WAEL HMIDI

ARTIFICIAL NEURAL NETWORK FOR
RECOGNITION OF GINGIVITIS

NEU
2017

ARTIFICIAL NEURAL NETWORK FOR RECOGNITION OF GINGIVITIS

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
WAEL HMIDI

In Partial Fulfilment of the Requirements for the Degree of Master of Science in Electrical and Electronic Engineering

NICOSIA, 2017

ARTIFICIAL NEURAL NETWORK FOR RECOGNITION OF GINGIVITIS

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
WAEL HMIDI

In Partial Fulfilment of the Requirements for the Degree of Master of Science in Electrical and Electronic Engineering

NICOSIA, 2017

WAEL HMIDI: ARTIFICIAL NEURAL NETWORK FOR RECOGNITION OF GINGIVITIS

Approval of Director of Graduate School
of
Applied Sciences

Prof. Dr. Nadire ÇAVUŞ

We certify this thesis is satisfactory for the award of the degree of Masters of Science in Electrical and Electronic Engineering

Examining Committee in Charge:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Wael Hmidi

Signature:

Date:

ACKNOWLEDGMENTS

I would like to express my sincere gratitude and thanks to my beloved supervisor, Assist. Prof. Dr. Kamil Dimililer , for his continuous guidance and assistance during my graduate studies. His inspiring knowledge and creative thinking have been source of encouragement throughout this work .

My deepest gratitude goes to my sisters, brother, and all my family, to whom I am most indebted. I thank them for the love, prayers, patience and support while I was studying. I know I can never come close to returning their favour upon me. I will always be thankful to my friends and colleagues for their unlimited support. I extend my thanks to all the Libyan community that gave me a second family away from home.

I would also like to express my gratitude to my wife for her support, patience, and feelings of love all over my life. Your constant love was the source of motivation for me to go on my way.

I present this work...

To the most precious woman; my dear mother (AL-Aaqela AL-Gizzawe) who has spared no effort in my upbringing and guidance

To the reason of my being in life, who stood with me all my life... My beloved father (Mustafa Hmidi)

May Allah grant them both his vast paradise.

To my parents and family….

ABSTRACT

Periodontal diseases are dangerous types of diseases that affect the teeth and the supporting bones and tissue. Gingivitis is one of these diseases that are caused by the existence of plaque between teeth. Plaque is the sticky waste of the bacteria and food remains that are not cleaned correctly. The gingivitis can cause permanent damage of teeth if not treated early and cause different losses of teeth. The early detection of this type of diseases is very important for the cure of patients against the loss or damage of their teeth. This work proposes the use of artificial neural networks for the diagnoses and early detection and prediction of the periodontal gingivitis disease. A back propagation based artificial neural network is used to identify and classify 160 different images into two classes. Different types of image filters were used in the image processing to compare their different results. Each one of the filters was applied separately to compare the performance of ANN with each filter. Weiner filter has proved it is offering the highest efficiency for the proposed system.

***Keywords*:** Artificial neural networks; back propagation; gingivitis; periodontal; teeth

ÖZET

Ağız ve damak hastalıkları, dişleri, onları destekleyen kemikleri ve dokuları etkileyen tehlikeli bir hastalık türüdür. Damak ağrıları dişler arasında oluşan taşların neden olduğu bu hastalıklardan birisidir. Taşlar, temizlenmeyen, yapışkan bakteri ve yiyecek kalıntılarıdır. Damak hastalıkları erken tedavisi yapılmazsa dişlerde kalıcı zararlara ve diş dökülmelerine neden olabilir. Bu tür bir hastalığın erken teşhisi, hastalarda dişlerde zarara veya diş dökülmelerinin tedavisi açısından çok önemlidir. Bu çalışmada, damak ve ağız hastalığının teşhisi, erken yakalanması ve olasılıkları ile ilgili yapay sinir ağlarının kullanılması önerilmektedir. Backpropagation temeline dayanan yapay sinir ağı 160 farklı görüntüyü belirleyip iki sınıfta toplamakta kullanılmaktadır. Farklı görünü filtreleri görüntüler arasındaki karşılaşrırmak için görünü almada kullanılmıştır. Her iki filtre, ANN performansını birbiriyle karşılaştırmak için ayrı ayrı uyulanmıştır.

Anahtar Kelimeler: Yapay sinir ağları; geri yayılım; fotograf filtrelemesi; veri tabanı; diş inflamasyon

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN: Artificial Neural Networks

AI: Artificial Intelligence

BPL: Back Propagation Learning

MIPAV: Medical Image Processing and Visualisation

MLP: Multi-Layer Perceptron

MSE: Mean Squared Error

RGB: Red, Green, and Blue

SLP: Single Layer Perceptron

CHAPTER 1

INTRODUCTION

1.1 Introduction

The goal of developing researches in the medical field is the development of accurate, high performance, and low cost algorithms and procedures to support the specialists in decision making. In this work, the investigation of artificial neural networks for the diagnosis of the Gingivitis teeth inflammation is proposed and implemented. Gingivitis is one of the teeth troubles that is not easy to be identified and diagnosed in its early stages of the disease. The use of artificial neural network in medical image analysis is increasingly gaining interest of researchers and scientists. It has proved its ability to achieve high performance and support the specialists in the diagnoses of different diseases like breast cancer, lung cancer, skin diseases and many other types of infections. The literature of the past few years shows the increasing interest of researchers in the development of a computerized gingivitis detection procedure. Many researchers have focused on the implementation of the neural network technology for the detection of the gingivitis.

Artificial neural networks have developed during the last two decades and penetrated many fields with its strong and reliable structure. It has proven its high capability of performing well in different applications and wide range of topics. ANN has been inspired from the structure of the human neural system. It is continuously developing and being updated to enhance its capability and make it more realistic and performing.

This research will focus on the implementation of a computer based gingivitis classification system based on an image processing approach and artificial neural network. A comparison of the performance of neural network under different image processing configurations will be carried out. Different types of filters like the adaptive Weiner and Median filter will be implemented and their results will be compared.

1.2 Gingivitis Inflammation of the Gums

Gingivitis is a periodontal disease or one of the familiar and soft forms of teeth gum infections. It can cause inflammation, irritation, red colour of gum and swelling of gingival. Gingival area is the area of the gum that surrounds the base of the teeth. Gingivitis is a serious infection that needs to be considered carefully and treated as fast as possible. This disease has the ability to damage the tissue of the area around the teeth. It is caused by the collection of plaque dumps between and around the teeth. Plaque is a textile that has adhesive nature, which is produced by a type of bacteria that lives on the food remains in the mouth. Plaque builds up around the teeth causing several dangerous diseases like gingivitis and may cause the teeth decay. Gingivitis is difficult to be detected in its early stages and needs to be carefully evaluated and diagnosed. The early treatment of such a disease is very important to protect the teeth from eternal damage and keep the nice look of the face.

1.3 Medical Image Processing Analysis and Visualization

The Medical Image Processing, Analysis, and Visualization (MIPAV) is referred to the use of different types of medical images like Magnetic resonance imaging, CT imaging, PET and their implementation in the visualisation and investigation of different types of diseases. MIPAV is becoming with higher importance in the analysis and diagnoses of all types of body troubles due to its high visual efficiency and accuracy in disease detection and evaluation. This tool helps scientist and medical researchers to find and monitor diseases with the least efforts and highest possible accuracy.

1.4 Literature Review

The implementation of ANN for the early diagnoses and detection of periodontal diseases has been the subject of different researches in the last few years. A neural network based system was proposed in (Thakur, Guleria, & Bansal, 2016). The research focused on the use of artificial neural networks in the early detection of periodontal diseases and gingivitis using their symptoms and risk factors. Collected data for 200 patients' symptoms and risk factors were used in the training of the ANN. Multilayer neural network was used with Levenberg-Marquardt training algorithm. In this research, sigmoid based ANN was used with one single hidden layer. The research has found the artificial neural network to be efficient with 82% correlation and a robust method for early detection of the gingivitis.

(Sudheera, Sajja, Kumar, & Rao, 2016) in their research proposed the implementation of K-mean method for the image segmentation and diagnoses of dental images for detection of plaque. K-mean method was applied on the data base images after being converted to Hue images. The research had a purpose of detecting the amount of plaque in the dentist area. In (Nagane, Dongre, Dhar, Jadh, & Burghate, 2016) a literature review about dental trouble and gingival problems was presented. The use of ANN for the detection of gingival diseases was also discussed and presented. No data or results were provided in this research. (Patil & Patil, 2011) has discussed the periodontal diseases like gingivitis and the use of biomedical methods for the early prediction of their existence.

In (Papantonopoulos, Takahashi, Bountis, & Loos, 2014); the authors presented a study the prediction methods of prediction for aggressive periodontal diseases. They claimed that the best results of prediction were obtained using the artificial neural network techniques and gave efficiency between 90-98%. They used an artificial network with one hidden layer to classify the dataset into two distinct groups. Their research has revealed that the use of ANN was of the best and highest performance method for the diagnoses of dental diseases.

CHAPTER 2

GINGIVITIS DISEASE

2.1 Introduction

This chapter will present the teeth disease of gingivitis as one of the infections that appear in the teeth and affect the gum. General information on the infection, diagnoses, causes and cautions as reported in medical reports will be provided. The second and third chapter of this work will also discuss the image collection phase and the different image processing steps implemented in the disease automated detection system.

2.2 Gingivitis Disease

Periodontal, or gum infection is a well known teeth problem that attacks the part of the gum that cover the structure that hold the teeth. Gingivitis, periodontal ligament, cementum, and the alveolar bone are of the main types of teeth or gum inflammation (Wilder & Moretti, 2016). Gingivitis is a periodontal disease or one of the familiar and soft forms of teeth gum infections. It can cause inflammation, irritation, red colour of gum and swelling of gingival. Gingival area is the area of the gum that surrounds the base of the teeth. Gingivitis is a serious infection that needs to be considered carefully and treated as fast as possible. It can cause very serious gum infections that are known as periodontitis and result in tooth losses. The main reason for the gingivitis is the lack of hygiene and bad oral habits. Daily brushing and flossing the teeth in addition to the periodic dental examinations and checkups can keep away the effects of gingivitis (Wilder & Moretti, 2016);(Clinic, 2017).

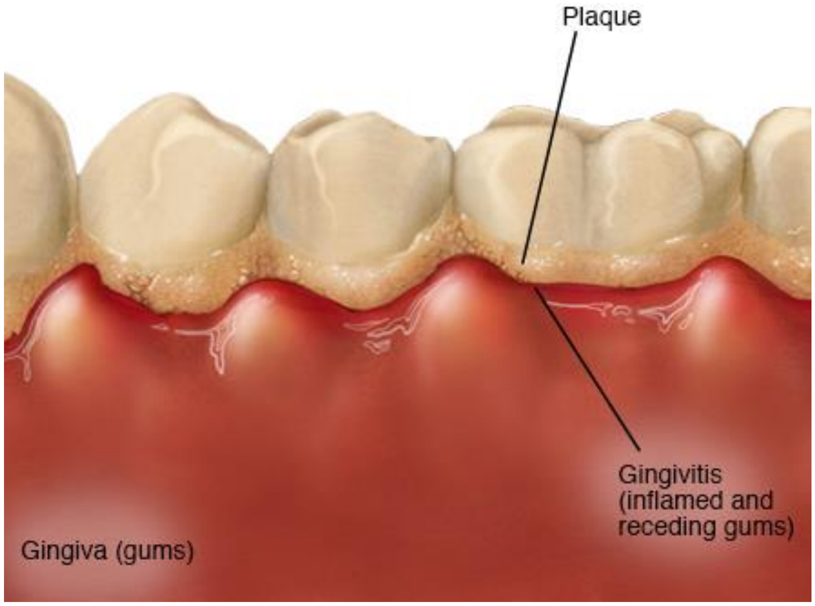


Figure 2.1: Symptoms of gingivitis in the teeth gum (Clinic, 2017)

Figure 2.1 presents the symptoms of the gingivitis in the gum of the teeth. Main symptoms of the gingivitis are swollen gums, dark red coloured teeth surrounding area, non-suitable breathing smell, thin and soft gum, and easily bleeding gum. On the contrary, the healthy gum is pink coloured, firm, well tightened around the teeth, and does not bleed easily.

2.3 Main Causes of Gingivitis

The main cause of gingivitis is the bad teeth cleaning habits during the day. This leads to the formation of plaque that can with time cause the gingivitis in the gum of the teeth. The gingivitis starts by the plaque that forms on the teeth. Plaque is a thin layer of bacteria. This bacteria form on the teeth when sugar or starches react with the mouth bacteria. If this plaque is not removed daily and cleaned properly, it starts to become hard and form tarter or coal. This creates a shield that protects bacteria again brushing tools and components. If the tarter and bacteria is not removed quickly, it very soon will become inflamed and cause gingivitis to appear in the gum. With time, the teeth will become swollen. There are different factors that affect the spread of inflammation and increase the chances of gingivitis in the gum; these factors are:

- bad teeth care practice

- Smoking
- Age factor
- Dry jaws
- bad nutrition habits, and lack of vitamin C
- broken or cracked teeth that are difficult to be cleaned
- Some types of treatments can also cause gingivitis
- Hormonal changes and its effects, especially in case of pregnancy, menstrual cycle and some types of birth control products
- Genetic factors can also affect the appearance of gingivitis

The treatment of gingivitis must be carried out as fast as possible in order to avoid the different complications of the disease. The gingivitis can spread into sub layers of the gum and even can affect the bones of the dental structure. Periodic teeth checkups are very vital for prevention of teeth diseases like gingivitis. The checkups include looking for cavities and different shapes in the gum. Some checkups may include X-rays radiography or other diagnostic procedures. Medical image processing techniques can also be implemented in the diagnoses of gingivitis disease.

