

ACCESSIBLE MELANOMA DETECTION USING SMART PHONE AND MOBILE IMAGE ANALYSIS

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in computer science and Engineering

by

Varri venkata madhavi (Reg. No. 37110809)

Vajralasahiti (Reg. No. 37110801)



**DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING SCHOOL OF
COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A"
by NAAC I2B Status by UGC Approved by AICTE
JEPPIAARNAGAR, RAJIV GANDHI SALAI, CHENNAI-600119

MARCH- 2021



SATHYABAMA



INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with "A" grade by NAAC 12B Status by UGC IAP
proved by AICTE
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Ms.V.V.madhavi (37110809) and V.sahiti (37110801) who carried out the project entitled "ACCESSIBLE MELONAMA DETECTION USING SMART PHONE AND MOBILE IMAGE ANALYSIS" under my supervision from november 2020 to march 2021

Internal guide

Mr.M.Kamalesh,M.E.,P

h.d

Head of the

Department

Dr.L.Lakshman,M.E.,Ph.D

Dr.S.Vigneswari,M.E.,Ph.D

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I **Varri Venkata Madhavi(37110809),Vajrala Sahiti(37110801)** hereby declare that the Project Report entitled “**ACCESSIBLE MELANOMA DETECTION USING SMART PHONE AND MOBILE IMAGE ANALYSIS**” is done by me under the guidance of **Mr.M.Kamalesh,M.E., Ph.D.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to Dr.T.Sasikala M.E.,Ph.D.,Dean,school of computing and Dr.L.Lakshman,M.E.,Ph.D and Dr.S.Vigneswari,M.E.,Ph.D.,Heads of the Department of computer science and engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide Mr.M.Kamalesh,M.E.,Ph.D for his valuable guidance,suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Computer Science and Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

Cellular gadgets, including smart phones, are being utilized by billions of humans everywhere in the global. This creates the opportunity to intend a huge style of cellular image applications, amongst many imaging applications; healthcare packages have drawn lots of attentions currently. Melanoma is particular of the lethal pores and skin disease inflicting defeat of days to heaps of humans each year. Early on recognition of disease is feasible via visible inspection of pigmented lesion over the pores and skin, handled with simple removal of the cancerous cells. Our system intends to restrict the pores and skin lesion through the assist of combining a lightweight method for pores and skin detection with a k means segmentation technique. In the proposed system K means segmentation version is designed henceforth the category is additional carried out via deep learning techniques to enhance the excellent of classification .By the use of the resilient neural network(RNN) it will check whether it is normal or abnormal of skin cancer.

TABLE OF CONTENTS

	ABSTRACT	v
	LIST OF FIGURES	vi
CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	1
	1.1 INTRODUCTION TO IMAGE PROCESSING	1
	1.2 TYPES OF DIGITAL IMAGE	2
	1.3 COLOR MODEL	2
	1.3.1 RGB COLOR MODEL	2
	1.3.2 CONVERTING COLOR TO GRAYSCALE	3
	1.3.3 GRAYSCALE AS SINGLE OR MULTIPLE CHANNEL COLOR IMAGES	4
	1.3.4 HISTOGRAM EQUALIZATION OF COLOR IMAGE	5
	1.4 DIGITAL IMAGE PREPROCESSING TECHNIQUES	6
2	LITERATURE REVIEW	9
	2.1 AUTOMATED MELANOMA RECOGNITION	9
	2.2 DEEP FEATURES TO CLASSIFY SKIN LESIONS	9
	2.3 EARLY MELANOMA DIAGNOSIS WITH MOBILE IMAGING	10
	2.4 ITERATIVE MULTIMODAL VISUAL SEARCH ON MOBILE	10
3	METHODOLOGY	12
	3.1 EXISTING SYSTEM	12
	3.2 PROPOSED SYSTEM	12
	3.3 MODULE DESCRIPTION	12
	3.4 K MEANS SEGMENTATION	13
	3.5 RESILIENT NEURAL NETWORK	14
	3.5.1 HISTORICAL BACKGROUND	14
	3.5.2 BASICS OF NEURAL NETWORKS	15
	3.5.3 NEURAL NETWORKS VS CONVENTIONAL COMPUTERS	16
	3.5.4 NETWORK LAYERS	17
	3.6 FEED-FORWARD NETWORKS	18
	3.6.1 FEEDBACK NETWORKS	18
	3.6.2 DEEP BELIEF NEURAL NETWORKS	18
	3.6.2 BACK-PROPAGATION MULTI-LAYER NEURAL NETWORKS	20

4	SYSTEM REQUIREMENTS	26
	4.1 HARDWARE REQUIREMENTS	26
	4.2 SOFTWARE REQUIREMENTS	26
	4.3 INTRODUCTION TO MATLAB	27
5	SYSTEM DESIGN	31
	5.1 ARCHITECTURE	31
	5.2 SEQUENCE DIAGRAM	32
	5.3 USE CASE DIAGRAM	33
	5.4 ACTIVITY DIAGRAM	34
6	RESULT AND DISCUSSION	35
7	CONCLUSION AND FUTURE WORK	42
	REFERENCES	43
	SOURCE CODE	44

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
1.1	RGB COLOR MODEL	3
1.2	CONVERTING COLOR TO GRAYSCALE	4
1.3	COMPOSITION OF RGB FROM 3 FRAYSCALE IMAGES	5
1.4	HISTOGRAM EQUALIZATION OF COLOR IMAGE	6
1.5	IMAGE REPRESENTATION	7
3.1	NEURAL NETWORK LAYER	15
3.6.2	DEEP BELIEF NETWORK LAYER	18
5.1	PROPOSED ARCHITECTURE	31
5.2	SEQUENCE DIAGRAM	31
5.3	USECASE DIAGRAM	32
5.4	ACTIVITY DIAGRAM	33
6.3	RESIZE IMAGE	37
6.4	RED PANEL	37
6.5	GREEN PANEL	38
6.6	BLUE PANEL	38
6.7	BACKGROUND IMAGE	39
6.8	MASKED IMAGE	39
6.9	TUMOR DETECTION	40

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO IMAGE PROCESSING

Image Processing is a method to enhance raw images received from cameras/sensors placed on satellites, space probes and aircrafts or pictures taken in normal day-to-day life for various applications. Various techniques have been developed in Image Processing during the last four to five decades. Most of the techniques are developed for enhancing images obtained from unmanned spacecrafts, space probes and military reconnaissance flights. Image Processing systems are becoming popular due to easy availability of powerful personnel computers, large size memory devices, graphics software's etc. Image processing is a physical process used to convert an image signal into a physical image. The image signal can be either digital or analog. The actual output itself can be an actual physical image or the characteristics of an image.

Image processing is photography. In this process, an image is captured or scans using a camera to create a digital or analog image. In order to produce a physical picture, the image is processed using the appropriate technology based on the input source type. In digital photography, the image is stored as a computer file. This file is translated using photographic software to generate an actual image. The colors, shading, and nuances are all captured at the time the photograph is taken the software translates this information into an image. When creating images using analog photography, the image is burned into a film using a chemical reaction triggered by controlled exposure to light. The image is processed in a darkroom, using special chemicals to create the actual image. This process is decreasing in popularity due to the opening of digital photography, which requires less effort and special training to product images.

Face recognition software, medical image processing and remote sensing are all possible due to the development of digital image processing. Specialized computer programs are used to enhance and correct images.

1.2 TYPES OF DIGITAL IMAGE

For photographic purposes, there are two important types of digital images: color and grayscale. Color images are made up of colored pixels while grayscale images are made of pixels in different shades of gray.

- **Grayscale Images:** A grayscale image is made up of pixels, each of which holds a single number corresponding to the gray level of the image at a particular location. These gray levels span the full range from black to white in a series of very fine steps, normally 256 different grays. Assuming 256 gray levels, each black and white pixel can be stored in a single byte (8 bits) of memory.

