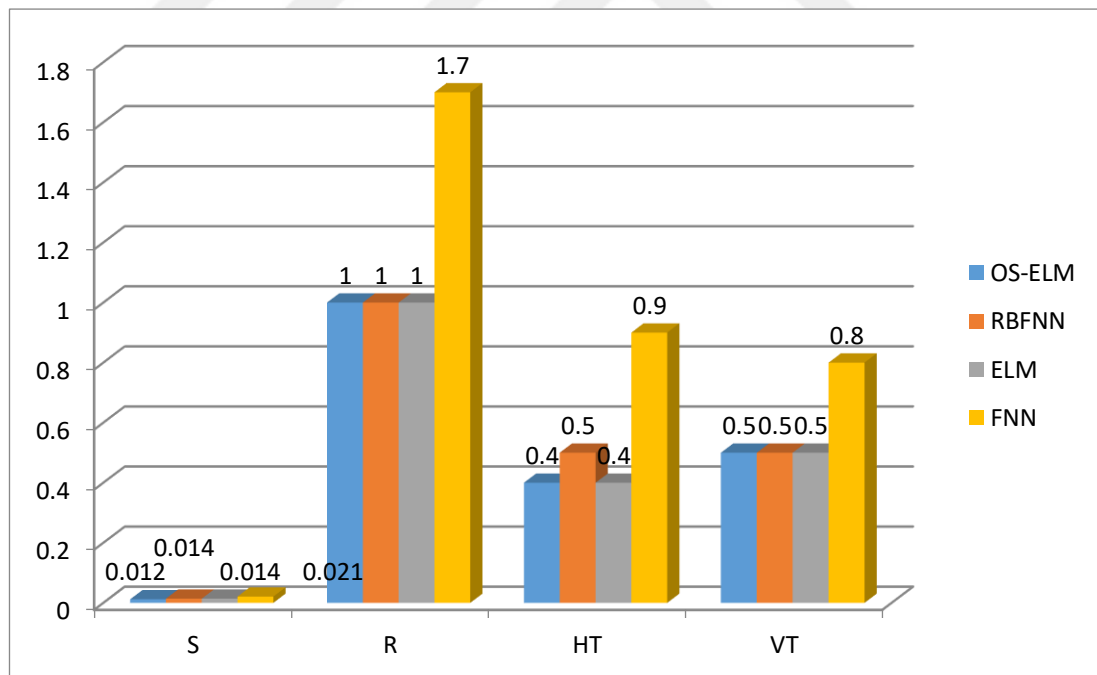


Figure 4.7 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the Moon image in MRCR

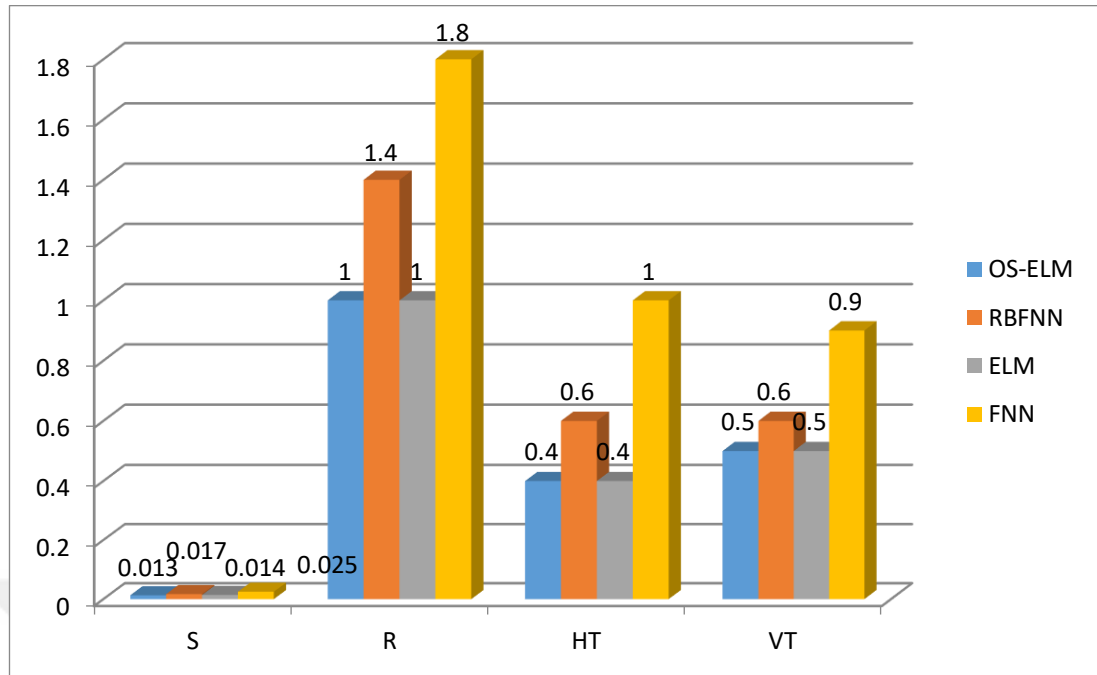


Figure 4.8 The mean absolute errors at 20 dB (ELM, OS-ELM, RBFNN and FNN) for the moon image in MRCR

