Republic of Iraq
Ministry of Higher Education
and Scientific Research
Al-Nahrain University
College of Science

Breast Cancer Tissues Recognition Using Fractal Dimension, Fuzzy Logic and Neural Network

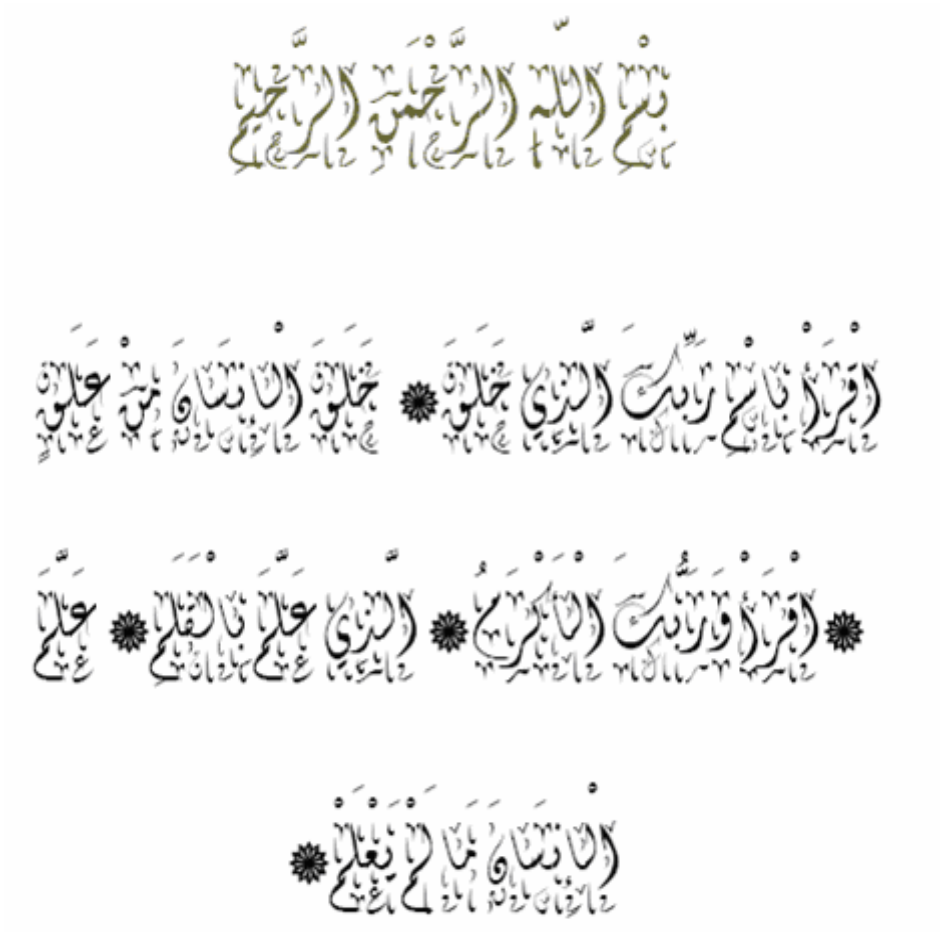
A Thesis Submitted to the College of Science, Al-Nahrain University in Partial Fulfillment of the Requirements for The Degree of Master in Computer Science

By
Hadeel Hatam J.
(B.Sc. 2003)

Supervised By

Dr. Loay E. George

2009 **1429**


بسم الله الرحمن الرحيم
اقرأ باسم ربك الذي خلق ۞ خلق الانسان من علق
۞ اقرأ وربك الاكرم ۞ الذي علم بالقلم ۞ علم
الانسان ما لم يعلم ۞

Supervisor Certification

I certify that this thesis was prepared under our supervision at the Department of Computer Science/College of Science/Al-Nahrain University, by **Hadeel Hatam Jassim** as partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Signature:
Name : **Dr. Loay E. George**
Title : **Senior Research**
Date : / / **2008**

In view of the available recommendations, I forward this thesis for debate by the examination committee.

Signature:
Name : **Dr. Taha S. Bashaga**
Title : **Head of the department of Computer Science,**
Al-Nahrain University.
Date : / / **2008**

COMMITTEE CERTIFICATION

We, the members of examining committee certify that after reading this thesis in title "***Breast Cancer Tissues Analysis Using Fractal Dimension, Fuzzy Logic and Neural Network***" and we have examining the student "***Hadeel H. Jassim***" in its contents, and in what is connected with it., and that in our opinion it meets the standard of thesis for the degree of master of science in computer science.

Signature:
Name: **Dr. Eiad AL-Any**
Title: Professor
Date: / /2009
{Chairman}

Signature:
Name: **Dr. Amer Sedeq**
Title: Assistant Professor
Date: / /2009
{Member}

Signature:
Name: **Dr. Mohamed Saheb**
Title: Lecturer
Date: / /2009
{Member}

Signature:
Name: **Dr. Loay E. George**
Title: Assistant Professor
Date: / /2009
{Supervisor}

Approved by the council of the collage of Science ***Al-Nahrain University.***

Signature:
Name: **Dr. Lith Al-Any**
Title: Assistant Professor
Date: / /2009

Acknowledgment

I would like to express my thanks and deep gratitude to my supervisor Dr. Loay E. George, for his invaluable advice, criticism, encouragement and continuous help and supervision throughout this work.

And I would like to thank the University of Al-Nahrain which offered me the chance to gain the master's degree.

My deep thanks to the Dean of the College of Science and to all staff of the Department of Computer Science.

Also I wish to express my thanks to my dear family for their patience and support in the preparation of this thesis.

Finally, thanks go to all my friends, and I would like to express my deep gratitude to all kind, helpful, and lovely people who helped me directly or indirectly to complete this work.

Abstract

In this research work, two recognition systems have been developed to recognize different breast tissues (normal or cancer infected). The multi-fractal dimensions have been used as textural discriminating feature. The difference between the two established systems is the first is based on using fuzzy logic to make recognition decision, while in the second system the feed forward neural network is used instead of fuzzy logic. The input to these systems is a breast tissue image. In this wok 24 type of breast images have been used, among this set one image represents the normal case (i.e., not cancer infected tissue image), while the other 23 images represent various types of breast tumors. At the beginning these images are analyzed using box-counting method to extract six fractal dimensions vectors. Then, these extracted feature vectors are used as input to the two recognition systems (fuzzy logic or ANN).

In recognition system based on fuzzy logic, two types of membership functions (MSF) have been used (i.e., triangle and trapezoidal membership functions) to represent the extracted rules, which are applied on the F_D-features. In recognition stage the sum of MSF for input samples is calculated and considered as recognition criteria, where class that shows the highest sum of MSF is considered as the class of the tested images. The experimental result showed that when value of number of bins parameter equal to 75 the recognition system show good performance in terms of recognition accuracy and processing time. Also, the result showed that the size of tested samples affects the recognition accuracy; when size of tested samples is large and near to the size used in training phase, then the rate of recognition becomes high.

In artificial neural network (ANN) the 24 considered classes were divided into two classes: (i.e., either cancer infected or not infected). The training process is done by taking samples from all classes, and training the network using backpropagation NN. Then some testing is made using both the training and testing sets of samples. The test result showed that when using multi-fractal dimensions (with number of bins equal to 150) in training phase the recognition result is acceptable when both the training and test sets are used as test samples. The recognition results when using combined sets of fractal dimensions vectors are better than the results obtained by using one or two feature vectors only. The test result showed that the use of 4 hidden nodes was adopted because it makes the NN works with less computational time. Also, the learning rate equal to 0.5 led to best performance, because at this time the network required less training time with insignificant degradation security. Also, the test results showed that the size of tested samples doesn't have significant effect on the recognition accuracy.

List of Abbreviations

ANN	Artificial Neural Network
BCM	Box-Counting Method
Bmp	Bitmap
BP	Backpropagation
BPNN	Back Propagation Neural Network
CT	Computerized Tomography
D	Dimension
FL	Fuzzy Logic
F_D	Fractal Dimension
IFS	Iterated function systems
MSF	MemberShip Function
MRI	Magnetic Resonance Image
NNs	Neural Networks

CONTENTS

Chapter Three: *Design and Implementation*

Chapter Four: *Experimental and Result*

Chapter One

General Introduction

Chapter One
General Introduction

1.1 Introduction

Texture analysis is one of the most important techniques used in analysis and classification of images presenting repetition of fundamental image elements. It is widely used in interpretation and classification of terrain images, radiographic and microscopic cell images, and many other domains of pattern recognition [12].

Texture could be recognized when it is seen, but it is a very difficult concept to define [30]. Texture is interesting because it allows easy discrimination of objects from background, using local features, and global information such as luminance [9]. The general approach to textural classification mainly based on global measures and feature extraction using statistical methods, structural techniques, and spectral techniques. Texture is an important characteristic for the analysis of many types of images including natural scenes, remotely sensed data and biomedical modalities. It is believed that the perception of texture plays an important role in the human visual system for recognition and interpretation [16, 43].

Fractal is a term coined by Mandelbrot, in 1975, it is "a rough or fragmented geometric shape that can be subdivided in parts, each of which is (at least approximately) a reduced-size copy of the whole". Because they appear similar at all levels of magnification, fractals are often considered to be infinitely complex. Natural objects that approximate fractals to a degree include clouds, mountain ranges, lightning bolts, coastlines, and snow flakes. However, not all self-similar objects are fractals, for example: the real line (a straight

Euclidean line) is formally self-similar but fails to have other fractal characteristics.

Fractals provide a description of the complex objects around us that are rough and fragmented with certain structure on many sizes [31]. Some studies have shown that fractal geometry can be used for describing the pathological architecture of tumors and, perhaps more surprising, for yielding insights into the mechanisms of tumor growth and angiogenesis that complement those obtained by modern molecular methods [6]. In medical image analysis, texture analysis is self-similar, so that, fractal extraction features can be used for a variety of classification tasks such as distinguishing normal from abnormal tissue or classifying different types of abnormal tissues.

1.2 Breast Cancer

Breast cancer is a disease in which malignant (cancer) cells form in the tissues of the breast. It is a heterogeneous disease, meaning that it is a different disease in different women, a different disease in different age groups and has different cell populations within the tumor itself. Generally, breast cancer is a much more aggressive disease in younger women. Women in the United States get breast cancer more than any other type of cancer except for skin cancer. It is second only to lung cancer as a cause of cancer death in women. It is expected that nearly 200,000 women will be diagnosed with breast cancer and more than 40,000 will die every year. Breast cancer is not exclusively a disease of women, however. Approximately 1,700 men will be diagnosed with breast cancer and 450 will die each year. The evaluation of men with breast masses is similar to that in women, including mammography.

So, there are many methods for determining breast cancer. In this research, it depend on the digital image (breast cancer tissues images) and using the

concept of fractal dimension in the color texture images to classify and recognize the breast cancer.

1.3 Computer Vision and Texture

Computer vision refers to the field of computer science that is concerned with the design and implementation of algorithms that allow the machines to simulate human's vision. Various fields related to computer processing of images are categorized according to the type of input it take and the type of results they produce [27].

In computer vision, texture is defined as all what is left after color and local shape has been considered, or defined by some term like structure and randomness. Many common textures are composed of small textons; usually they are too great in number to perceive as isolated objects. The elements can place more or less regularly or randomly. They can be almost identical or subject to large variations in their appearance and pose.

Textures are replications, symmetries and combinations of various basic patterns or local functions, usually with some random variation. Texture is the most important visual cue in identifying the types of homogeneous regions in the image; this called texture classification. The goal of texture classification is to produce a classification map of the input image where each uniform texture region is identified with the texture class it belongs to.

Texture based approaches in extracting relevant feature sets from images listed in databases gave very promising results in addressing this problem.

1.4 Medical Image Analysis

The automatic analysis of medical images had led to a vast improvement in both speed and accuracy of the diagnosis of many diseases. Medical images,

such as ultrasound, MRI and X-ray images, are often unclear and distorted, and in general do not contain clear separation of important objects. These characteristic make such image unsuitable for many traditional object recognition and classification techniques which rely on the detection of object boundaries. Texture analysis, with no such reliance on these features, had been shown to give good results in many medical applications [36, 15].

In this field, there are hundreds of examples such as [37]:

1. The classification of pulmonary disease using texture features. Some disease, such as interstitial fibrosis, affect the lungs in such a manner that the resulting changes in the X-ray images are texture changes are opposed to clearly delineated lesions. In such applications, texture analysis methods are ideally suited.
2. The extraction of various first-order statistics (such as mean gray level in a region) and second-order statistics (such as gray level co-occurrence matrices) could be utilized to differentiate different types of white blood cells.
3. The textural features existing in ultrasound images could be used to estimate tissue scattering parameters. The related studies have made significant use of the knowledge about the physics of the ultrasound imaging process and tissue characteristic to design the textural model.

1.5 Literature Survey

Among the published papers in the field of using textural analysis concepts for medical diagnosis, in the following details about some relevant studies are given:

1. Sedivy et al. [38], they applied a box-counting algorithm to determine the fractal dimension of atypical nuclei in dysplastic cervical epithelium. An automatic algorithm was used to determine the fractal dimension of nuclei in order to prevent errors from manual segmentation. Four groups of patients with 10 subjects each were examined. In total, the fractal dimensions of 1200 nuclei were calculated. In their results, they found that the fractal dimensions of the nuclei increased as the degree of dysplasia increased. They conclude that the fractal dimension is a valuable tool for detecting irregularities in atypical nuclei of the cervix uteri and thus allows objective nuclear grading.

2. Iftekharuddin et al. [17] developed three models to detect tumor in MR brain images using fractal dimension analysis. A multimedia web-based application was developed for tumor detection application.

3. Dragomir et al. [8] used box-counting method in order to analyze CT and MR brain images. They first developed a computer application offering several tools, most of them based on box counting method. They concluded that their proposed algorithms could be used as medical tools to help the diagnosis process.

4. Da Silva et al. [7], in their paper, they studied the application of box counting method (BCM) to estimate the fractal dimension of 3D plant foliage. They use artificial crowns with known theoretical fractal dimension to characterize the accuracy of the BCM and they extend the approach to 3D digitized plants. In particular, errors are experimentally characterized for the estimated values of the fractal dimension. Results show that, with careful

protocols, the estimated values are quite accurate. This analysis is used to introduce a new estimator, derived from the BCM estimator, whose behavior is characterized.

5. K. H.Sager [37] had suggested the use of new set of fractal parameters to identify visual color textures. A testing procedure had conducted to evaluate the degree of sensitivity for each kind of fractal dimension spectrums to the visual variation of different textures. The proposed classification methods have been applied on two important set of images. The first set consists of remote sensing images, and the second set consist of breast tumors images. The recognition accuracy for the first set was (100%), while for the second set it was (96.8%). He concluded that the fractal dimension spectrum can provide substantial tools to discriminate texture, and hence they are useful for discrimination and classification of color textures.

1.6 Aim of Thesis

The main objectives of this research can summarize by the following tasks:

1. Provide a review about fractal dimension concept and some relevant texture analysis subjects.
2. Utilize the box-counting method to determine the multi-scale fractal dimension vector.
3. Design a pattern recognition system based on the determination of the fractal dimension. The concepts of fuzzy logic and neural network are going to be use to improve and simplify the recognition decision task.
4. Apply the developed method to the problem of breast cancer diagnosis to recognize and classify breast cancer tissues.

1.7 Thesis Outline

This thesis deals with the development of classification of breast cancer images using the fractal geometry concept. The remaining part of this thesis is structured into four chapters:

Chapter Two "*Theoretical Background*", covers the basics of image and texture analysis related disciplines, followed by fractal theory and an introduction to fuzzy concepts.

Chapter Three "*Design and implementation*", in this chapter the design and implementation steps of the proposed system are presented.

Chapter Four "*Experimental Results*", in this chapter some samples of the conducted tests have been listed, they discussed to evaluate the performance of the proposed system.

Chapter Five "*Conclusion and Future Work*", in this chapter a list of some derived conclusions are given, beside to some recommendations for the future work.

Chapter Two

Theoretical Background

Chapter Two
Theoretical Background

2.1 Introduction

This chapter is dedicated to review some related issues in the field of texture analysis. Following this, the concept of fractal geometry is presented, its importance in the fields of computer vision and image processing is illustrated, especially in the field of texture images analysis. Moreover, the concept of fractal dimension is given, and its measurement idea is explained with focus on the most popular measurement method, i.e., box-counting method.

Then the terms fuzzy logic and neural network are investigated, they are reviewed as a system concept, and as a method of classification.

2.2 Image Analysis

Image analysis is primarily a data reduction process. Image contains enormous amount of data, typically of the order of hundreds of kilobytes or even megabytes. Since much of this information is not necessary to solve a specific computer-imaging problem, so the primary task of the image analysis is to determine exactly what information is necessary [42].

The spectrum of techniques in image analysis is divided into three basic areas, they are: (i) low-level processing, (ii) intermediate-level processing, and (iii) high-level processing.

Figure (2.1) illustrates the relationship between these areas.

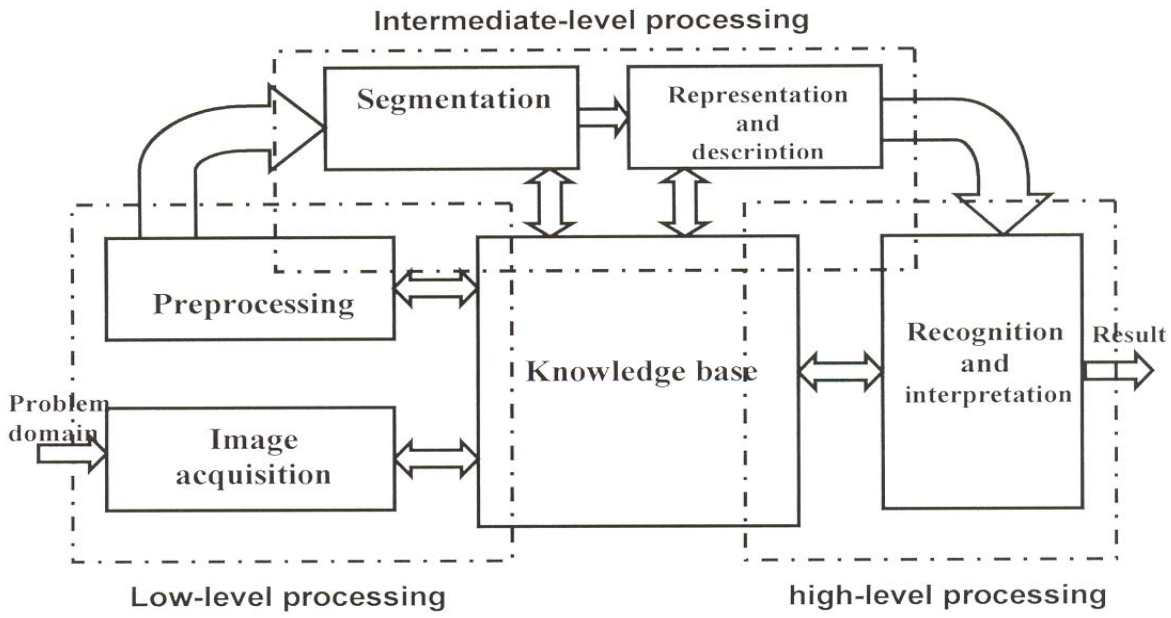


Figure (2.1) Elements of image analysis [13]

The *low-level processing* stage deals with functions that may be viewed as automatic reactions, requiring no intelligence, on the part of the image analysis system [13]. Functions that can be treated as low-level processing are:

1. Image acquisition: is to acquire a digital image. This requires a use of an imaging sensor (such as, a camera, thermal sensor,…) and the capability to digitize the signal produced by the sensor [13].
2. Preprocessing: the preprocessing algorithms, techniques, and operators are used to perform initial processing that makes the primary data reduction and analysis task easier. They include operation related to *extracting regions of interest, performing basic algebraic operations on images, enhancing specific features*, and reducing data in both resolution and brightness [42].