- **Color Images:** A color image is made up of pixels, each of which holds three numbers corresponding to the red, green and blue levels of the image at a particular location. Assuming 256 levels, each color pixel can be stored in three bytes (24 bits) of memory. Note that for images of the same size, a black & white version will use three times less memory than a color version.

- **Binary Images:** Binary images use only a single bit to represent each pixel. Since a bit can only exist in two states- ON or OFF; every pixel in a binary image must be one of two colors, usually black or white. This inability to represent intermediate shades of gray is what limits their usefulness in dealing with photographic images.

1.3 COLOR MODEL:

1.3.1 RGB Color Model:

A representation of additive color mixing. Projection of primary color lights on a screen shows secondary colors where two overlap; the combination of all three of red, green, and blue in appropriate intensities makes white.



Fig 1.1 RGB color model

The RGB color model is an additive color model in which red, green, and blue light is added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue. The main purpose of the RGB color model is for the sensing, representation, and display of images in electronic systems, such as televisions and computers, though it has also been used in conventional photography.

1.3.2 CONVERTING COLOR TO GRAYSCALE

Conversion of a color image to grayscale is not unique; different weighting of the color channels effectively represents the effect of shooting black-and-white film with different-colored photographic filters on the cameras.

To convert any color to a grayscale representation of its luminance, first one must obtain the values of its red, green, and blue (RGB) primaries in linear intensity encoding, by gamma expansion. Then, add together 30% of the red value, 59% of the green value, and 11% of the blue value (these weights depend on the exact choice of the RGB primaries, but are typical). Regardless of the scale employed (0.0 to 1.0, 0 to

255, 0% to 100%, etc.) the resultant number is the desired linear luminance value; it typically needs to be gammacompressed to get back to a conventional grayscale representation.



Fig.1.2 converting color to grayscale image

This is not the method used to obtain the luma in the YUV and related color models, used in standard color TV and video systems as PAL and NTSC, as well as in the L^*a^*b color model. These systems directly compute a gamma-compressed luma as a linear combination of gamma-compressed primary intensities, rather than use linearization via gamma expansion and compression.

To convert a gray intensity value to RGB, simply set all the three primary color components red, green and blue to the gray value, correcting to a different gamma if necessary.

1.3.3 GRAYSCALE AS SINGLE CHANNELS OF MULTICHANNEL COLOR IMAGES

Color images are often built of several stacked colorchannels, each of them representing value levels of the given channel. For example, RGB images are composed of three independent channels for red, green and blue primarycolor components; CMYK images have four channels for cyan, magenta, yellow and black inkplates, etc.

Here is an example of color channel splitting of a full RGB color image. The column at left shows the isolated color channels in natural colors, while at right there are their grayscale equivalences:

The reverse is also possible: to build a full color image from their separate grayscale channels. By mangling channels, using offsets, rotating and other manipulations, artistic effects can be achieved instead of accurately reproducing the original image.

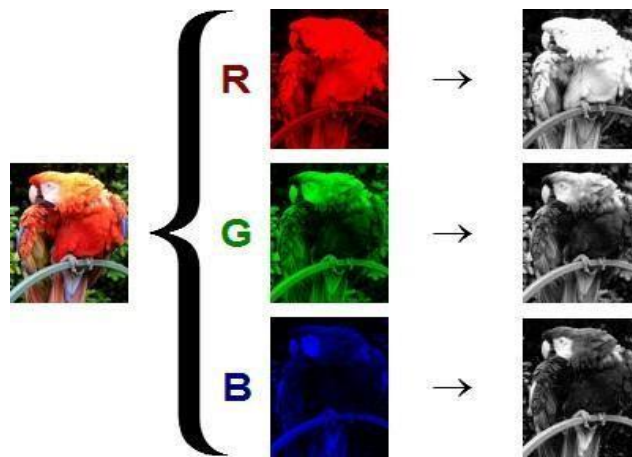
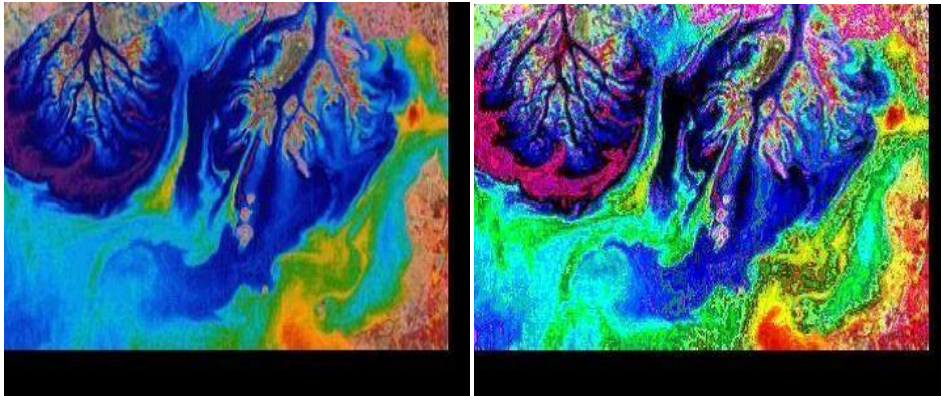


Fig 1.3 Composition of RGB from 3 Grayscale images

1.3.4 HISTOGRAM EQUALIZATION OF A COLOR IMAGE:

Figure shows a color image with 768 x 512 pixels and 16 levels for each band. As a result of the color histogram equalization, Figure 3 shows more diversity in color than. Due to lack of space, the three-dimensional histogram is not depicted here. In fact, it is equalized uniformly. It took 412.6 seconds to equalize the histogram with nonzero values at 1501 colors out of 4096. Consecutive probabilities in the histogram would be further investigated. The results of the histogram equalization for two color images are displayed.



(a) Original image

(b) Color-equalized image

Fig 1.4. Histogram equalization of a color image.

1.4 DIGITAL IMAGE PROCESSING TECHNIQUES

The various Image Processing techniques are:

- Image representation
- Image preprocessing
- Image enhancement
- Image restoration
- Image analysis
- Image reconstruction
- Image data compression

IMAGE REPRESENTATION

An image defined in the "real world" is considered to be a function of two real variables, for example, $f(x,y)$ with f as the amplitude (e.g. brightness) of the image at the *real* coordinate position (x,y) . The effect of digitization is shown in Figure.

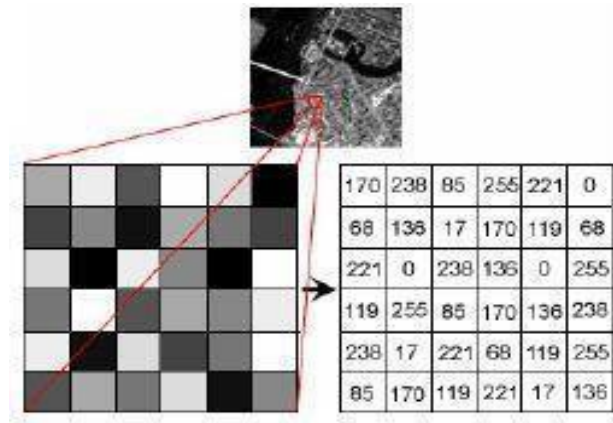


Fig.1.5 Image Representation

The 2D continuous image $f(x,y)$ is divided into N rows and M columns. The intersection of a row and a column is called as *pixel*. The value assigned to the integer coordinates $[m,n]$ with $\{m=0,1, 2,...,M-1\}$ and $\{n=0,1,2,...,N-1\}$ is $f[m,n]$. In fact, in most cases $f(x, y)$ which we might consider to be the physical signal that impinges on the face of a sensor. Typically an image file such as BMP, JPEG, TIFF etc., has some header and picture information. A header usually includes details like format identifier (typically first information), resolution, number of bits/pixel, compression type, etc.