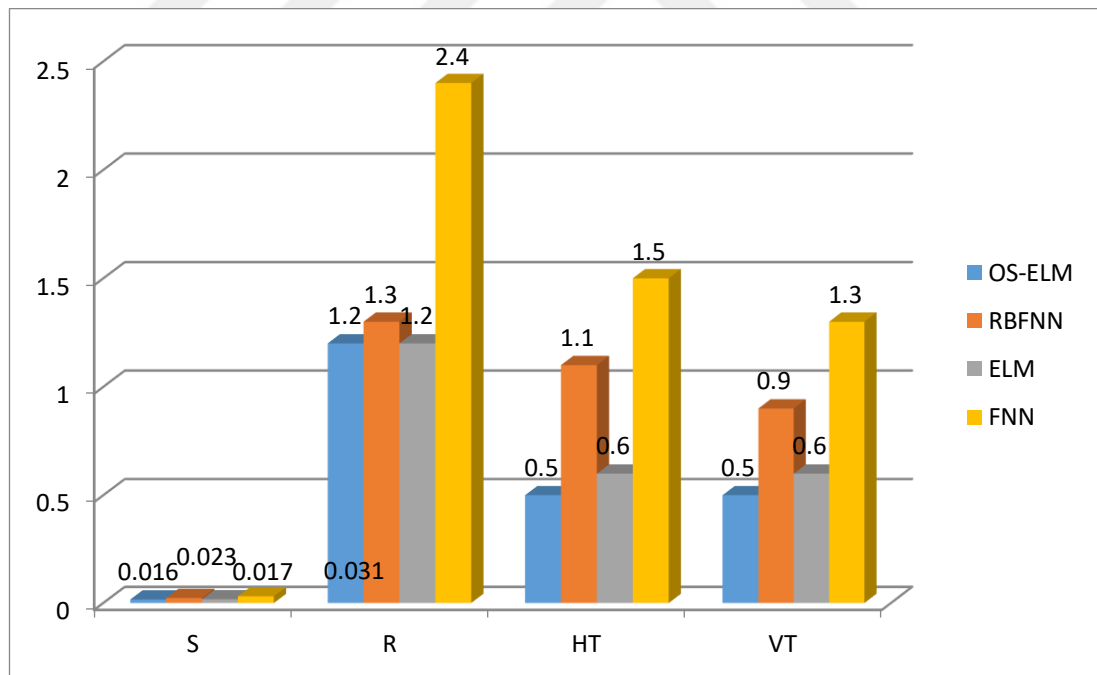


Figure 4.9 The mean absolute errors at 5 dB (ELM, OS-ELM, RBFNN and FNN) for the moon image in MRCR