2.4 Diagnosis of Gingivitis

Diagnosis of this disease depends on the clinical examination of the mouth and gums in terms of shape and colour as well as pain associated with the condition. It is assumed that the healthy gums are pink, natural, shiny, and smooth and have pores. It is also well known that the healthy gums extend from the neck of the tooth. It covers the entire bone while the infected gums appear swollen dark red covering the visible part of the tooth with bleeding. As a result of the aggravation of the disease and the bone erosion it shows most of the walls of the tooth. The diagnosis also depends on data taken from the patient. The extent and severity of bulging and dental pockets are detected by using periodontal probes. Figure 2.2 shows different shapes of periodontal probes. In the clinical test, the dentist inserts the probe pursing the pockets of the teeth and measuring the depth of that pocket. The depth is measured in gradual manner to see how the swell exceeds the depth of the teeth. According to the opinion of dentists, this test is one of the painful and difficult tests that are carried out in their clinics. Finding new automated technologies that can help in the detection of the disease will be considered an important favour for dentists.

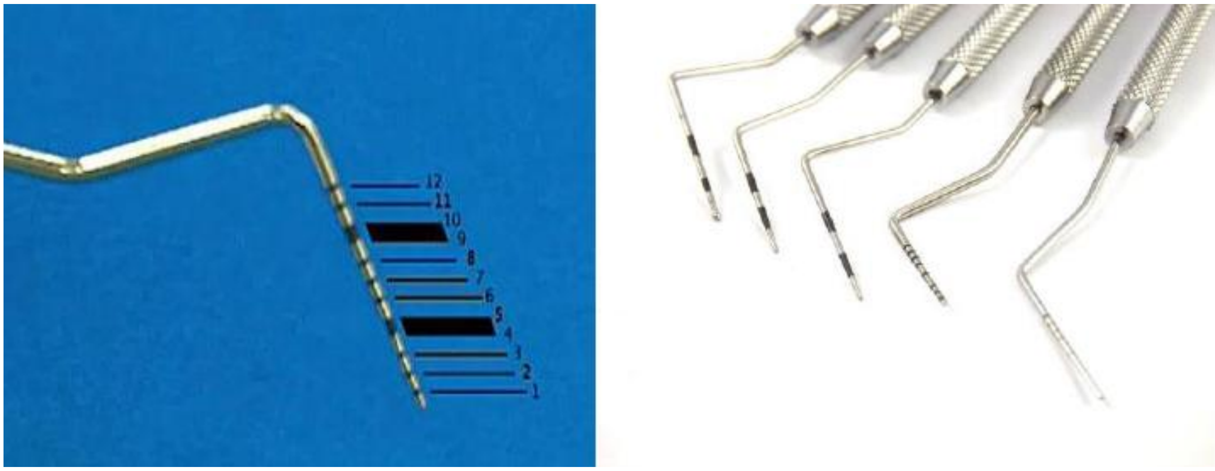


Figure 2.2: Periodontal probes used in the diagnosis of gingivitis

CHAPTER 3
DIGITAL IMAGE PROCESSING

3.1 Medical Image Processing

Medical image processing is the process in which medical images are used and treated automatically or manually to detect different diseases. There exist many data collections on the web for different types of diseases. The development of digital electronics and computer software has encouraged the development of different automated medical software. The use of automated disease detection processes is gaining more and more interest in scientific medium. An image dataset is a compilation of images or medical images with some connotation, usually implemented for a given process. In many cases, image collections have very vital importance for illustration of some visual concepts in different communities.

3.2 Image Processing and Database Collection

In order to apply the automated detection of disease using medical images, different image collection and image processing steps need to applied prior to the beginning of the work. The first step of the image processing is the collection of the suitable images that can be used in the system as examples of both healthy and infected gum. At the next steps the image processing phase starts preparing for the application of recognition system. The bloc diagram of Figure 3.1 presents the different steps of image processing starting from the database collection and ending by constructing the input image for the ANN.

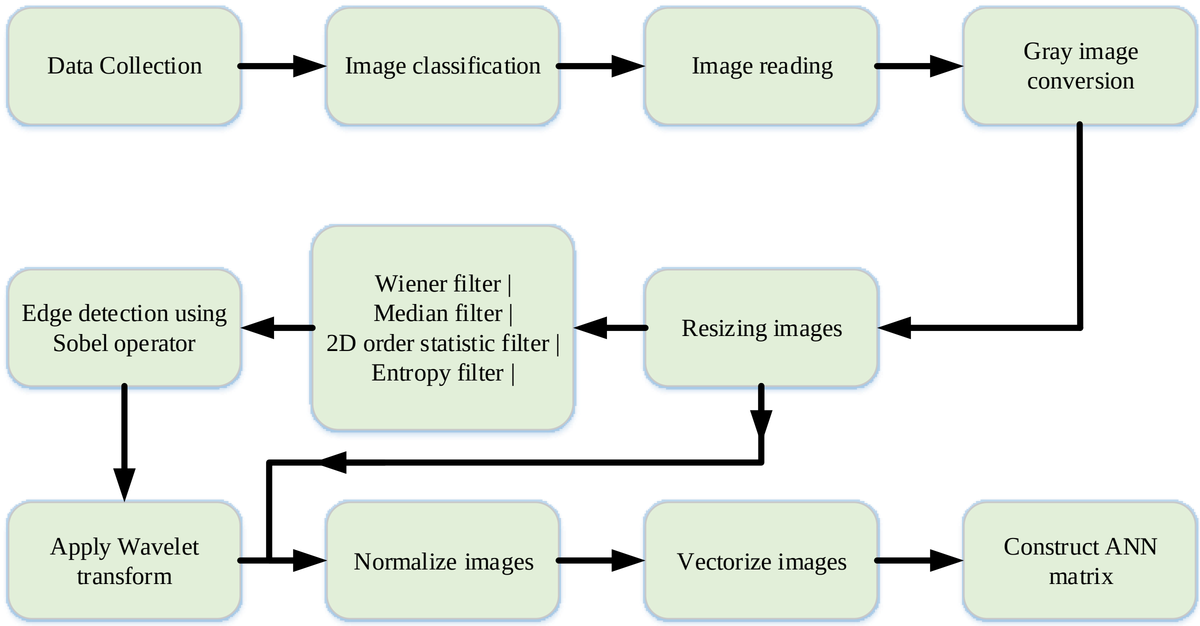


Figure 3.1: Bloc diagram of image processing phase of the system

3.2.1 Data collection of gingivitis images

To obtain the required image database, different sources were explored looking for suitable database images. This data base was collected from internet medical sources and from some dentist clinics in the Libyan Republic and under supervision of dentist. The collected database includes 60 normal healthy teeth and gum images in addition to 100 gingivitis-infected images. The infected images were all diagnosed to be infected by specialist dentist and confirmed to be infected. Figure 3.2 presents the collection of all the infected images in RGB format. The images show variety of shapes and degrees of sickness and their classification is a real task that implies special efforts. In some cases, infected teeth look like normal teeth with some difference in the colour, which is difficult to be detected.

Figure 3.2: The collection of the 99 infected images

3.3 Image Processing Techniques

Collected images were all arranged in two separate folders. One of these folders contains the normal images while the other contains the infected images as diagnosed by dentists. The images were stored in JPG format with RGB scale. The RGB images contain information about different frequency spectrums of the visual light. RGB images can provide human eyes with more details than any other type of image can provide. However, for computer vision the colours represent less importance as computer does not understand colours. Computer is more affected by light intensity in each point of the image. This is the reason why most computerized classification and recognition tasks are using the gray scale image format. Gray scale image format contains details of light intensity of each pixel in an image.

(a) Normal image (b) Infected image

Figure 3.3: Original images of normal and infected images

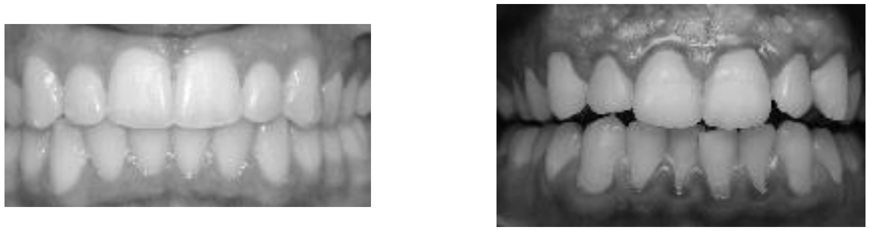
RGB images contain three colour density matrices for the Red, Blue, and Green colours. The colour concentration of these three colours creates the difference in the human visual system that help to identify and distinct between different images. The human eye is sensitive to the colour details of an image. For this reason, coloured images are nowadays used in all visual systems and imaging applications, video streaming etc. however, computer or machine vision applications are not sensitive for colour details in general. The computer is less interested in the beauty of the green colour of grass in an image of the nature. For most of automated image processing and recognition tasks, the coloured images are converted into another form or scale that incorporates less data processing and avoid the extra details. For this reason, the images of our experiment were all converted into gray scale images to simplify the computer recognition task applied.

3.3.1 Gray scale image

In the gray scale image, the intensity of white light is expressed in the form of integer number in each pixel. The pixel light intensity expresses the range of light intensity between white and black colours. Hence, the name gray scale is used because it is the colour that resides between the black and white. Gray scale images are widely implemented in computerized recognition processes due to their low processing cost compared with the RGB images. The gray image is created by converting the RGB image intensity matrices into one intensity matrix using the next formula (Zollitch, 2016). Figure 3.4 presents a sample of the database images after being converted into gray scale images. The gray scale images presented in Figure 3.4 are visually not clear compared to the original RGB images. However, the computer does not require the colour details for increasing its performance in image processing.

$$g = 0.299 * R + 0.587 * G + 0.114 * B \tag{3.1}$$

Where; R, G, and B are the intensity values of red, green, and blue colours. The ‘g’ refers to the gray scale intensity of the image. This formula is not designed based on the equal division of three colour intensities. It is built around the sensitivity of human eye for each one of the colours. It was found that each colour has specific sensitivity in the human eye. The above formula describes the best the weight of image colour as it is seen by the human eye (Zollitch, 2016).



(a) Normal image (b) Infected image

Figure 3.4: Gray scale version of database images

3.3.2 Filtering the gray scale images

The images are generally subject to different types of noises and deformations due to multiple factors. Noise and deformation in the image can be the result of capturing devices, user movements during the image capturing, or other sources of noise caused by storage and transmission processes of images. In order to ensure the removal of the important noises from the image pixels, special processes should be applied that can reject different types of noises or reduce their intensity. Rendering the original image from a deformed version of that image is also concerned by the filtering process. Filtering is a special mathematical process applied on the image to reject the noise and restore a clean noise free image. The bad, pulsing, and irrelevant details can be removed from a digital image through the process of image filtering. There exist many types of filters that can be applied on digital images to remove noise such as median filter, Gaussian filter, Wiener adaptive filter, and many other different types of filters. Each one of these filters has its own characteristics and drawbacks. In our thesis work, wiener filter will be implemented and used as it offer best performance as sited by (Mohan, Mridula, & Mohanan, 2016). Other filters will be also implemented and discussed in this work for comparison and performance experimentation.

3.3.3 Wiener filter

Wiener filter is a type of the adaptive filters that is based on mean squared error approximation. It is a type of stationary linear filter that is used to enhance degraded images by types of additive noise or blur. It is a frequency domain filter that is applied on the discrete Fourier transform of an image. The original spectrum of the image is found by convolution of the Wiener filter with the DFT of the image. The image is then constructed by applying the inverse form of DFT to recalculate the estimate of the image.

Figure 3.5 presents a sample of the normal and infected images after applying Wiener adaptive filter. Weiner filter was proposed as the best solution for different image restoration and filtering applications in (Fechner, 1993; Khajwaniya & Tiwari, 2015; Wang, Peng, Wang, & Peng, 2015). Weiner filter is a low pass filter that is applied for gray scale images affected by additive noise types. It uses a pixel wise wiener approach using an estimation of the local surrounding pixels of each element of the image. Using wiener filter, the local mean and standard deviation values are estimated around each pixel. Based on these estimated values, the filter creates a new value of the pixel as shown in the next formulas.

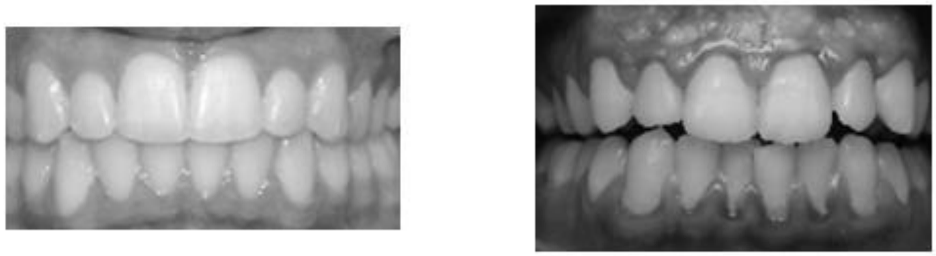
$$m = \frac{1}{MN} \sum_{n1,n2} a(n1,n2) \tag{3.2}$$

$$\sigma^2 = \frac{1}{MN} \sum_{n1,n2} a^2(n1,n2) - m \tag{3.3}$$

Where; *M* and *N* are the length and width of the wiener filter window, m and σ^2 are the mean and standard deviation values of wiener window pixels. The estimate of the new pixel value is then given by:

$$x(n1,n2) = m + \frac{\sigma^2 - v^2}{\sigma^2}(\Delta m) \tag{3.4}$$

Where; the term Δm refers to the difference between the pixel value and the local mean of that value or the deviation of the pixel from its local mean value. It is important to remember here that the Wiener filter is based on an estimation of the noise in the local neighbourhood of the concerned pixel.