The *intermediate-level processing* stage deals with the task of extracting and characterizing components (regions) in an image resulting from a low-level process [13]. Intermediate-level processes encompass segmentation and description:

1. Segmentation: the goal of image segmentation is to find regions that represent objects [42].
2. Representation and Description: representing a region involve either of two choices [13]:
   a. the region is represented in terms of its external characteristics (its boundary).
   b. or in terms of its internal characteristics (e.g., the pixels comprising the region).

   For example, a region may be represented by its boundary, where the boundary may described by its features (such as its length, the orientation of the straight line joining the extreme points, and the number of concavities in the boundary). Generally, an external representation is chosen when the primary focus is on shape characteristics. An internal representation is selected when the primary focus is on reflectivity properties (such as color and texture).

Finally, the *high-level processing* stage involves with recognition and interpretation [13].

2.3 Feature Extraction

The main task of image analysis system is to extract useful information for solving application based problems. Feature extractions refer to the process of

finding a mapping that reduces the dimensionality of the pattern by extracting some numerical measurements from raw input pattern.

There are many different feature-extraction definitions, each attempted by some computer vision researchers; all of them carry the same idea. The following are some of these definitions:

- It is the process of reducing the representation of an image to a small number of components carrying enough discriminating information [13].
- It is the process of finding a mapping that reduces the dimensionality of the patterns by extracting some numerical measurements from raw input pattern [33].
- It is the process of forming a new (often smaller) set of features (refined representation) from the original feature set [24].

Feature extraction can avoid the curse of dimensionality, improve the generalization ability of classifiers, and reduce the computational requirements of the classifier [24].

Exactly what is done with features is application dependent. If it is working on a computer vision problem, the end goal may be the generation of a classification rule in order to identify objects. But, if there is a need to develop a new image compression algorithm, it may need to determine what image data are important; then the insignificant information can be coarsely compressed or eliminated completely [42].

2.4 Texture

Quantitative study of images is often concerned with four types of parameters, which are of fundamental importance. These four types [4]:

1. Contrast: it is an important measure in image processing, which often determines the quality of an image.
2. Color: it adds more useful discriminatory information to the image.
3. Shape: it is a measure used to recognize various objects contained in an image.
4. Texture: describe the spatial distribution of tonal value within band, and provide useful information for performing automatic interpretation and recognition.

The computational processing of textures can be divided into two main problem areas, segmentation and classification. Texture classification consists of taking whole images and grouping them into texture classes or categories, so as to be able to rapidly detect whether two texture samples are alike or dissimilar. Texture segmentation is, on the other hand, the process by which an image is partitioned into regions of homogeneous texture patterns. Segmentation is a more complex problem than classification since it involves discriminating textures and being able to tell them apart. But it also involves the optimal detection of boundaries between nonhomogeneous texture regions.

Although texture can be recognized when it is seen, it is very difficult to define. This difficulty is demonstrated by the number of different texture definitions attempted by vision researchers. The following are some of these definitions:

- ”It is what continues a macroscopic region. Its structure is simply attributed to the repetitive patterns in which elements or primitives are arranged according to a placement rule” [40].

- "It is non figurative and cellular. An image texture is described by the number and types of its tonal primitives and the spatial organization or layout of its tonal primitives. The fundamental characteristic of texture is that it cannot be analyzed without a frame of reference of tonal primitive being stated or implied. For any smooth gray-tone surface, there exists a scale, such that when the surface is examined it has no texture. Then, as resolution increases, it takes on a fine texture and then a coarse texture" [16].
- "A region in an image has constant texture if a set of local statistics or other local properties of the picture function are constant, slowly varying, or approximately periodic" [41].

These definitions of texture indicate that the key point of texture definition is that the basic pattern elements (texels) must appear with a characteristic repetition inside a given area [30].

2.4.1 Recognition Based on Texture Analysis

Visual texture had been a topic of great interest in the field of image processing up to this moment. While traditional object recognition and classification techniques often rely on areas of constant intensity for object separation and description, this is rarely the case when dealing with most real-world images. Texture analysis attempts to solve this problem by supplying a description of areas which have no uniform intensity, but rather those areas which are easily recognized by human observers as belonging to same object or class. Typically, such areas contain a uniformly varying, periodic or structured appearance. One of the important problems in the development of image

recognition systems is the design of complex image analysis mechanisms capable to discriminate the different regions. The estimation of image characteristics requires significant effort in order to depict an image in terms of a representation which best matches its information content [37].

2.4.2 Texture Characteristics

Texture is thought of being composed of elements that are replicated over the textured region. These textural elements are called texels, such that:

1. They have various sizes and degree of uniformity.
2. They are oriented in different directions.
3. They may be spaced at varying distances in different directions.
4. Their contrast may have various magnitudes and variation.
5. Various amount of background may be visible between texels.
6. Composing the texture from texels will allow for varying degrees of regularity or randomness.

A good texture feature (attribute) should satisfy the following [39]:

1. The feature should enable discrimination of different textures.
2. The feature should be independent of the window size, location and orientation (if the window is representative).

Computational texture features are in most cases the attempt to capture certain stochastic or deterministic similarity criteria. One should always take into consideration that different texture features perform differently on different textures [39].

Uniformity, density, coarseness, roughness, regularity, linearity, directionality, frequency, and phase are important properties used to describe textures. Some of these properties are not independent. For example, frequency is not independent of density, and the direction property only applies to directional textures. The fact that the perception of texture has so many different dimension is an important reason why there is no single method of texture representation that is adequate for variety of textures. For that reason, different types of textures may need different features to represent and classify them [35].

2.4.3 Texture Properties

Identifying the perceived qualities of texture in an image is an important preliminary step toward building mathematical models for texture. In spite of the fact that there is no general definition of texture; texture has number of properties that are assumed to be true. These properties are [41]:

1. Texture is a property of area. So texture is a contextual property where its definition must involve gray values in spatial neighborhood. The size of this neighborhood depends upon the true texture type, or size of primitive defining texture.
2. Texture involves spatial distribution of gray levels. Thus, 2-D histograms or co-occurrence matrices are reasonable texture analysis tools.
3. Texture in an image can be perceived at different scales or level of resolution. For example, consider the texture formed by individual bricks in the wall; at low resolution the interior details in brick are lost. While at

higher resolution, when only a few bricks are in the field of view, the received texture shows the details in the bricks.

4. A region is perceived to have texture when the number of primitive objects in the region is large. If only a few primitive objects are presented, then a group of countable objects is perceived instead of a textured image. In other words, a texture is presented when significant individual “forms” are not presented.

2.5 Fractals

Fractal, in mathematics, is a geometric shape that is complex and detailed in structure at any level of magnification. In 1975, Mandelbrot coined the word “*Fractal*” to describe the object that is too irregular to fit into a traditional geometric setting. The word fractal is derived from the Latin word “*fractus*”, meaning "broken" or "fractured"

Generally, a fractal could be defined "as a rough or fragmented geometric shape that can be subdivided in parts, each of which is a reduced-size copy of the whole" [26].

Fractal should have the following properties [1]:

1. Fractal has details at every scale.
2. Fractal is exactly, approximately, or statistically self-similar.
3. The fractal dimension is greater than its topological dimension.
4. Its formation is by iteration.
5. There is a simple algorithmic to description fractal.

Because they appear similar at all levels of magnification, fractals are often considered to be infinitely complex in informal terms. Natural objects that

approximate fractals to a degree include clouds, mountain regions, lightning bolts, coastlines, and snow flakes. However, not all self-similar objects are fractals for example, the Treal lineT, a straight Euclidean line, is formally self-similar but fails to have other fractal characteristics.

2.5.1 Classification of Fractals

Fractals can be classified according to their self-similarity. Three types of self-similarity can be found in fractals [37]:

1. Exact self-similarity: it is the strongest type of self-similarity, the fractal appears identical at different scales. Fractals defined by iterated function systems often display exact self-similarity.
2. Quasi-self-similarity: it is a loose form of self-similarity, the fractal appears approximately (but not exactly) identical at different scales. Quasi-self-similar fractals contain small copies of the entire fractal in distorted and degenerate forms. Fractals defined by recurrence relations are usually quasi-self-similar but not exactly self-similar.
3. Statistical self-similarity: it is the weakest type of self-similarity, the fractal has numerical or statistical measures which are preserved across scales. Most reasonable definitions of "fractal" trivially imply some form of statistical self-similarity. Fractal dimension itself is a numerical measure which is preserved across scales. Random fractals are examples of fractals which are statistically self-similar, but neither exactly nor quasi-self-similar.

2.5.2 Self-Similarity/ Semi-Self Similarity

An important defining property of a fractal is self-similarity, which refers to an infinite nesting of structure on all scales, so fractal made of parts similar to the whole. In self-similarity, mathematical fractal has some infinitely repeating pattern, and can be made by the iteration of a certain rule [22].

Consider the fern as a typical pattern of many plants in that it exhibits self similarity. A fern consist of a leaf, which is made up of many similar, but smaller leaves, each one, in turn, is made of even smaller leaves. More details could be found at closer look; one will see the overall theme of repeating leaves. Each smaller leaf looks similar to the larger leaf. Looking a little closer, one can see that those small leaves are made up from even smaller leaves closer still and finer detail is visible no matter how close he looks, more detail is always apparent as shown in the Figure (2.2).

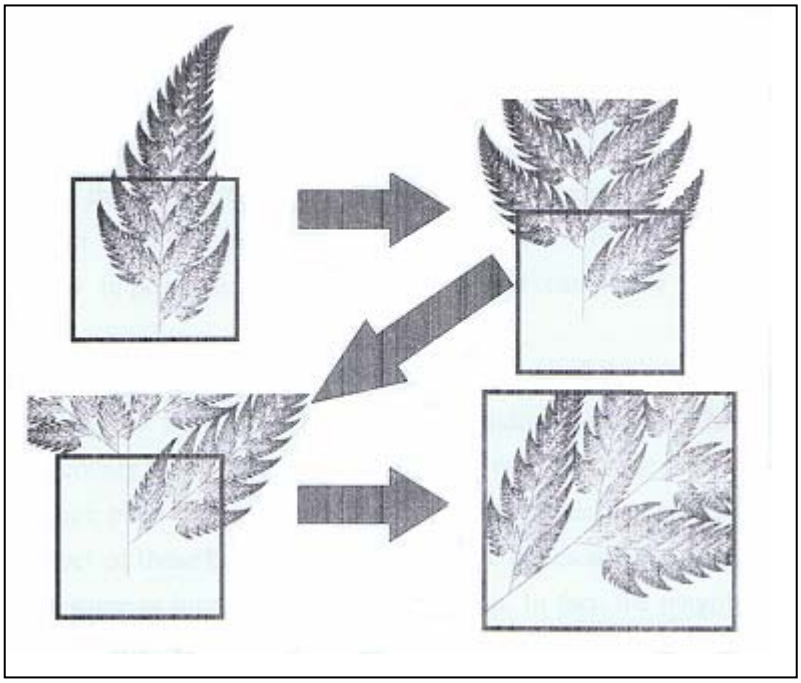


Figure (2.2) Self similarity property of fractals [37]

Of course, in reality, fern does not have a smallest leaf. What is interesting is that the program generates this image is only a few lines long. Same trend is true for all fractals. A very simple algorithm can produce infinitely complex objects.

2.5.3 Fractal Geometry

Fractal geometry is the branch of mathematics which studies the properties and behavior of fractals. It describes many situations which cannot be explained easily by classical geometry, and has often been applied in science, technology, and computer-generated art. The roots of fractals concept is relevant to problem of measuring the size of some objects for which the traditional definitions that based on Euclidean geometry or calculus fail. The basic concept of fractals is that they contain a large degree of self-similarity. This means that they usually contain little copies of themselves buried deep within the original, and they have infinite detail. [37].

While the classical Euclidean geometry works with objects which exist in integer dimensions, fractal geometry deals with objects in non-integer dimensions. Fractal geometry, however, is described by algorithms (a set of instructions on how to create a fractal object), while Euclidean described by simple formula.

One of the simpler fractal shapes is the Von Koch snowflak, the method of creating this shape is to repeatedly replace each line segment with four line segments. The process starts with a single line segment and continues for ever. [2].

2.5.4 Types of Fractals

According to their synthesis techniques, fractals are classified into:

1. Escape-time fractals: they are defined by a recurrence relation at each point in a space (e.g., application of certain mathematical constraint on the points of a complex plane). The Mandelbrot set is an example of escape-time fractals.
2. Iterated function systems (IFS): they have a fixed geometric replacement rule. Cantor set and Koch snowflake are some examples of such fractals.
3. Random fractals: they are generated by stochastic rather than deterministic processes. Fractal landscapes and Brownian tree are examples belong to this kind of fractals.

2.5.5 Monofractals / Multifractals [44]

Monofractals is a small group of fractals have one certain fractal dimension, which is scale invariant. Most of natural fractals have different fractal dimensions depending on the scale. They are composed of many fractals with the different fractal dimension. They are called multifractals (e.g. set of the different coastlines). To characterize a set of multifractals there is no need to determine all their fractal dimensions, it is enough to evaluate their fractal dimension at the same scale.

2.5.6 Fractal in Image Analysis

Various fractal techniques are used in image analysis, they used [34]:

1. To characterize the overall spatial complexity of an image.
2. To supply image classification with textual information.

3. To describe the geometric complexity of the shape of feature classes in a classified image.
4. To examine the scaling behavior of environmental phenomena.

2.6 Fractal Dimension

Fractal dimension is a measure of how 'complicated' a self-similar object is. In a rough sense, it measures 'how many points' lie in a given set. Also, fractal dimension captures the notion of 'how large a set is'. Fractal dimension measures the rate of addition of structural detail with increasing magnification, scale or resolution, therefore, serves as a quantifier of complexity [22].

Fractal dimension is a popular parameter for explaining certain phenomena, and for describing natural textures, and it represents an important feature of textural images. Hence, it is used to characterize roughness and self-similarity in a picture. This feature is used in texture description, classification, shape analysis and other computer vision and image processing problems. From digital point of view, the fractal dimension is a measure of how close together the pixels in a binary image are [37].

Figure (2.4) illustrates the geometric concept of fractal dimension. A line segment is divided into N equal parts, each of them scaled by a factor $r=1/N$. Similarly, for a plane the scale is $r=1/N^{1/2}$ and for a 3D shape $r=1/N^{1/3}$. In general, for a D-dimensional self-similar object it is $N=1/r^{D}$. Hence, the ratio $D=log(N)/log(1/r)$ is called similarity (or fractal) dimension (D) in self-similarity sets. [2].

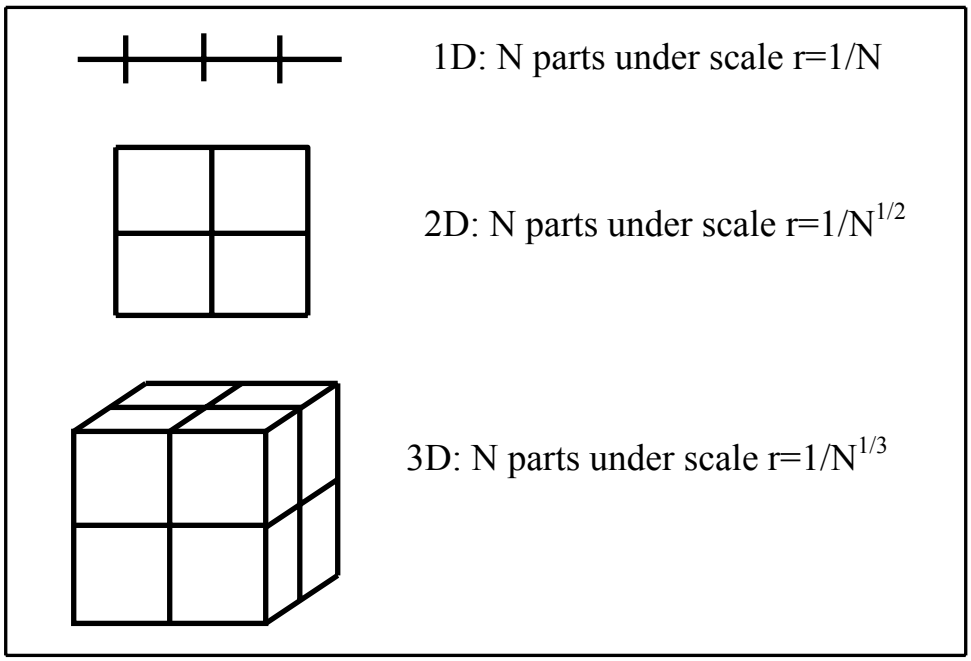


Figure (2.4) Estimation of fractal dimension [2]

2.6.1 Measuring Fractal Dimension

The most well known method for estimating the fractal dimension of digital image data, is called box-counting method (BCM). This method is characterized as [44]:

1. The most accurate method used to assess fractal dimension.
2. It determines the fractal dimension of black & white digitized images of fractals.
3. It works by covering fractal area (its image) with boxes (squares) and then evaluating how many boxes are needed to cover fractal completely. Repeating this measurement with different sizes of boxes will result into a relationship between the logarithm of box size (i.e, log(s) where $s=L^2$) and the logarithm of the number of boxes needed to cover fractal. The slope of

this linear relationship is referred as box dimension. Box dimension is taken as an appropriate approximation of fractal dimension.

In the example shown in Figure (2.5) the upper-left image (the letter 'A') involves (4) boxes. In the upper-right image the letter 'A' involves (5) boxes. In the lower-left image (11) boxes are involved. While, in the lower-right image (35) boxes are involved [25].

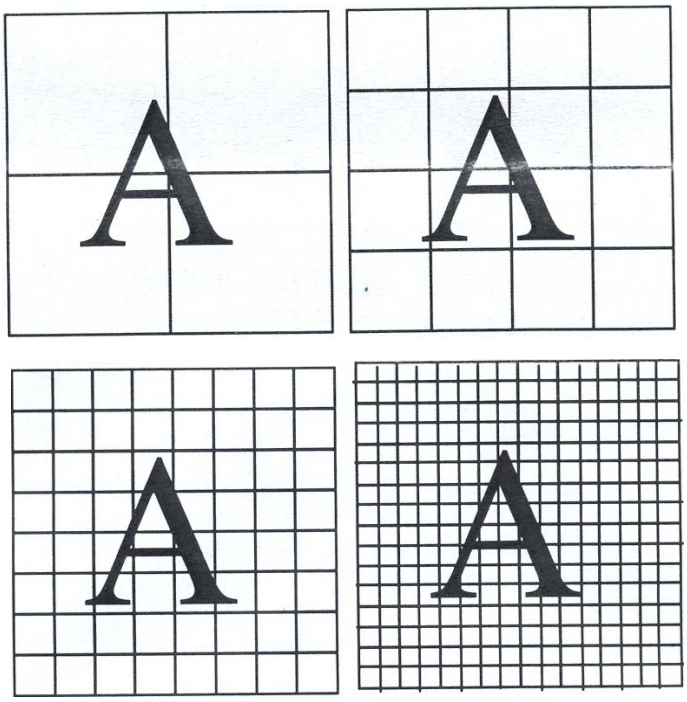


Figure (2.5) An example for box counting method [25]

Consider a bounded set (A) in Euclidean n-space. The set A is said to be self-similar when (A) is the union of (N) distinct (non-overlapping) copies of itself; each has been scaled down by a ratio (r) in all coordinates. Simply, the relation gives the fractal (or similarity) dimension of A is,

$$D = \frac{\log(N)}{\log(\frac{1}{r})} \quad ,\ldots\ldots (2.1)$$

Suppose one can cover the set A with n-dimensional boxes of size (L_{max}). If the set (A) is scaled down by the ratio (r), then there are ($N=r^{-D}$) subsets, and so the number of boxes of size ($L=rL_{max}$) needed to cover the whole set is given by:

$$N\,(L) = \frac{1}{r^D} = [\frac{L_{max}}{L}]^D \quad ,\ldots\ldots (2.2)$$

The simplest way to estimate (D) from the above equation is to divide the n-dimension space into a grid of boxes with side length (L), and to count the number of non-empty boxes. If N(L) is computed for several values of (L), then (D) can be estimated as the slope of the best linear fit between the corresponding pairs of the data set {ln (L), -ln(N(L))} [29].