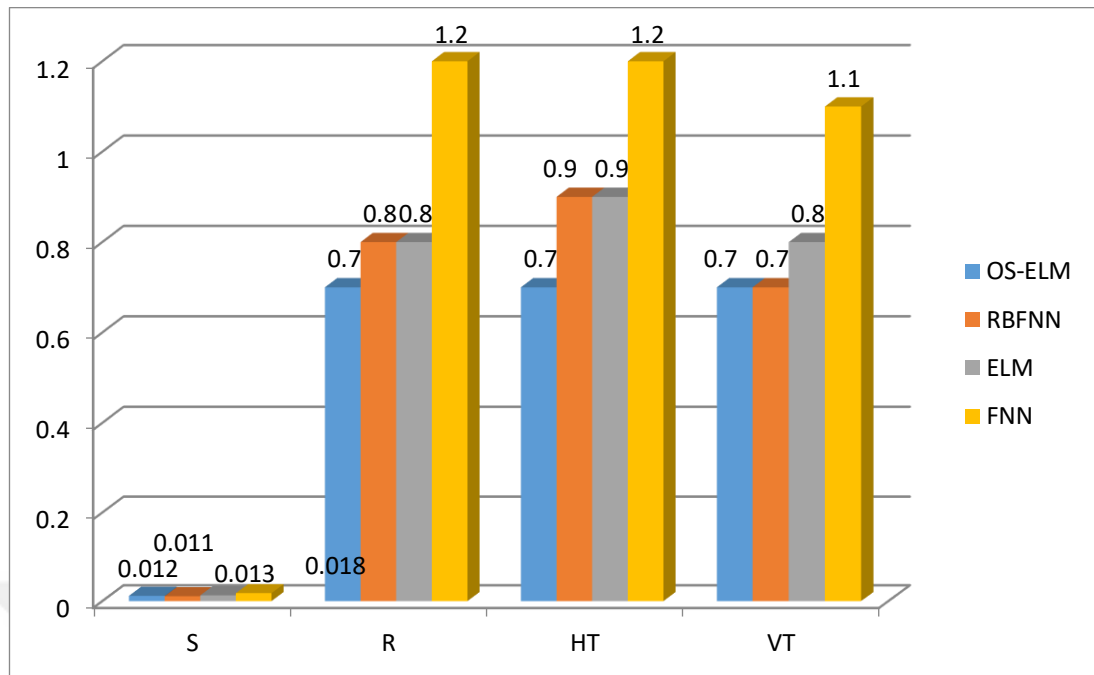


Figure 4.10 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in MRCR

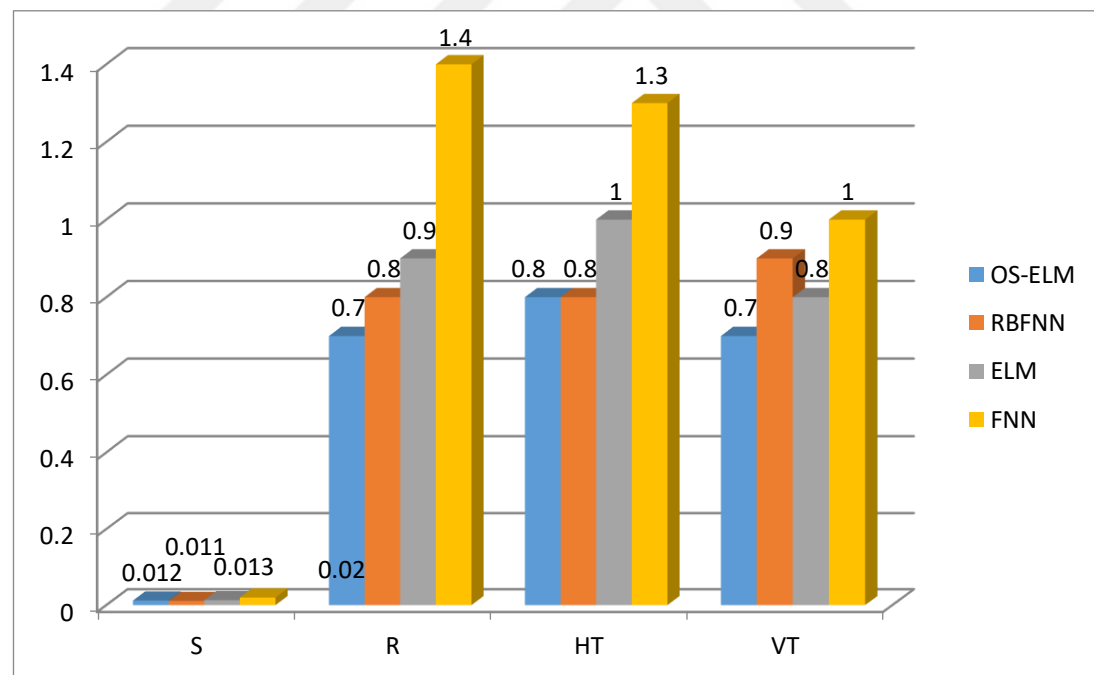


Figure 4.11 The mean absolute errors at 20 dB (ELM, OS-ELM, RBFNN and FNN) for the girl image in MRCR

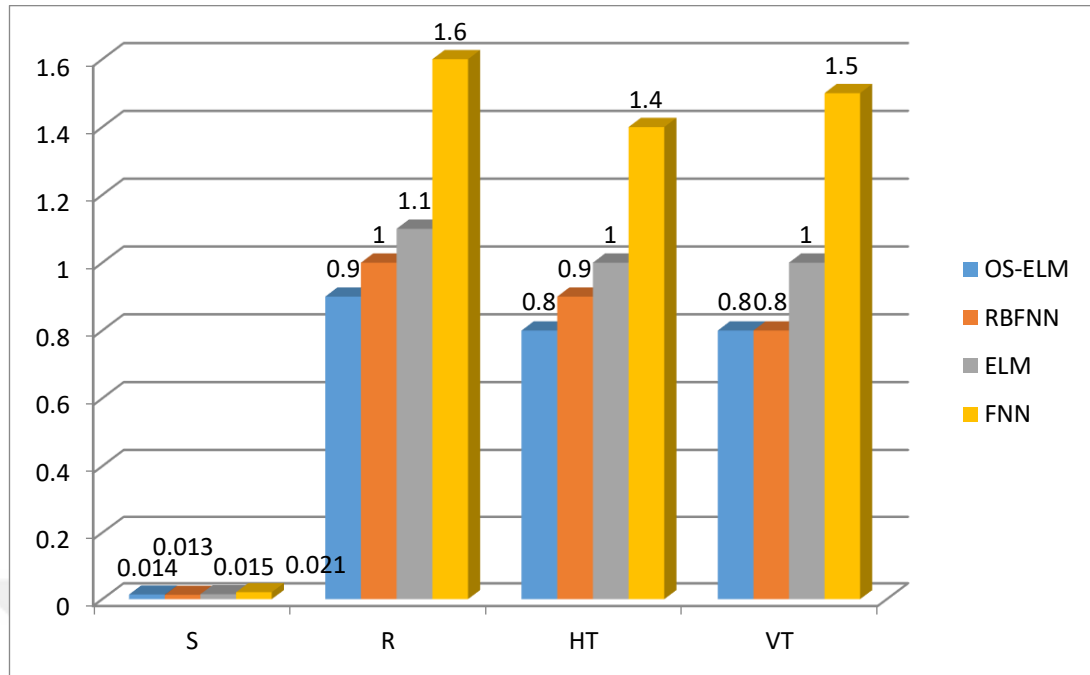


Figure 4.12 The mean absolute errors at 5 dB (ELM, OS-ELM, RBFNN and FNN) for the girl image in MRCR