(a) Normal image (b) Infected image

Figure 3.5: Wiener filter application on the images

3.3.4 Median filter

Median filter is one of the most famous filters in image processing techniques. It uses the well known statistical median term to remove impulsive noises from the images and reject all extremely different values from their neighbourhood. Median filter is used to smoothen or removing blur from images. A lot of smoothening techniques implement a type of low pass filter on the images as a solution or they apply a median process for them (Tantua, 2015). Non linear filters like the middle value or median filter are the best choice for the smoothening of images. Median filter is implemented to cancel the impulsive type noise from 1D and 2D signal like images. Median filter does the filtering process while not affecting the original image feature (Stella & Trivedi, 2012).

The idea of median filter is very simple and clever. A window that contains a specific number of image elements horizontally and vertically is set up (Helwan, 2014). The filter then applies statistic median operation of the window and replaces centre value with the median. The window continues moving horizontally and vertically to be applied to the whole image. This way all value that have great differences from their neighbours are eliminated and replaced with more realistic values. Median filter is most effective at removing salt and pepper noise type case they affect special pixels of the image. The median is found by arranging all the pixels of the concerned window in increasing order, and then substituting the considered pixel with the middle or median value. The result of applying median filter on 2D vectors is shown in Figure 3.6. All the pixels with values that are not near to their neighbours were simply eliminated as the figure illustrates.

5	2	6	5	5	5	5	2	6	5
10	4	3	6	5	1	10	4	3	6
1	6	5	4	5	2	1	6	5	4
8	6	1	1	3	1	8	6	1	1
4	12	3	6	4	1	4	12	3	6
1	6	6	6	2	5	1	6	6	6
5	2	6	5	5	5	5	2	6	5
10	4	3	6	5	1	10	4	3	6
1	6	5	4	5	2	1	6	5	4
8	6	1	1	3	1	8	6	1	1

(a) Original 2D vector

0	3	3	5	5	5	2	3	3	0
2	5	5	5	5	5	4	5	5	4
4	5	4	4	3	3	4	5	4	1
4	5	5	4	3	3	4	5	5	1
4	6	6	3	3	3	5	6	6	1
2	5	6	5	5	4	5	5	6	5
2	5	6	5	5	5	5	5	6	5
2	5	5	5	5	5	4	5	5	4
4	5	4	4	3	3	4	5	4	1
0	1	1	1	1	1	1	1	1	0

(b) 2D vector after median filter

Figure 3.6: Application of median on 2D vectors

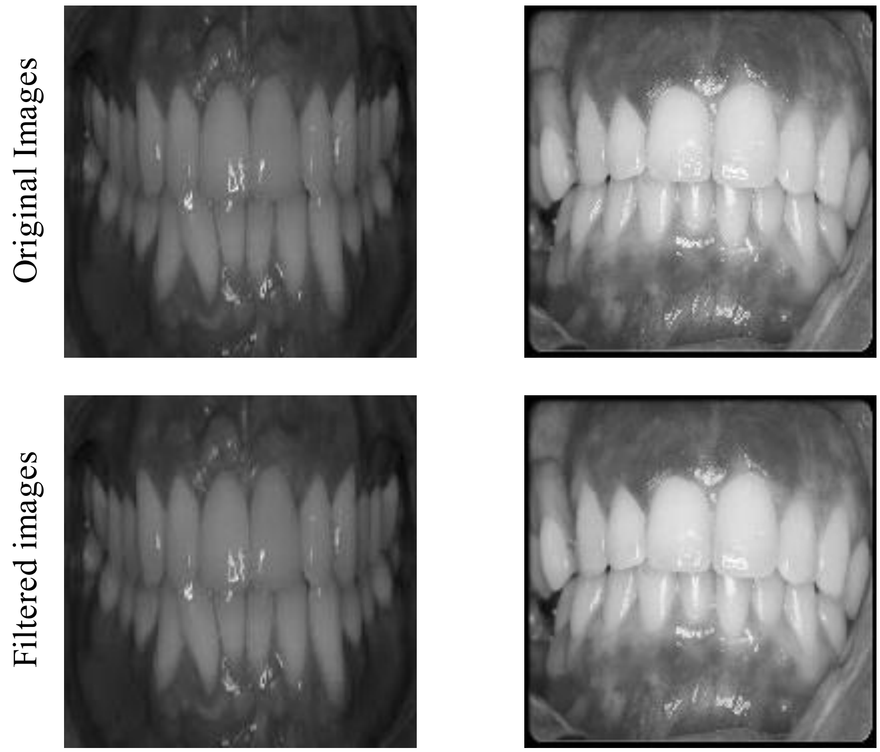
Figure 3.7 presents the application of median filter on the noisy image and the filtered image. It is obvious that the filter has removed all the noise from the image and restored it to its clear state without noise. The application of median filter on out image database will reduce all types of impulsive noise that may be contained within (Gonzalez & Woods, 2001).

(a) Salt & pepper noisy image

(b) Median filtered image

Figure 3.7: Applying median filter on gray scale image

The Application of Median filter on the different database images will ensure the removal of all non-required noise and effects and create clearer image. This is very important to avoid differences in raining and test database due to different noise types. The removal of noise prior to test and training increases the chances of success of classification task. Figure 3.8 shows the result of applying Median filter on the database images.

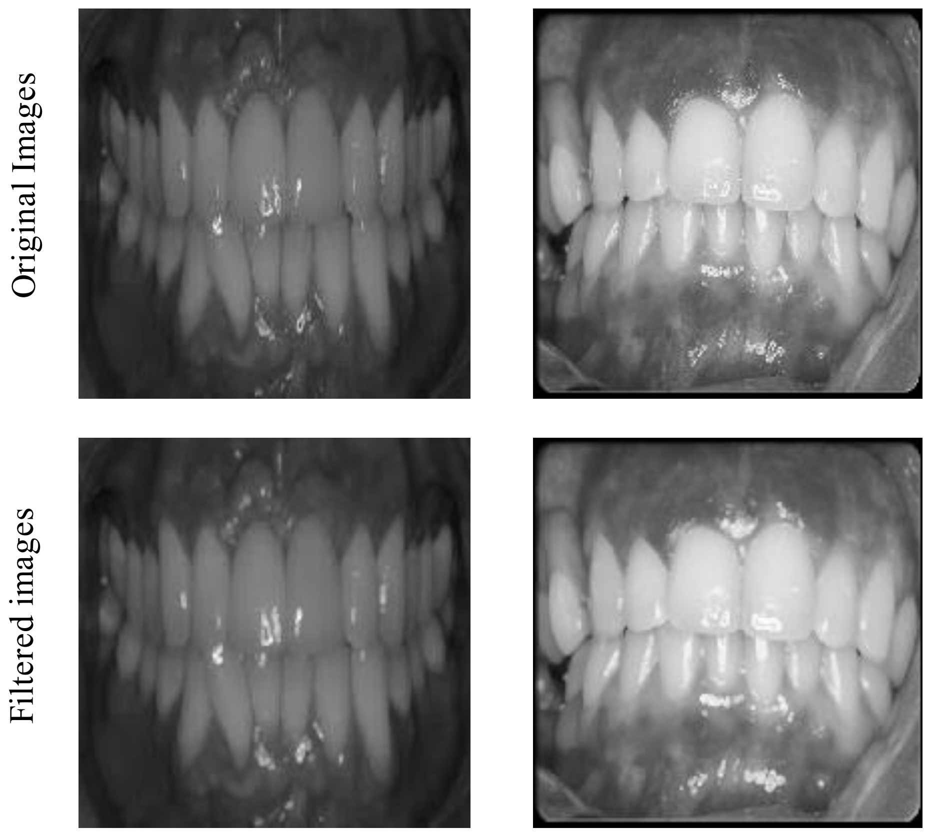


(a) Normal image (b) Infected image

Figure 3.8: Median filtered applied to the database images

3.3.5 2D Order statistic filters

Order statistics filters are classified as spatial nonlinear filters. The output of this type of filters is a function of arranging the local pixels in the area of filter application. The centre pixel is then replaced by the result of ranking. In this definition, it is clear that Median filter is considered as an order statistic filter type. It replaces the centre pixel by the Medial of the local area in which the filter is applied (Gonzalez & Woods, 2001). The 2D order statistic filter is similar to the Median filter as it uses a ranking approach to arrange the elements of the local area of the concerned pixel. This filter then replaces that pixel by one of the elements of the arranged vector based on the specified order. Suppose choosing rectangular area of 3x3 elements to be specified around the treated pixel. The first order is the minimum, the last is the maximum. If the middle was chosen, the filter will be then a 2D order statistic filter of type Median (Stella & Trivedi, 2012). Figure 3.9 illustrates the result of using 2D order statistic filter on the database images. Statistic order filters can be applied on kernels with 3x3, 5x5, and 7x7 elements (Stella & Trivedi, 2012). Different kernel sizes of 3x3 and 5x5 were used and experimented with different orders. The minimum and maximum orders were also experimented in this work. The use of 3x3 elements kernel with the 7th element has given a training efficiency of 65% while the 9th element has given a training performance of 58.75%. After executing many experiments the filter was applied on a window of 5x5. The 20th element was chosen to be replacing the centre of the filter kernel. In our work the 5x5 kernel was found to be the best choice to be used.



(a) Normal image (b) Infected image

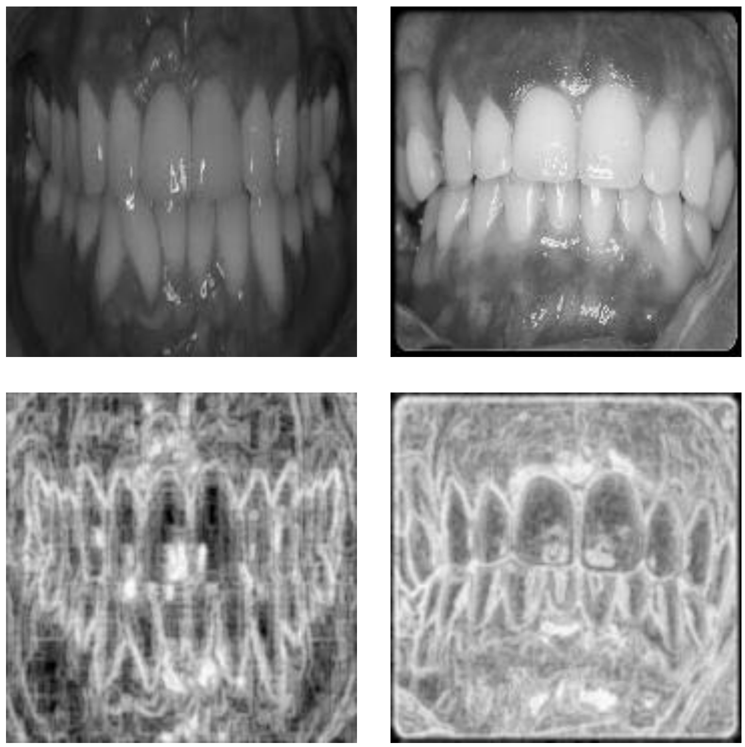
Figure 3.**9**: Application of 2D order statistic filter on images

3.3.6 Entropy filter

The Entropy in image processing is the relationship between the pixel and its neighbor pixels complexity. This type of filter is able to detect very small variations in the local gray scale allocation of the image. Entropy is one of the statistic types of random degree measurement that is implemented to specify the quality of the image (Gonzalez & Woods, 2001). The entropy of the pixel neighborhood is defined as a logarithmic for the base 2 of the histogram of the image. It is defined by:

$$E = -sum(p.*\log_2(p)) \tag{3.5}$$

Where; the *P* term in this equation refers to the histogram of the image at the specified area of neighbour pixels. The kernel of image was chosen to 3x3 in this application. Figure 3.10 presents the results of applying Entropy filter on the database images.



(a) Normal image (b) Infected image

Figure 3.10: Application of Entropy filter on the images

3.3.7 Image segmentation

Image segmentation is the process of extracting the important features and lines from an image. It is important in reducing the amount of details in the image that are not important for the recognition system. Image segmentation is a vast topic that covers many methods and algorithms. Generally, segmentation of special purpose images is done manually because there is no special method that can offer the best automatic segmentation. In different applications, special segmentation approaches are used for each individual project. The segmentation method that is used for an image or a pattern of images can be useless for other pattern of images. Sobel edge detection method is a gradient based edge detection approach that takes as input the gray scale image. It defines the pixels of image where the gradient is maximal. This means the pixels where sudden change happens.

In Sobel edge detection, the gradient of each pixel in the image is found. Two masks for x and y directions are given by:

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad ; \quad G_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{3.6}$$

The simplest way of finding the gradients is to apply these two masks for each 3*3 window of the image. Suppose having the next matrix of 4*4 randomly chosen values, the Sobel edge detection is applied as follows:

$$A = \begin{bmatrix} 2 & 3 & 2 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 4 & 4 & 2 \\ 4 & 1 & 4 & 3 \end{bmatrix} \tag{3.7}$$

The gradient of first pixel in the x direction is calculated by taking the first 3*3 window:

$$\begin{aligned} G_x(1,1) &= (A_{31} + 2A_{32} + A_{33}) - (A_{11} + 2A_{12} + A_{13}) \\ G_y(1,1) &= (A_{13} + 2A_{23} + A_{33}) - (A_{11} + 2A_{21} + A_{31}) \end{aligned} \tag{3.8}$$

From which the magnitude of gradient is found by:

$$G = \sqrt{G_x^2 + G_y^2} \tag{3.9}$$

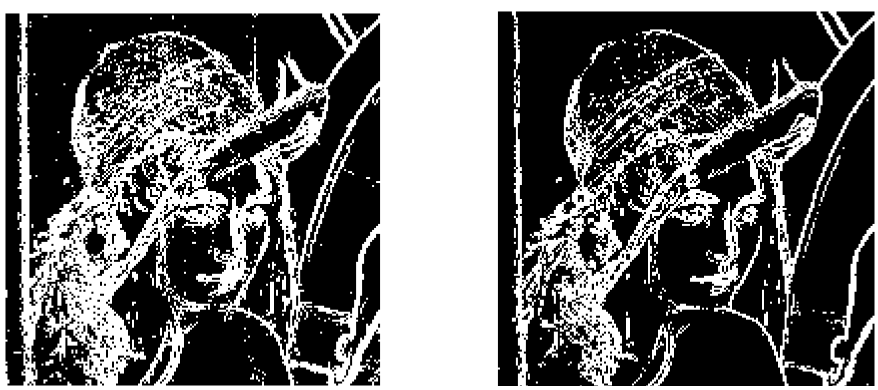
The gradient matrix of the matrix A is then

$$G = \begin{bmatrix} 5.8310 & 7.2111 \\ 7.6158 & 4.4721 \end{bmatrix} \tag{3.10}$$

After finding the gradient of the matrix, a threshold is used to identify the points that are to be considered and the points that are to be suppressed of the image. In order to show the effect of edge detection on the images, the well known image 'lena.jpg' is going to be used and Sobel filter will be applied. Figure 3.11 shows the gradient image of Lena using Sobel approach. The gradient image is not clear until a suitable threshold is applied to focus the pixels with high values and reject the gradients of low value. After applying threshold of suitable value, the image can have distinctive pixels like those in Figure 3.12.