Accordingly, in the case of digital gray image, for every pixel in the image of size (M×N) with the largest box size to be used (set as L_{max}), the number of points (m) within each box of size (L) centered at the pixel(x,y, f(x,y)) is counted and recorded as m(L). The centering and counting activity is restricted to pixels having all their neighbors' inside the image. The image is then divided into overlapping windows. The overlap is decided by the increment steps between successive windows. For digital image surface, the values of (m) range from (1 to L^2) for a cube of side (L). The estimation of fractal dimension is the slope obtained by performing the least squares fit upon the data set {$\ln(L^2)$,-ln(m(L))} [21].

Box-counting dimension is much widely used than other self-similarity dimension, since the box-counting dimension can be measured for pictures that are not self-similar, taking into consideration that most of the real life applications are not self-similar.

2.6.2 Problems with Measuring Fractal Dimension

1. It is necessary to remember that the image, whose fractal dimension is going to be calculated, represent a real textural image with a certain scale (resolution). In this connection, texture of smaller scales generally displays more details of the texture than those of bigger ones. This is quit similar to the distance of the observer from real object: If one stands far away from something it can be hardly recognized; getting closer, which means increasing the scale of the scene, the details become visible. Certain details can only be identified from a certain distance or scale on. Thus translating the original texture into a digital image, first of all, means to define the distance or the scale-range with the aid of the biggest and smallest recognizable detail. Using box-size beyond this range will lead to falsified results, no additional details of the real texture are included into the calculation because the image does not contain them [37].
2. The quality of the image itself affects the accuracy of measuring fractal dimension. Spatial image quality may describe by the number of pixels per inch. Also, the "quality" means how many details are getting lost because of the preparation of the texture by scanning and after treatment, and not because of the distance (i.e., scale) [5].

3. When one digitizes a shape or a texture for subsequent fractal analysis, the way the image is generated by the scanner is not usually regarded as a trivial detail. The length, orientation and placement of an image with respect to the initial box are all potential causes of error, which can cause a significant bias in estimation of fractal dimension. This is important since the placement of the initial box is normally either arbitrary or determined by field exposure. Moreover, the presence of unconnected parts in the same image must be accounted for, since these parts do not necessarily belong to the same higher order shape. Finally, the process of digitization itself is a major cause of error.

2.7 Fuzzy Logic

Fuzzy logic (FL) is basically a multivalued logic that allows intermediate values to be defined between conventional evaluations like yes/no, true/false, black/white, etc. Notions like rather warm or pretty cold can be formulated mathematically and processed by computers. In this way an attempt is made to apply a more human-like way of thinking in the programming of computers.

Some of the essential characteristics of FL are related to the following facts [10]:

1. In FL, exact reasoning is viewed as limiting case of approximate reasoning.
2. In FL, everything is a matter of degree of decision maker.
3. In FL, knowledge is interpreted as a collection of elastic or, equivalently, fuzzy constraint on a collection of variables.
4. Inference is viewed as a process of propagation of elastic constraints.
5. Any FL system can be fuzzified.

There are two main characteristics of fuzzy systems that give them better performance of specific applications [10]:

1. Fuzzy systems are suitable for uncertain or approximate reasoning, especially for the system whose mathematical model is difficult to drive.
2. FL allows decision making with estimated values under incomplete or uncertain information.

Here is a list of general remarks about fuzzy logic:

1. FL is conceptually easy to understand: the mathematical concepts behind fuzzy reasoning are very simple. Fuzzy logic is a more intuitive approach without the far-reaching complexity.
2. FL is flexible: with any given system, it is easy to layer on more functionality without starting again from scratch.
3. FL is tolerant of imprecise data: everything is imprecise when looking closely enough, but more than that, most things are imprecise even on careful inspection. Fuzzy reasoning builds this understanding into the process rather than taking it onto the end.
4. FL is based on natural language: the basis for fuzzy logic is the basis for human communication. This observation underpins many of the other statements about fuzzy logic. Because fuzzy logic is built on the structures of qualitative description used in everyday language, fuzzy logic is easy to use.

2.7.1 Fuzzy Set

The concept of the set is one of the most basic concepts in both logic and mathematics [19]. In a classical set theory, a set is normally defined as a

collection of objects, which can be finite, countable, or uncountable. An element x can be either belong to or not belong to a set A. If set A represent the concepts, for example, "male" and "female", a person is either a "male" or "female" because the concepts "male" and "female" are well defined without any ambiguity. This can be totally different if we consider the concepts "tall "and" short". A person have a height of 1.95 meter is commonly regarded as tall. But a person with height of 1.7 meter can be said to be tall for Asian with very high certainty, while for Europeans and American, this person is not considered to be tall. To handle these ambiguous concepts, the conventional set theory was extended to fuzzy set theory [32]. Fuzzy sets are sets (or classes) without sharp boundaries; the transition between membership and nonmembership of a location in the set is gradual. A fuzzy set is characterized by a fuzzy membership grade (also called a *possibility*) that ranges from 0.0 to 1.0, indicating a continuous increase from nonmembership (0) to complete membership (1).

2.7.2 Membership Functions

A *membership function* (MSF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership), whose value extend between 0 and 1.

Accordingly, a person is a member of the set “tall people” to the degree to which he / she meets the operating concept of “tall”. The degree of membership of individual in a fuzzy set expresses the degree of compatibilities of the concept represented by the fuzzy set. Each fuzzy set, A, is defined in terms of a relevant universal set, X, by a function analogous to the characteristic function.

This function, called a membership function, assigns to each element x of X a number, A(x), in the closed unit interval [a, b] that characterizes the degree of membership of x in A. MSF are thus functions of the form [19]:

$$A: X \longrightarrow [a, b] \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.3)$$

In conventional fuzzy set classifiers, the range of each input variable is divided into several intervals. Then, each interval is considered to be a fuzzy set, and an associated MSF is defined. Thus, the input space is divided into several subregions that are parallel to input axis. For each subregion a fuzzy rule is defined; if the input is in the sub region, then the input belongs to the class associated with the subregion. The degrees of membership of an unknown input for all fuzzy sets are calculated and the input is classified as a member of the class with the maximum degree of membership. Therefore, the MSFs directly influence the performance of the fuzzy classifier [3].

There are many types of membership functions, the most simple and widely used types are the Triangular, Trapezoidal, and Bell-Shape functions.

A. Triangular Membership Functions

The triangular membership function have shape similar to that shown in Figure (2.6). At x=c (i.e., the center of the membership function) the degree of membership is 1, and it decreases as x moved away from c, and reaches 0 when $x \leq v$ or $x \geq V$. The center c is not necessarily the middle point between v and V. the membership function m(x) for the input variable x can be defined as follows [3]:

$$m(x)=\begin{cases} 0 & \text{for } x<v. \\ 1-\dfrac{c-x}{c-v} & \text{for } v\le x\le c, \\ 1-\dfrac{x-c}{V-c} & \text{for } c<x\le V, \\ 0 & \text{for } x>V, \end{cases} \qquad \text{,……. (2.4)}$$

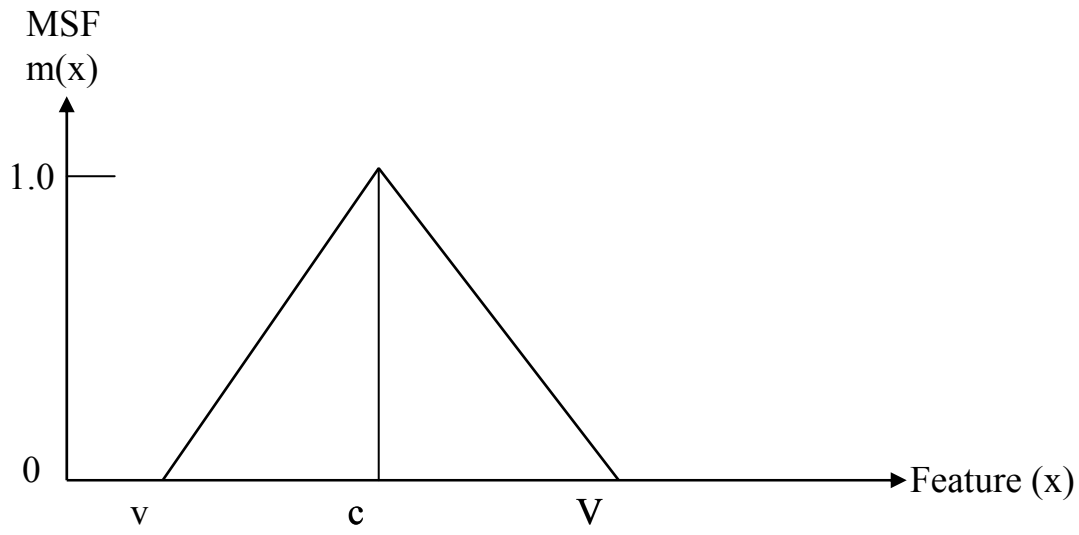


Figure (2.6) Triangular membership function

B. Trapezoidal Membership Functions

Another important class of MSFs has trapezoidal shape, which is illustrated by the generic graphical representation that shown in Figure (2.7). Each function in this class is fully characterized by four parameters (a, b, c and d) such that [19]:

$$m(x) = \begin{cases} \frac{x-a}{b-a} & \text{when } a \le x \le b \\ 1 & \text{when } b \le x \le c \\ \frac{d-x}{d-c} & \text{when } c \le x \le d \\ 0 & \text{otherwise} \end{cases} \quad ,\ldots\ldots\ldots\ldots(2.5)$$

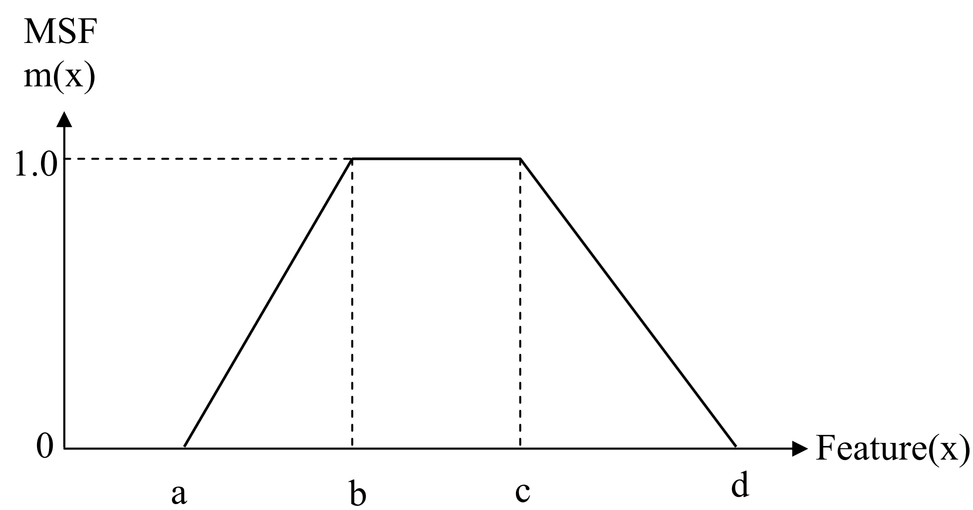


Figure (2.7) Trapezoidal membership function

2.7.3 Fuzzy Rules [14]

The most common fuzzy system uses rules to formally represent knowledge. Although a single type of rule does not exist, if-then representation is a standard. A basic if- then rule can be written in the form:

IF a_1 AND a_2 … AND a_n THEN b ,……………………………… (2.6)

Where $\{a_i, 1\le i \le n\}$, and *b* are fuzzy propositions. Each rule has an antecedent part and a consequent part. The antecedent part is the collection of

conditions connected by AND, OR, NOT logic operators, and the consequent part represents its action.

Each rule represents some reasonable assumptions about the actions (output values) that should be taken in the case of that state of the system (input values); both input and output are described by fuzzy sets.

2.8 Artificial Neural Networks (ANN) [20]

An artificial neural network (or simply a neural network) is a biologically inspired computational model that consists of processing elements (neurons) and connections between them, as well as of training and recall algorithms.

The structure of an artificial neuron is defined by inputs, having weights bound to them; an input function, which calculates the aggregated net input signal to a neuron coming from all its inputs; an activation (signal) function, which calculates the activation level of a neuron as a function of its aggregated input signal and (possibly) of its previous state. The output signal equal to the activation value that emitted through the axon of the neuron. In general, an artificial neuron model is based on the following parameters:

1. Input connections (or inputs): $x_1, x_2, \ldots, x_n$. There are weights bound to the input connections: $w_1, w_2, \ldots, w_n$; one input to the neuron, called a bias, has a constant value of 1 and is usually represented as a separate input, say x_0, but for simplicity it is treated here just as an input, clamped to a constant value.
2. Input function f, calculates the aggregated net of the input signal to the neuron u=f(x, w), where x and w are the input and weight vectors, respectively; f is usually the summation function: $u = x_0 + \sum_{i=1\ldots n} x_i . w_i$.

3. An activation (signal) function s calculates the activation level of the neuron a = s(u).
4. An output function calculates the output signal value emitted through the output (the axon) of the neuron: o = g(a); the output signal is usually assumed to be equal to the activation level of the neuron, that is, o = a.

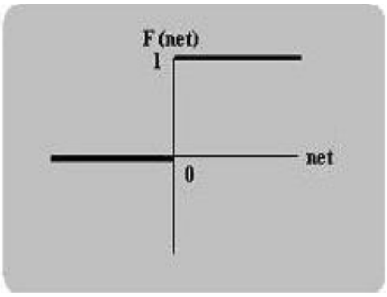
2.8.1 Types of Neural Network Classifiers

There are two types of ANN classifiers:

1. *Supervised neural network classifiers:* supervised learning algorithms include a special case of reinforcement learning, where a given pattern is identified as a member of already known classes. Multilayer feed forward have emerged as a popular model for pattern classification tasks. The most important attribute of multilayer feed forward is that it can learn mapping of any complexity [28].
2. *Unsupervised neural network classifiers:* in such kind of classifiers, the classes are determined by locating clusters in input (feature) space, and assuming that each cluster corresponds to a class. The problem becomes a clustering identification. Competitive neural networks (such as self-organization map and Kohonen 's LVQ "Learning Vector Quantizer ") are often used to cluster input patterns. Despite their attractive architectures and biological / neurophysiological plausibility, the updating procedures used in these neural-based clustering methods are quite similar to some classical partition clustering approaches. The final step in the unsupervised classification is to decide which cluster represents which physical class [24].

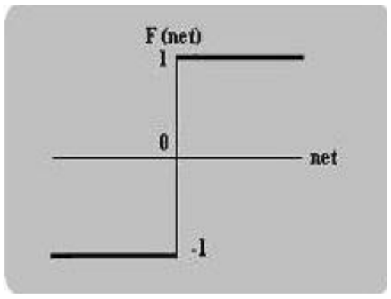
2.8.2 Types of Activation Functions

Various activation functions can be used to define a neuron's output, and the used activation function which was utilized often depends on the learning rule. The back propagation-learning rule, which was utilized in this study and will be described later, requires differentiable function such as the sigmoid function. Figure (2.8) shows the commonly employed activation functions and their outputs, also their mathematical expressions are given in equations (2.7-10) [45]. In this study the unipolar sigmoid activation function was used which is presented by equation (2.9).



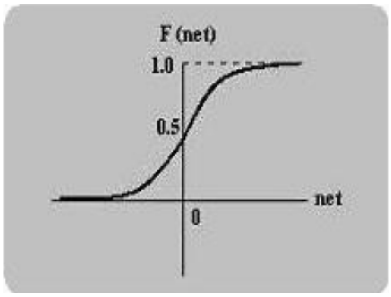
$$F(net)=\begin{cases} 0 \text{ if } net < 0 \\ 1 \text{ if } net > 0 \end{cases} \quad \text{,……(2.7)}$$

Unipolar Binary Function



$$F(net)=\begin{cases} 1 \text{ if } net \geq 0 \\ -1 \text{ if } net < 0 \end{cases} \quad \text{,……....(2.8)}$$

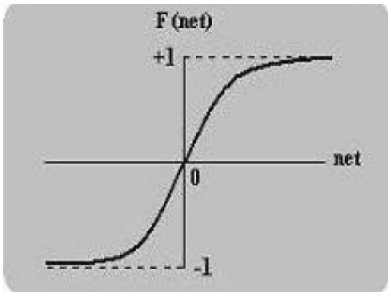
Bipolar Binary Function



$$F(net)=\frac{1}{1+e^{-\lambda net}} \text{,……(2.9)}$$

Where $\lambda \geq 1$

Unipolar Sigmoid Function



$$2F(net) = \frac{2}{1+e^{-net}}$$

Figure (2.8) Common activation functions [45]

2.8.3 Neural Network Classification Phase

Classification by Neural Networks (NNs) has two phases [23]:

1. *Learning Phase:* in this phase, the ANN weights are adjusted (trained) so that the input patterns are more likely to be correctly classified.
2. *Retrieving phase* (recognizing phase): the objective of this phase is to determine to which class the input pattern belongs, this decision depends on the output values. The output values depend on the input values and network weights (determined in training phase), the function used in output nodes is called discrimination function.

2.8.4 FeedForward Neural Network (Back-Propagation)

Feedforward NNs with backpropagation (BP) constitute one of the most popular neural network models. Their supervised learning procedure is based on a simple idea: if the network gives a wrong answer, the weights on the connections are adjusted to reduce this error and increase the likelihood of a correct response in the future. The model is typically composed of three types of layers (each of simple units), namely; the input layer, the hidden layer, and

the output layer. Each layer is fully connected to the previous and next layers through weighted connections that propagate the signal in a forward direction from the input to the output; Figure (2.9) presents the Multilayer-Feedforward Neural Network Architecture. No processing takes place at the input layer. That is, the input vector, provided to the input layer, is directly propagated to the hidden layer through the weighted connections. Assuming *p* input units and *q* hidden units, each hidden unit *j* computes its activation level a_j, according to the following formula [18]:

$$a_j = \frac{1}{(1+e^{-s_j})} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots(2.11)$$

Where,

$$S_j = \sum_{i=1\ldots p} W_{ij} X_i + X_{0j}, \quad j = 1,\ldots,q \quad ,\ldots\ldots\ldots\ldots\ldots\ldots(2.12)$$

In this formula, X_i is the i th component of the input vector *X* (or, equivalently, the activation level of input unit *i*),W_{ij} is the connection weight between input unit *i* and hidden unit *j*, and *j* is the input bias of unit *j*. The activation level of each hidden unit is in the interval [0,1] due to the sigmoid transformation. These activation levels are then propagated to the output units through the weighted connections between the hidden and output layers. The same processing takes place at each output unit, and the final activation levels of these units (i.e., the output vector) represent the response of the network to the input vector *X*.

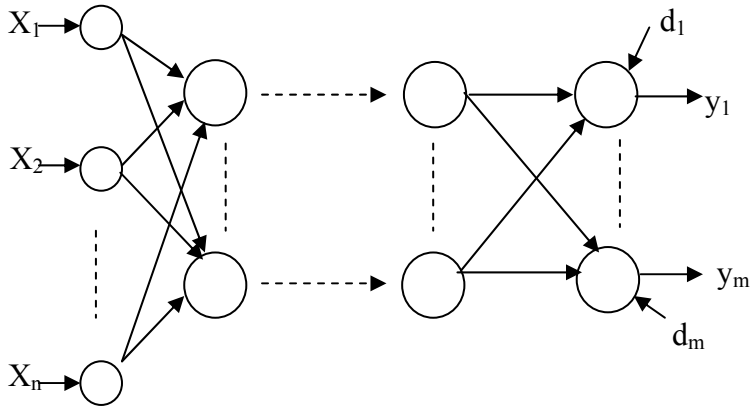


Figure (2.9) Multilayer-Feedforward Neural Network Architecture [18]

The power of this model comes from its ability to adjust the connection weights to perform a particular task. The BP learning algorithm uses a training set of examples that characterizes the task. In this set, each input vector is associated with a desired response or output vector. These examples are processed one by one by the network; and the weights are adjusted accordingly. The mean-squared error between the actual outputs (calculated by the network) and the desired outputs (over all examples in the training set) is typically used to monitor the network performance. The training stops when the error measure becomes stable.