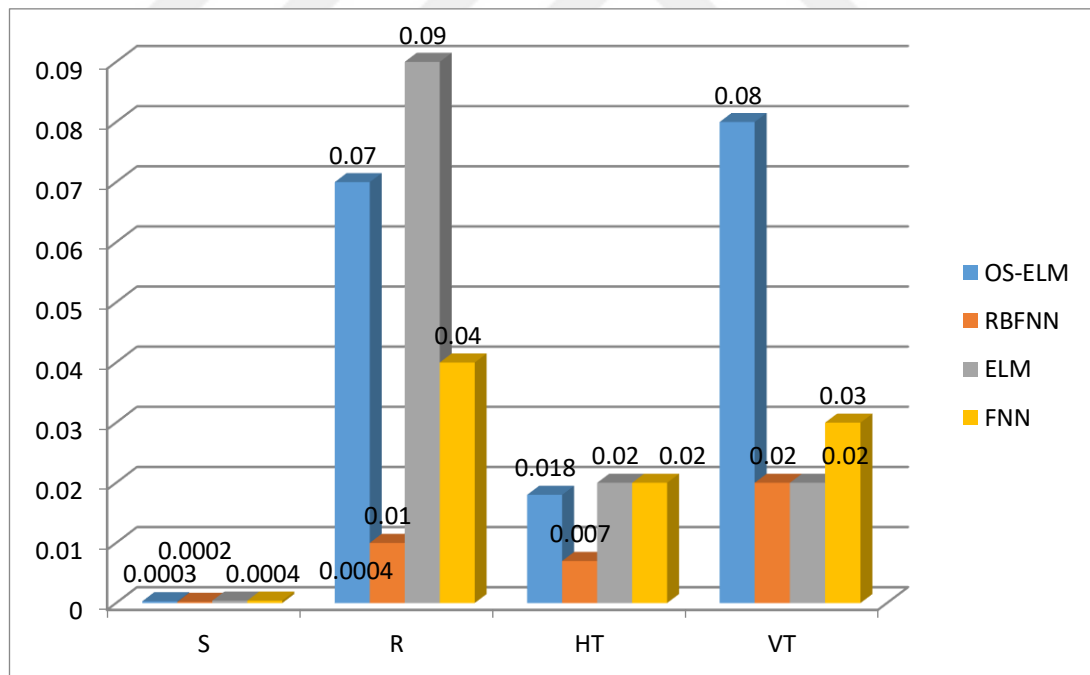


Figure 4.13 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the aerial image in LRFR

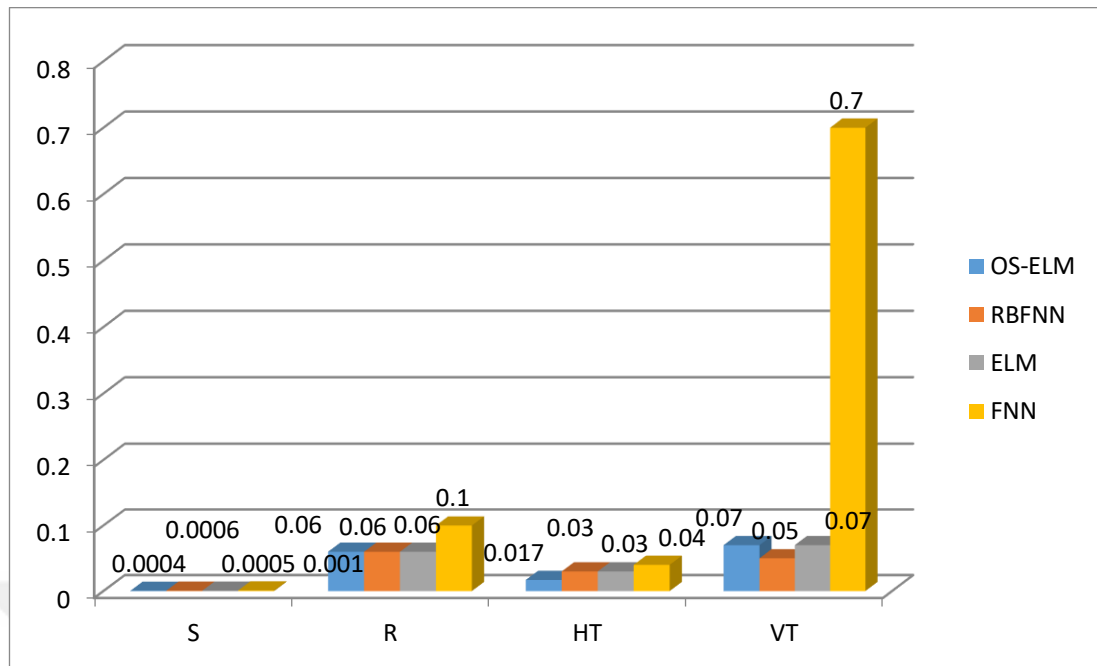


Figure 4.14 The mean absolute errors at 20 dB (ELM, OS-ELM, RBFNN and FNN) for the aerial image in LRFR