(a) Lena RGB image (b) Gradient image

Figure 3.11: Lena image gradient using Sobel



(a) Threshold = 40 (b) Threshold = 60

Figure 3.12: Sobel thresholding of gradient image

Sobel edge detection method was applied on the training and test images of our system as shown in Figure 3.13. The figure shows the Sobel result with two different threshold values. The use of the threshold of 60 has given better results for this image; however, the result may differ from image to another based on the contents of the image.

(a) Normal image

(b) Infected image

Figure 3.13: Application of Sobel operators on training and test images

3.3.8 Image normalization

As the images used are 8-bit images, the value of each pixel is represented as a concentration value between 0 and 255. The ANNs are better used with values whose absolute value is always less than 1. For this reason, it is preferred to normalize the used training and testing images such that they are comprised in the range (0, 1). This task is accomplished by applying the normalization formula on each image before being used in the neural network. The normalization is applied on the images after the end of all image processing steps like filtering and edge detection. The obtained values are then fed to the neural network in double format for better results. The easiest way of normalization of an image is the division by the value of 255; this is a linear function that keeps constant ratio space between the original gray images after normalization. The normalized version of an 8-bit image is formed by:

$$Normalized \quad image \quad = double(Normal\ image)/255 \tag{3.11}$$

The division by 255 is done because the images are 8 bit resolution images whose pixel values are between 0 and 255. The normalisation is done to map the pixel values to the range between 0 and 1. After normalizing all the images, each one of the images will be converted to a vector by arranging image columns in one-dimensional column. The new vector is going to be assembled with all other image vectors preparing to be submitted to the network structure. The images will be fed to the network in the form of one grand matrix that will contain all training images. Practical application of the ANN on the processed images in addition to the results will be discussed later in this thesis work.

CHAPTER 4

ARTIFICIAL NEURAL NETWORKS

The artificial neural networks abbreviated (ANN) are the planned structures to be implemented in our work. They are going to be used to overcome the limitations in the normal human based classification methods and to test the capability of automated structures in solving classification problems. A part of this work will be also concerned by the development of a semi-automated process for the training of artificial network. In this chapter, the basic concepts of the artificial neural network technology will be introduced and discussed. A semi-automatic ANN recognition system will be developed in this work as well. That is, the chapter will discuss the main structure of artificial neural network and its basic components. Different types of ANN structures will also be presented and discussed. Training process of artificial neural networks and its mathematical development will also be presented and studied in this chapter.

4.1 Main Concepts of ANN

The brain in humans is a system apt of making decisions. It is constructed of a huge collection of smaller neurons that create complex networks in structure and function. The human brain is unique in its very high speed that is incomparable with any fast computer. It is able to handle complicate problems in a simple way and minimum effort. Very hard tasks like images recognition or classification and person identification can be processed in the human brain with the least effort. Logical thinking is also a complicated process that is carried out by the human brain in an autonomous unbelievable simplicity and speed (Bishop, 1995).

The human brain in believed to contain different layers of complex neural structures that interconnect with each other in parallel structures. This signifies that each one of the neurons is able to handle inputs from all the neurons of other layers that have direct connection with its layer. Its output signals are also sent to all the neurons of the next layers. It is interconnected in a manner such that information is passing in all directions all the time and processed by all the neurons of the layer. The continuous processing forward and backward of information between the different layers of the brain creates image of these information in the form of memory. This memory can affect the processing of the next incoming information by providing the suitable decision according to the form of the information. The brain is still more complicated than the explanation can cover or imagine. It has a very powerful capability of solving problems and making hard decisions with high precision. The main property of the brain that gave it its uniqueness is its ability to gain the experience and learn new things. Humans can learn from their own experiences or even from others experiences and can create new information in a simple way (Clabaugh, Myszewski, & Pang, 2000).

All the above mentioned capabilities of the human brain led to the development of systems that can mimic the brain structure in an artificial way. The creation of a machine that can think has occupied an area in the literature of humans since long time. However, it was not until the revolution of the industrial technologies that opened the doors for such dreams to become true. The artificial neural networks appeared and evolved during the second half of the 20th century and the first decades of this century. Nowadays, these structures are gaining more and more interest in the world of science. These structures were given the name of artificial neural networks.

ANNs are genius mathematical models supported with complex mathematical algorithms. They consist mainly of three parts: Input layers, Output layers, and hidden or intermediate layers. The input layers are the receiving parts of the network. The heart or core of the neural network is the hidden layers part. This part is more flexible than the other parts as it can contain one or more layers with any number of neurons. It is separated from the inputs and outputs. All mathematical operations happen inside the neuron that process inputs and generate correspondent outputs of the network. Like the way a real biological system in the brain, the hidden layer neuron obtains and conveys a set of numbers from the precedent layer to the coming layer, respectively. The output values of the neuron differ from the inputs according the given weight of the neuron. The process of passing the information from one layer to the next until finding the output of the neural network is known as the feed forward. An example of the feed forward neural network is present in the Figure 4.1.

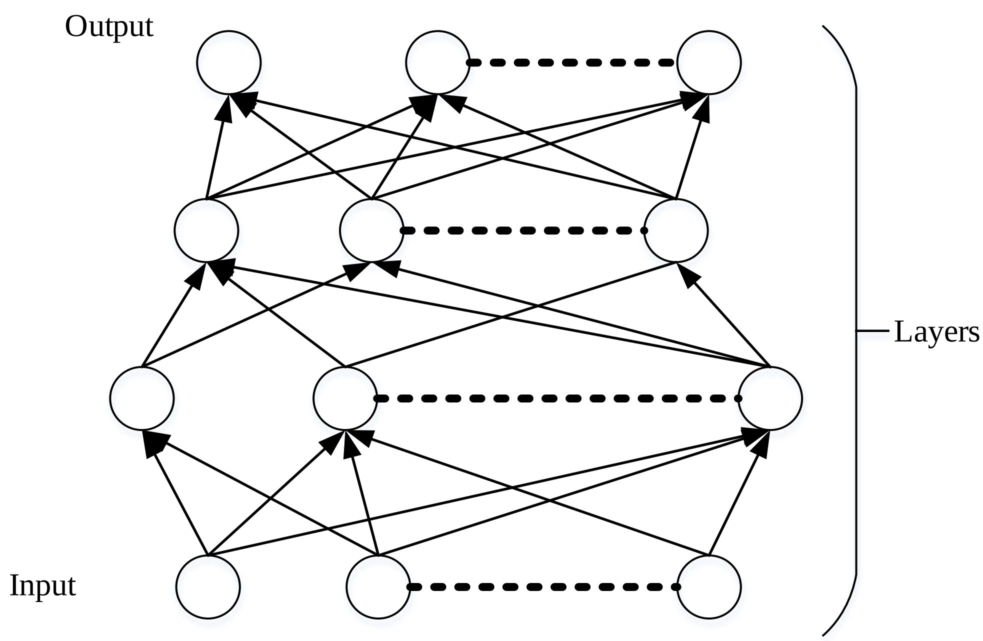


Figure 4.1: Feed Forward neural network example

In Figure 4.1, the network input vector and output vector are represented. The neural network is an example of optimization process. The weights of the neurons are the design values that need to be searched. The minimization function is generally the mean squared error function for most types of neural networks. It is defined by the difference between the actual output of the network and the exact output expected from that network. The general optimization formula of the network can be given by (Gupta, 2006):

$$\begin{aligned} &Find: \; W \in R^{n} \\ &To\,\mathrm{min}imize \;\; MSE = \sum_{i=1}^{N} (T_i - O_i)^2 \end{aligned} \tag{4.1}$$

Where; W refers to the weight matrix of the neural network, R is the group of real numbers, T is the target output vector, while O is the actual output vector of the neural network. Nowadays, there exist many models of the artificial neural networks. The basic ones that consist of single neuron and the very complex neurons exist. The complexity of a neural structure is a function of the application in which it is implemented. In some applications, huge and complex networks are required while in others, simple neural networks can do better job. The simplicity of a problem varies according to the number of outputs and inputs of the network whose relations are to be learned.

4.2 Types of Artificial Neural Networks

There are many different styles of neural networks that were developed to be implemented in various applications. Sometimes, same application uses different types of ANN to solve the same problem in order to optimize the results and increase the efficiency. The selection of the appropriate network, network size, training functions, and parameters is very critical on the efficiency of the system. The selection of these parameters is done after various experiments until finding the best for each parameter of the above mentioned. In this part, main types of artificial networks are going to be explored and discussed.

4.2.1 Feed forward networks

Feed forward artificial network like the one presented in Figure 4.2, is merely of the commonly used and the first invented shape of neural networks. Inputs of the network are enveloped in the input layer. The inputs reach the hidden layer throughout the weights of neurons. Hidden neurons are shown in the form of circles including transfer functions. The output of hidden layers reaches the output layer through its weights and passes also from transfer functions. In general, there are different types of transfer functions that will be discussed later on (Bishop, 1995).

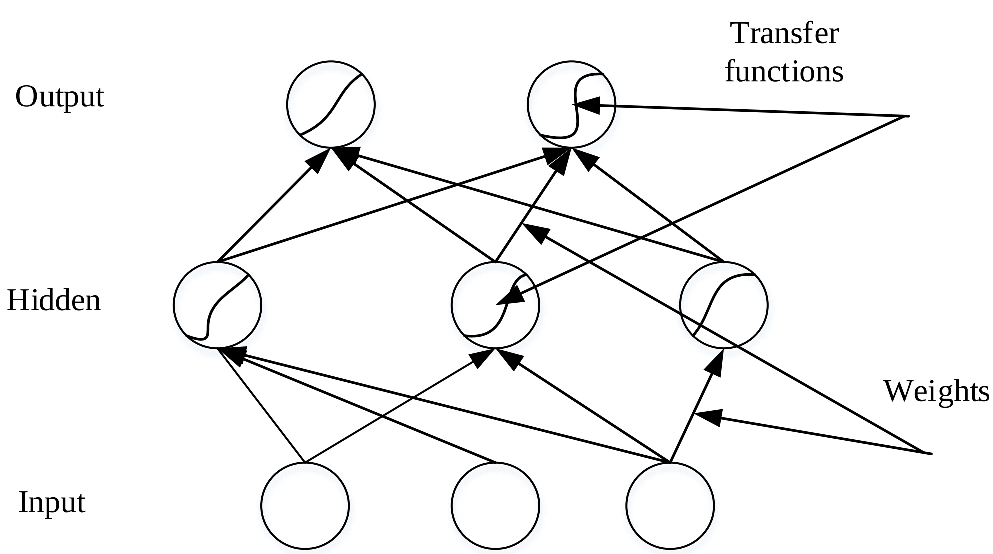


Figure 4.2: Feed forward network structure

Feed forward neural networks are widely implemented because of its efficiency both in classification and regression applications. It can also be used in pattern recognition applications with good performance. It has different advantages such as the generalization of systems for any inputs even those that are not part of the training elements. Once the training of the FFAN is finished, the network will be able to generate proper output for inputs that are not existent in the training set. The limitations of such networks reside in the time issues and memory consumption. In some situations, the network can be misled to inaccurate results because of local minimum situation (Anderson & McNeill, 1992).

4.2.2 Radial bases neural networks

Radial bases neural network is another type of artificial neural network that is widely implemented in different fields and applications. In addition to the output layer and input layer in this network, it consists of an additional hidden layer. Figure 4.3 presents the general form of the RBNN structure.

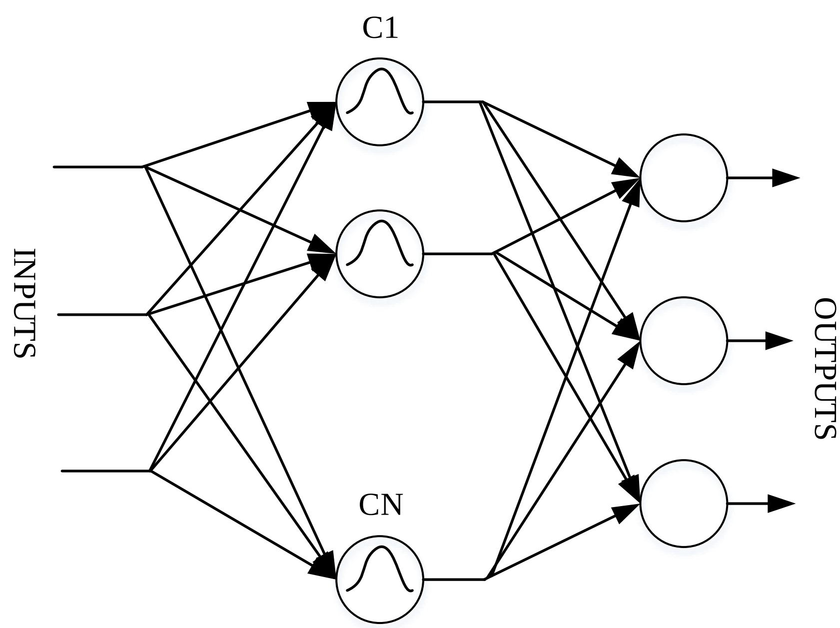


Figure 4.3: Radial basis neural network

The training idea of the RBNN is based on a minimization problem like that of the feed forward neural network. The MSE is to be reduced to a minimum value through continuous iteration and weight update. The weight updating process is looking for optimum values of the weight matrix that generates the suitable output for the training input.