IMAGE ENHANCEMENT

Sometimes images obtained from satellites and conventional and digital cameras lack in contrast and brightness because of the limitations of imaging sub systems and illumination conditions while capturing image. Images may have different types of noise. In image enhancement, the goal is to accentuate certain image features for subsequent analysis or for image display. Examples include contrast and edge enhancement, pseudo-coloring, noise filtering, sharpening, and magnifying. Image enhancement is useful in feature extraction, image analysis and an image display.

The enhancement process itself does not increase the inherent information content in the data. It simply emphasizes certain specified image characteristics. Enhancement algorithms are generally interactive and application-dependent.

IMAGE SEGMENTATION:

Image segmentation is the process that subdivides an image into its constituent parts or objects. The level to which this subdivision is carried out depends on the problem being solved, i.e., the segmentation should stop when the objects of interest in an application have been isolated e.g., in autonomous air-to-ground target acquisition, suppose our interest lies in identifying vehicles on a road, the first step is to segment the road from the image and then to segment the contents of the road down to potential vehicles.

CHAPTER 2

LITERATURE REVIEW

2.1 AUTOMATED MELANOMA RECOGNITION

Author Name:Harald Ganster, Axel Pinz, Reinhard Röhner, Ernst Wildling, Michael Binder, Harald Kittler

Year of Publishing: 2014

Abstract:

A system for the computerized analysis of images obtained from ELM has been developed to enhance the early recognition of malignant melanoma. As an initial step, the binary mask of the skin lesion is determined by several basic segmentation algorithms together with a fusion strategy. A set of features containing shape and radiometric features as well as local and global parameters is calculated to describe the malignancy of a lesion. Significant features are then selected from this set by application of statistical feature subset selection methods. The final kNN classification delivers a sensitivity of 87% with a specificity of 92%.

2.2 DEEP FEATURES TO CLASSIFY SKIN LESIONS

Author Name:Jeremy Kawahara, Aicha BenTaieb, and Ghassan Hamarneh

Abstract:

Diagnosing an unknown skin lesion is the first step to determine appropriate treatment. We demonstrate that a linear classifier, trained on features extracted from a convolutional neural network pretrained on natural images, distinguishes among up to ten skin lesions with a higher accuracy than previously published state-of-the-art results on the same dataset.

Further, in contrast to competing works, our approach requires neither lesion segmentations nor complex preprocessing. We gain consistent additional improvements to accuracy using a per image normalization, a fully convolutional network to extract multi-scale features, and by pooling over an augmented feature space. Compared to state-of-the art, our proposed approach achieves a favorable accuracy of 85.8% over 5-classes (compared to 75.1%) with noticeable improvements in accuracy for underrepresented classes (e.g., 60% compared to 15.6%). Over the entire 10-class dataset of 1300 images captured from a standard (non-dermoscopic) camera, our method achieves an accuracy of 81.8% outperforming the 67% accuracy previously reported.

2.3EARLY MELONAMA DETECTION USING SMART PHONE

Author Name:Thanh-Toan, Yiren Zhou, Haitian Zheng, Ngai-Man Cheung, Dawn Koh

Year of Publishing:2015

Abstract:

We research a mobile imaging system for early diagnosis of melanoma. Different from previous work, we focus on Smartphone-captured images, and propose a detection system that runs entirely on the Smartphone. Smartphone-captured images taken under loosely-controlled conditions introduce new challenges for melanoma detection, while processing performed on the Smartphone is subject to computation and memory constraints. To address these challenges, we propose to localize the skin lesion by combining fast skin detection and fusion of two fast segmentation results. We propose new features to capture color variation and border irregularity which are useful for Smartphone-captured images. We also propose a new feature selection criterion to select a small set of good features used in the final lightweight system. Our evaluation confirms the effectiveness of proposed algorithms and features. In addition, we present our system prototype which computes selected visual features from a user-captured skin lesion image, and analyzes them to estimate the likelihood of malignance, all on an off the- shelf smart phone.

2.4INTERACTIVE MULTIMODAL VISUAL SEARCH

Author Name:Priyanka Pakhare Karishma Mor eKaran Nagane

Year of Publishing:2015

Abstract:

This paper gives information about interactive multimodal visual search on mobile device. We mainly focus on the color features; color presents an interesting problem (namely the 'cross-talk' of features). Multimedia includes a combination of text audio, still images, animation, video or interactivity content form. The searching method will search a database of multimedia object to locate objects that match a query object exactly. Multimedia search queries as textual request and through other media. And this process is often referred to as search by example, because the typical interaction consists submitting a piece of information (e.g., video, an image or piece of audio) at the purpose of finding similar multimedia items

CHAPTER 3

DESIGN METHDOLOGY

3.1 EXISTING SYSTEM

In the existing design a detection system that is optimized to run entirely on the resource constrained smart phone. Our system intends to localize the skin lesion by combining a lightweight method for skin detection with a hierarchical segmentation approach using two fast segmentation methods.

3.2 PROPOSED SYSTEM:

In the proposed system K means segmentation model is designed henceforth the classification is further done by Resilient neural network(RNN) to improve the quality of classification .A morphological study is been developed here for deep understanding of the image features etc.

3.3MODULE DISCRIMINATION

Module I: Preprocessing

If the input images are color images means we are convert to gray scale from that color images.

Module II: Segmentation

The image features like color, weight and pixel information to apply before the classifier (Resilient neural network).Image segmentation is typically used to locate objects and boundaries in images. Here we used the K means segmentation algorithm is used in order to segment the portion of defected areas.

Module III: Classification

In this module the image data are being stored up in an array, and channel values of the images are being compared and stored in temporary arrays.

3.4 K MEANS SEGMENTATION

Image segmentation is an important preprocessing operation in image recognition and computer vision. We use K -means image segmentation method, which generates accurate segmentation results with simple operation. Since the simplest form of K -means algorithm, it is used by most of the researchers in the field of data mining. The process of k -Means follows eminent and simple way to classify a given data set by means of a certain number of clusters. The K -means clustering is a method used to divide an image as n patterns $\{x_1, x_2, \dots, x_n\}$ in d dimensional space into k clusters (assume k clusters). The result is a set of k clusters based on k centres, each of which is located at the centroid of the separated dataset.

The basic idea of K -means algorithm is to cluster the objects closest to them by clustering the K points in the space. Iteratively, the values of centroid of clusters are updated one by one until the best clustering results are obtained. K -means algorithm is a typical representative of the clustering method based on the prototype function. It takes the distance from the data point to the prototype as the objective function of optimization. The adjustment rules of iterative operation are obtained by the method of finding extreme values of functions. The K -means algorithm takes Euclidean distance as the similarity measure, which is to find the optimal classification of an initial cluster center vector, so that the evaluation index is minimum. The error square sum criterion function is used as a clustering criterion function. Although the algorithm of K -means is efficient, value of K should be given in advance, and the selection of K value is very difficult to estimate. In many cases, it is unknown in advance how many categories the given data set should be divided into. As we mentioned, K -means is one of the classical clustering algorithms in the partitioning method. The efficiency of this algorithm is high, but due to the need to determine the number of clusters K , it brings certain difficulties for automated calculations. This algorithm can be shortened in the following steps:

The steps involved in images by k -Means algorithm are given below.

Step 1: Insert the original images as input.

Step 2: Convert the rgb to grayscale image.

Step 3: Cluster dataset images.

Step 4: Find out the 'k' in image by algorithm itself.

Step 5: Randomly choose the k cluster centers.

Step 6: Calculate area or depth calculation in segmented image..