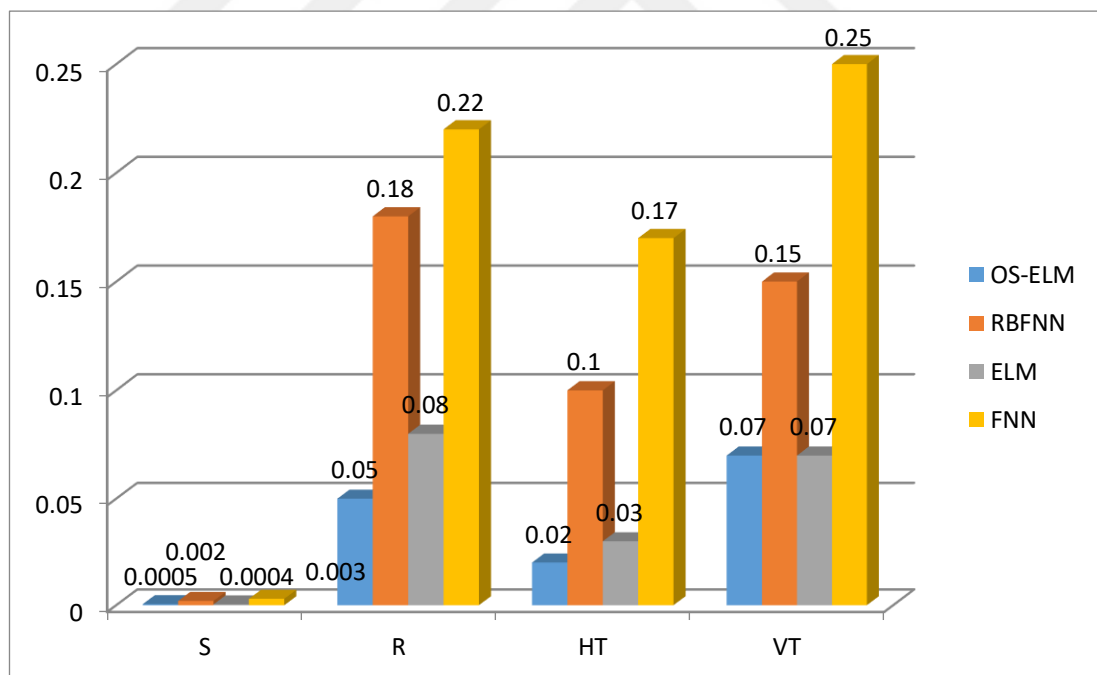


Figure 4.15 The mean absolute errors at 5 dB (ELM, OS-ELM, RBFNN and FNN) for the aerial image in LRFR

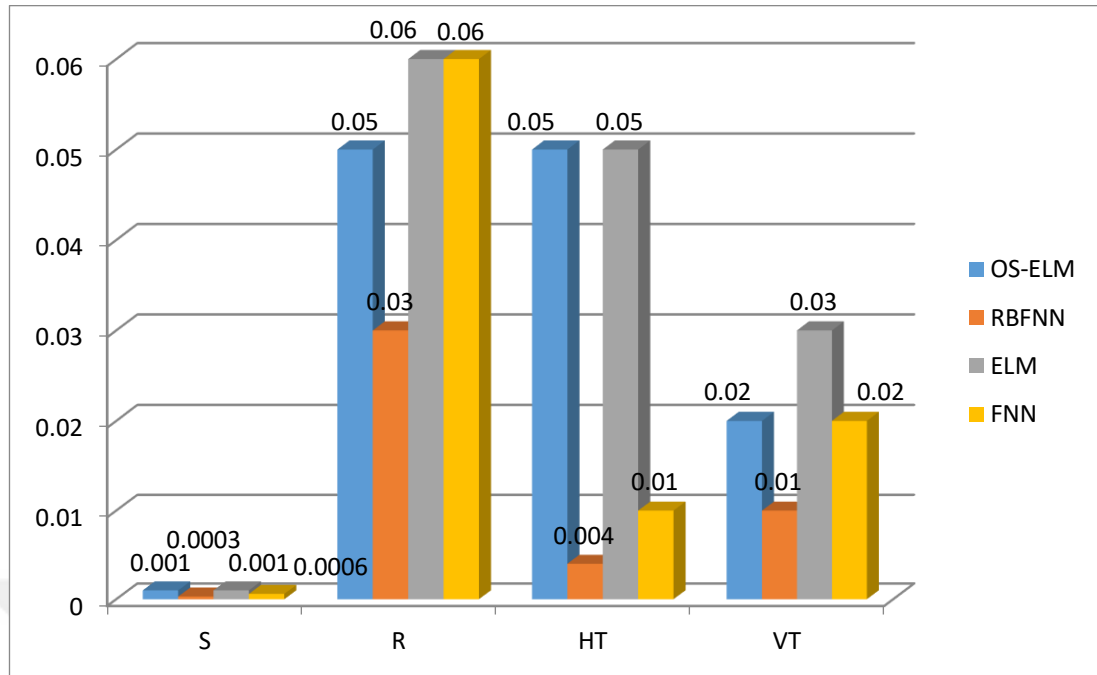


Figure 4.16 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the moon image in LRFR