RBNN provides many advantages that make it very powerful type of neural networks in different fields. It is accurate in finding the best results for a given set; it also doesn't suffer the problem of local minimum, the time and memory consumption issues are also eliminated in this type of network as it uses less number of layers and weights (Bataineh, 2012).

4.3 Formulation of the FFAN

The feed forward ANN has a simple processing idea that will be presented in the next part of this chapter. This type of the neural network combined with the training process called back propagation is going to be implemented in the work of this thesis. The FFAN shown in Figure 4.4 has d inputs and c outputs. The output of the hidden layer j is found by calculating a weighted output of the d values and adding biases to the result. The formula can be expressed by:

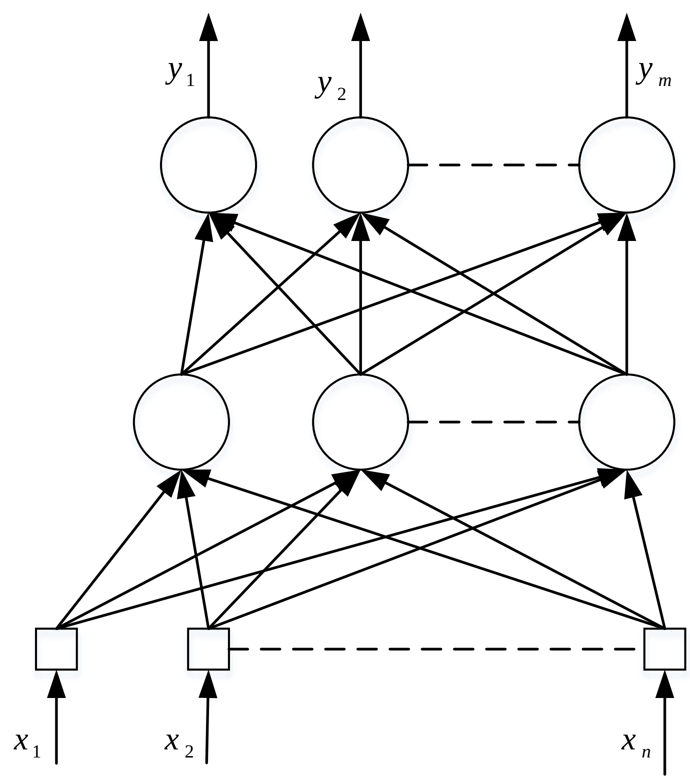


Figure 4.4: FFAN with n inputs and m outputs

$$a_j^n = \sum_{i=1}^{d} x_i \omega_{ji}^n + b_j^n \tag{4.2}$$

Where, the notation n signifies the layer number, ω is the weight matrix of the layer n and x is the input vector of the concerned layer while b is a vector of bias values added to the network to increase stability. the obtained values of this formula need to be passed through a transfer function that reshapes the outputs and give the final output value of the layer. Transfer functions have different forms like hard function, linear function, sigmoid function, bell function and others. The output of the layer is then given by:

$$o_j = f\ (a_j^n) \tag{4.3}$$

4.3.1 Transfer functions

Activation or transfer functions are mathematical equations that are implicated in making a decision on how the signal will be processed in a network. The transfer function decides whether the correspondent neuron is activated or deactivated based on a threshold value.

In the applications of the artificial neural networks, there exist many types of transfer functions that are applied on the weighted inputs of the network. These functions differ in the shape of the curve, the output range, and the slope. Some of the transfer functions are logical that have two states like the hard limit functions. Other transfer functions are continuous and can have infinite number of outputs in a given range like linear functions and sigmoid algorithms.

When deciding the type of transfer function one must take care of different factors. In the case of multilayer neural networks, the use of linear functions is preferred to be avoided because relations are non-linear. The stability of the network is another important aspect as the signals are summarized. The value of the output can increase and reach infinite numbers affecting the overall stability of the network. This is to be avoided by using limited versions or saturation of the transfer functions. In most of applications, it is recommended to make use of sigmoid functions that are limited naturally and give smooth output curves. Another important and basic concept that needs to be considered especially when using back propagation training algorithm is the continuity of the transfer function. These functions should be continuous and derivable over the definition period to accomplish the requirements of the training algorithm.

4.3.1.1 Hard limit functions

Hard limits are of the first types of transfer functions that were used due to the simple output form. They decide whether the output exists or not. Sometimes gives a level of the output among two different levels. Generally, hard limits can have the outputs 0 and 1 based on a threshold function. The output of a threshold function is at one level if the weighted sum of inputs is higher than the threshold, it goes to the other level if this condition in not fulfilled.

4.3.1.2 Linear Transfer function

Linear transfer function is a function where the output is a linear curve of the weighted sum of the inputs. The slope of the curve controls the outputs and decides the output variation following the variation of the input. Some types of linear functions are limited by saturation values. The saturation prevents the output from exceeding an upper and lower limit that is set by user.

4.3.1.3 Sigmoid transfer function

The outputs of such functions are smooth and limited from both sides by upper and lower limits. These types of functions are derivable and continuous in the period of the inputs. The slope of these functions can be controlled until their output can be made sharp like the hard limit functions. A tangent sigmoid function is ranged between (0, 1) and can be given by:

$$f\,(a) = \frac{1}{1+e^{-a}} \tag{4.4}$$

Another function can be implemented as a transfer function in artificial neural networks that uses the (tan h) mathematical function. The hyperbolic tangent ranges between -1 and 1 and is defined by:

$$f\,(a) = \frac{e^{a} - e^{-a}}{e^{a} + e^{-a}} \tag{4.5}$$

The main advantages of these functions reside in the ease of their implementation, their smooth output, and the simple calculation of their derivatives.

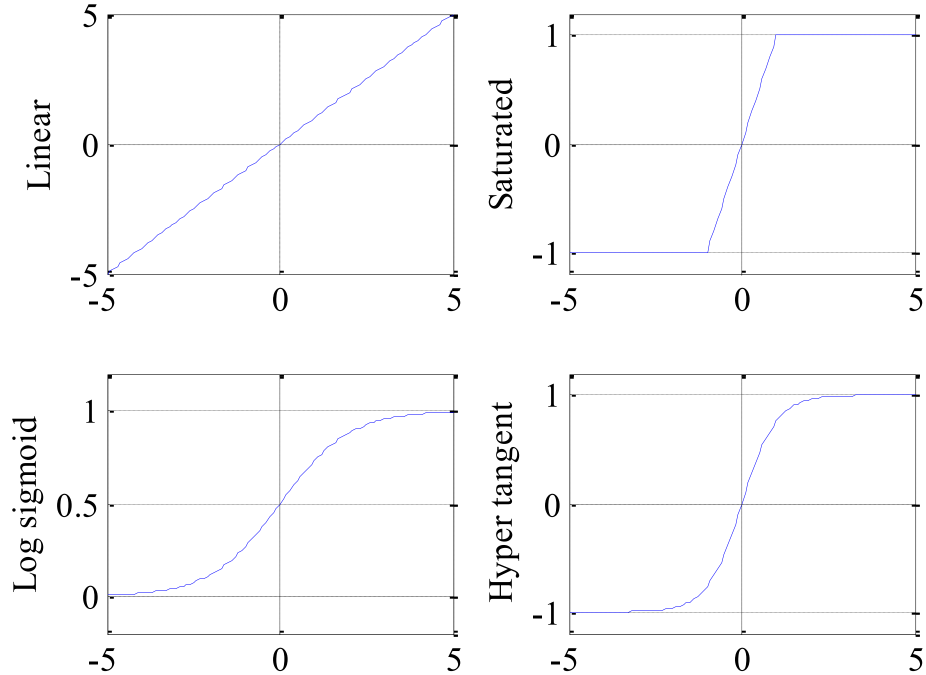


Figure 4.5: Shape of different transfer functions

4.3.2 Back propagation learning and error back propagation

The learning ability of the artificial neural networks is amazing and it gives it all its huge functions in different fields. The learning procedure is completed by regulating the weight values systematically. These weight adjustments are done to control the way in which the network processes its inputs. It is important to find an accurate way to adjust the weights in a manner that help reducing the output errors. The main purpose of any learning process is to lead the network to generate desired outputs and converge toward them. The selection of the right way for training a network and the suitable tuned parameters is a stone key in the success of the process.

The training process is carried out based on predefined set of input output data used to examine the convergence of the network. These sets of data are called training data. The training data should be chosen in the best way to represent all variations of the data. This process is classified in the supervised learning processes. The supervised learning is the learning in which a teacher is required. The teacher is someone who has the means of controlling whether a lesson or a pattern was correctly understood and can be applied for different situations.

The back propagation is an example of the supervised learning of multi-layer neural networks. It is one of the oldest learning algorithms that have its advantages and drawbacks. However, its simple implementation and ease of use have given it a very vast area of applications. The learning procedure involves the use of an error function to give a measure of the amount of convergence of the network toward its desired version. By using this error, the weights of the different layers can be easily adjusted to suit the needs for convergence. The mean squared error function is generally implemented in the neural networks back propagation as it offers the possibility of finding the best way to reduce the error.

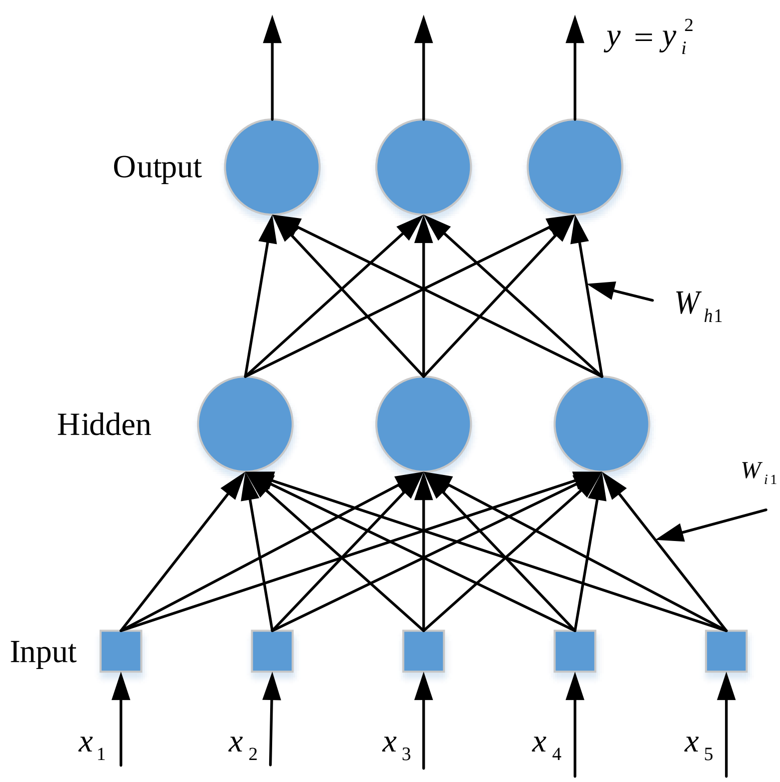


Figure 4.6: Supervised BPNN learning network

Figure 4.6 presents a simple neural network that consists of one input layer, one hidden layer, and one output layer. The input layer is a 5 neurons layer; the hidden contains 3 neurons while the output has 3 neurons. The next equations describe the outputs of each one of the layers.

$$y_i^2 = \sum_j W_{ji}^2 y_j^1 + b_i^2 \tag{4.6}$$

$$y_i^1 = f\,(\sum_j W_{ji}^1 x_j + b_i^1) \tag{4.7}$$

Where; the term x is used to describe the input of the respected layer, W is the weight matrix associated to that layer, b is the bias vector and y is the output obtained from that layer. After the final output is found, the error can be calculated using the formula:

$$e^k = (T_1^k - y_1^k)^2 + (T_2^k - y_2^k)^2 + (T_3^k - y_3^k)^2 \tag{4.8}$$

Where; k is the iteration number. This formula can be generalized for a network with *N* outputs to be:

$$e^k = \sum_{i=1}^{N} (T_i^k - y_i^k)^2 \tag{4.9}$$

Where; *T* and *y* are used to describe the desired and actual output of the neural network. This error is calculated and back propagated to the previous layers in order to be used in updating their weights. The implementation parameters and structures of this algorithm are going to be presented and discussed in the discussions of the methods of the thesis.

CHAPTER 5

RESULTS AND DISCUSSIONS

5.1 Introduction

This chapter is designed to discuss the results of the proposed system's efficiency. The work of this thesis was designed to compare the efficiency of the neural network detection of gingivitis based on different types of image filtering and processing techniques. Four types of image filters are going to be employed separately on each one of the images. The filtered image of each step is going to be associated with a neural network structure. Wiener filter, two dimensional order statistics filter, median filter, and entropy filter were used in this work for the filtering of the database images. Images were also treated in different ways to examine the effect of different types of image processing techniques on the performance of the artificial neural network.

In the first section of this chapter, the performance of the system processed using Wiener filter is going to be presented.