Step7: Calculate the distance between each pixel to each cluster centre. Step8: If the distance is near to the centre then move to that cluster. Step 9: Otherwise move to next cluster.

Step 10: Repeat the process until the segmentation portion comes.

3.5 RESILIENT NEURAL NETWORK:

A Resilient Neural Network (RNN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. NNs, like people, learn by example. An NN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons.

3.5.1 HISTORICAL BACKGROUND:

Neural network simulations appear to be a recent development. However, this field was established before the advent of computers, and has survived at least one major setback and several eras. Many important advances have been boosted by the use of inexpensive computer emulations. Following an initial period of enthusiasm, the field survived a period of frustration and disrepute.

These pioneers were able to develop convincing technology which surpassed the limitations identified by Minsky and Papert. Minsky and Papert, published a book (in 1969) in which they summed up a general feeling of frustration (against neural networks) among researchers, and was thus accepted by most without further analysis. Currently, the neural network field enjoys a resurgence of interest and a corresponding increase in funding

3.5.2 BASICS OF NEURAL NETWORK

Neural networks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. Patterns are presented to the network via the 'input layer', which communicates to one or more 'hidden layers' where the actual processing is done via a system of weighted 'connections'. The hidden layers then link to an 'output layer' where the answer is output as shown in the graphic below.

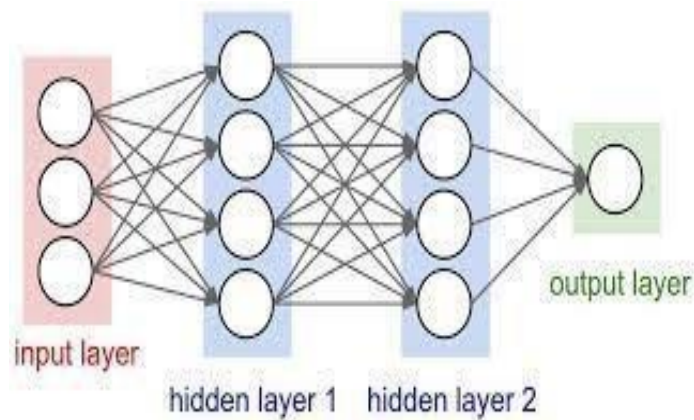


Fig.3.1 Neural network layer

Use of neural networks:

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze.

Other advantages include:

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
2. Self-Organization: An NN can create its own organization or representation of the information it receives during learning time.
3. Real Time Operation: NN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.

3.5.3 NEURAL NETWORKS VERSUS CONVENTIONAL COMPUTERS

Neural networks take a different approach to problem solving than that of conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Solve the problem. That restricts the problem solving capability of conventional computers to problems that we already understand and know how to solve. But computers would be so much more unless the specific steps that the computer needs to follow are known the computer cannot useful if they could do things that we don't exactly know how to do.

Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem. Neural networks learn by example.

They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly. The disadvantage is that because the network finds out how to solve the problem by itself, its operation can be unpredictable.

On the other hand, conventional computers use a cognitive approach to problem solving; the way the problem is to be solved must be known and stated in small unambiguous instructions. These instructions are then converted to a high level language program and then into machine code that the computer can understand. These machines are totally predictable; if anything goes wrong is due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the neural network) in order to perform at maximum efficiency.

3.5.4 NETWORK LAYERS

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units.

The activity of the input units represents the raw information that is fed into the network.

The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

We also distinguish single-layer and multi-layer architectures. The single-layer organization, in which all units are connected to one another, constitutes the most general case and is of more potential computational power than hierarchically structured multi-layer organizations. In multi-layer networks, units are often numbered by layer, instead of following a global numbering

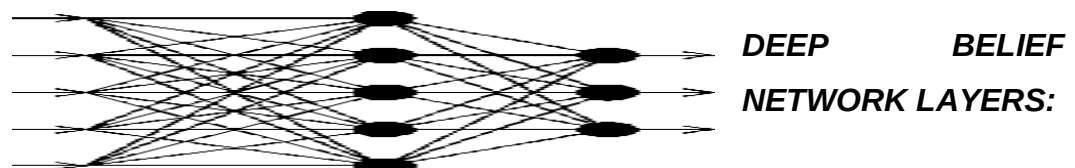
3.6 FEED-FORWARD NETWORKS:

Feed-forward ANNs allow signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organization is also referred to as bottom-up or top-down.

3.6.1 FEEDBACK NETWORKS:

Feedback networks (figure 1) can have signals traveling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organizations.

3.6.2



The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "**input**" units is connected to a layer of "**hidden**" units, which is connected to a layer of "**output**" units.

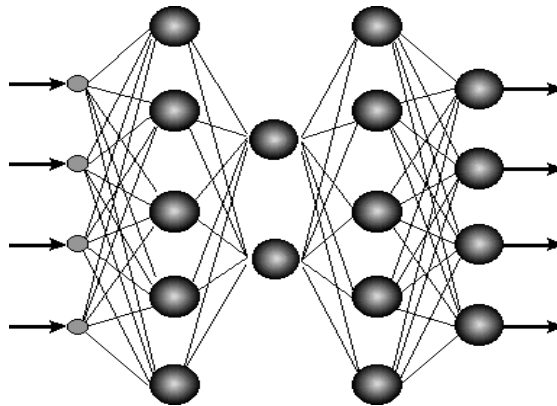
The activity of the input units represents the raw information that is fed into the network.

The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

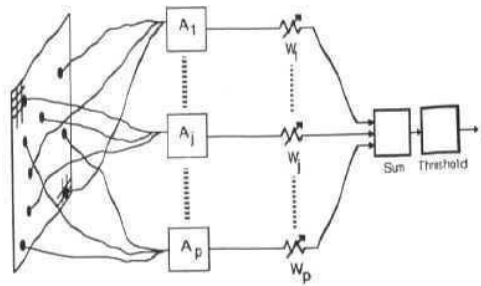
This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

We also distinguish single-layer and multi-layer architectures. The single-layer organization, in which all units are connected to one another, constitutes the most general case and is of more potential computational power than hierarchically structured multi-layer organizations. In multi-layer networks, units are often numbered by layer, instead of following a global numbering.



The most influential work on neural nets in the 60's went under the heading of 'perceptrons' a term coined by Frank Rosenblatt. The perceptron turns out to be an MCP model (neuron with weighted inputs) with some additional, fixed, pre-- processing. Units labeled A_1 , A_2 , A_j , A_p are called association units and their task is to extract specific, localized features from the input images. Perceptrons mimic the basic idea behind the mammalian visual system. They were mainly used in pattern recognition even though their capabilities extended a lot more.

In 1969 Minsky and
which they described
layer Perceptrons. The
had was tremendous
neural network



Papert wrote a book in
the limitations of single
impact that the book
and caused a lot of
researchers to lose their

interest. The book was very well written and showed mathematically that single layer perceptrons could not do some basic pattern recognition operations like determining the parity of a shape or determining whether a shape is connected or not. What they did not realize, until the 80's, is that given the appropriate training, multilevel perceptrons can do these operations.

3.6.3 INTRODUCTION TO BACK-PROPAGATION MULTI-LAYER NEURAL NETWORKS:

Lots of types of neural networks are used in data mining. One of the most popular types is multi-layer perceptron network and the goal of the manual has is to show how to use this type of network in Kocker data mining application.

Multi-layer perceptron is usually used for classification or prediction methods of data mining. User usually wants the network to split input data into some groups (classify them). There are always some necessary steps in lifetime cycle:

- Preparing data (and also understanding them)
- Preparing network
- Training network
- Classifying

We presume that you have prepared data. This document will help you with preparing and training network to get the best network for classification as possible. After all, data classification is very simple.