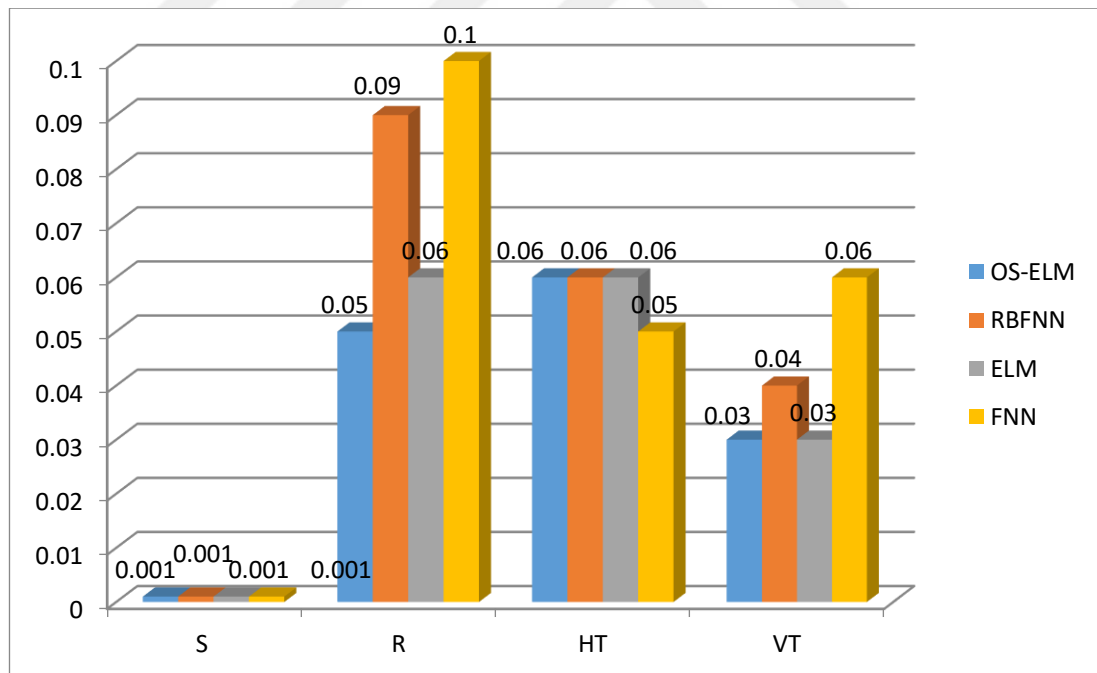


Figure 4.17 The mean absolute errors at 20 dB (ELM, OS-ELM, RBFNN and FNN) for the moon image in LRFR

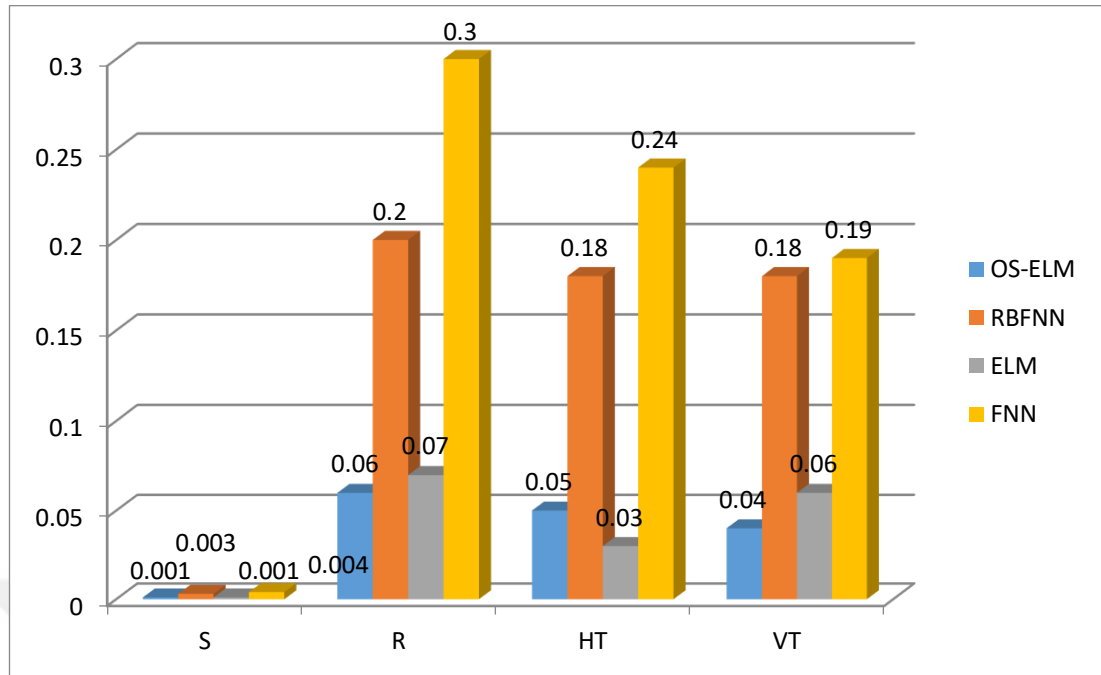


Figure 4.18 The mean absolute errors at 5 dB (ELM, OS-ELM, RBFNN and FNN) for the moon image in LRFR

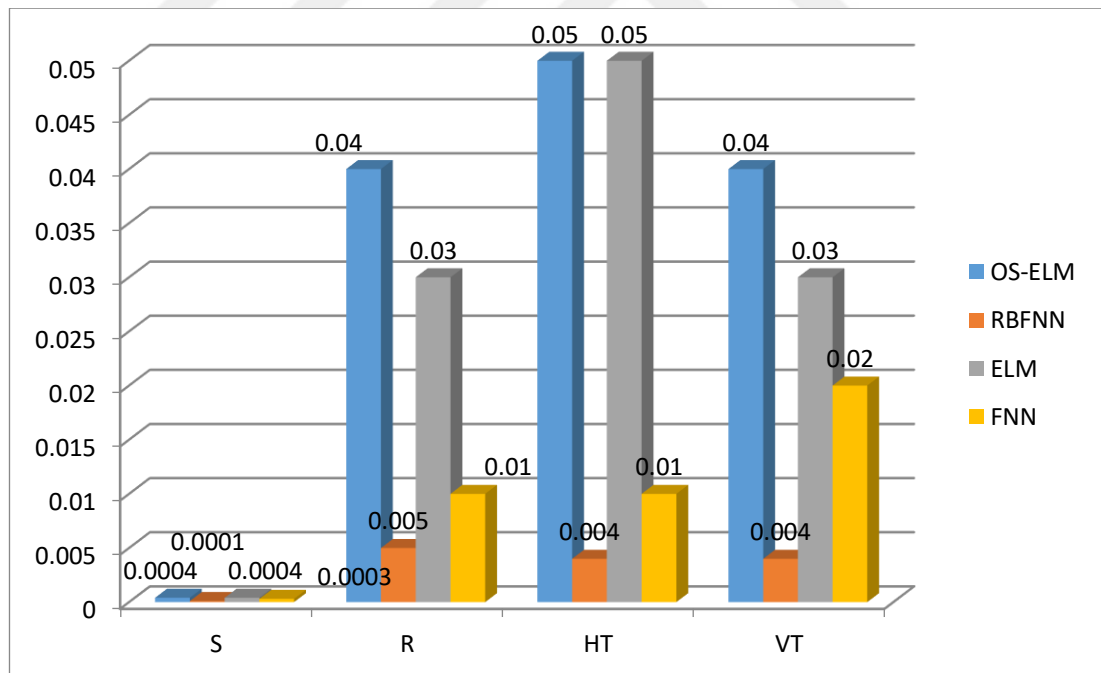


Figure 4.19 The mean absolute errors without noise (ELM, OS-ELM, RBFNN and FNN) for the girl image in LRFR