The neural network will be investigated under the same parameters for all different types of filters to establish the comparison between the results. The network parameters are going to be used as illustrated in the Table 5.1. Two hidden layers neural network structure was used for the training of the system. The output of the system was coded using two binary values of 01 and 10 in the form of column. The tangent sigmoid function was chosen to be used in the training of the system for its performance and good results.

Table 5.1: The parameters of the neural network

Parameter	Value	Parameter	Value
Input layer size	2500	Out Transfer function	tangent
Output layer size	2	Hidden 1 transfer	tangent
Hidden layer size	250, 250	Hidden 2 transfer	tangent
Moment factor	0.2	MSE goal	0.004
Learning rate	0.01	[infected] [healthy]	[1; 0] [0;1]

5.2 Training of Raw Images without Filtering

The images contained in the dataset will be presented to the neural network in the training and test with no prior filtering of these images. The images were provided to the network after the gray scale conversion and size reduction directly. The training of the images has given a training performance of 91.25%. 73 images of the 80 training images were classified correctly during the training. 73 images out of the 80 test images were also classified correctly with a performance of 91.25% as well. Figure 5.1 presents the raw image in its small size before being fed to network.

Figure 5.1: Raw images used in the training

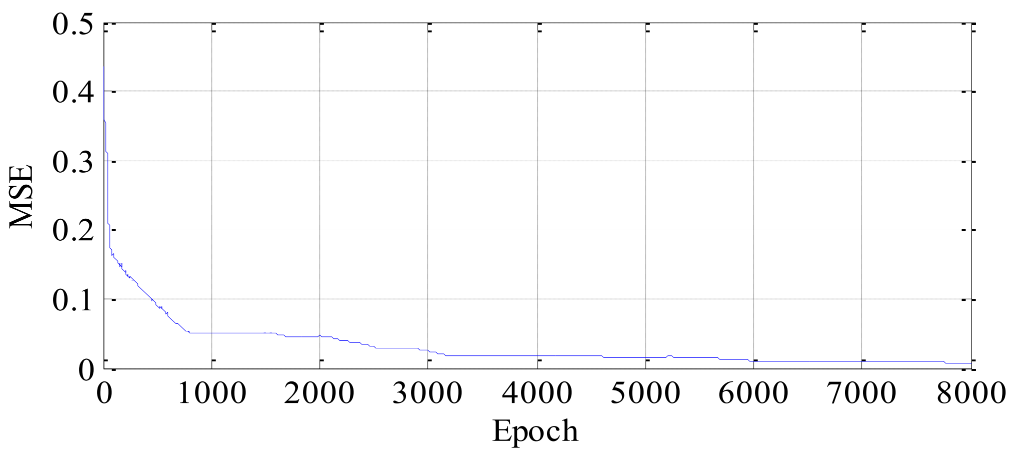


Figure 5.2: MSE curve of the training (Raw image training)

Table 5.2: Training results sample of ANN (Raw images)

0,856	0,031	0,424	0,855	0,387	0,033	0,957	0,567	0,793	0,121
0,125	0,985	0,502	0,251	0,692	0,973	0,024	0,586	0,143	0,895
0,971	0,010	0,333	0,111	0,948	0,037	0,919	0,153	0,998	0,974
0,034	0,982	0,384	0,917	0,066	0,970	0,085	0,745	0,001	0,024

0,829	0,372	0,921	0,035	0,880	0,135	0,939	0,109	0,972	0,011
0,184	0,551	0,090	0,984	0,085	0,878	0,045	0,934	0,027	0,995
0,410	0,117	0,826	0,088	0,366	0,032	0,891	0,062	0,971	0,122
0,841	0,877	0,140	0,945	0,617	0,977	0,132	0,939	0,026	0,872
0,893	0,024	0,073	0,038	0,021	0,188	0,847	0,001	0,444	0,084
0,093	0,973	0,963	0,968	0,883	0,829	0,151	0,995	0,249	0,930
0,425	0,016	0,906	0,230	0,913	0,189	0,813	0,042	0,408	0,886
0,666	0,992	0,104	0,886	0,078	0,824	0,172	0,936	0,433	0,200
0,132	0,102	0,769	0,797	0,587	0,116	0,981	0,380	0,926	0,008
0,803	0,873	0,247	0,417	0,578	0,893	0,024	0,825	0,099	0,991
0,956	0,003	0,132	0,105	0,296	0,232	0,064	0,702	0,919	0,009
0,033	0,997	0,700	0,883	0,378	0,746	0,739	0,518	0,063	0,990

Table 5.2 and Table 5.3 illustrates the results of training and test of the ANN using raw image files. Tables show that most of the training and test images were recognized correctly. The training MSE curve is presented in Figure 5.2.

Table 5.3: Test results sample of the Raw images

0,963	0,102	0,886	0,287	0,716	0,141	0,977	0,078	0,937	0,021
0,035	0,873	0,107	0,808	0,099	0,843	0,050	0,908	0,051	0,981
0,921	0,055	0,903	0,289	0,722	0,100	0,945	0,860	0,989	0,103
0,081	0,820	0,061	0,472	0,238	0,894	0,056	0,405	0,014	0,898
0,989	0,013	0,989	0,002	0,989	0,287	0,989	0,060	0,989	0,074
0,014	0,980	0,014	0,999	0,014	0,705	0,014	0,914	0,014	0,828
0,989	0,062	0,989	0,019	0,989	0,018	0,989	0,016	0,989	0,384
0,014	0,938	0,014	0,995	0,014	0,983	0,014	0,975	0,014	0,297
0,989	0,020	0,989	0,001	0,989	0,392	0,989	0,036	0,989	0,106
0,014	0,989	0,014	0,970	0,014	0,317	0,014	0,895	0,014	0,911
0,989	0,054	0,989	0,019	0,989	0,012	0,989	0,092	0,989	0,018
0,014	0,905	0,014	0,989	0,014	0,999	0,014	0,859	0,014	0,981
0,989	0,690	0,989	0,016	0,989	0,080	0,989	0,155	0,989	0,003
0,014	0,740	0,014	0,988	0,014	0,986	0,014	0,884	0,014	0,998
0,989	0,016	0,989	0,033	0,989	0,018	0,989	0,010	0,989	0,007
0,014	0,988	0,014	0,973	0,014	0,981	0,014	0,990	0,014	0,988

5.3 Wiener Filter Processing Technique

In this part, the images of database were treated and filtered using an adaptive Wiener filter. Adaptive Wiener filter was applied to remove noise from the database images and simplify the neural network task. Figure 5.3 presents the wiener-filtered image. Figure 5.4 below confirms the neural network-training tool of MATLAB during the training of the network. The figure illustrates the two hidden layers and the output layer of the network.

Figure 5.3: Images filtered using Wiener filter

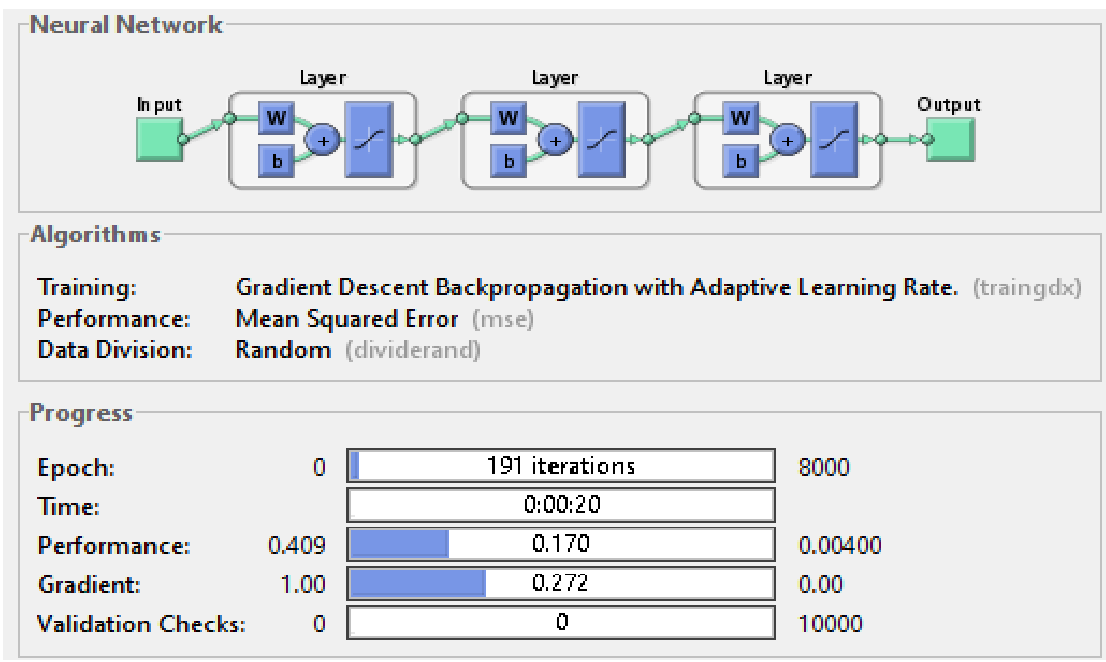


Figure 5.4: Training tool of artificial neural network

After the separation of results and comparison with the expected targets of training and test images, it was found that 98.75% of the training images were correctly classified. This means that the program was able to correctly identify 79 images out of the 80 training images. This rate was less in the test set of images. 77 images out of the 80 training images were correctly classified. The test rate was approximately 96.25%. Figure 5.5 illustrates the evolution of the MSE over the iteration number during the training. The MSE was decreasing during the training until the set goal of 0.00388 was reached after 1280 iterations. The elapsed time during the training of this set of images with the network was 56 seconds.

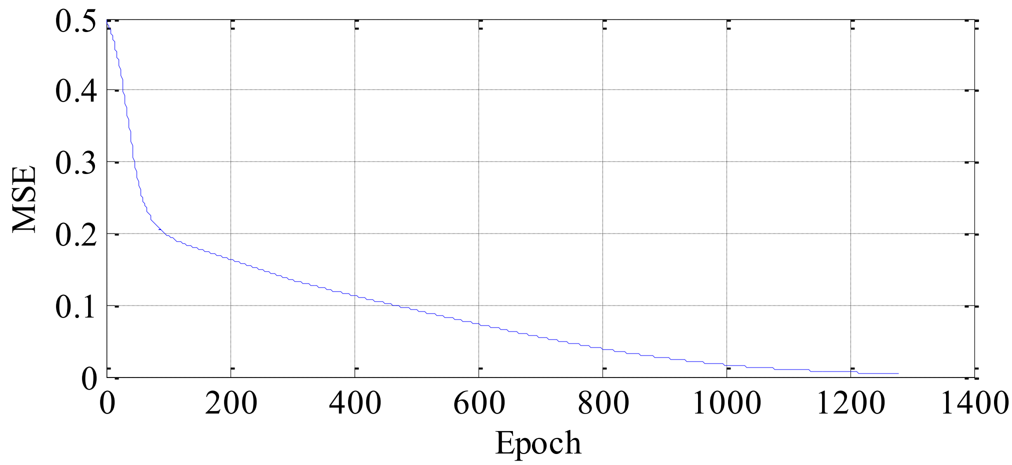


Figure 5.5: Evolution of the MSE over training epochs

5.4 Median Filter Processing Technique

The processing of data base images in this part was carried out with help of median filter type. The median filter was applied using a moving window of 9 elements (3x3). Median filter is known for its performance in the rejection of pulsed noise. Any strange pixels of the image are being easily removed from the images using median filter of two dimensions. After applying the median filter in addition to the other image processing techniques mentioned earlier in this work, the processed images were all passed through the phase of training and test of the artificial neural network. The same parameters as those used in the previous section were applied in this section. The training of the system took around 2 minutes and 30 seconds to converge to the present MSE of 0.004. This goal was met after a total of 2937 training iterations. Figure 5.7 demonstrates the curve of the mean squared error during the training of the network in this configuration. The training results of the network shows that 78 images out of the 80 training images were recognized successfully. The training rate obtained using this configuration was approximately 97.5%. Figure 5.6 shows the image after being filtered using Median filter.

Figure 5.6: Median filtered image

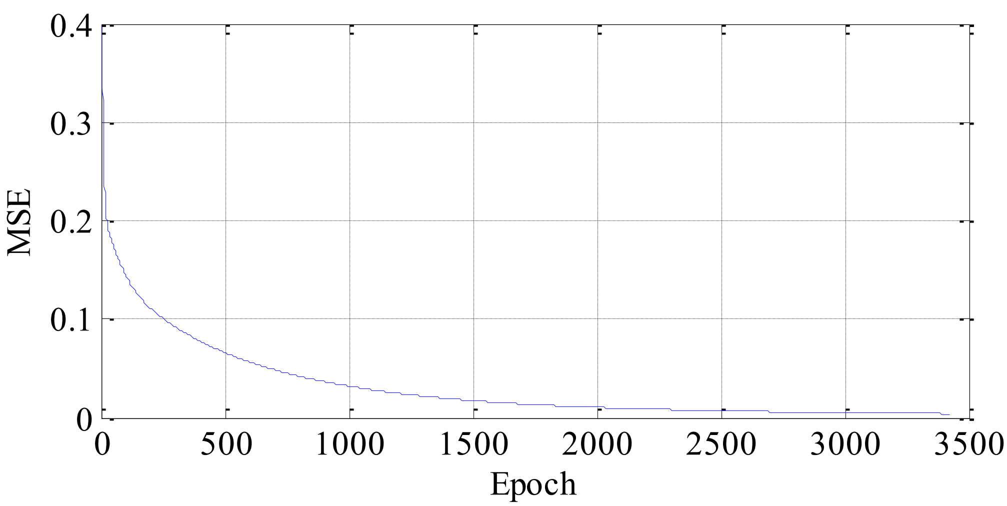


Figure 5.7: Evolution of the MSE curve in the training

From the test results, it can be demonstrated that 73 images of the test set were classified correctly as they were expected. The test rate of this configuration was 91.25%. The obtained results prove that the network generalization was satisfactory using this configuration.