1. Strategies of Adjusting the Weights

There are three strategies for adjusting the weights of BP network [47, 46]:

a. On-Line: the weights are adjusted after the entry of each training pattern.

b. Batch (Off-Line): the weights are adjusted after the entry of all training patterns.

c. Block Update: a set of random patterns are chosen, and after they entered the system should adjust its weights.

2. The BP Algorithm Steps

The terms used in the equations of the applied algorithm are defined in table (2.1). The algorithm consists of the following steps [45, 11]:

Table (2.1) Terms used in back propagation algorithm

Terms	Description
i	Index to input neurons.
j	Index to hidden neurons.
k	Index to output neurons.
n_{in}	Number of input nodes.
n_{hid}	Number of hidden nodes.
n_{out}	Number of output nodes.
X_i	Input of node (i) in the input layer including bias node (X_0=1).
S_i	Output of node (i) in the input layer including bias node (S_0).
h_j	Output of node (j) in the hidden layer including bias node (h_0=1).
O_k	Output of node (k) in the output layer.
net_j	The activation output of hidden node (j) in the hidden layer.
net_k	The activation output of output node (k) in the output layer.
V_{ij}	The weight of the connection from node (i) in the input layer to node (j) in the hidden layer including bias node weight (V_{0j}).

W_{jk}	The weight of the connection from node (j) in the hidden layer to node (k) in the output layer including bias node weight (W_{0k}).
$V_{ij}^{(t)}$	The weight of the connection from node (i) in the input layer to node (j) in the hidden layer including bias node weight (V_{0j}) at time (iteration) t.
$V_{ij}^{(t+1)}$	The weight of the connection from node (i) in the input layer to node (j) in the hidden layer including bias node weight (V_{0j}) at time t+1 (or t+1 iteration).
$W_{jk}^{(t)}$	The weight of the connection from node (j) in the hidden layer to node (k) in the output layer including bias node weight (W_{0k}) at time (iteration) t.
$W_{jk}^{(t+1)}$	The weight of the connection from node (j) in the hidden layer to node (k) in the output layer including bias node weight (W_{0k}) at time t+1 (or t+1 iteration).
δ_k	The error of the output node (k).
δ_j	The error of the hidden node (j).
d_k	Desired output of node (k) in the output layer.
H	Learning rate ($0< \eta <1$).
A	A constant specifying the momentum ($0<\alpha<1$)
$\Delta W_{jk}^{(t)}$	Change in weight of the connection from node (j) in the hidden layer to node (k) in the output layer including bias node weight (W_{0k}) at time (iteration) t.
$\Delta V_{ij}^{(t)}$	Change in weight of the connection from node (i) in the input layer to node (j) in the hidden layer including bias node weight (V_{0j}) at time (iteration) t.

$\Delta W_{jk}^{(t+1)}$	Change in weight of the connection from node (j) in the hidden layer to node (k) in the output layer including bias node weight (W_{0k}) at time t+1 (or t+1 iteration).

1. **Initialization:** Set all weights and biases of the network to small random numbers. Note that the activation of node biases is fixed at 1.
2. **Presentation of training examples:** The network is trained using some selected training samples (or instances), it is expected that the network will learn in one cycle of the algorithm. Steps (3 and 4) listed below are performed for every training sample (instance).
3. **Forward Pass (calculation of activation):**
   a. The activation level of an input unit (S_i) is determined by the instance presented to the network, as given in following:

$$S_i = X_i \qquad 0 \leq i \leq n_{in} \qquad ,\ldots\ldots(2.13)$$

   b. The activation level (h_j) of a hidden unit and the output (O_k) of an output unit is determined by the equations given below, as mentioned in section (2.8.2) the unipolar sigmoid function (presented in equation 2.9) is used:

$$h_j = f(net_j) = f\left(\sum_{i=0}^{n_{in}} V_{ij} X_i\right) \qquad 0 \leq j \leq n_{hid} \qquad ,\ldots\ldots(2.14)$$

$$O_k = f(net_k) = f\left(\sum_{j=0}^{n_{hid}} W_{jk} h_j\right) \qquad 1 \leq k \leq n_{out} \qquad ,\ldots\ldots(2.15)$$

4. Backward Pass (Weight Training)

a. Compute the difference between the actual output and the desired output in order to calculate the errors of the output neurons (δ_k) and the errors of the hidden neurons (δ_j), as described by the following equations:

$$\delta_k = O_k(1-O_k)(d_k - O_k) \quad ,\ldots\ldots(2.16)$$

$$\delta_j = h_j(1-h_j)\sum_{k=1}^{n_{out}} \delta_k w_{jk} \quad ,\ldots\ldots(2.17)$$

b. Use the resulting differences δ_k and δ_j to calculate the changes in weight ΔW_{jk} and ΔV_{ij}, as shown by the following equations:

$$\Delta W_{jk}^{(t)} = \eta.\delta_k.h_j \quad ,\ldots\ldots(2.18)$$

$$\Delta V_{ij}^{(t)} = \eta.\delta_j.S_i \quad ,\ldots\ldots(2.19)$$

Sometimes, the convergence need to be faster, this could be done by adding a momentum term to both of ΔW_{jk} and ΔV_{ij}, as given in the following equations:

$$\Delta W_{jk}^{(t+1)} = \eta.\delta_k.h_j + \alpha.\Delta W_{jk}^{(t)} \quad ,\ldots\ldots(2.20)$$

$$\Delta V_{ij}^{(t+1)} = \eta.\delta_j.S_i + \alpha.\Delta V_{ij}^{(t)} \quad ,\ldots\ldots(2.21)$$

c. Adjust the weights by using the following equations:

$$W_{jk}^{(t+1)} = W_{jk}^{(t)} + \Delta W_{jk}^{(t)} \quad ,\ldots\ldots(2.22)$$

$$V_{ij}^{(t+1)} = V_{ij}^{(t)} + \Delta V_{ij}^{(t)} \quad ,\ldots\ldots(2.23)$$

5. Iteration: Steps 2-4 are iterated until reaching the required convergence (in terms of the selected error criterion).

Chapter Three

Design and Implementation

Chapter Three
Design and Implementation

3.1 Introduction

This chapter presents the design considerations and implementation steps of the establishment of two recognition systems for breast cancer tissue images using fractal geometry concepts. The two systems are designed to investigate the idea of using fractal features as textural features. In each of these two systems, the recognition of breast cancer tissues is based on one of the two considered types of decision making methods: fuzzy logic, and artificial neural network.

3.2 Breast Tissue Recognition Based on Fuzzy Logic

Figure (3.1) shows the layout of the first proposed system. This system consists of two phases:

Enrollment Phase: the input to this phase is a set of typical breast tissue images whose types are well known. The first module in this phase performs a sequence of preprocessing operations which includes: (1) read image header (2) convert the color image to gray image and (3) image smoothing using average filter. The second module is for feature extraction, this module includes two stages: (1) application of box counting method to determine the multi-fractal dimension and (2) building the membership functions. The result of this phase is a database contains: (1) the extracted fractal dimensions histograms and (2) fuzzy sets (rules).

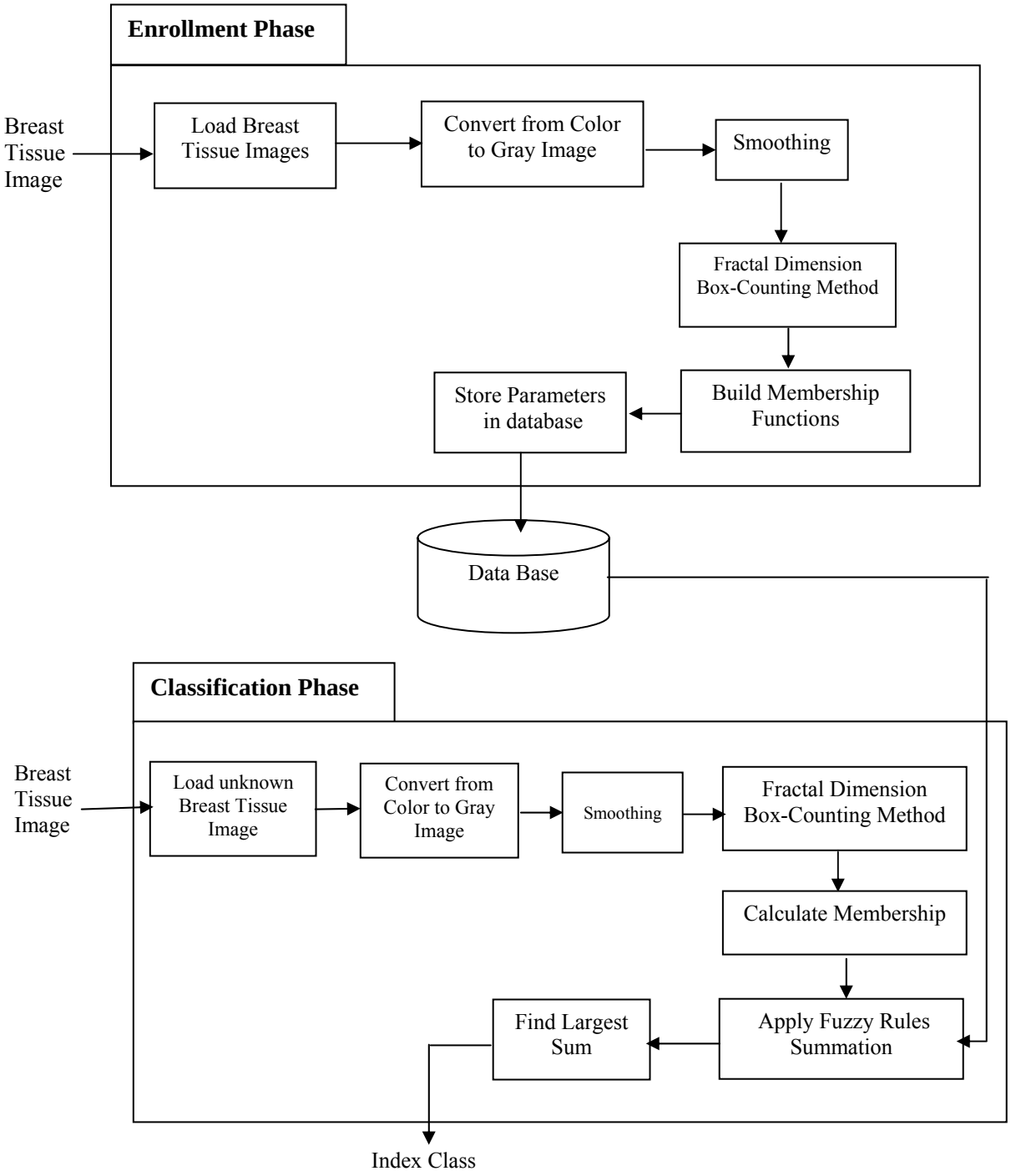


Figure (3.1) The Breast Recognition System Based on Fuzzy Logic

Classification Phase: in this stage the input is an unknown color breast tissue image. The same preprocessing operations and features extraction stages that performed in enrollment phase are implemented here. The next stage in this phase is the calculation of the sum of the membership values of all extracted fractal features from the unknown image, using the fuzzy sets (which have been built and stored in database as the output of the enrollment phase). This sum of membership is determined for all considered classes (or types) of breast tissues. The index of the class that corresponds to highest collected sum of the membership function is considered as the class index of the tested breast tissue.

3.2.1 Preprocessing Stage

In this stage, a bitmap (Bmp) image file of a colored breast tissue is input to the system. This stage includes reading the header of the BMP image file format. This header contains the required information about the image attributes (such as, height and width of the image, data type, etc). These information are very important to load the image data. The next step in this stage is the computation of image data width (W) using the following formula:

$$W = 4 \times \lfloor (W_p \times C_r + 31)/32 \rfloor - 1 \quad , \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.1)$$

Where, W_p represents the width of image in pixels. C_r represents the image color resolution or depth (i.e., bits per pixel), and $\lfloor x \rfloor$ denotes the higher integer value that is smaller than or equal to (x).

The next step is the conversion of the color image data component to gray. The image pixels are directly represented by 24-bits/pixel (which means it has

no RGB palette), where 8 bits (one byte) are used to represent each of the three primary color components (i.e., red, green, blue). The conversion to gray is done using the following equation:

$$g(x,y) = \frac{R(x,y) + G(x,y) + B(x,y)}{3} \qquad \text{,…….. ..(3.2)}$$

Where, *R*, *G*, *B* are the red, green blue components values, respectively, at $(x,y)^{th}$ pixel, and *g* is the corresponding gray level value.

The last step of preprocessing stage is the image smoothing, where the average filter (3×3) was used. The average filtering process could be done using the following:

$$g_{smooth}(x,y) = \frac{1}{9}\sum_{i=x-1}^{x+1}\sum_{j=y-1}^{y+1} g(i,j) \qquad \text{,……...(3.3)}$$

Where, *g(i,j)* is the gray level value, and g_{smooth} is the corresponding smooth value.

3.2.2 Feature Extraction

After performing the steps of preprocessing stage, the feature extraction stage begins. This stage includes the following steps:

A. Determination of Fractal Dimension Vector

The box-counting method (BCM) is the most conventional and accurate method for calculating fractal dimension. It relies on the digitized representations of the image. In this research, it is used to compute the fractal

dimension of the gray images of breast tissues. The size of box was varied to get many fractal dimensions, each one is measured at different scale. The values of fractal dimensions at each pixel in the image are determined.

The inputs to BCM are: (1) the gray image (i.e., breast tissue sample image), (2) L_{max} value which refer to the largest considered box size and (3) L_{stp} value refer to the increment step of the box size. The output of this algorithm is a file contains the histograms of the determine fractal dimensions for all points of the input image (except those lay at the margins of the image). The histogram elements values are saved inside a file to be used in the next stage.

The histogram of each extracted fractal dimension, at specific box size, is taken as a feature vector, which consists of N_{bins} histogram elements. This number represents the number of quantization bins (in our research research, N_{bin} is set 100) used to quantize the values of determined fractal dimensions. The quantization process is applied to make the size of feature vector, for all scales and for all tested samples, is same. The quantization process was performed by applying the following equation:

$$F_q=\text{round}(N_{bins}\times (F_D-2)) \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.4)$$

Where, N_{bin} is the number of quantization bins

F_D is the actual fractal dimension value.

F_q is the quantization index of the fractal dimension.

Algorithm (3.1) shows the steps taken to compute the fractal dimension for a specific scale using box-counting method.

Algorithm (3.1) Compute Fractal Dimension (box-counting method)

Input: *Img* // *is the array of the gray image*
L_{max} // *is the maximums box size*
L_{stp} // *is the step size for increasing box size*
Width // *is the width of the image*
Height // *is the height of the image*
N_{bins} // *is the number of quantization bins*
Output: *His* // *is the histogram of the fractal dimension for each box*

Procedure:

```
loop for Y=Lmax to Wight-Lmax
    loop for X=Lmax to Hight-Lmax
        loop for L= 1 to Lmax
            set N [L] ← 0
            loop for Xp =X-L to X+L
                loop for Yp = Y-L to Y+L
                    if | Img (Xp, Yp) – Img (X, Y) | ≤ L then
                        set N [L] ← N [L] +1
                endloop Yp
            endloop Xp
        endloop L
        set sm1 ← 0
        set sm2 ← 0
        loop for M = 0 to Lmax
            set LogL ← 2× Log (2×M+1)
            set LogNL ← Log (N [M])
            set sm1 ← sm1 + LogL × LogN
            set sm2 ← sm1 + LogL × LogN
            If (((M Div Lstp) = 0) And (M>0)) Then
                set Fdim ← round ((1-Sm1/Sm2) × Nbins)
                set His (Fdim, M/Lstp) ← His (Fdim, M/Lstp) + 1
            endif
        endloop M
    endloop X
endloop Y
save His() elements in "Histogram" file
end
```

B. Building Membership Functions (Fuzzy Sets)

After the construction of histogram file (as a last step in the stage of fractal dimension calculation), the number of elements of each histogram vector is

N_{bin} elements. The value of N_{bin} may be set large and in such case it causes an increase in computation time (if it is used directly in matching process). For this reason fuzzy logic in adopted to bypass this situation. Two simple types of membership functions (i.e., Triangular and Trapezoidal membership functions) have been utilized in this research. The result of the stage of building membership functions is a list (or file) contains sets of coefficients; each set contains either four or five integer values. The first value represents the membership function type (triangular or trapezoidal) and others (three or four values) represent the coefficients of the membership function. These sets of values are stored in database as fuzzy sets (rules) parameters.

Algorithm (3.2) shows the implemented steps for determining the optimal values of the membership function. The input to this algorithm is the histogram data (which contains the histogram vectors). In this algorithm each vector is checked to know whether it could be better represented in terms of triangular function or trapezoidal function. This is done by checking the sum of absolute errors for both types, individually. The input vector is classified to either type according to location of lowest founded sum of absolution errors. Then, the result (i.e., the type of membership function, and its coefficients) is saved as a fuzzy set in file.

Algorithm (3.2) The steps of finding the best fuzzy logic representation.

Algorithm (3.2) Fuzzy Logic

Input: *His_FD // is the histogram files of fractal dimension.*
L_{max} // is the maximums box size
L_{stp} // is the step size for increasing box size
Output : *FFL // is the file contains the coefficient of the best found membership function.*

```
Procedure:
        N=Lmax / Lstp
        loop for J =1 to 6
              loop for I = 0 to 100
                  set PDF (I) ← His_FD(I,N)
             nextloop I
             FitTriangularMemberShip ErrorTrng, P1Trng, P2Trng, P3Trng, PDF
             FitTrapozoidalMemberShip ErrorTrpz, P1Trpz, P2Trpz, P3Trpz, PDF, P4Trpz
             if ErrorTrng < ErrorTrpz Then
                  printin file "1", P1Trng, P2Trng, P3Trng
             else
                  printin file "2", P1Trpz, P2Trpz, P3Trpz, P4Trpz
             end if
      nextloop J
end.
```

B.1 Triangular Membership Function

It is one of the simplest types of functions used to represent the membership degree of a feature to certain class. Figure (3.2) illustrates the triangular membership function (MSF).

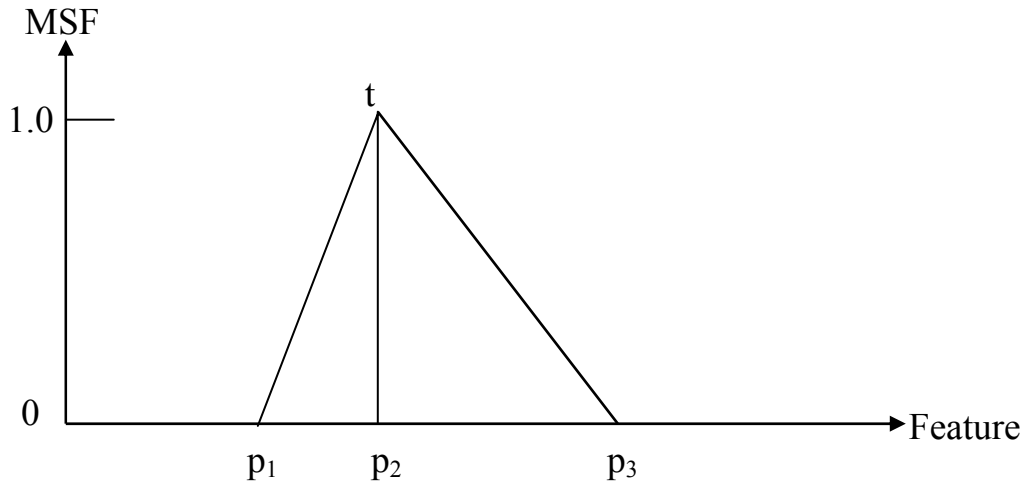


Figure (3.2) Triangular membership function

In the proposed system, each breast tissue class is represented by N histograms, each histogram corresponds to one of fractal features that determined using certain box size. In our work, the number of taken box sizes is 6 (i.e., L_{max} = 9, 17, 25, 33, 41, 49), which means that N=6. To find the best triangular representation the optimal values of p_1, p_2 and p_3 need to be found. For determining the values of p_1, p_2, p_3 the exhaustive search based method was adopted, the search criteria is finding the set of (p_1, p_2, p_3) that lead to lowest possible sum of absolute error between the membership function values and the corresponding normalized histogram values of the fractal dimension. This best found set of (p_1, p_2, p_3) values and the corresponding error are given as the output of the exhaustive search (or fitting) algorithm.