The network is trained using Back-propagation algorithm with many parameters, so you can tune your network very well. Remember, you can use only numbers (type of integers, float, double) to train the network! And it is presumed that all data are normalized into $<0,1>$ interval. The better you prepare your data, the better results you get.

Resilient Neural Network Models:

Regression models describe the relationship between a *dependent variable*, y , and *independent variable* or variables, X . The dependent variable is also called the *response variable*. Independent variables are also called *explanatory* or *predictor variables*. Continuous predictor variables might be called *covariates*, whereas categorical predictor variables might be also referred to as *factors*. The matrix, X , of observations on predictor variables is usually called the *design matrix*.

A multiple linear regression model is

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \dots + \beta_p X_{ip} + \varepsilon_i, \quad i = 1, \dots, n,$$

where

- y_i is the i th response.
- β_k is the k th coefficient, where β_0 is the constant term in the model. Sometimes, design matrices might include information about the constant term. However, `LinearModel.fit` or `LinearModel.stepwise` by default includes a constant term in the model, so you must not enter a column of 1s into your design matrix X .
- X_{ij} is the i th observation on the j th predictor variable, $j = 1, \dots, p$.
- ε_i is the i th noise term, that is, random error.

In general, a linear regression model can be a model of the form

$$y_i = \beta_0 + \sum_{k=1}^K \beta_k f_k(X_{i1}, X_{i2}, \dots, X_{ip}) + \varepsilon_i, \quad i = 1, \dots, n,$$

where $f(\cdot)$ is a scalar-valued function of the independent variables, X_{ij} s. The functions, $f(X)$, might be in any form including nonlinear functions or polynomials. The linearity, in the linear regression models, refers to the linearity of the coefficients β_k . That is, the response variable, y , is a linear function of the coefficients, β_k .

Some examples of linear models are:

$$y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \beta_3 X_{3i} + \varepsilon_i$$

$$y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \beta_3 X_{1i}^3 + \beta_4 X_{2i}^2 + \varepsilon_i$$

$$y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \beta_3 X_{1i} X_{2i} + \beta_4 \log X_{3i} + \varepsilon_i$$

The following, however, are not linear models since they are not linear in the unknown coefficients, β_k .

$$\log y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \varepsilon_i$$

$$y_i = \beta_0 + \beta_1 X_{1i} + \frac{1}{\beta_2 X_{2i}} + e^{\beta_1 X_{1i} X_{2i}} + \varepsilon_i$$

The usual assumptions for linear regression models are:

- The noise terms, ε_i , are uncorrelated.
- The noise terms, ε_i , have independent and identical normal distributions with mean zero and constant variance, σ^2 . Thus

$$\begin{aligned} E(y_i) &= E\left(\sum_{k=0}^K \beta_k f_k(X_{i1}, X_{i2}, \dots, X_{ip}) + \varepsilon_i\right) \\ &= \sum_{k=0}^K \beta_k f_k(X_{i1}, X_{i2}, \dots, X_{ip}) + E(\varepsilon_i) \\ &= \sum_{k=0}^K \beta_k f_k(X_{i1}, X_{i2}, \dots, X_{ip}) \end{aligned}$$

and

$$V(y_i) = V\left(\sum_{k=0}^K \beta_k f_k(X_{i1}, X_{i2}, \dots, X_{ip}) + \varepsilon_i\right) = V(\varepsilon_i) = \sigma^2$$

So the variance of y_i is the same for all levels of X_{ij} .

- The responses y_i are uncorrelated.

The fitted linear function is

$$\hat{y}_i = b_0 + \sum_{k=1}^K b_k f_k(X_{i1}, X_{i2}, \dots, X_{ip}), \quad i = 1, \dots, n,$$

where $\hat{y}_i$ is the estimated response and b_k s are the fitted coefficients. The coefficients are estimated so as to minimize the mean squared difference between the prediction vector $b f(X)$ and the true response vector y , that is $\hat{y} - y$. This method is called the *method of least squares*.

In a linear regression model of the form $y = \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p$, the coefficient β_k expresses the impact of a one-unit change in predictor variable, X_j , on the mean of the response, $E(y)$ provided that all other variables are held constant. The sign of the coefficient gives the direction of the effect. For example, if the linear model is $E(y) = 1.8 - 2.35X_1 + X_2$, then -2.35 indicates a 2.35 unit decrease in the mean response with a one-unit increase in X_1 , given X_2 is held constant.

If the model is $E(y) = 1.1 + 1.5X_1^2 + X_2$, the coefficient of X_1^2 indicates a 1.5 unit increase in the mean of Y with a one-unit increase in X_1^2 given all else held constant. However, in the case of $E(y) = 1.1 + 2.1X_1 + 1.5X_1^2$, it is difficult to interpret the coefficients similarly, since it is not possible to hold X_1 constant when X_1^2 changes or vice versa.

Train the network:

Once the network weights and biases are initialized, the network is ready for training. The multilayer feed forward network can be trained for function approximation (nonlinear regression) or pattern recognition. The training process requires a set of examples of proper network behavior network inputs p and target outputs t . The process of training a neural network involves tuning the values of the weights and biases of the network to optimize network performance, as defined by the network performance function net .

The default performance function for feed forward networks is mean square error mse—the average squared error between the networks outputs and the target outputs t . It is defined as follows:

$$F = mse = \frac{1}{N} \sum_{i=1}^N (e_i)^2 = \frac{1}{N} \sum_{i=1}^N (t_i - a_i)^2$$

There are two different ways in which training can be implemented: incremental mode and batch mode. In incremental mode, the gradient is computed and the weights are updated after each input is applied to the network. In batch mode, all the inputs in the training set are applied to the network before the weights are updated. This topic describes batch mode training with the `train` command.

Incremental training with the `adapt` command is discussed in Incremental Training with `adapt`. For most problems, when using the Neural Network Toolbox™ software, batch training is significantly faster and produces smaller errors than incremental training.

For training multilayer feed forward networks, any standard numerical optimization algorithm can be used to optimize the performance function, but there are a few key ones that have shown excellent performance for neural network training. These optimization methods use either the gradient of the network performance with respect to the network weights, or the Jacobian of the network errors with respect to the weights. The gradient and the Jacobian are calculated using a technique called the back propagation algorithm, which involves performing computations backward through the network.

CHAPTER 4

SYSTEM REQUIREMENT

4.1 HARDWARE REQUIREMENTS

- Hard Disk : 40GB and Above
- RAM : 512MB and Above
- Processor : Pentium III and Above

4.2 SOFTWARE REQUIREMENTS

- MATLAB R2018a
- Windows Operating System XP and Above

MATLAB (**matrix laboratory**) is a multi-paradigm numerical computing environment and fourth-generation programming language. A proprietary programming language developed by Math Works, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the Mu PAD symbolic engine, allowing access to symbolic computing abilities. An additional package, Simulink, adds graphical multi-domain simulation and model-based design for dynamic and embedded systems.

The basic data structure in MATLAB is the array, an ordered set of real or complex elements. This object is naturally suited to the representation of images, real-valued ordered sets of color or intensity data.

MATLAB stores most images as two-dimensional arrays (i.e., matrices), in which each element of the matrix corresponds to a single pixel in the displayed image. (Pixel is derived from picture element and usually denotes a single dot on a computer display.)

Some images, such as RGB, require a three-dimensional array, where the first plane in the third dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities.

This convention makes working with images in MATLAB similar to working with any other type of matrix data, and makes the full power of MATLAB available for image processing applications. For example, you can select a single pixel from an image matrix using normal matrix subscripting.

4.3 INTRODUCTION TO MATLAB R2018a:

The MATLAB System

The MATLAB system consists of these main parts:

Desktop Tools and Development Environment

This is the set of tools and facilities that help you use and become more productive with MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, an editor and debugger, a code analyzer and other reports, and browsers for viewing help, the workspace, files, and the search path.