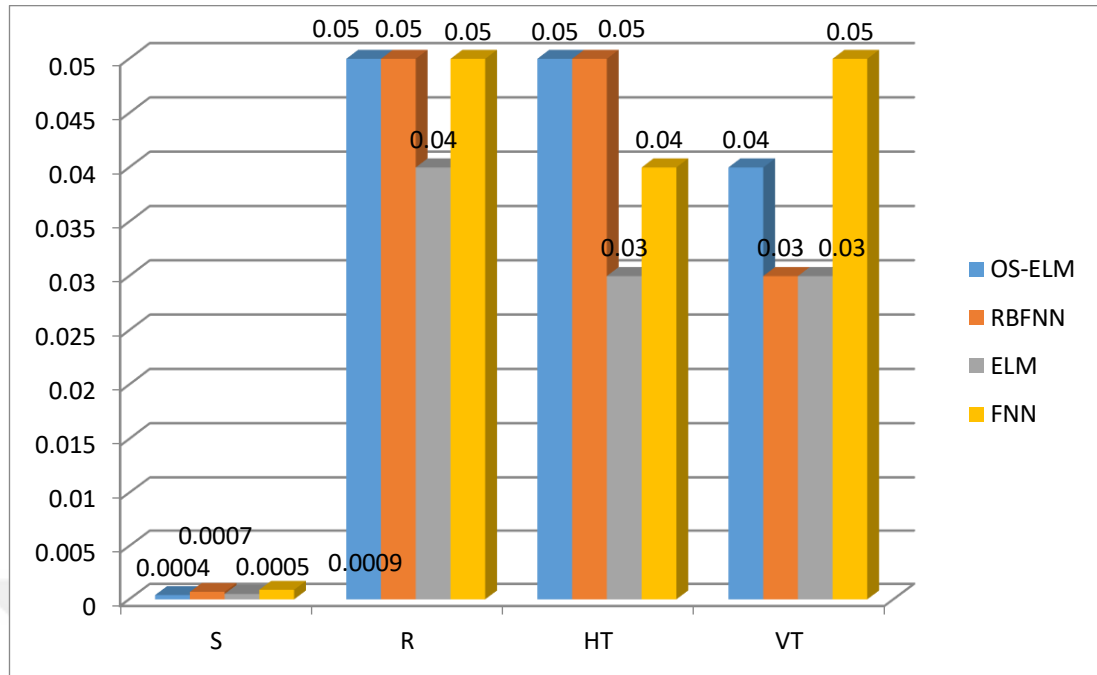


Figure 4.20 The mean absolute errors at 20 dB (ELM, OS-ELM, RBFNN and FNN) for the girl image in LRFR

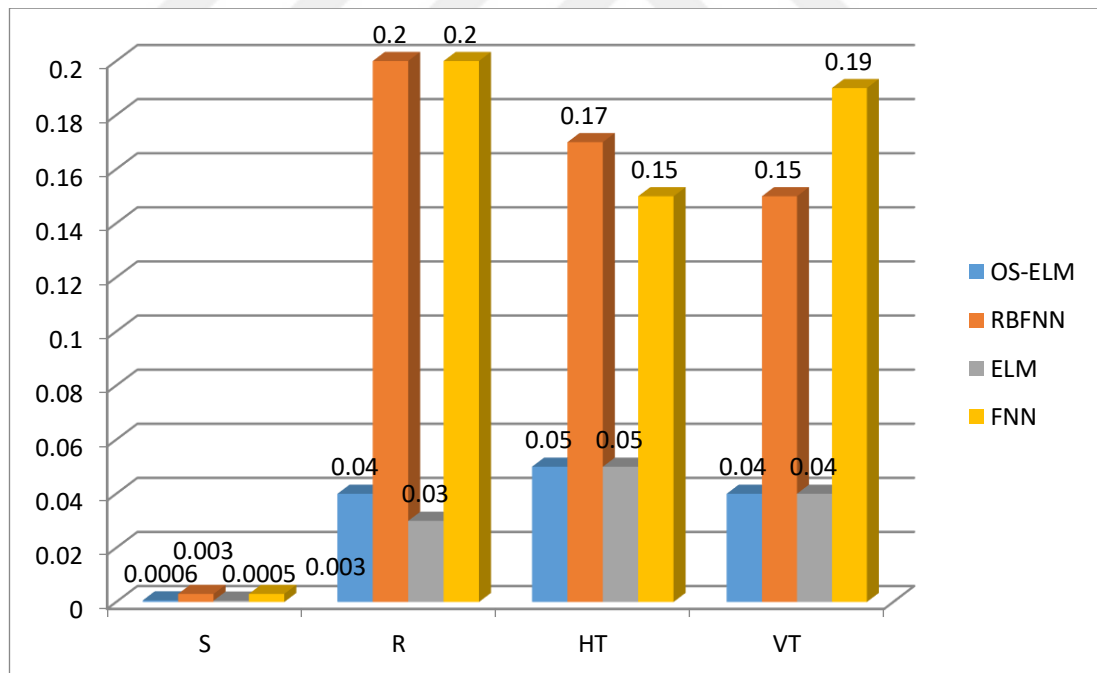


Figure 4.21 The mean absolute errors at 5 dB (ELM, OS-ELM, RBFNN and FNN) for the girl image in LRFR