To speed up the involved computations of the exhaustive search algorithm for each possible combination of (p_1, p_2, p_3) the following steps were taken:

1. The accepted values of p_1, p_2, p_3 should satisfy the following condition: $0 \le p_1 < p_2 < p_3 \le N_{bins}$, where N_{bin} is the number of quantization bins of the extracted fractal dimension.
2. For all feature values that are greater than p_3 or less than p_1 the corresponding membership value are equal to zero, so the sum of errors at this range of values is equal to:

$$e^2 = \sum_{i=0}^{p_1} His_n(i) + \sum_{p_3}^{N_{bin}} His_n(i) \qquad \text{,……..(3.5)}$$

Where $His_n(i)$ is the normalized histogram which is determined using the following equation:

$$His_n(i) = \frac{His(i)}{\max_{i=1,2,..N_{bin}}(His(i))} \quad 0 \le i \le N_{bin} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.6)$$

3. But, when the feature value *(f)* lay within the interval $[p_1+1,p_2]$ the membership function value is given by the following line $\overline{p_1 t}$ whose equation is:

$$M_1(f) = \frac{f - p_1}{p_2 - p_1} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.7)$$

and the corresponding error term is (e_1), where

$$e_1^2(f) = [His_n(f) - M_1(f)]^2 \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.8)$$

4. In similar way, for the feature values *(f)* lay within the interval $[p_2+1,p_3-1]$, the corresponding membership function is given by the following line $\overline{tp_3}$ whose equation is:

$$M_2(f) = \frac{p_3 - f}{p_3 - p_2} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.9)$$

and, the corresponding error term for each feature value *(f)* within the interval $[p_2,p_3]$ is:

$$e_2^2 = (His_n(f) - M_2(f))^2 \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.10)$$

5. The total sum of errors for specific set of (p_1, p_2, p_3) values is:

$$\chi^2 = e^2 + \sum_{f=p_1+1}^{p_2} e_1^2(f) + \sum_{f=p_2+1}^{p_3-1} e_2^2(f) \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.11)$$

Algorithm (3.3) shows the implemented steps to compute the optimal set of the coefficients of triangular membership function for a specific normalized histogram vector of the extracted fractal dimension.

Algorithm (3.3) Fit Triangle Membership Function

Input: *His* // *is the Histogram vector*
N_{bins} // *is the number of quantization bins*
Output : *P1,P2,P3,* // *is the rule of triangle membership*
sum_error // *is the summation of error*

Procedure:

```
'Find Max(His()), and normalize His()
set Max ← His(0)
  loop for I = 1 to Nbins
    if Max < His(I) then set  Max ← His(I)
  nextloop I
  loop for I = 0 to Nbins
    set A(I) ← His(I) / Max
  nextloop I

'Find the triangle points
set Sum ←  9.9E+19  'as a very large value
loop for P1 = 0 to Nbins-2
  set Sum1 ←  0
  loop for I = 0 to P1
    set Sum1 ←  Sum1 + A(I) × A(I)
  nextloop I

  loop  for P2 = P1 + 1 to Nbins -1
    set  Sum2 ←  Sum1
    set One ← 1 / (P2 - P1)
    loop for I = P1 + 1 to P2
      set  Two ← A(I) - One × (I - P1)
      set Sum2 ←  Sum2 + Two × Two
    nextloop I
    loop  for P3 = P2 + 1 to Nbins
      set Sum3 ←  Sum2
      set  One ←  1 / (P3 - P2)
      loop for I = P2 + 1 to P3
        set Two  ←  A(I) - One × (P3 - I)
```

```
                    set Sum3 ← Sum3 + Two × Two
                nextloop I
                loop for I = P3 + 1 to N_bins
                   set Sum3 ← Sum3 + A(I) ×A(I)
                nextloop I
                if Sum > Sum3 then
                    set P1opt ← P1
                    set P2opt ← P2
                    set P3opt ← P3
                    set Sum ← Sum3
                endif
            nextloop P3
        nextloop P2
    nextloop P1
erase A
end
```

B.2 Trapezoidal Membership Function

It is one of the functions used to define the degree of membershipness of the input feature value to certain class. Figure (3.3) shows the trapezoidal membership function (MSF).

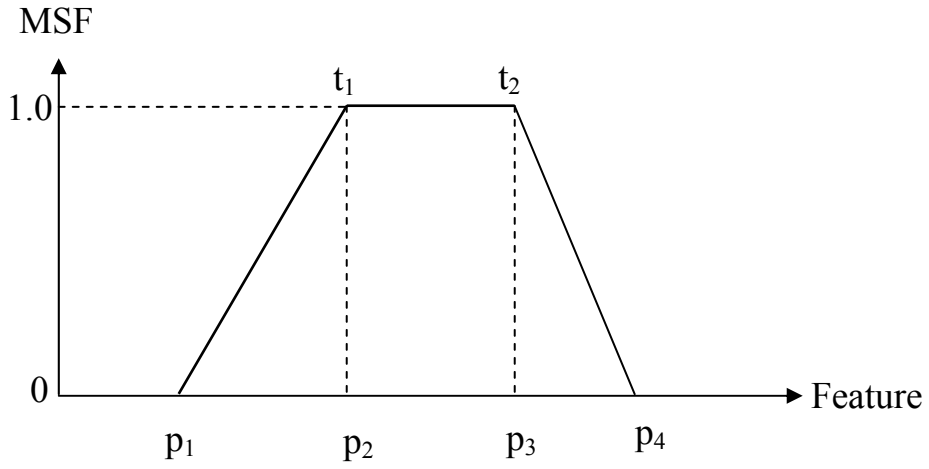


Figure (3.3) Trapezoidal membership function

To define the suitable trapezoidal membership function the values of its parameters (p_1, p_2, p_3, p_4) have to be found, they required to represent the involved rules of each trapezoidal membership function. A similar algorithm to that followed with triangular membership function is adopted to handle the trapezoidal case. It is also based on exhaustive search method, where all possible values of p_1, p_2, p_3, p_4 have been considered to find out the best set of p's values that lead to lowest possible error between the actual values of the normalized histogram and the corresponding values determined from the trapezoidal function.

To compute the best set of p's values the following remarks have been taken in consideration:

1. The coefficients $\{p_1, p_2, p_3, p_4\}$ should satisfy the condition: $0 \le p_1 < p_2 < p_3 \le N_{bins}$.
2. For the feature values *(f)* lay within the intervals $[0, p_1]$ and $[p_4, N_{bins}]$ the error between trapezoidal function and the actual normalized histogram is given by the following expression:

$$e^2 = \sum_{f=0}^{p_1} His_n^2(f) + \sum_{p_4}^{100} His_n^2(f) \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.12)$$

3. But, when the feature value is within the interval $p_1 < f < p_2$ the membership function is given by the line $\overline{p_1 t_1}$, which is described by the following equation:

$$M_1(f) = \frac{f - p_1}{p_2 - p_1} \qquad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.13)$$

and the contribution term to the total error is:

$$e_1^2(f) = [His_n(f) - M_1(f)]^2 \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.14)$$

4. If the feature value lay within the interval $p_2 \leq f \leq p_3$ then the value of the membership function is equal to one, and the corresponding error contribution is given as:

$$e_2^2(f) = (His_n(f) - 1)^2 \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.15)$$

5. If the value of the feature is within the interval $p_3 < f < p_4$ then the membership function is described by the line $\overline{t_2 p_4}$, which is given by the following equation:

$$M_2(f) = \frac{p_4 - f}{p_4 - p_3} \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.16)$$

and the corresponding error term is:

$$e_3^2(f) = (His_n(f) - M_2(f))^2 \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.17)$$

7. So, the total error is determined using the following expression:

$$e_{Tot} = e^2 + \sum_{f=p_1+1}^{p_2-1} e_1^2(f) + \sum_{f=p_2}^{p_3} e_2^2(f) + \sum_{f=p_3+1}^{p_4-1} e_3^2(f) \quad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.18)$$

Algorithm (3.4) shows the implemented steps to find the best trapezoidal membership function that represent the normalized histogram of the extracted features vector.

Algorithm (3.4) Fit Trapezoidal Membership Function

Input: *His* // *is the histogram vector*
N_{bins} // *is the number of quantization bins*
Output: *P1, P2, P3, P3* // *is the rule of triangle membership*
Error_sum // *is the summation of error*

Procedure:

```
'Find Max(His()), and normalize His()
set Max ← His (0)
loop for I = 1 to Nbins
    if Max < His (I) then
       set Max ← His (I)
nextloop I
loop for I = 0 to Nbins
    set A (I) ← His (I) / Max
nextloop I
"Find the trapezoidal points
 set Sum ← 9.9E+19
loop for P1 = 0 to Nbins -3
    set Sum1 ← 0
    loop for I = 0 to P1
        set Sum1 ← Sum1 + A (I) × A (I)
    nextloop I

    loop for P2 = P1 + 1 to Nbins -2
       set Sum2 ← Sum1
       set One ← 1 / (P2 - P1)
       loop for I = P1 + 1 to P2
           set Two ← A (I) – One × (I - G1)
           set Sum2 ← Sum2 + Two × Two
       nextloop I

       loop for P3 = P2 + 1 to Nbins -1
            set Sum3 ← Sum2
            loop for I = P2 + 1 to P3
                set Two ← 1 - A (I)
                set Sum3 ← Sum3 + Two × Two
            nextloop I
           loop for P4 = P3 + 1 to Nbins
                 set Sum4 ← Sum3
                set One ← 1 / (P4 - P3)
                loop for I = P3 + 1 to P4
                      set Two ← A (I) – One × P4 - I)
```

```
                    set Sum3 ← Sum3 + Two × Two
                nextloop I
                loop for I = P4 + 1 to N_bins
                    set Sum3 ← Sum3 + A (I) × A (I)
                nextloop I
                if Sum > Sum then
                    set P1opt ← P1
                    set P2opt ← P2
                    set P3opt ← P3
                    set P4opt ← P4
                    set Sum ← Sum3
                endif
            nextloop P4
        nextloop P3
    nextloop P2
nextloop P1
erase A
end
```

3.2.3 Recognition Using Fuzzy Logic

The aim of recognition stage is to identify the class index of an input image (i.e., unknown sample image of a breast tissue), under the condition that the class of the input image should be one of classes that the recognition system had previously trained on them. "Fuzzy Logic" had been used to find out the class index of the feature vector extracted from the unknown image sample by applying the fuzzy rules which already have been established in the enrollment stage. The index of the class that scored the highest sum of individual membership values is considered as the class index of the unknown tested sample.

Algorithm (3.5) was implemented to identify the class index of an input sample image, taking into consideration that its class should be one of the classes which have been already introduced to the system during the enrollment

stage. The input to this algorithm is the fuzzy data (including type of membership functions and their rules) and the tested sample image whose class should be identified. The first step in this algorithm is using box-counting on the sample image to calculate its fractal dimension. The sum of membership function values of the extracted features is determined for all considered classes of breast tissues. Then, the collected highest sum is allocated, and its class index is considered as the class index of the tested sample. Algorithm (3.5) illustrates the implemented steps of the classification process. Algorithm (3.6), used in algorithm (3.5), illustrates the steps of calculating the value of triangular membership function. Algorithm (3.7), used in Algorithm (3.5), illustrates the steps taken for calculating the value of trapezoidal membership function.

Algorithm 3.5 // Recognition of unknown sample image

Input: *FFL // the file of fuzzy data*
L_{max} *// is the maximums box size*
L_{stp} *// is the step size for increasing box size*
// the input sample image
Output: *IndexPos // is the class index*

Procedure:

```
set iLmax ← Lmax / Lstp
loadBitmapFile
   loop for I = 1 to NoClasses
        set SumMem(I) ← 0
   nextloop I
   loop for Y = Lmax to Hgt - 1 - Lmax
      loop for X = Lmax to Wid - 1 - Lmax
        Call BoxCounting Lmax, Lstp, X, Y, F
        loop for I = 1 to NoClasses
          loop for iLmax1 =1 to iLmax
            if Typ(I, iLmax1) = 1 then
               set Memb ← GetTrng(F, Gf(I, iLmax1, 1), Gf(I, iLmax1, 2),
                     Gf(I,iLmax1,3))
```

```
                    else
                        set Memb ← GetTrpz(F, Gf(I, iL_max1, 1), Gf(I, iL_max1, 2),
                                Gf(I, iL_max1, 3), Gf(I, iL_max1, 4))
                    endif
                    set SumMem(I) ← SumMem(I) + Memb
                nextloop iL_max1
            nextloop I
        nextloop X
    nextloop Y
    set Max ← SumMem(1)
    set IndexPos ← 1
    loop for I = 2 to NoClasses
        if Max < SumMem(I) then
            set Max ← S umMem(I)
            set IndexPos ← I
        endif
    nextloop I
  output IndexPos  // as the class index
end.
```

Algorithm 3.6: Get Triangle Membership Function

```
Input:  X                          //  is the feature value
        P1Trng, P2Trng, P3Trng     // is the rule of the MSF
Output: Meb                        //  is the membership function value
```

```
Procedure:
        if (X < P1Trng) or (X > P3Trng) then
            set Meb ← 0
        else
        if (X < P2Trng) then
            set Meb ← (X - P1Trng) / (P2Trng - P1Trng)
        else
            set Meb ← (P3Trng - X) / (P3Trng - P2Trng)
        end If
        set GetTrng ←  Meb
end.
```

Algorithm 3.7 // Get Trapezoidal Membership Function

Input: *X* // *is the feature value*
P1Trpz, P2Trpz, P3Trpz, P4Trpz // *is the rule of the MSF*
Output: *Meb* // *is the membership function value*

Procedure:

```
if (X < P1Trpz) or (X > P4Trpz) then
   set Meb ← 0
else
if X <= P2Trpz then
    set Meb ← (X - P1Trpz) / (P2Trpz - P1Trpz)
else
if X <= P3Trpz Then
    set Meb ← 1
else
     set Meb ← (P4Trpz - X) / (P4Trpz - P3Trpz)
endif
set GetTrpz ← Meb
end.
```

3.3 Breast Tissue Recognition Based on Neural Network

Figure (3.4) shows the layout of the second proposed system. This system consists of two phases:

Enrollment Phase: the input to this phase is the breast tissue images. The first stage in this phase is the preprocessing stage, it includes: (1) reading color image data, and (2) conversion of color image to gray image. The second stage is the feature extraction stage, it includes two parts (1) application of box counting method to determine the multi-fractal dimensions, and (2) applying neural network backpropagation training algorithm. The result of this module is a database contains: (1) the multi-fractal dimensions

histograms (2) the values of the nodes' weights of the trained neural network.

Classification Phase: in this phase the input is an unknown color breast tissue image. The same preprocessing operations that performed in enrollment stage are applied here. The next stage in this phase is the calculation of multi-fractal dimensions of the unknown input image. The third stage is the identification stage, which is performed to assign to class index of the tested image from the neural network output nodes after passing the extracted feature vector through the trained network.

3.3.1 Preprocessing Stage

In this stage the input is an unknown color breast tissue image. The same preprocessing operations which performed in the first system are implemented here. Details about this preprocessing stage are given in section (3.2.1).

3.3.2 Feature Extraction

After performing the preprocessing stage, the feature extraction stage begins. In this stage the feature vector (i.e., histograms of multi-fractal dimensions) is determine.

The BCM was used to determine the fractal dimensions. The same inputs, outputs and operations that performed in the first proposed system are implemented here. Details about the calculation of multi-fractal dimension using BCM are given in section (3.2.2.A).

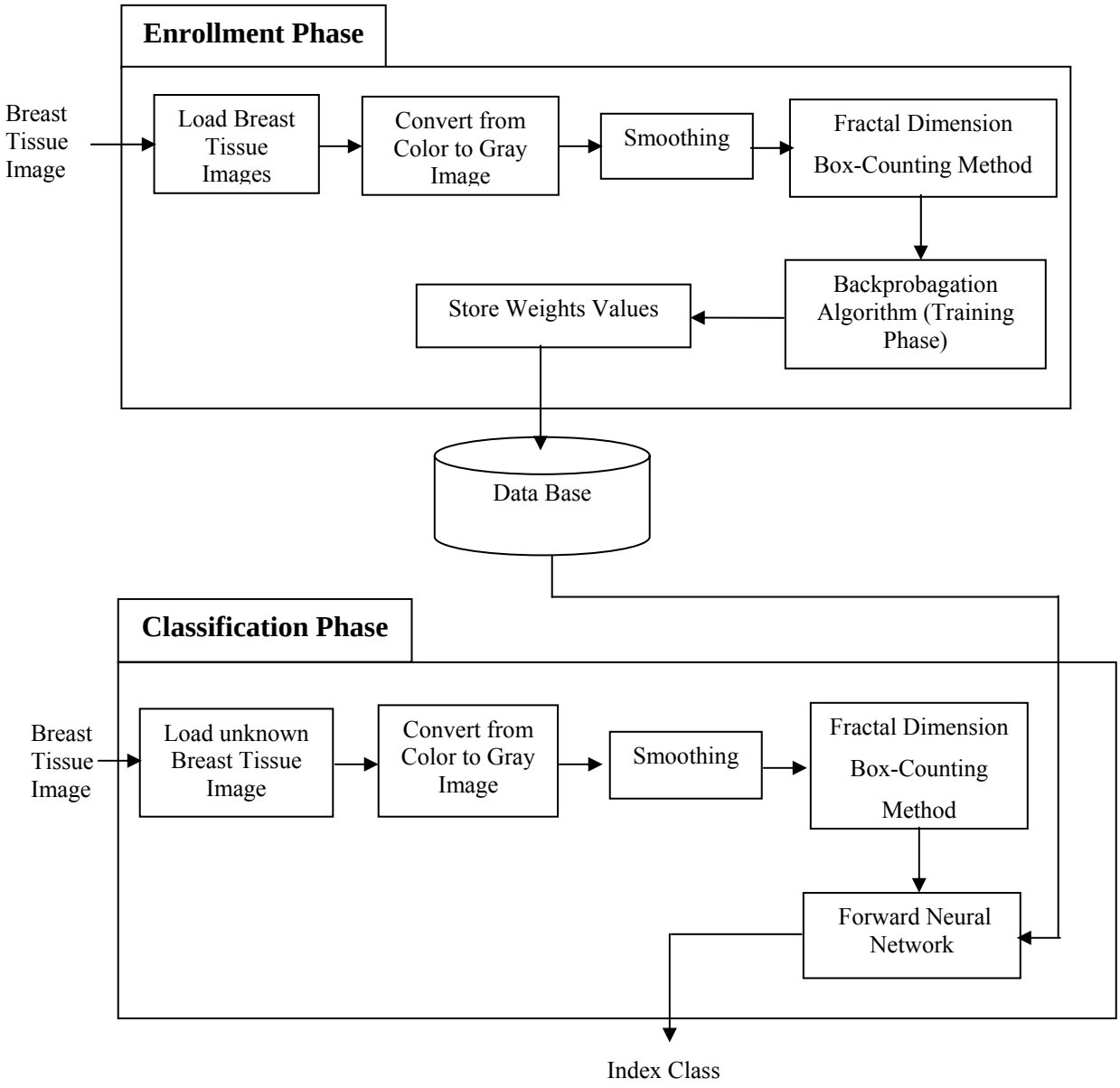


Figure (3.4) The Block Diagram of Breast Tissue Recognition System Based on Neural Network

3.3.3 Preparation of Artificial Neural Network

In this proposed system three layers feed forward neural network architecture had been adopted. In such kind of neural network there is one

input layer, one output layer, and one hidden layer. In the first step the network architecture is defined by assigning its parameters (i.e., the number of input nodes, the number of output nodes, and the number of hidden nodes).

1. Number of Input Nodes: the number of input nodes are equal to number of quantization bins of the multi fractal dimension, this parameter represents the length of each histogram vector, taking into consideration that these histogram vectors are fed as input to neural network.