Mathematical Function Library

This is a vast collection of computational algorithms ranging from elementary functions, like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix Eigen values, Bessel functions, and fast Fourier transforms.

The Language

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create large and complex application programs.

Graphics

MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

External Interfaces

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), for calling MATLAB as a computational engine, and for reading and writing MAT-files.

Image Processing Toolbox

The Image Processing Toolbox is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations, including:

- Spatial image transformations
- Morphological operations
- Neighborhood and block operations

- Image analysis and enhancement
- Image registration
- De blurring
- Region of interest operations

Many of the toolbox functions are MATLAB M-files, a series of MATLAB statements that implement specialized image processing algorithms. You can extend the capabilities of the Image Processing Toolbox by writing your own M-files, or by using the toolbox in combination with other toolboxes, such as the Signal Processing Toolbox and the Wavelet Toolbox.

Computer Vision Toolbox

The field is of importance for such various applications as autonomous vehicles, navigating with the help of images, captured by a mounted camera, and high precision measurements using images, taken by calibrated cameras. In this paper, we will present a number of numerical routines, implemented in MATLAB, that are useful in a variety of computer vision applications.

The collection of routines will be called the Computer Vision Toolbox. One of the main problems in Computer Vision is to calculate the 3D-structure of the scene and the motion of the camera from measurements in the images taken from different viewpoints in the scene. This problem is called structure and motion, referring to the fact that both the structure of the scene and the motion of the camera are calculated from image measurements only.

A number of different sub problems, arising from different knowledge of the intrinsic properties of the camera appear. Other important problems are to calculate the structure of the 3D-scene given the motion of the camera and to calculate the motion of the camera given the structure of the scene. These problems are somewhat simpler than the general structure and motion problem, but are nevertheless important for navigation and obstacle avoidance.

A related problem is to calibrate a camera, i.e. calculate the focal distance, the principal point etc., from images of a known scene.

The problem of estimating structure and motion from image sequences are closely related to the field of closed range photogrammetric.

In this field images are used to make precise measurements of three-dimensional objects from a number of images, often taken by calibrated cameras. The main difference is that in computer vision, the focus is not on accuracy, but instead on reliability and speed of calculation. In all these problems different kind of so called features can be used.

The simplest feature is points that are easily detected in the sequence, such as corners or painted marks. Other types of features are lines and curves, which are easier to detect and track but more difficult to use in structure and motion algorithms. In order to use our computer vision routines these features must be detected in advance and the correspondence between different images must be established. The main emphasis of our computer vision toolbox is to use these detected features to solve for structure and motion.

Image Acquisition Toolbox

The Image Acquisition Toolbox's is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image acquisition operations including

- Acquiring images through many types of image acquisition devices, from professional grade frame grabbers to USB-based Webcams
- Viewing a preview of the live video stream
- Triggering acquisitions (includes external hardware triggers)
- Configuring call back functions that execute when certain events occur
- Bringing the image data into the MATLAB workspace