CHAPTER FIVE

CONCLUSION

The research work of this thesis is related with the image registration which was carried out by various types of neural networks. Ensembled ELM and OS-ELM were proposed instead of RBFNN and FNN to estimate the registration parameters in MRCC and LRFR. In the experiment, all used neural networks were fed by DCT features as global features. By use of three different images all four networks were compared with each other at various noise conditions and various range of registration.

The experimental results show that in estimation of affine transformation parameters, the use of ELMs instead of FNN and RBFNN gives more accurate results and shows good robustness under noisy conditions. Advantage of ELM are not limited to its better performance, but this system is also easy to implement. The FNN has long training duration and has network generalization problem experienced during the training session. On the other hand, RBFNN fast in training session, gives good performance in noise free conditions, but requires the selection of optimized spread value. In the ELM approach, the hidden node parameters are randomly selected and only the output weights are calculated. This provides fast learning and better generalization capability. Generating the weights and basis randomly within a range reduces the effect of the noise. The ensemble of the ELM boosts the accuracy of the results. The training of the ELM and RBFNN is similar. Both of them require the calculation general linear system.

REFERENCES

- Ahmed, N., Natarajan, T., & Rao, K. R. (1974). Discrete cosine transform. *IEEE Transactions on Computers*, 100(1), 90-93.
- Abche, A. B., Yaacoub, F., Maalouf, A., & Karam, E. (2006). Image registration based on neural network and Fourier transform. *In 2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, 4803-4806.
- Bakshi, S., Lamba, R. G., & Singh, M. D. (2012). Designing a Framework for Noise dependent Filter Selection Algorithm. *Doctoral dissertation*.
- Brown, L. G. (1992). A survey of image registration techniques. *ACM computing surveys (CSUR)*, 24(4), 325-376.
- Dabbaghchian, S., Aghagolzadeh, A., & Moin, M. S. (2007). Feature extraction using discrete cosine transform for face recognition. *In 2007 9th International Symposium on Signal Processing and Its Applications*, 1- 4.
- Doré, V., Moghaddam, R. F., & Cheriet, M. (2011). Non-local adaptive structure tensors: application to anisotropic diffusion and shock filtering. *Image and Vision Computing*, 29(11), 730-743.
- Dudek, G. (2016). Extreme learning machine as a function approximator: Initialization of input weights and biases. *In Proceedings of the 9th International Conference on Computer Recognition Systems CORES 2015*, 59-69.
- Elhanany, I., Sheinfeld, M., Beck, A., Kadmon, Y., Tal, N., & Tirosh, D. (2000). Robust image registration based on feedforward neural networks. *In Smc 2000 conference proceedings. 2000 IEEE international conference on systems, man and cybernetics 'cybernetics evolving to systems, humans, organizations, and their complex interactions' (2)*, 1507-1511.

- Fausett, L. (1994). *Fundamental of neural network architectures, algorithm and application*. USA: Prentice hall international edition.
- Gadde, P., & Yu, X. H. (2016, July). Image registration with artificial neural networks using spatial and frequency features. *In 2016 International Joint Conference on Neural Networks (IJCNN)*, 4643-4649.
- Ham, F. M. & Kostanic, I. (2001). *Principles of neurocomputing for science and engineering*, Singapore: Mcgraw Hill International Edition.
- Haykin, S. (1999). *Neural networks a comprehensive foundation (2nd edition)*. United States of America: Prentice Hall International, Inc.
- Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. *Neural networks*, 2, 985-990.
- Kokila, R. (2018). A study on Image Registration schemes. *International Journal of Engineering Techniques*, 4(1), 575- 581.
- Park, H., & Kim, K. (2001). Visible watermarking using verifiable digital seal image. *In Symposium on Cryptography and Information Security* 103-108.
- Piraino, D., Kotsas, P., Richmond, B., Recht, M., & Kormos, D. (1994). Three dimensional image registration using artificial neural networks. *In Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, 4017-4021.
- Qian, Z., & Li, J. G. (1997). Use of hopfield neural network for complex image registration. *In Proceedings Ninth IEEE International Conference on Tools with Artificial Intelligence*, 204-207.

- Saxena, S., & Singh, R. K. (2014). A survey of recent and classical image registration methods. *International journal of signal processing, image processing and pattern recognition*, 7(4), 167-176.
- Sarnel, H., & Senol, Y. (2011). Accurate and robust image registration based on radial basis neural networks. *Neural Computing and Applications*, 20(8), 1255-1262.
- Salomon, D., & Motta, G. (2010). *Handbook of data compression*. London: Springer Science & Business Media.
- Xu, A., Jin, X., Guo, P., & Bie, R. (2006, July). KICA feature extraction in application to FNN based image registration. *In The 2006 IEEE International Joint Conference on Neural Network Proceedings*, 3602-3608.
- Zitova, B., & Flusser, J. (2003). Image registration methods: a survey. *Image and Vision Computing*, 21(11), 977-1000.