2. Number of Output Nodes: the output layer consists of a number of nodes, the output of each node represents one digit in the binary output number. The binary rounded values of these nodes are combined to produce one output integer number which represents the winner class index.

3. Number of Hidden Nodes: Initially, it is consist of a specific number of nodes. Different methods have been used to determine the proper number of hidden neurons in feed forward network (like, trial-and-error, evolutionary algorithms and simulated annealing). Since the main focus here is on investigating the strength and limitations of using different types of input approaches with the back propagation algorithm rather than testing a constructive or stochastic search algorithm, the trial-and error method has been followed to determine the number of hidden neurons.

It is difficult to give a formula that can precisely determine the number of hidden nodes; however a little possible number of neurons was

presumed. There are two main reasons for this: the first is concerned with the computation time; because more hidden neurons in the network needs longer time in network training phase. The second reason is concerned with the possibility of overfitting, where using too many hidden neurons results in high accuracy on the training set but a high error rate on the test set.

4. Learning Rate: it is an important training parameter since it controls two conflicted requirements: the fast convergence and stable weights estimations.

Algorithm (3.8) illustrates the implemented steps for the creation of a forward feedback neural net. The inputs to this algorithm are: (1) the learning rate, and (2) array of layers. In this algorithm the structure of neural network will built (after assigning the number of layers and its nodes). Bias and weights values are initialized to have random values restricted to be within the range [-1,1], the GetRand function have been used to gave this random values. The result of this algorithm is a binary integer number: either (0) which means unsuccessful creation of the network, or (1) which means successful creation of the network.

Algorithm (3.8) Create a forward feedback neural net

Input: *LearningRate // is the learning rate*
ArrayOfLaye // is the array of layers
Output: *0 // Unsuccesful*
1 // Successful

```
Procedure:
        set Network.LayerCount ← ubound(ArrayOfLayers)
        if Network.LayerCount < 2 then
            set CreateNet ← 0     'Unsuccessful
                exit function
        endif
        set  Network.LearningRate ← LearningRate
        loop  for I = 1 to ubound(ArrayOfLayers)
            set Network.Layers(I).NeuronCount ← ArrayOfLayers(I)
            loop for J = 1 to ArrayOfLayers(I)
                  if I = ubound(ArrayOfLayers) then
                  set Network.Layers(I).Neurons(J).Bias ← GetRand
                  set Network.Layers(I).Neurons(J).DendriteCount ← ArrayOfLayers(I -
                       1)
                  loop for K = 1 to ArrayOfLayers(I - 1)
                       set Network.Layers(I).Neurons(J).Dendrites(K).Weight
                            ← GetRand
                 nextloop K
                endif
                 if I<>1 then
                   set Network.Layers(I).Neurons(J).Bias ← GetRand
                   set Network.Layers(I).Neurons(J).DendriteCount ←
                        ArrayOfLayers(I - 1)
                   loop for K = 1 to ArrayOfLayers(I - 1)
                        set Network.Layers(I).Neurons(J).Dendrites(K).
                            Weight ← GetRand
                   nextloop K
                 endif
            nextloop J
        nextloop I
        set CreateNet ← 1     'successful
```

After the successful creation of the network, and assigning the values of its parameters it becomes ready to accept any input vector. Algorithm (3.11) illustrates the steps taken of run the designed ANN for specific input. The inputs to this algorithm is an array of histogram vectors (which represent the extracted features from specific sample). In our adopted system the sigmoid activation function was used.

Algorithm (3.9) Run the designed ANN for specific input

Input: *ArrayOfInputs // is the array of specific input*
Output: *OutputResult // is the array of output*

```
Procedure:
    loop for I = 1 to Network.LayerCount
        loop for J = 1 to Network.Layers(I).NeuronCount
            if I = 1 then
                set Network.Layers(I).Neurons(J).Value ← ArrayOfInputs(J)
            else
                set Network.Layers(I).Neurons(J).Value ← 0
                loop for K = 1 to Network.Layers(I - 1).NeuronCount
                    set Network.Layers(I).Neurons(J).Value ← _
                        Network.Layers(I).Neurons(J).Value + _
                        Network.Layers(I - 1).Neurons(K).Value ×_
                        Network.Layers(I).Neurons(J).Dendrites(K).Weight
                nextloop K
                set Network.Layers(I).Neurons(J).Value ← _
                    Activation(Network.Layers(I).Neurons(J).Value + _
                    Network.Layers(I).Neurons(J).Bias)
            endif
        nextloop J
    nextloop I
    loop for I = 1 to (Network.Layers(Network.LayerCount)
                        .NeuronCount)
        set OutputResult(I) ← (Network.Layers(Network.LayerCount)
                                .Neurons(I).Value)
    nextloop I
    set RunANN ← OutputResult
endfunction
```

3.3.4 The Operation Phases of ANN

The proposed neural-based breast tissue recognition system have two operational phases:

1. Training Phase

The main goal of network training is to adjust the weight and bias value of each neural network neurons such that the network should be capable to accurately recognition different classes of samples. The complete steps of backprppagation training algorithm are illustrated in chapter two.

The training process begins with the calculation of the required number of output nodes, which depends on the number of classes, more precisely it is equal to:

$$N_{output} = \lceil \log_2 (N_c) \rceil \qquad ,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.22)$$

Where, N_c is the number of classes, which should be recognized by the system.

N_{output} is the number of output nodes.

As a second step a neural network that have the required architecture is created. After creation of the network, the network training loop is started, at each loop (or iteration) the input vectors are passed through the input layer, and the output of this layer pass through the hidden layer toward output layer, then the sum of differences between the desired output and the actual output, at the output layer nodes, is addressed to re-adjust the weights of the network nodes. At the end of each loop the stopping conditions are checked to decide whether there is need to continue training with a new iteration or to stop training loop. In our proposed system the training process would be stopped if the error become smaller than or equal to a predefined value of permissible error level, or the number of iterations passes the predefined maximum number of allowed iterations. After the completion of training phase, the corrected weights are stored in an array of weights to be used as a knowledge in testing phase. A

number (N_s) of samples are taken from each class of breast tissues as training data representing that class. Also, other set, consist of M_s samples, is taken to be the test material in the testing phase. Algorithm (3.10) illustrates the implemented steps of training process using backpropagation algorithm.

Algorithm (3.10) Training processes

Input: *different files of fd (samples for each files)*
Output: *final weights*

Procedure:

```
set NoBinsP ← tNoBins + 1
if Flgtrain then
    set NoOutputNodes ← 1
    set N ← 2
    while N < tNoClasses
        set N ← N ×2
        set NoOutputNodes ← NoOutputNodes + 1
    wend
    call CreateNet LearningRate, Array(NoBinsP, NoHiddenNodes, NoOutputNodes)
    set mErr ← 9.8E9
    loop for Iter = 1 to NoTrainIter
        loop for I = 1 to tNoClasses
            loop for K = 1 to NoOutputNodes
                select case K
                    case 1: if (I and 1) = 0 then oDat(1) = 0 else oDat(1) = 1
                    case 2: if (I and 2) = 0 then oDat(2) = 0 else oDat(2) = 1
                    case 3: if (I and 4) = 0 then oDat(3) = 0 else oDat(3) = 1
                    case 4: if (I and 8) = 0 then oDat(4) = 0 else oDat(4) = 1
                    case 5: if (I and 16) = 0 then oDat(5) = 0 else oDat(5) = 1
                endselect
            nextloop K
            loop for J = 1 to tNoPatternsPerClass
                loop for K = 1 to NoBinsP
                    set iDat(K) ← FDtrain(I, J, K)
                nextloop K
                call TrainANN iDat, oDat
            nextloop J
        nextloop I
        set Err ← 0
```

```
            loop for I = 1 to tNoClasses
             loop for J = 1 to NoOutputNodes
                select case J
                  case 1: if (I and 1) = 0 then oDat(1) = 0 else oDat(1) = 1
                  case 2: if (I and 2) = 0 then oDat(2) = 0 else oDat(2) = 1
                  case 3: if (I and 4) = 0 then oDat(3) = 0 else oDat(3) = 1
                  case 4: if (I and 8) = 0 then oDat(4) = 0 else oDat(4) = 1
                  case 5: if (I and 16) = 0 then oDat(5) = 0 else oDat(5) = 1
                endselect
             nextloop J
             loop for J = 1 to tNoPatternsPerClass
                loop  for K = 1 to NoBinsP
                   set iDat(K) ← FDtrain(I, J, K)
                nextloop K
                set X ←  Call RunANN(iDat)

                loop for K = 1 to NoOutputNodes
                   set Ee ←  oDat(K) - X(K)
                   set Err ←  Err + Ee ×Ee
                nextloop K
             nextloop J
          nextloop I
          set Err ←  Err / tNoClasses / tNoPatternsPerClass
          if Err < mErr then
             set aIter ←  Iter
             set mErr ←  Err
             set OptNetwork ←  Network
             if Err <= MinErr then
                set Iter ← NoTrainIter
          endif
    nextloop Iter
endif
endsub
```

2. Testing Phase

Testing phase is the second operational phase of the proposed neural network based system for breast tissue recognition. The testing procedure starts by taking the training samples (N_s) and testing samples (M_s) for each class and

pass each one of them to calculate the neural output (i.e., the assessed class index of the input vector). Then, this output is compared with actual class index of the input to determine the identification accuracy of the system.

Algorithm (3.11) illustrates the steps taken to conduct testing of the established network using the set of training data.

Algorithm (3.11) Test Training Data

Input: *set of training sample from each file of fd for each class*
Output: *succ // is the success rate of identification*

```
Procedure:
    if Flgtrain and Flgtrained and (NoClasses = tNoClasses) and (NoBins = tNoBins)
       then
          set Nn ← CLng(tNoClasses) ×tNoPatternsPerClass - 1
    set Suc ← 0
    set Nn ← -1
    set NoBinsP ← tNoBins + 1
    loop for I = 1 to tNoClasses
       loop for J = 1 to tNoPatternsPerClass
          loop for K = 1 to NoBinsP
             set iDat(K) ← FDtrain(I, J, K)
          nextloop K
          set X ← call RunANN(iDat)
          set M ← 1
          set N ← 0
          loop for K = 1 to NoOutputNodes
             if X(K) >= 0.5 then
                set N ← N + M
             endif
             set M ← M ×2
          nextloop K
          if N = I then
             set Suc ← Suc + 1
             set Nn ← Nn + 1
       nextloop J
    nextloop I
  endif
endsub
```

Algorithm (3.12) shows the implemented steps for testing the established network using the set of test data.

Algorithm (3.12) Test Testing Data

Input: *set of testing samples from each file of fd for each class*
Output: *succ // is the success rate of identification*

```
Procedure:
    if Flgtest and Flgtrained and (NoClasses = sNoClasses) and (NoBins = sNoBins)
        then
            set Nn ← CLng(sNoClasses) ×sNoPatternsPerClass - 1
    set Suc ← 0
    set Nn ← -1
    set NoBinsP ← sNoBins + 1
    loop for I = 1 to sNoClasses
      loop for J = 1 to sNoPatternsPerClass
        loop for K = 1 to NoBinsP
          set iDat(K) ← FDtest(I, J, K)
        nextloop K
        set X ← call RunANN(iDat)
        set M ← 1
        set N ← 0
        loop for K = 1 to NoOutputNodes
          if X(K) >= 0.5 then
            set N ← N + M
          endif
         set M ← M ×2
        nextloop K
        if N = I then
          set Suc ← Suc + 1
          set Nn ← Nn + 1
      nextloop J
    nextloop I
   endif
endsub
```

Chapter Four

Experimental and Result

Chapter Four
Experimental Result

4.1 Introduction

This chapter is dedicated to demonstrate the results of the conducted tests to assess the performance of the two established breast cancer recognition systems. At first, the test material of breast tissues images is described, and then the results of the tests performed on the first recognition system (which uses fuzzy logic) are given. Finally, the test results of the second system (which uses artificial neural network) are presented. The tests have been conducted to investigate the output of the three stages of the two recognition system, which are: preprocessing stage, feature extraction stage and recognition stage.

4.2 Test Material

In this study, 24 type of breast tissues images, were taken as test material. These images are taken from “The Color Atlas of Tumor Histopathology”. One of these images represents the normal (i.e., non-infected breast tissue, while the other 23 tissue images present various types of tumor. The images files have bitmap format, with resolution 24 bit/pixel, and the size of each image is 890x560 pixels. Figure (4.1) represent breast tissues images were image from C1 to C23 represent various classes of breast tumor, while image C24 represent normal case (non-infected breast tissue).

4.3 Results of Preprocessing Feature Extraction Stages

The input image was a color image, the first step is the conversion of the color image to gray image and then, using the average filter to smooth the produced gray image. Figure (4.2) shows a sample image after its conversion from color to gray representation, and its appearance after smoothing its pixels values.

The next stage after preprocessing stage is "feature extraction", it implies the use of box counting method (BCM) to extract the multi-fractal dimension distribution from the tested image. In BCM the value of box size has critical importance on the results. On one hand, it should be reasonably big to get an accurate estimation of the local texture characteristics, and to suppress region border effects. On the other hand, the box size should be small in order to capture the local texture characteristics. In this research a range started from (9×9 with L_{max} equal to 4) to (49×49 with L_{max} equal to 24) pixels for the box size was tested.

After the calculation of multi-fractal dimension for each pixel in the image, its probability density function (PDF) is determined to represent the statistical behavior of the extracted fractal dimension values. Figure (4.3) shows the probability density function (PDF) for the determined fractal dimensions of the tested breast tissue images and for all considered box sizes. From these figures it was noticed that:

1. When L_{max} value is taken small the shape of PDF is not smooth.
2. When L_{max} value is taken high the shape of PDF shifts toward high fractal dimension values, and become smother.
3. When L_{max} is given medium values, the shape of PDF is smooth and nearly distributed over all the dynamic range of fractal dimensions.

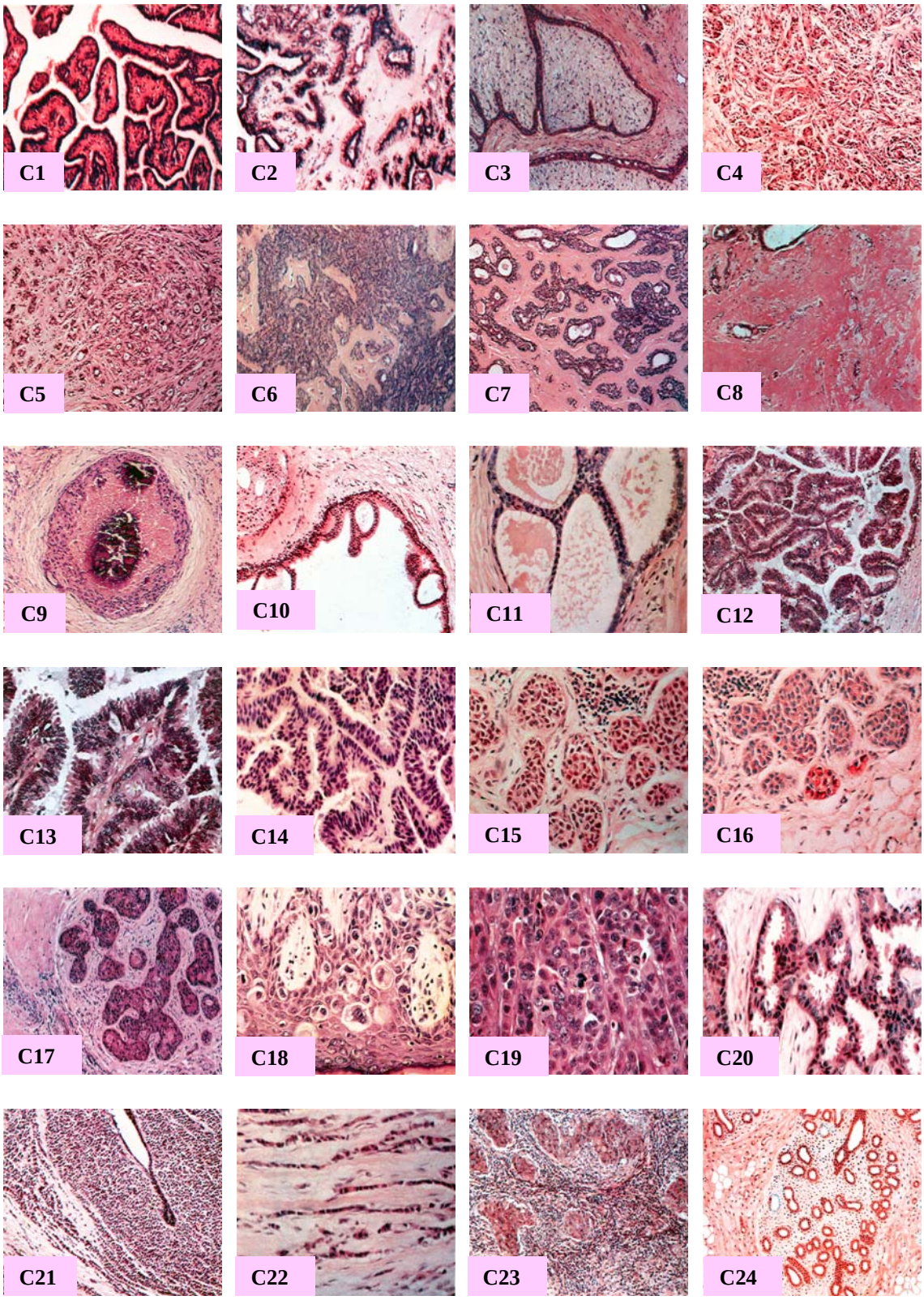
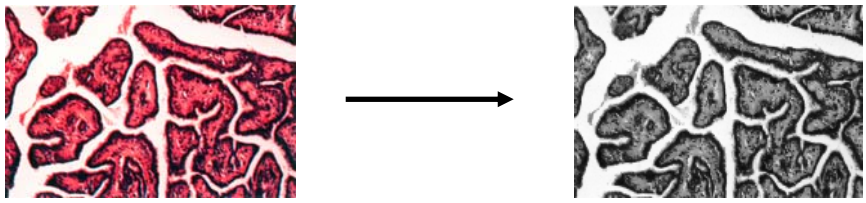


Figure (4.1) Samples of the tested breast tissues images



(a) (b)

Figure (4.2) The conversion of the color image to gray image,
(a) color image, (b) smooth gray image

4.4 Results of Recognition Tests

There are two sets of tests results, each set belong to one of the decision machines (either fuzzy logic or ANN):

1. **Recognition Decision Based on Fuzzy Logic:** After the extraction of probability density function of fractal dimension, the fuzzy logic stage begins; the first step in this stage is building the membership functions. Table (4.1) lists the parameters values of the determined membership functions (i.e., their types and coefficients values). From the results listed in table (4.1), the mostly suitable type of MSF is triangle.

   The last step in fuzzy based recognition system is testing the accuracy of recognition process. In this set of tests, five samples have been extracted to represent each breast tissue class, the number of bins was varied, and the sample size (i.e., height and width) was also varied. Taking into consideration that the whole size of each breast tissue image is (890×560). Table (4.2) shows the recognition results when the number of bins is taken 100, 75 and 50.