CHAPTER 5

SYSTEM DESIGN

5.1 ARCHITECTURE:

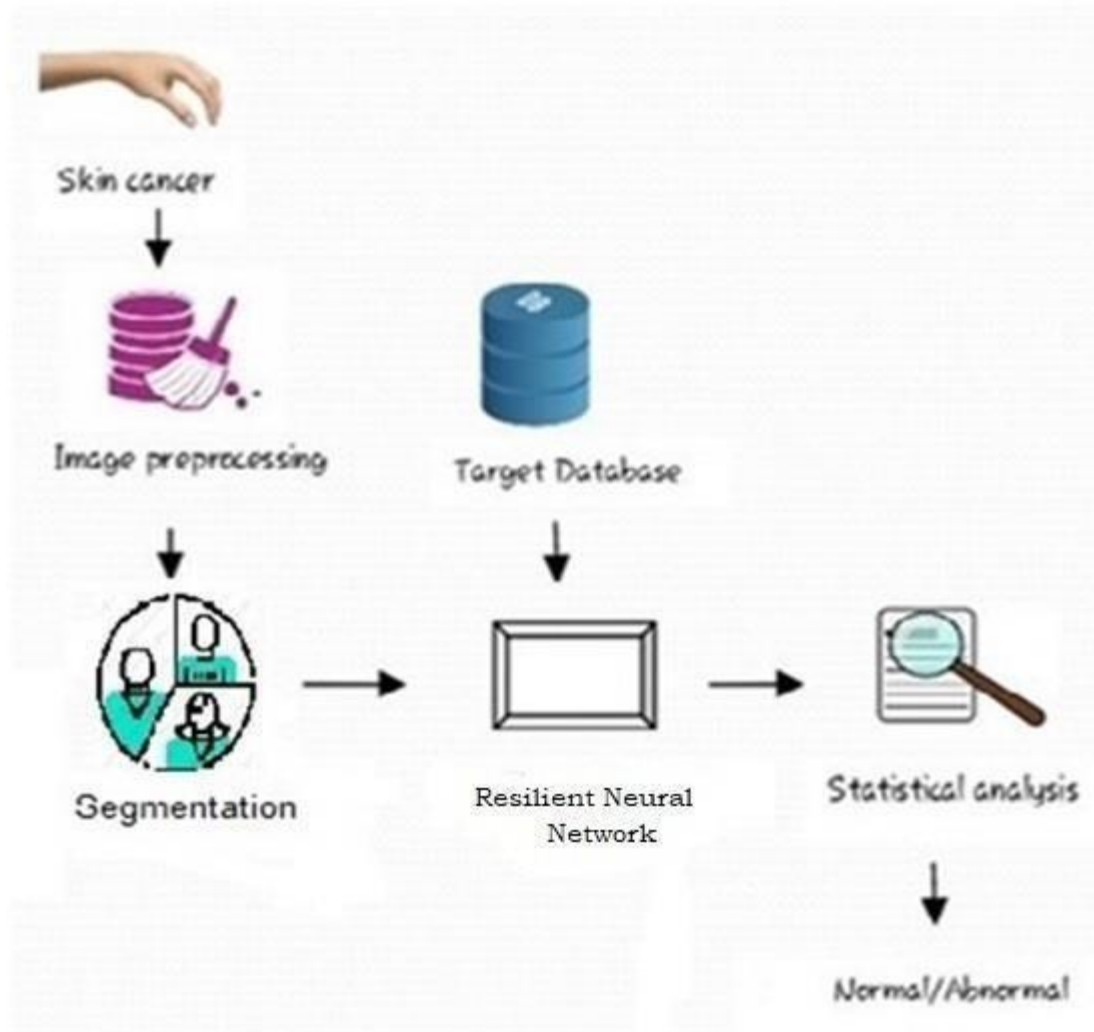


Fig.5.1 Proposed architecture

5.2 SEQUENCE DIAGRAM:

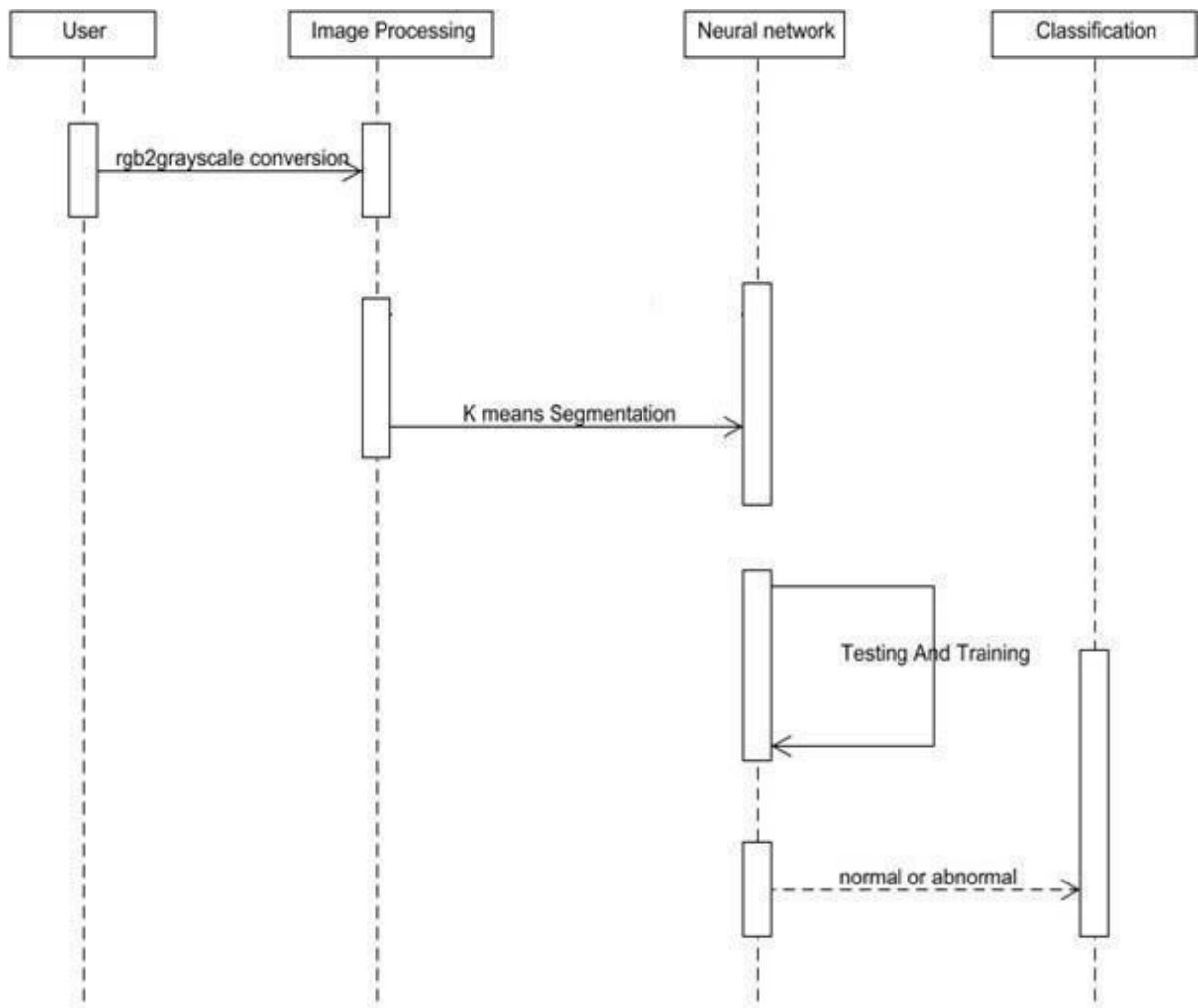


Fig.5.2 Sequence diagram

Initially the user will send the input image. If the input images are color images means we are convert to gray scale from that color images. The back propagation neural network concept for training the image and testing the image with the help of weight estimating classifier. The result image will compared with the dataset images and it will display whether it is normal or abnormal.

5.3 USE CASE DIAGRAM:

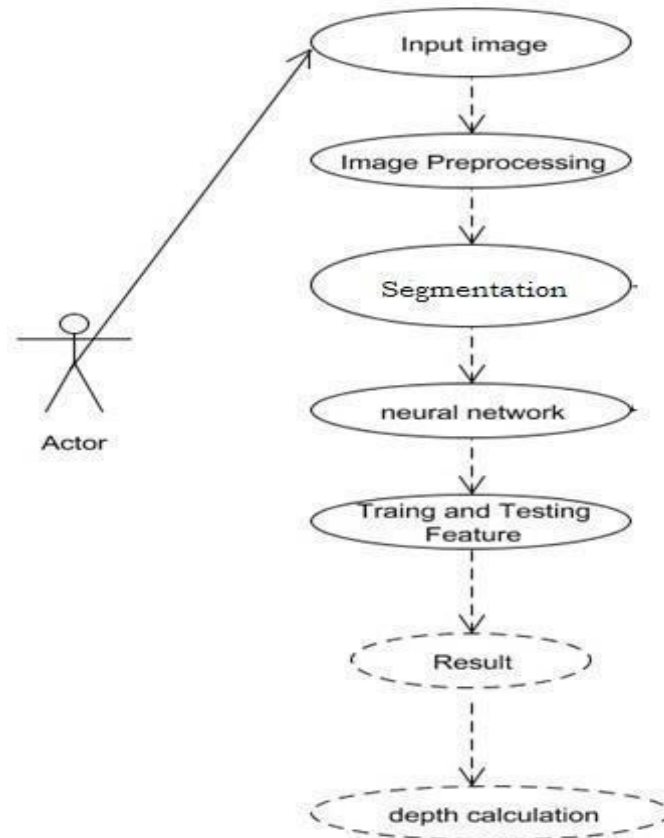


Fig.5.3 Use case Diagram

The actor will send an input image. If the input images are color images means we are convert to gray scale from that color images. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries in images. The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units. By using the resilient neural network classifier it will display whether it is normal or abnormal.

5.4 ACTIVITY DIAGRAM:

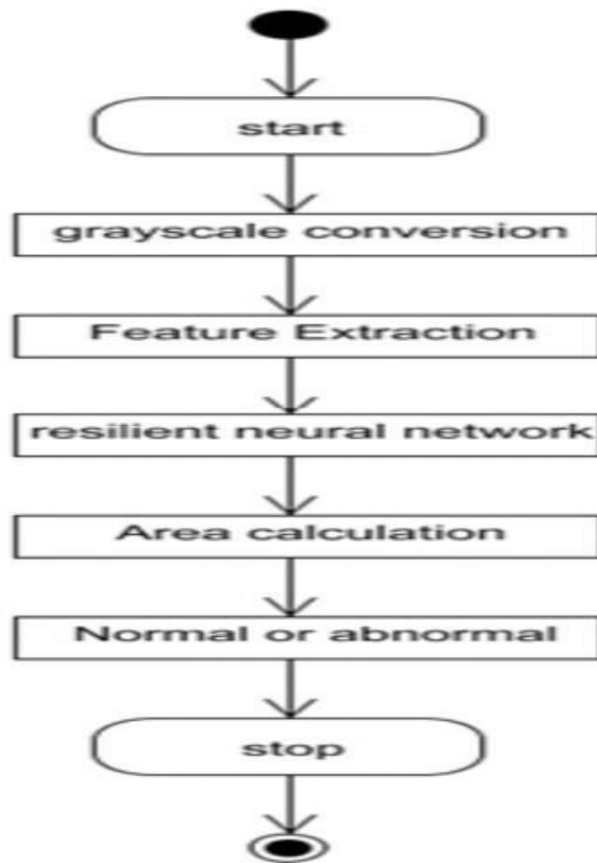
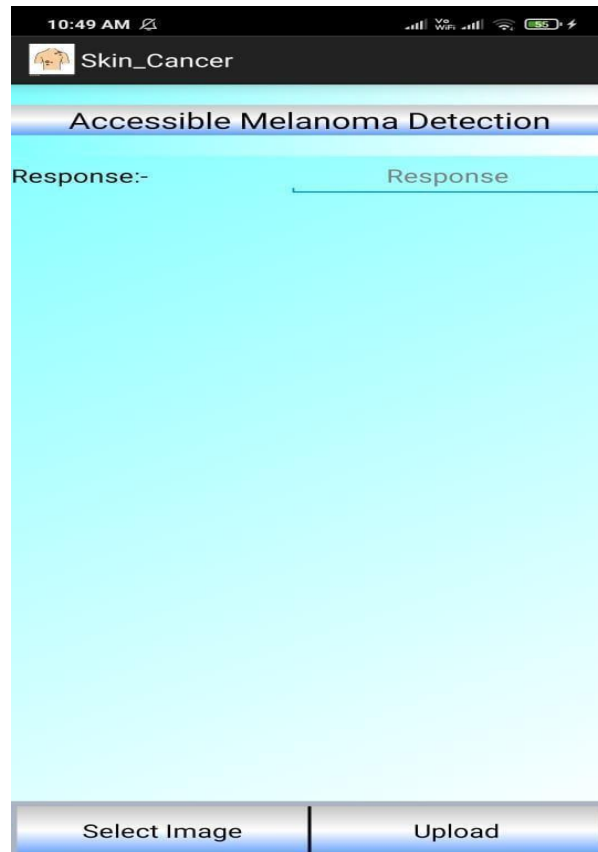