5.5 Entropy Filter Processing Technique

The training of the neural network structure was carried out using the same parameters used in the previous two sections. Database images were all filtered using entropy filter. The training of this structure has shown low efficiency in terms of mean square error and convergence speed. The training took longer time and iterations during the training than expected. Training time was approximately 340 seconds without reaching the expected MSE value. Training of the system was stopped after 8000 epochs at an MSE of 0.0132. After the end of the training process, the training output was obtained as result for the training data set. It is found that this data set has good performance with considerable errors from the target output. Output error of 0.551 was observed in the output of the network. From the results, it was found that 77 images out of the 80 training images were accepted to be true with a training performance of 96%. 3 images of this dataset were not correctly classified and there for rejected during the result observation.

From the test results, 72 images out of these test images were correctly classified with performance of 90%. 8 outputs weren’t accepted as the error between them and the targets was large. The training mean squared error during the training of the system is shown in Figure 5.8. The curve is showing good decreasing form with low rate. The final achieved MSE value was not as expected because the maximum epoch value was reached.

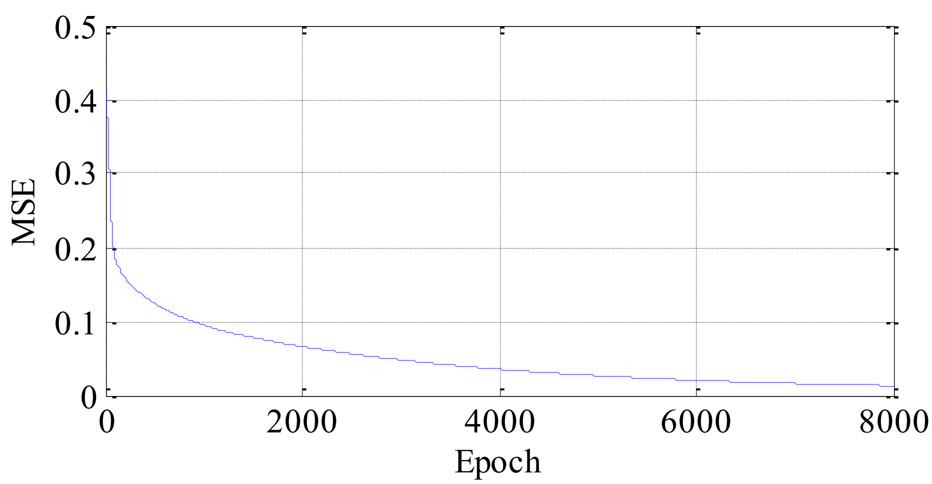


Figure 5.8: MSE curve development over the training

5.6 2D Order Statistics Filter Processing Technique

The application of ANN on the images treated using 2D order statistic filter was carried out in this part of the thesis. The output of the ANN for the training and test datasets was obtained and analysed. Figure 5.10 illustrates the curve of the MSE error obtained during the training. Statistics of the results and MSE show that the training reached 8000 iteration in 352 seconds of time. The obtained MSE value was 0.0102, which is higher than the desired value. Training performance of this experiment was 96.25% while test performance of 92.25% was obtained.

Figure 5.9: Statistic order filtered image

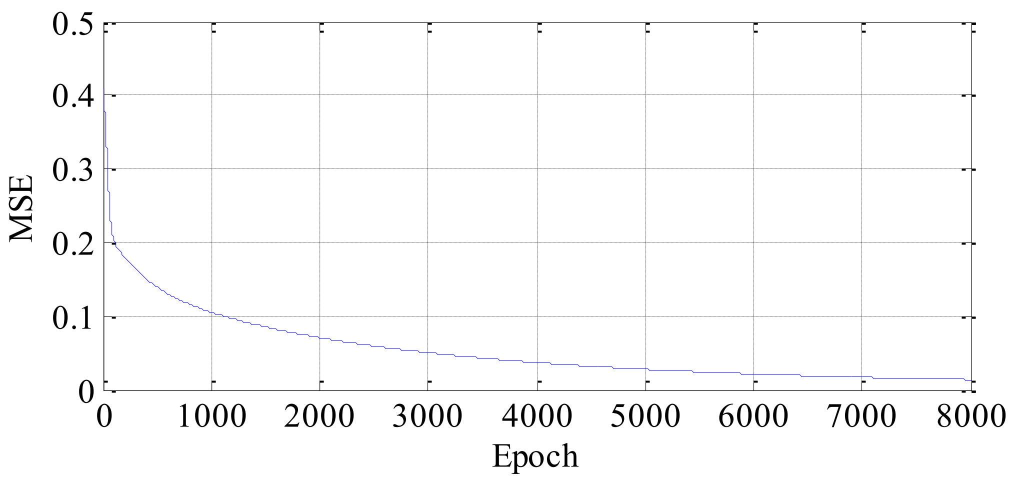


Figure 5.10: Curve of the mean squared error

5.7 Comparison of the Obtained Results

After training the system using different image processing and filtering techniques, the obtained results showed that the use of different types of filters have slight effect on the training of the neural network. Table 5.4 presents a comparison between the results of different configuration of the neural network and image processing. The comparison between the different obtained results reveals that the image filtering using Weiner filter has given the best results during the training and the test of the neural network. As seen from the table that the training time of the Weiner based configuration was the minimum with 56s of training time. The median filter has come the second in terms of training time followed by raw images in the third class. Entropy filter and statistic filter have been the last with 340s and 352s.

From the point of view of MSE, the Weiner filter and Median filter also was the first as they both reached the desired MSE goal. Weiner configuration has reached the goal in 1280 iterations while Median filter configuration has reached that value in 2937 iterations. The other three configurations did not reach the desired goal after the end of the maximum iteration numbers. Comparison of training performance of the five configurations shows that the best training performance was achieved in Weiner configuration with 98.5%. Median was the second with performance of 97.5% followed by the statistic filter configuration with 96.25%. Entropy filter was the fourth with performance of 96% while raw images configuration did not exceed the 91.25% performance.

Table 5.4: Comparison of the obtained results

	RAW image	Weiner	Median	Entropy	2D Statistic
Time (s)	306	56	150	340	352
Iterations	8000	1280	2937	8000	8000
MSE	0.007	0.004	0.004	0.0132	0.0102
Train perf.	91.25%	98.5%	97.5%	96%	96.25%
Test perf.	91.25%	96.25%	91.25%	90%	92.25%

Test performance of the different configurations shows that the Weiner filter was the best configuration with 96.25% performance. The nearest performance as obtained using statistic filter configuration and reached a performance of 92.25%. Median filter and raw image configurations have shown similar results with 91.25% test performance while the use of Entropy filter have given the least test performance with 90%.

CHAPTER 6

CONCLUSIONS AND FUTURE WORKS

The work presented in this thesis proposes the use of an image processing approach combined with artificial neural network structure in the classification of Gingivitis teeth disease. The studied algorithm investigates in the back propagation neural technique and ANN for the classification of the disease. Different image processing techniques were used in order to increase the efficiency of the proposed system. Database composed of 160 images was implemented in the system to validate and test the different configurations of the used system.

The proposed system uses a series of image processing techniques such as scale conversion, averaging process to reduce image size, segmentation of the images, and wavelet transform to extract image features.

This work has studied five different configurations to evaluate the performance of the ANN classification system under different conditions. These configurations use different types of filters to validate the performance of the system and find the best configuration. The five configurations include using raw images in the training and test of neural networks, the use of Weiner filter, Median filter, Entropy filter, and Statistic filter in the processing of images. The results obtained for each one of these five configurations were obtained and discussed.

The comparison of the different obtained results revealed that the training of the neural network for training of raw images has given a performance of 91.25% in the training and test of the neural network. The investigation of the Entropy filtering method has increased the training performance to 96% while decreased the test performance to 90%. 2D statistic filter has given a performance of 96.25% in the training and 92.25% in the test. Median filter has given better performance during the training with 97.5%. The best results of training and test were obtained using Weiner filter for the filtering of images. It has given a training performance of 98.5% and test performance of 96.25%.

The use of Weiner filter has proven higher performance in terms of training time as training time of Weiner configuration was 3 times faster than the Median filter configuration and 6 times faster than raw images configuration. From the point of view of the mean squared error, the Weiner filter and Median filter have given the best results and the best MSE values. The next of filters have given higher level of MSE within the time and iteration limits. The use of artificial intelligence for the detection and classification of gingivitis can be very useful for dentist. It introduces the modern technologies instead of the traditional methods used for the gingivitis detection. As future plans, the implementation of other neural network structures like deep learning neural network with massive and huge amount of data is proposed.

REFERENCES

Anderson, G., & McNeill, D. (1992). *Artificial Neural Networks Technology*.

Bataineh, M. (2012). *Artificial neural network for studying human performance*. University of Iowa.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Clarendon Press.

Clabaugh, C., Myszewski, D., & Pang, J. (2000). Neural Networks. Retrieved May 1, 2017, from http://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/index.html

Clinic, M. (2017). Gingivitis. Retrieved March 10, 2017, from http://www.mayoclinic.org/diseases-conditions/gingivitis/home/ovc-20305807

Fechner, T. (1993). Nonlinear noise filtering with neural networks: comparison with Weiner optimal filtering. In *Third International Conference on Artificial Neural Networks* (pp. 143–147). IET.

Gonzalez, R. C., & Woods, R. E. (2001). *Digital Image Processing* (2nd Editio). New Jersey: Prentice-Hall.

Gupta, C. (2006). *Implementatıon of Back Propagatıon Algorıthm (of Neural Networks) In VHDL*.

Helwan, A. (2014). Iris Tumor Detection System Using Image Processing Techniques. *International Journal of Science and Engineering Research*, *30*, 75–80.

Khajwaniya, K. K., & Tiwari, V. (2015). Satellite image denoising using Weiner filter with SPEA2 algorithm. In *9th International Conference on on Intelligent Systems and Control* (pp. 1–6). India. https://doi.org/10.1109/ISCO.2015.7282324

Mohan, R., Mridula, S., & Mohanan, P. (2016). Speckle Noise Reduction in Images using Wiener Filtering and Adaptive Wavelet Thresholding. *IEEE*, 2860–2863. https://doi.org/10.1109/TENCON.2016.7848566

Nagane, K., Dongre, N., Dhar, A., Jadh, D., & Burghate, B. (2016). An Analytical Study for Gum Disease Detection Systems. *International Journal of Science Technology & Engineering*, *3*(5), 94–97.

Papantonopoulos, G., Takahashi, K., Bountis, T., & Loos, B. G. (2014). Artificial Neural Networks for the Diagnosis of Aggressive Periodontitis Trained by Immunologic

Parameters. *PLOS ONE*, *9*(3), 1–8. https://doi.org/10.1371/journal.pone.0089757

Patil, P. B., & Patil, B. R. (2011). Saliva: A diagnostic biomarker of periodontal diseases. *Indian Society of Periodontology*, *15*(4), 310–317.

Stella, A., & Trivedi, B. (2012). Implementation of Order Statistic Filters on Digital Image and OCT Image: A Comparative Study. *International Journal of Modern Engineering Research*, *2*(5), 3143–3145.

Sudheera, P., Sajja, V. R., Kumar, S. D., & Rao, N. G. (2016). Detection of dental plaque using enhanced K-means and silhouette methods. In *International Conference on Advanced Communication Control and Computing Technologies* (pp. 559–563). IEEE. https://doi.org/10.1109/ICACCCT.2016.7831702

Tantua, P. D. (2015). *Intelligent Knee Rheumatoid Arthritis Identification System Based on Image Processing and Neural Classification*. NEU.

Thakur, A., Guleria, P., & Bansal, N. (2016). Symptom & risk factor based diagnosis of Gum diseases using neural network. In *2016 6th International Conference Cloud System and Big Data Engineering* (pp. 101–104). https://doi.org/10.1109/CONFLUENCE.2016.7508095

Wang, J., Peng, Y., Wang, X., & Peng, Y. (2015). Study on algorithm of image restoration based on Stochastic Resonance and Weiner filtering. In *5th International Conference on Electronics Information and Emergency Communication* (pp. 244–247).