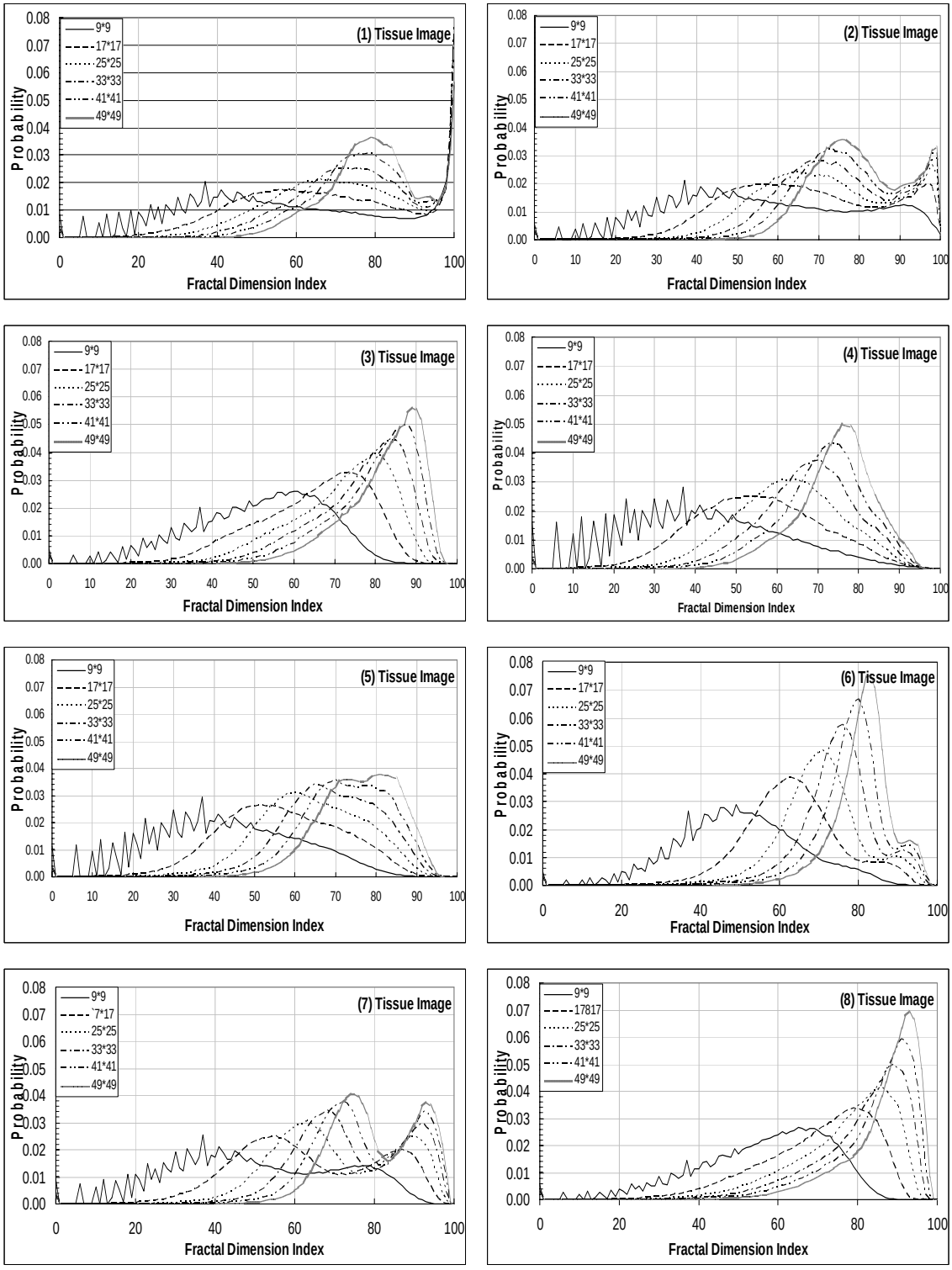
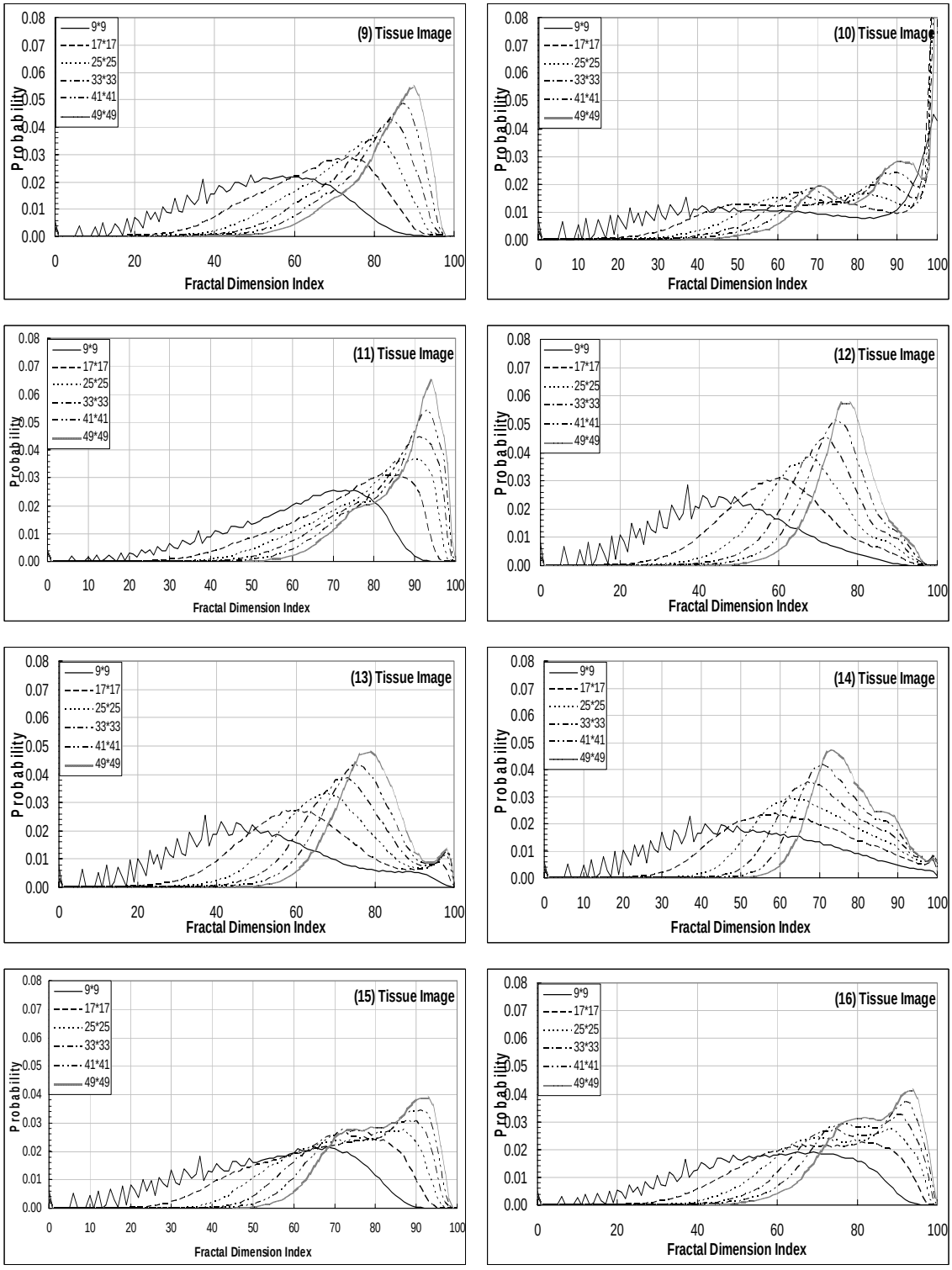
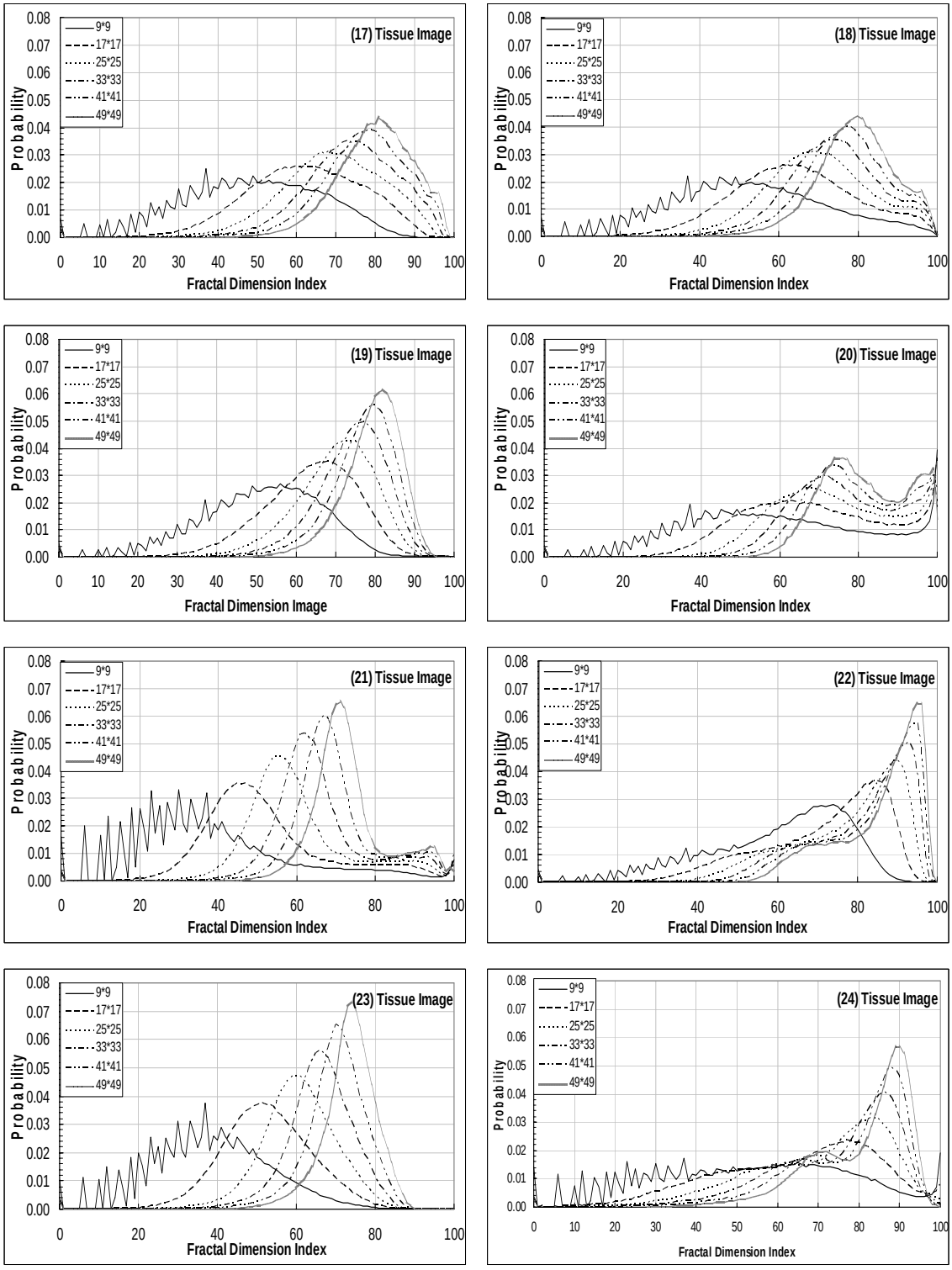


Figure (4.3) The probability density function for the multi-fractal dimensions



The continue of Figure (4.3)



The continue of Figure (4.3)

Table (4.1) Type and parameters of the membership functions (in type field the symbol 1 means triangle, and 2 means trapezoidal)

Class No.	Vector No.	MSF Type	Rules			
			P_1	P_2	P_3	P_4
1	1	1	96	99	100	
	2	1	97	99	100	
	3	1	98	99	100	
	4	1	67	72	78	
	5	1	64	76	89	
	6	1	65	76	100	
2	1	1	13	44	100	
	2	2	29	53	99	100
	3	1	22	99	100	
	4	1	45	72	100	
	5	2	52	72	99	100
	6	1	54	77	100	
3	1	1	8	64	85	
	2	1	29	29	90	
	3	1	41	83	93	
	4	1	50	86	95	
	5	1	57	89	96	
	6	1	62	91	97	
4	1	1	4	37	81	
	2	1	19	52	95	
	3	1	31	63	94	
	4	1	41	69	93	
	5	1	48	74	93	
	6	1	54	77	93	
5	1	1	9	37	83	
	2	1	24	52	96	
	3	1	36	61	97	
	4	1	43	68	98	
	5	1	46	75	98	
	6	1	49	81	97	
6	1	1	19	47	87	
	2	1	40	61	90	
	3	1	51	70	90	
	4	1	59	76	90	
	5	1	64	80	91	
	6	1	69	83	92	
7	1	1	13	37	100	
	2	1	29	53	100	
	3	1	41	61	100	
	4	1	49	67	100	
	5	1	42	95	99	
	6	1	48	95	99	
8	1	1	14	71	88	
	2	1	36	83	94	
	3	1	49	89	96	
	4	1	59	92	98	
	5	1	67	94	98	
	6	1	73	95	99	
9	1	1	5	60	89	
	2	1	26	75	92	
	3	1	40	82	94	
	4	1	50	86	96	
	5	1	57	89	97	
	6	1	62	91	97	
10	1	1	90	99	100	
	2	1	96	99	100	
	3	1	97	99	100	
	4	1	98	99	100	
	5	1	98	99	100	
	6	1	98	99	100	
11	1	1	16	76	92	
	2	1	34	89	96	
	3	1	46	93	98	
	4	1	55	95	99	
	5	1	62	96	99	
	6	1	69	96	100	
12	1	1	11	45	83	
	2	1	29	59	92	
	3	1	42	66	93	
	4	1	50	71	93	
	5	1	57	74	94	
	6	1	61	77	94	
13	1	1	13	42	94	
	2	1	29	58	100	
	3	1	41	66	100	
	4	1	49	71	100	
	5	1	55	74	100	
	6	1	59	77	100	
14	1	1	9	46	100	
	2	1	25	59	100	
	3	1	39	64	100	
	4	1	48	67	100	
	5	1	55	70	100	
	6	1	59	73	100	

Class No.	Vector No.	MSF Type	Rules			
			P_1	P_2	P_3	P_4
15	1	1	0	70	92	
	2	1	17	82	96	
	3	1	29	87	98	
	4	1	36	91	98	
	5	1	43	93	98	
	6	1	50	93	99	
16	1	1	1	71	99	
	2	2	30	65	93	94
	3	1	32	89	100	
	4	1	41	92	100	
	5	1	47	94	100	
	6	1	53	95	100	
17	1	1	8	50	90	
	2	1	26	62	99	
	3	1	39	70	100	
	4	1	46	76	100	
	5	1	52	80	100	
	6	1	56	83	100	
18	1	1	9	48	100	
	2	1	28	62	100	
	3	1	40	69	100	
	4	1	49	73	100	
	5	1	54	77	100	
	6	1	58	78	100	
19	1	1	13	57	85	
	2	1	34	69	89	
	3	1	46	74	91	
	4	1	53	78	92	
	5	1	59	80	93	
	6	1	63	83	93	
20	1	1	38	46	61	
	2	1	42	58	100	
	3	1	44	68	100	
	4	1	35	99	100	
	5	2	55	72	99	100
	6	2	59	74	99	100
21	1	1	7	30	62	
	2	1	26	45	73	
	3	1	39	55	75	
	4	1	48	62	77	
	5	1	53	67	80	
	6	1	58	71	82	
22	1	1	19	77	90	
	2	1	40	88	95	
	3	1	51	92	97	
	4	1	59	95	98	
	5	1	67	96	99	
	6	1	72	97	99	
23	1	1	11	37	66	
	2	1	28	51	79	
	3	1	42	60	82	
	4	1	50	66	84	
	5	1	56	71	85	
	6	1	61	74	87	
24	1	1	2	65	96	
	2	1	15	80	100	
	3	1	33	86	97	
	4	1	47	89	96	
	5	1	56	91	96	
	6	1	62	92	96	

Table (4.1) Continue

From the listed results in table (4.2), the decrease in number of bins causes a decrease in recognition accuracy, and the values of success rate is almost close to each other when the number of bins is taken 100 or 75. So, the default value of number of bins parameter was taken 75 because at this value the recognition system shows good performance in terms of recognition accuracy and processing time.

Table (4.2) Recognition accuracy when the number of bins is 100, 75 or 50

Height×Width	No. Bins	Sample No.	Success ratio
890×560	100	5	100%
	75	5	100%
	50	5	96%
700×500	100	5	96%
	75	5	96%
	50	5	93%
500×500	100	5	91%
	75	5	89%
	50	5	88%
450×300	100	5	75%
	75	5	75%
	50	5	72%
300×200	100	5	68%
	75	5	68%
	50	5	63%
100×100	100	5	40%
	75	5	39%
	50	5	35%

2. Recognition Decision Based on Neural Network

In the first trial of this research work, the neural network was trained to recognize the 24 considered classes of breast tissues, but the attained recognition rates were low and far from being acceptable. This failure is due to the statistical behavior of the fractal dimensions. There is a wide variability in their values, which caused a significant overlapping in the dynamic range of fractal dimension values extracted from different classes of breast tissue images. This overlapping causes a significant loss in recognition accuracy. For this reason the collected samples of breast tissues have been categorized into two categories: normal (class 24) and infected (class 1-23). Then, the neural network was trained to recognize the samples whether they are normal or infected. The number of collected samples to conduct the ANN

training task was 230; where 115 samples extracted from normal image (i.e., class 24) and 5 samples were taken from each one of the 23 images which represent the infected classes (i.e., classes 1 to 23). Two sets of samples were prepared the first set was for training purpose (which consists of 230 sample), and the second set is for testing purpose only. Since, there are two types of images to be recognized, the number of output nodes was taken 2, and the wining node mechanism was applied.

To define the behavior of neural network during the training the following parameters have been determined:

1. The attained minimum training error.
2. Number of iterations and training time, which are correlated, are conducted in training stage.

Table (4.3) describes the involved ANN parameters in the training phase of the neural network. ANN can be successfully trained when using a proper set of its parameters. The suitable values of these parameters can be found by experimentation, because they strongly depends on the nature of the problem to be examined.

After reaching a successful training state, the trained network could be used for breast tissue detection. Table (4.4) presents the adopted default parameters values of the used neural network for breast tissue recognition task.

Table (4.3) The training parameters of the artificial neural network

Parameter name	Description
Number of hidden neurons	The number of hidden neurons, determined by trail and error procedure
Range of initial weights	The range [-r,r] of the initial weights (randomly selected); it has a significant effect on the learning speed and on the ultimate solution.
Initial learning rate(η)	The rate of gradient descent. Bigger learning rate, causes larger changes in the weights (i.e. speed up the convergence), while a small value has a complementary effect.
Critical error (ε)	The desired error (E) for stopping network training; if the actual error is equal to or less than this error then the training will be terminated.
Maximum number of iterations (I)	Maximum limit for training process, it is defined by the user.

In the remaining part of this chapter the effects of the following system parameters on the performance parameters have been investigated:

1. Box size parameter.
2. Sample size.
3. Number of hidden nodes.
4. Learning rate.

A. Effects of Box Size Parameter

Table (4.5) shows the results of the conducted tests to study the effect of box size parameter, where the size of box was varies from 9×9 to 49×49, while the values of other involved parameters have be fixed at their default values. The size of the sample images used in this training was 400×700; and stopping error=1×10^{-6}.

Table (4.4) The default values of the neural network parameters used in training stage

Parameter	Value
Number of input neurons	150
Number of hidden neurons (computed by Using trial and error mechanism)	8
Number of output neurons	2
Range of initial weights	[-0.1,+0.1]
Initial learning rate (η)	0.8
Minimum error (ε)	1×10^{-3}
Maximum number of iteration (I)	10000
Number of training patterns	230
Number of testing patterns	120

Table (4.5) The training results when the box size parameter is varied (when N_{bins} =25)

Box size	Min Error	Iteration no.	Training time (H:M:S)	Recognition Success
9×9	**0.278269**	**1648**	**00:04:32**	**72.5 %**
17×17	**0.115962**	**5978**	**00:13:35**	**82.5 %**
25×25	**0.673604**	**1035**	**00:02:40**	**40 %**
33×33	**0.797089**	**15**	**00:00:25**	**4 %**
41×41	**0.758735**	**10**	**00:00:11**	**4 %**
49×49	**0.766243**	**9506**	**00:22:43**	**14.1%**

The results listed in the above table indicate that the box sizes (9×9) and (17×17) led to recognition success rates 72.5% and 82.5%, which are the best in comparison with attained values when other box sizes are used. Since, the training ratio is small; we tried to apply the idea of using more than one set of fractal dimensions to improve the accuracy results. Two types of combinations of F_D sets were tested. In the first combination type the six F_D dimension sets (which were obtained by changing the value of box size to be L_{max} = 9, 17, 25, 33, 41, 49), were combined and fed to ANN as input layer,

the number of bins was taken 25. The obtained successful recognition rate was 96.6%. In the second combination type, the F_D vectors belong to (L_{max} = 9 and 17), have been assembled in one input vector to ANN; the number of bins was taken 50; and the obtained successful rate was 93.3%. Table (4.6) shows the results of the conducted tests to study the effects of combing the six F_D sets (where L_{max} = 9, 17, 25, 33, 41, 49), and combing of two F_D sets (where L_{max} = 9, 17), while the values of other involved system parameters have been set to have their default values.