Wilder, R. S., & Moretti, A. J. (2016). Gingivitis and periodontitis in adults: Classification and dental treatment. Retrieved March 10, 2017, from http://www.uptodate.com/contents/gingivitis-and-periodontitis-in-adults-classification-and-dental-treatment?source=search_result&search=Gingivitis+and+periodontitis+in+adults&selectedTitle=1~133

Zollitch, C. (2016). Stemmer Imaging. Retrieved March 15, 2017, from http://www.stemmer-imaging.co.uk/en/knowledge-base/grey-level-grey-value/

APPENDIX

PROGRAM LISTING

```
clc
clear all
close all
var.net = 'W';
var.Save = true;
var.size = [512 512];
var.level = 0.4;
var.wnsize = [2 2] ;
var.path = cd ;
var.normal_folder = strcat(var.path,'\Normal\Original\');
var.anormal_folder = strcat(var.path,'\Anormal\Original\');
var.X = dir(var.normal_folder);
var.normal_image_names = var.X(3:end);
var.X = dir(var.anormal_folder);
var.anormal_image_names = var.X(3:end);
cd(var.normal_folder);
pause_Time = 0;

%Read normal images
for i=1:length(var.normal_image_names)
  disp(strcat('Reading the normal image number : ',num2str(i)));
  var.nImage{1,i} = imread(var.normal_image_names(i).name);
  pause(pause_Time);
  clc
end

if(length(var.normal_image_names)<length(var.anormal_image_names))
    for i=length(var.normal_image_names)+1:length(var.anormal_image_names)
       var.nImage{1,i} = var.nImage{1,i-1} ;
    end
end

%Resize normal images
for i=1:100
   disp(strcat('Resizing the normal image number : ',num2str(i)));
   pause(pause_Time);
   var.nImage{2,i} = imresize(var.nImage{1,i},var.size);
   if(var.Save)
       var.nameTosave = strcat([var.path,'\Normal\Size image\']);
       if(exist(var.nameTosave)~= 7)
           mkdir(var.nameTosave);
       end
       if(i <= length(var.normal_image_names))
           var.nameTosave = 
strcat([var.nameTosave,var.normal_image_names(i).name,'.jpg']);
           imwrite(var.nImage{2,i},var.nameTosave,'jpg');
       end
   end
   clc
end
```

```
%Converting images to gray scale
for i=1:100
  disp(strcat('Converting the normal image : ',num2str(i),' to gray
scale'));
  pause(pause_Time);
  if(size(var.nImage{2,i},3)==3)
    var.nImage{3,i} = rgb2gray(var.nImage{2,i});
  else
    var.nImage{3,i} = mat2gray(var.nImage{2,i});
  end
  if(var.Save)
      var.nameTosave = strcat([var.path,'\Normal\gray image\']);
      if(exist(var.nameTosave)~= 7)
          mkdir(var.nameTosave);
      end
      if(i <= length(var.normal_image_names))
          var.nameTosave =
strcat([var.nameTosave,var.normal_image_names(i).name,'.jpg']);
          imwrite(var.nImage{3,i},var.nameTosave,'jpg');
      end
  end
  clc
end

%%Applying filters
for i=1:100
  disp(strcat('Applying Weiner filter to the normal image : ',num2str(i)));
  pause(pause_Time);

  var.nImage{4,i} = var.nImage{3,i};
  % var.nImage{4,i} = medfilt2(var.nImage{3,i});
  % var.nImage{4,i} = ordfilt2(var.nImage{3,i},25,true(5));
  % var.entropy = entropyfilt(var.nImage{3,i});
  % var.nImage{4,i} = uint8(var.entropy*max(max(var.entropy)));

   var.nImage{5,i} = wiener2(var.nImage{4,i},var.wnsize);
   if(var.Save)
       var.nameTosave = strcat([var.path,'\Normal\Wiener image\']);
       if(exist(var.nameTosave)~= 7)
           mkdir(var.nameTosave);
       end
       if(i <= length(var.normal_image_names))
           var.nameTosave =
strcat([var.nameTosave,var.normal_image_names(i).name,'.jpg']);
           imwrite(var.nImage{5,i},var.nameTosave,'jpg');
       end
   end
   clc
end

for i=1:100
  disp(strcat('Applying Sobel edge detection to the normal image :
',num2str(i)));
  pause(pause_Time);
   % Canny edge detection
```

```
    var.nImage{6,i} = 100 * edge(var.nImage{3,i},'Sobel');
    %Threshold the image read from the folder
    bw = im2bw(var.nImage{3,i},var.level);
    var.nImage{7,i} = bwareaopen(bw, 5);
    if(var.Save)
        var.nameTosave = strcat([var.path,'\Normal\Sobel image\']);
        if(exist(var.nameTosave)~= 7)
            mkdir(var.nameTosave);
        end
        if(i <= length(var.normal_image_names))
            var.nameTosave =
strcat([var.nameTosave,var.normal_image_names(i).name,'.jpg']);
            imwrite(var.nImage{7,i},var.nameTosave,'jpg');
        end
    end
    clc
end

for i=1:100
  disp(strcat('Applying Wavelet transform to the normal image :
',num2str(i)));
    %wavelet compression
    cd(var.path);
    var.nImage{8,i} = wavelet(var.nImage{3,i},3,512,9);
    %cd(var.normal_folder);
    if(var.Save)
        var.nameTosave = strcat([var.path,'\Normal\Wavelet image\']);
        if(exist(var.nameTosave)~= 7)
            mkdir(var.nameTosave);
        end
        if(i <= length(var.normal_image_names))
            var.nameTosave =
strcat([var.nameTosave,var.normal_image_names(i).name,'.jpg']);
            imwrite(var.nImage{8,i},var.nameTosave,'jpg');
        end
    end
    pause(pause_Time)
    clc
end

for i=1:100
  disp(strcat('Resizing image number : ',num2str(i)));
    %wavelet compression
    cd(var.path);
    var.nImage{9,i} = double(imresize(var.nImage{3,i},[50 50]))/255;
    %cd(var.normal_folder);
    if(var.Save)
        var.nameTosave = strcat([var.path,'\Normal\Size 50x50\']);
        if(exist(var.nameTosave)~= 7)
            mkdir(var.nameTosave);
        end
        if(i <= length(var.normal_image_names))
            var.nameTosave =
strcat([var.nameTosave,var.normal_image_names(i).name,'.jpg']);
            imwrite(var.nImage{9,i},var.nameTosave,'jpg');
```

```
        end
    end

    pause(pause_Time)
    clc
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

cd(var.path);
cd(var.anormal_folder);
for i=1:length(var.anormal_image_names)
  disp(strcat('Reading the Anormal image number : ',num2str(i)));
  var.anImage{1,i} = imread(var.anormal_image_names(i).name);
  pause(pause_Time)
  clc
end

for i=1:length(var.anormal_image_names)
  disp(strcat('Resizing the Anormal image number : ',num2str(i)));
  var.anImage{2,i} = imresize(var.anImage{1,i},var.size);
  if(var.Save)
      var.nameTosave = strcat([var.path,'\Anormal\Size image\']);
      if(exist(var.nameTosave)~= 7)
          mkdir(var.nameTosave);
      end
      if(i <= length(var.anormal_image_names))
          var.nameTosave =
strcat([var.nameTosave,var.anormal_image_names(i).name,'.jpg']);
          imwrite(var.anImage{2,i},var.nameTosave,'jpg');
      end
  end
  pause(pause_Time);
  clc

end

for i=1:length(var.anormal_image_names)
  disp(strcat('Converting the Anormal image : ',num2str(i),' to gray
scale'));
  if(size(var.anImage{2,i},3)==3)
     var.anImage{3,i} = rgb2gray(var.anImage{2,i});
  else
     var.anImage{3,i} = mat2gray(var.anImage{2,i});
  end
  if(var.Save)
      var.nameTosave = strcat([var.path,'\Anormal\Gray image\']);
      if(exist(var.nameTosave)~= 7)
          mkdir(var.nameTosave);
      end
      var.nameTosave =
strcat([var.nameTosave,var.anormal_image_names(i).name,'.jpg']);
      imwrite(var.anImage{3,i},var.nameTosave,'jpg');
  end
```

```
  pause(pause_Time)
  clc
end

for i=1:length(var.anormal_image_names)
 disp(strcat('Applying Weiner filter to the Anormal image : ',num2str(i)));
  % median filter
  var.anImage{4,i}=var.anImage{3,i};
  %var.anImage{4,i} = medfilt2(var.anImage{3,i});
  %var.anImage{4,i} = ordfilt2(var.anImage{3,i},25,true(5));
  %var.entropy = entropyfilt(var.anImage{3,i}); var.anImage{4,i} =
uint8(var.entropy*max(max(var.entropy)));

  var.anImage{5,i} = wiener2(var.anImage{4,i},var.wnsize);
  if(var.Save)
      var.nameTosave = strcat([var.path,'\Anormal\Wiener image\']);
      if(exist(var.nameTosave)~= 7)
          mkdir(var.nameTosave);
      end
      var.nameTosave =
strcat([var.nameTosave,var.anormal_image_names(i).name,'.jpg']);
      imwrite(var.anImage{5,i},var.nameTosave,'jpg');
  end
  pause(pause_Time)
  clc
end

for i=1:length(var.anormal_image_names)
  disp(strcat('Applying Sobel edge detection to the Anormal image :
',num2str(i)));
  % Canny edge detection
  var.anImage{6,i} = 255 * edge(var.anImage{5,i},'Sobel');
  %
  bw = im2bw(var.anImage{4,i},var.level);
  var.anImage{7,i} = bwareaopen(bw, 10);
  if(var.Save)
      var.nameTosave = strcat([var.path,'\Anormal\Sobel image\']);
      if(exist(var.nameTosave)~= 7)
          mkdir(var.nameTosave);
      end
      var.nameTosave =
strcat([var.nameTosave,var.anormal_image_names(i).name,'.jpg']);
      imwrite(var.anImage{7,i},var.nameTosave,'jpg');
  end
  pause(pause_Time)
  clc

end

for i=1:length(var.anormal_image_names)
  disp(strcat('Applying Wavelet transform to the Anormal image :
',num2str(i)));
  %wavelet compression
  cd(var.path);
  var.anImage{8,i} = wavelet(var.anImage{3,i},3,512,9);
```

```
  if(var.Save)
      var.nameTosave = strcat([var.path,'\Anormal\Wavelet image\']);
      if(exist(var.nameTosave)~= 7)
          mkdir(var.nameTosave);
      end
      var.nameTosave =
strcat([var.nameTosave,var.anormal_image_names(i).name,'.jpg']);
      imwrite(var.anImage{8,i},var.nameTosave,'jpg');
  end
  %cd(var.anormal_folder);
  pause(pause_Time);
  clc
end

for i=1:length(var.anormal_image_names)
  disp(strcat('Resize Anormal image : ',num2str(i)));
  %wavelet compression
  cd(var.path);
  var.anImage{9,i} = double(imresize(var.anImage{3,i},[50 50]))/255;
  %cd(var.anormal_folder);
  if(var.Save)
      var.nameTosave = strcat([var.path,'\Anormal\Size 50x50\']);
      if(exist(var.nameTosave)~= 7)
          mkdir(var.nameTosave);
      end
      var.nameTosave =
strcat([var.nameTosave,var.anormal_image_names(i).name,'.jpg']);
      imwrite(var.anImage{9,i},var.nameTosave,'jpg');
  end
  pause(pause_Time);
  clc
end

clear('i');
cd(var.path);
var.n = length(var.normal_image_names);
var.an = length(var.anormal_image_names);

clc
disp('End of image processing ');
disp(' '); pause(2);
disp('Artificial neural Networks are being prepared');
disp(' ');pause(2);
disp('The Artificial Neural Network is starting');
pause(2);
neural_net
```

%neural_net

```
clc
nn.x = length(var.nImage);
nn.y = length(var.anImage);
if(var.net=='RAW')
    imageIndex = 3 ;
else
    imageIndex = 8 ;
```

```
end
nn.imLength = [16 50 64 512];
i=1;
nn.xy = min(nn.x,nn.y);
nn.maxxy = max(nn.x,nn.y);
nn.input=[]; nn.output=[];
nn.testInput =[]; nn.testOutput=[];
nn.trImg = 50;
msg__='Training is not complete, MSE is high, repeat training';
 nn.no = [1 0]';
 nn.ano = [0 1]';
while(i<=nn.trImg)
    x = imresize(var.nImage{imageIndex,i},[nn.imLength(1) nn.imLength(1)]);
    nn.a = reshape(x,nn.imLength(1)^2,1);
    nn.input =[nn.input nn.a];
    nn.output=[nn.output nn.no];
    x = imresize(var.anImage{imageIndex,i},[nn.imLength(1)
nn.imLength(1)]);
    nn.a = reshape(x,nn.imLength(1)^2,1);
    nn.input =[nn.input nn.a];
    nn.output=[nn.output nn.ano];
    i = i+1;
end
if(nn.trImg<nn.xy)
  for i=nn.trImg+1:nn.xy
      x = imresize(var.nImage{imageIndex,i},[nn.imLength(1)
nn.imLength(1)]);
    nn.a = reshape(x,nn.imLength(1)^2,1);
    nn.testInput = [nn.testInput nn.a];
    nn.testOutput = [nn.testOutput nn.no];
    x = imresize(var.anImage{imageIndex,i},[nn.imLength(1)
nn.imLength(1)]);
    nn.a = reshape(x,nn.imLength(1)^2,1);
    nn.testInput = [nn.testInput nn.a];
    nn.testOutput = [nn.testOutput nn.ano];
  end
end
nn.allInput=[nn.input nn.testInput];
%nn.allInput = double(nn.allInput);
nn.allOutput = [nn.output nn.testOutput];
layer_size = [250 250];
transfer_functions = {'tansig','tansig','tansig'};
load('data');
switch(var.net)
    case 'W',
      nets = networks.networkW;
    case 'O',
      nets = networks.networkO;
    case 'E',
      nets = networks.networkE;
    case 'M',
      nets = networks.networkM;
    case 'RAW',
      nets = networks.networkRAW;
    otherwise,
      error('An error encountered, please check the type of network');
```

```
end

nets.trainParam.goal = 0.004;
nets.trainParam.lr = 0.002;
nets.trainParam.epochs = 8000;
[netR,TR] = train(nets,nn.allInput,nn.allOutput);
if(min(TR.perf)<0.05)
    result = sim(netR, nn.input);
    testResult = sim(netR, nn.testInput);
    error1 = result(:,1:80) - nn.output(:,1:80);
    er1 = sum(abs(error1));
    tr_rate = 100 * sum((abs(sum(error1)))<0.2)/80;
    disp(strcat('training rate is :  ',num2str(tr_rate),'%'));
    disp('  ');
    error = nn.testOutput(:,1:80) - testResult(:,1:80);
    test_rate = 100 * sum((abs(sum(error)))<0.1)/80;
    disp(strcat('test rate is :  ',num2str(test_rate),'%'));
    tab_tr=[];
    ko=1;
    for i=1:8
        tab_tr=[tab_tr ; result(:,ko:ko+9)];
        ko=ko+10;
    end
    tab_test=[];
    ko=1;
    for i=1:8
        tab_test=[tab_test ; testResult(:,ko:ko+9)];
        ko=ko+10;
    end
else
    disp(msg__);
end
```