Table (4.6) The training results when the combination of F_D sets is used

No. of bins	Box size	Min Error	Training time (H:M:S)	Iteration no.	Training ratio
150	**All F_{Ds}**	**0.028959**	**00:22:21**	**1632**	**96.6%**
50	**9 and 17**	**0.068185**	**00:06:57**	**2867**	**93.3%**

The results listed in table (4.6) indicate that the use of multi-fractal dimensions leads to better recognition accuracy.

In testing stage the idea of using one and more than one set of fractal dimensions was also tested. Table (4.7) lists the obtained recognition rates when different samples sizes have been used, with different values of box size and different sets of fractal dimensions. The results listed in table (4.7) indicate that the use of multi-fractal dimensions would also lead to better recognition accuracy. For this reason the multi-fractal dimensions (i.e., the combined sets of F_D) are used in the remaining tests to find out the effects of hidden nodes and learning rates on the performance of the recognition system.

Table (4.7) The recognition success rates when using different box-size values, applied on the test set of samples

L_{max}	Sample size					
	100×100	300×200	450×300	500×500	700×500	890×560
9	65.8%	40.8%	70%	73.3%	72.5%	58.3%
17	50.8%	41.6%	73.3%	87.5%	85.8%	60%
25	40%	55%	53.3%	52.5%	47.5%	55%
33	4.1%	4.1%	4.1%	4.1%	4.1%	4.1%
41	4.1%	4.1%	4.1%	4.1%	4.1%	4.1%
49	14.1%	17.5%	14.1%	12.5%	8.3%	10.8%
$AllF_{Ds}$	95%	95.8%	96.6%	96.6%	97.5%	95.8%
Both 9 and 17	55.8%	30.8%	55.8%	79.1%	80%	66.6%

B. Effects of the Number of Hidden Neurons

The trail-and-error mechanism was adopted in this research work to find out the proper number of hidden neurons. Various values of hidden neurons numbers were tested to examine the effect of this parameter on ANN performance. Table (4.8) summarizes the results of these conducted tests. The test results indicate that the use of four hidden nodes or higher leads to best performance results. The use of 4 hidden nodes was adopted because it makes the neural network works with less computational time in comparison with the cases of using higher numbers of hidden nodes.

Table (4.8) shows the results of the conducted tests on the training set to study the effect of hidden nodes parameter, the number of hidden nodes was varied from 2 to 8, and the values of other involved parameters were set to have their default values, and the size of image training samples was taken 400×700; the training stopping error was set 10^{-3}; the training stopping difference error=10^{-6}. Table (4.9) shows the obtained correct recognition

rates when the number of hidden nodes is varied, this set of tests was carried on the test set whose samples size was varied.

Table (4.8) The training results when number of hidden nodes is varied

Hidden nodes	Min Err	Iteration no.	Training time (H:M:S)	Recognition success ratio
2	**0.035387**	**1154**	**00:04:39**	**95.83%**
4	**0.032755**	**1508**	**00:09:32**	**96.66%**
6	**0.032862**	**1579**	**00:14:43**	**96.66%**
8	**0.028959**	**1632**	**00:22:02**	**96.66%**

Table (4.9) The effect of number of hidden nodes parameter on the system performance (for the test set)

No. of Hid. Nodes	Sample Size					
	100×100	300×200	450×300	500×500	700×500	890×560
2	**95.8%**	**95%**	**93.3%**	**95.8%**	**96.6%**	**96.6%**
4	**95%**	**95.8%**	**95.8%**	**96.6%**	**97.5%**	**95.8%**
6	**95%**	**95.8%**	**95.8%**	**96.6%**	**97.5%**	**95.8%**
8	**95%**	**95.8%**	**96.6%**	**96.6%**	**97.5%**	**95.8%**

C. Effects of Learning Rate Parameter

The learning rate has important effect on the progress of training process. When it is small it used to prevent the occurrence of large oscillations in error during the training stages. Also it reduces the likelihood that the network will find weights that correspond to main error minima. Different learning rate values were tested (i.e., 0.1, 0.3, 0.5, 0.8, 0.9) to find

out which values leads to the best performance. Table (4.9) shows the training results for various values of learning rate.

Table (4.10) The training results when the value of learning rate parameters is varied

Learning rate	Min Err	Iteration no.	Training time (H:M:S)	Training success ratio
0.9	**0.032631**	**1563**	**00:18:44**	**95.83%**
0.8	**0.038715**	**1632**	**00:22:02**	**96.66%**
0.5	**0.021473**	**1319**	**00:16:39**	**95.83%**
0.3	**0.032137**	**2068**	**00:24:41**	**96.66%**
0.1	**0.027316**	**2798**	**00:74:51**	**96.66%**

The results listed in table (4.9) show that the learning rate equal to 0.5 led the best performance, because at this time the network required less training time with insignificant degradation security. Table (4.10) presents the effect of learning rate of recognition accuracy, when the set of test samples is used as test material.

Table (4.11) the effect of learning rate of recognition rate (when different sample sizes are used)

Learning rate	Sample Size					
	100×100	300×200	450×300	500×500	700×500	890×560
0.9	**95%**	**94.1%**	**95.8%**	**96.6%**	**97.5%**	**95.8%**
0.8	**95%**	**95.8%**	**96.6%**	**98.3%**	**98.3%**	**95.8%**
0.5	**95.8%**	**92.5%**	**93.3%**	**95.8%**	**96.6%**	**96.6%**
0.3	**95%**	**94.1%**	**95.8%**	**96.6%**	**97.5%**	**95.8%**
0.1	**95%**	**94.1%**	**95.8%**	**96.6%**	**97.5%**	**96.6%**

Chapter Five

Conclusion and Future Work

Chapter Five
Conclusions and Future Work

5.1 Introduction

In this research work, two recognition systems have been developed to recognize different breast tissues (normal or cancer infected). The multi-fractal dimensions have been used as textural discriminating features. The difference between the two established systems is the first is based on using fuzzy logic to make recognition decision, while in the second system the feed forward neural network is used as recognition engine instead of using fuzzy logic.

5.2 Conclusions

Many conclusions have been derived during the analysis of the test results, among these conclusions are the following:

1. The size of box, in box counting method, plays an important role in producing fractal dimension. When L_{max} is given values (17 or 25) the shape of PDF become smooth and more distributed over the dynamic range of fractal dimensions.
2. In the fuzzy based system, it was found that triangular shape of MSF is suitable, and there is no need to use more complicated types.
3. In the fuzzy based system, it was found that the use of number of bins equal to 75 (or higher) leads to best recognition results, so setting N_{bin} parameter equal to 75 bins is a suitable choice because less

computational load it required (in comparison with the cases of using higher number of bins).

4. In neural network based system, setting the number of bins equal to 25 leads to unsatisfactory recognition accuracy. But when the number of bins is increased to be 50 (with L_{max} 9 or 17) or higher the recognition accuracy becomes more convincing.
5. In ANN based system, the concatation of all F_D vectors (with N_{bins} =25 for each F_D vector) was superior recognition accuracy in comparison with other combination cases. So, this setup is recommended.
6. The size of samples is not too effective on the recognition accuracy of ANN-based system, while it's so effective on the accuracy of fuzzy logic based system.

5.3 Suggestions for Future Works

In the following, a number of suggestions are listed for further relevant works:

1. Implementing other kinds of fractal dimension estimating methods and compare their effectiveness on recognition accuracy with the obtained results in this research.
2. Using other textural attributes in addition to fractal dimension to improve the recognition accuracy.
3. Using other methods of membership functions (such as, the bell shaped membership functions) to improve the performance of fuzzy logic-based system.

References

References

- [1] E. Al-Hilo, ”*Fractal Colored Image Compression Using Moments Features*”, PhD thesis, Al-Mustansiriyah University, College of Science, 2007.
- [2] J. Amini & M. Rahnemonfar, "*Determination of the fractal dimension in digital terrain model for rough area",* Department of Geomatic Engineering, Faculty of Engineering, University of Tehran, Iran, 2005. *URL: http://www.mapindia.org/2005/papers/pdf/12.pdf* .
- [3] S. Abe, ” *Patteren Classification Neuro-Fuzzy Method and Their comparison*”, Springer-Verlag London Limited, 2001.
- [4] L. Al-Ani, "*Classification of Digital Satellite Images*", PhD thesis, Al-Nahrain University, College of Science, 1996.
- [5] P. Bourke, "*Fractal Dimension Calculator*”, 2003. *URL:http://astronomy.swin.edu.au/~pbourke/fractals/* .
- [6] J. W. Baish & R. K. Jain, "*Fractals and Cancer*", Department of Radiation Oncology, Massachusetts General Hospital, Harvard Medical School, Boston, Massachusetts 02114 [R. K. J.], Cancer Research 60, 3683-3688, 2000. *URL:http//www.**cancer**res.aacrjournals.org/cgi/content/full/60/14/3683*
- [7] D. Da Silva, F. Boudon, C. Godin, O. Puech, C. Smith, & H. Sinoquet, "*A Critical Appraisal of the Box Counting Method to Assess the Fractal Dimension of Tree Crowns*", Proceedings of ISVC, vol. 291, pp. 751-760, 2006.
- [8] R. Dragomir, "*The Analysis of Ct and MR Brain Images Using Box Counting-Type Methods*", Centre hospitalier Louis Pasteur, Service

d'imagerie, Cherbourg France, Mircea Olteanu "Politehnica" University Bucharest, Math. Dept., 2003.
URL:http://www.vectron.mathem.pub.ro/dept/colloq04/M-OLM.PDF

- [9] N. R. Dupaguntla & V. Vemuri, "*A Neural Network Architecture for Texture Segmentation and Labeling*", in Proc. Int. Joint Con. Neural Networks, Washington, Dc, vol. 1, June, pp. 127-133, 1989.
- [10] R. Fuller, "*Introduction to Neuro-fuzzy system*", Heidelberg, 2000.
- [11] L. Fausett, "*Fundamental of Neural Network*", Prentice-Hall International, Inc, 1994.
- [12] J. Guibert & D. C. He, L. Wang., "*Texture Discrimination Based on Optimal Utilization of Texture Features*", Pattern Recognition, vol. 21, no. 2, pp. 141-146, 1988.
- [13] R. C. Gonzalez, *"Digital Image Processing*", Addison-Wesley Publishing Company, 1987.
- [14] Geraldo, *"Fuzzy Logic"*, PhD thesis, Federal University, College of Engineering , Brazil, 2000.
- [15] H. Harms, U. Gunzer & H. M. Aus, "*Combined Local Color and Texture Analysis of Stained Cells*", Computer Vision, Graphics, and Image Processing, vol. 33, pp. 364- 376, 1986.
- [16] R. M. Haralick, "*Statistical and Structural Approaches to Texture*", Proceeding of the IEEE, vol. 67, no. 5, pp. 786-804, 1979.
- [17] K. M. Iftekharuddin, W. Jia & R. Marsh, "*A Fractal Analysis Approach to Identification of Tumor in Brain MR Images*", ISIP lab, ECE Department, University of Memphis, Memphis, TN38152.
*URL:http://www.ee.memphis.edu/people/faculty/iftekhar/area_pub_files/**t umor**.pdf*

- [18] L. C. Jain & N. M. Martin, "*Fusion of Neural Networks, Fuzzy Systems and Genetic Algorithms: Industrial Applications*", CRC Press LLC, 1998.
- [19] G. J. Klir, ”*Fuzzy set Theory Foundation and Application*”, Prentice Hall, 1997.
- [20] N. K. Kasabov, "*Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering*", A Bradford Book, 1996.
- [21] J. M. Keller & S. Chen, “*Texture Description and Segmentation through Fractal Geometry*”, Computer Vision, Graphics, and Image Processing, vol. 45, pp. 150-166, 1989.
- [22] W. Klonowski, **“***Signal And Image Analysis Using Chaos Theory and Fractal Geometry***”**, Lab. of Biosignal Analysis Fundamentals Institute of Biocybernetics and Biomedical Engineering, Polish Academy of Sciences, 2000.
  URL:http://www.hrabia.ibib.waw.pl/~lbaf/PDF_Doc/gkpo2000.pdf.
- [23] S. Y. Kung, "*Digital Neural Networks*", PTR Prentice Hall, 1993.
- [24] J. Mao & A. K. Jain, "*Neural Networks and Pattern Recognition*", Comp. Intell.: Imitating Life, J. M. Zurada, R. J. Marks II, C. J. Robinson (Eds.), IEEE Inc., pp. 194-212, 1994.
- [25] R. Marsh, “*Fractal Net: A Neural Network Approach to Fractal Geometry*”, Report to Computer Science Department University of North Dakota Grand Forks, 701-777-4013.
- [26] B. B. Mandelbrot, ”*The Fractal Geometry of Nature*”, 3rd Edition, W. H. Freeman, San Francisco, 1983.
- [27] W. Niblack, "*An Introduction to Digital Image Processing*", Prentic-Hall International, (UK) LTD, 1986.

- [28] D. W. Patterson, "*Artificial Neural Network Theory and Application*", Printice Hall, 1996.
- [29] C. Pickover & A. Khorasani, "*Fractal Characterization of Speech Waveform Graphics*" Computer Graphics, vol. 10, no. 1, pp. 51-61, 1986.
- [30] J. J. Roan, J. K. Aggarwal & W. N Martin, "*Multiple Resolution Imagery and Texture Analysis*". Pattern Recognition, vol. 20, no. 1, pp. 17-31, 1987.
- [31] A. J. Roberts, "*The Importance of Beings Fractal*", Department of Mathematics & Computing, University of Southern Queens, 2003. *URL:http:\\www.ricardo.ecn.wfu.edu/LaTeX/**fractal**s .pdf*
- [32] R. Saadi, "*Image Segmentation Using Fuzzy and Neural Networks*", M. Sc thesis, Al-Nahrain University, College of Science, 1999.
- [33] R. Setiono & H. Liu, "*Neural Network Feature Selectior*", IEEE Tran. on Neural Network, vol. 8, no. 3, May, pp. 654-662, 1997.
- [34] W. Sun, G. XU, P. GONG, & S. LIANG, "*Fractal Analysis of Remotely Sensed Images: A Review of Methods and Applications*", International Journal of Remote Sensing , vol. 27, no. 22, 20, pp. 4963–4990, November 2006.
- [35] V. W. Samawi, *"An Investigation into the use of Neural Networks in Texture Classification*", PhD thesis, Al-Nahrain University, College of Science, 1999.
- [36] R. Sutton & E. L. Hall, "*Texture Measures for Automatic Classification of Pulmonary Disease*". IEEE Transaction on computers, vol. 21, pp. 667-676, 1972.
- [37] K. H. Sager, *"Fractal Based Classification for Color Textural Images*", PhD thesis, Baghdad University, College of Science, 2006.

- [38] R. Sedivy, "*An Objective Method for Identifying A Typical Nuclei in Dysplastic Lesions of the Cervix Uteri*", Gynecologic Oncology, vol. 75**,** no.1, pp. 78–83, 1999.
- [39] K. D. Toennies, "*Pattern Recognition in Image Analysis*", 2007. *URL:http://isgwww.cs.uni-magdeburg.de/bv/skript/pr/PRIA03.pdf*
- [40] H. Tamura, S. Mori & T. Yamawaki, "*Textural Features Corresponding to Visual Perception*", IEEE Transactions on Systems, Man, and Cybernetics, SMC-8, pp. 460-473, 1978.
- [41] M. Tuceryam & A. K. Jain, "*Texture Analysis*", Handbook of *Pattern Recognition and Computer Vision*, C. H. Chen, L. F. Pau and P. S. P Wang (eds.), World Scientific Publishing Company, pp. 235-276, 1993.
- [42] S. E. Umbaugh, "*Computer Vision and Image Processing*", Prentice-Hell, 1998.
- [43] M. Unser, "*Sum and Difference Histograms For Texture Classification*", Pattern Recognition Lett., vol.1, pp. 43-50, 1986.
- [44] O. Zmeskal, M. Vesely, M. Nezadal & M. Buchnicek, "*Fractal Analysis of Image Structures***",** HarFA - Harmonic and Fractal Image Analysis journal vol. 2001, pp. 3–5. HarFA e-journal, *URL:http://www.fch.vutbr.cz/lectures/imagesci/harfa.htm 11*
- [45] J. M. Zurada, "*Introduction to Artificial Neural Systems*"; JAICO Publishing House, 1997.
- [46] V. B. Rao, *"C++ Neural Networks and Fuzzy Logic"*, IDG Books Worldwide, 1995.
- [47] F. M. Silva & L. B. Almeida, "*Speed Up Back Propagation*"; Advanced Neural Computers, Eckmiller R. (Editor); pp. 151-158; 1990.

الخلاصة

في هذا البحث تم استخدام منظومتان لتمييز الصور النسيجية لسرطان الثدي. في المنظومة الأولى تم اعتماد المنطق المضبب، أما في المنظومة الثانية فقد تم استخدام الشبكات العصبية. في المنظومتين اختبار ٢٤ صوره نسيجيه مثلت ٢٣ منها صنوف مختلفة من سرطان الثدي، وصوره واحده تمثلت الحالة الطبيعية الغير مصابه. في البداية تم تحليل هذه الصور باستخدام البعد ألكسوري المتعدد. وقد تم استخراج ستة مصفوفات أحاديه الأبعاد لتمثل الخصائص الأولية للصورة، واستخدمت هذه المصفوفات كمدخلات أساسيه لكلا المنظومتين.

في المنطق المضبب استخدمت الدالتين العضويتين (المثلثة والشبه منحرفة) لتمثيل درجة العضوية لكل قيمة مستخرجة للبعد ألكسوري. في مرحلة التمييز يتم حساب مجموع العضوية للعينة المراد اختبارها بالاعتماد على القواعد المستنبطة، فالصنف الذي سيؤشر أعلى مجموع لدرجة العضوية سيمثل الصنف الذي تنتمي له العينة المراد اختبارها. أظهرت النتائج ان $N_{bins} = 75$ يعطي نتائج جيده لكل من الدقه والوقت. كما واظهرت النتائج ان حجم العينة المراد اختبارها يؤثر على نتائج التمييز حيث يتناسب طرديا مع كبر حجم العينه المراد اختبارها.

في المنظومة الثانية تم استخدام الشبكة العصبية من النوع "التغذية الأمامية المنتقلة". فقد تم تقسيم المدخلات (٢٤ صوره) إلى صنفين (أما صوره مصابه أو غير مصابه). تم أخذ عينات من جميع الصور وتدريب الشبكة عليها، ومن ثم تم اختبارها على مجموعتين من العينات "المدربة والمختبرة". أظهرت النتائج إن استخدام البعد الكسوري المتعدد N_{bins} يساوي 150 يعطي أفضل نتائج للتمييز بالنسبة للعينات المدربة والمختبرة، كما واظهرت النتائج ان استخدام hidden nodes = 4 و learning rate =0.5 يعطي افضل النتائج

من ناحية الوقت والدقه.كما وأظهرت النتائج إن حجم العينات المراد اختبارها لا يؤثر على نتائج التمييز.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة النهرين
كلية العلوم

تحليل أنسجة سرطان الثدي باستخدام البعد ألكسوري، المنطق المضبب والشبكات العصبية

رسالة مقدمه إلى كلية العلوم في جامعة النهرين كجزء من متطلبات نيل درجة الماجستير في علوم الحاسبات

من قبل

هديل حاتم جاسم

(بكالوريوس جامعة النهرين ٢٠٠٣)

أشراف

د. لؤي أدور جورج

1429 **2009**