Fig.5.4 Activity Diagram

Initially at the start if the input images are color images means we are convert to gray scale from that color images. The back propagation neural network concept for training the image and testing the image with the help of weight estimating classifier. The result image will compared with the dataset images and it will display whether it is normal or abnormal. At the end process will be stopped.

CHAPTER 6

RESULT & DISCUSSION



10:49 AM

Vo WiFi 56



Skin_Cancer

Accessible Melanoma Detection

Response:-

Response



Select Image

Upload

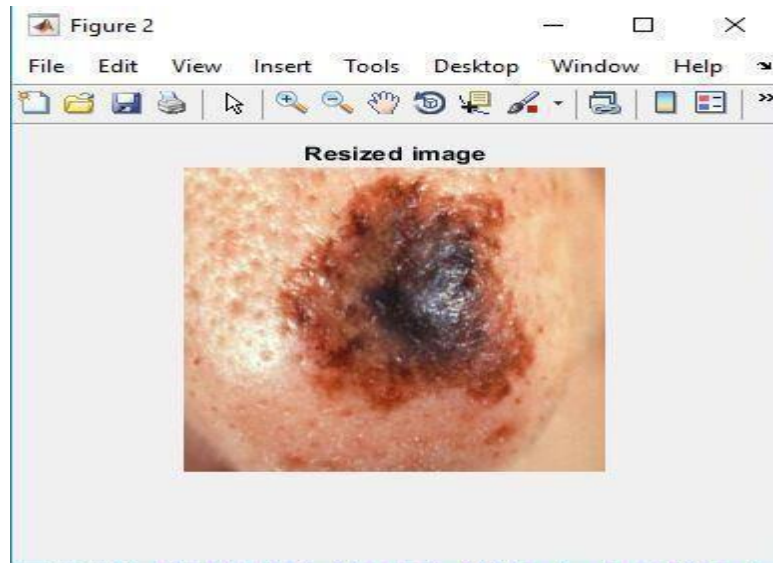


Fig.6.3 Resize image

From the input image and it will change the size of an image. This method takes the number of rows and columns and fits the original input image to an output image having the specified number of rows and columns.

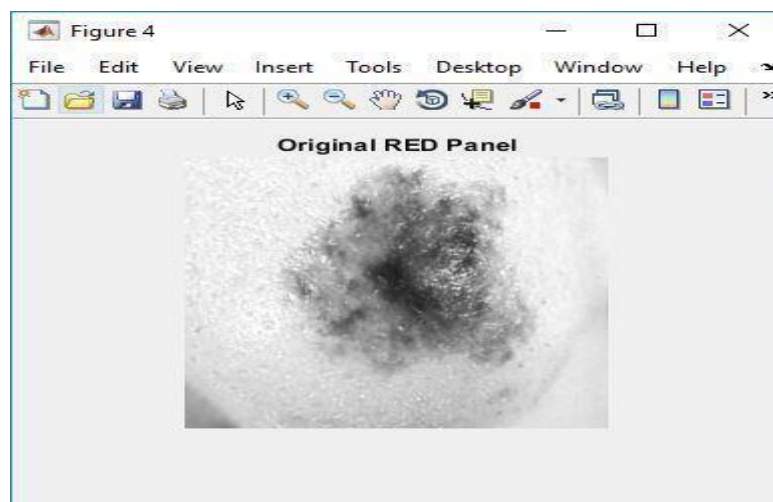


Fig.6.4 Red Panel

Grayscale image contains only brightness information. Each pixel value in grayscale image corresponds to an amount or quantity of light. The brightness graduation can be differentiated in grayscale image. variation from 0 to 255 where '0' represents black and '255' represents white.

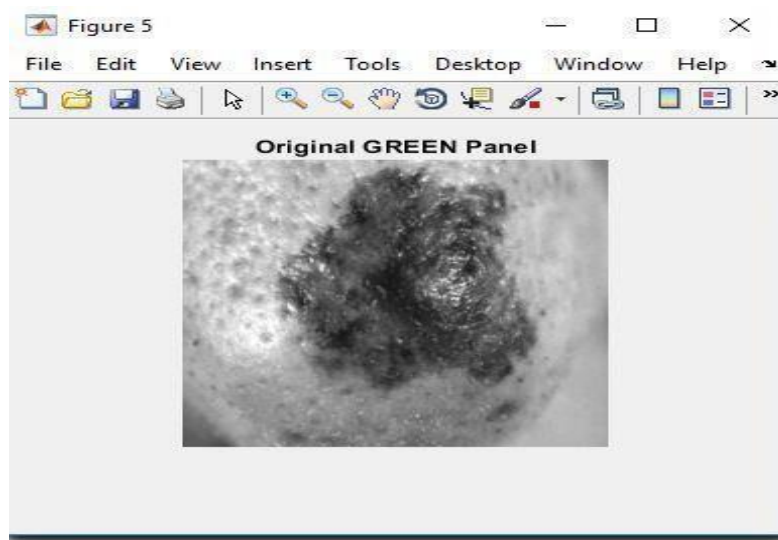


Fig.6.5 Green Panel

A colored image can be represented as a 3 order matrix. The first order is for the rows, the second order is for the columns and the third order is for specifying the color of the corresponding pixel. Here we use the RGB color format, so the third order will take 3 values of Red, Green and Blue respectively.

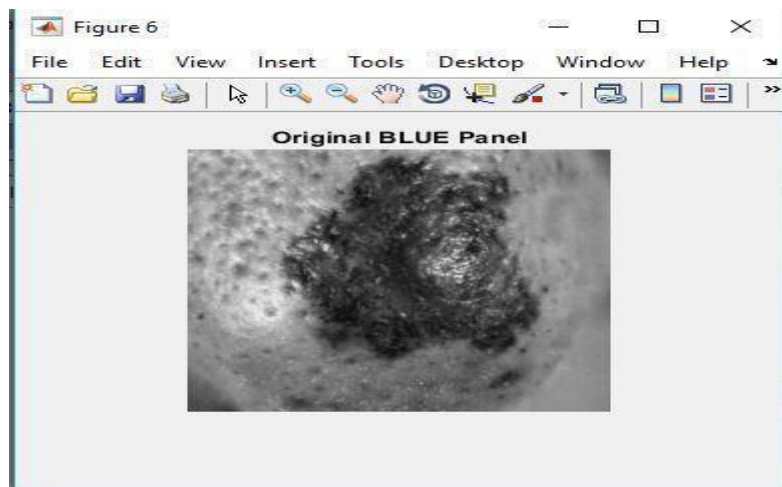


Fig.6.6 Blue Panel

A colored image can be represented as a 3 order matrix. The first order is for the rows, the second order is for the columns and the third order is for specifying the color of the corresponding pixel.

Here we use the RGB color format, so the third order will take 3 values of Red, Green and Blue respectively.

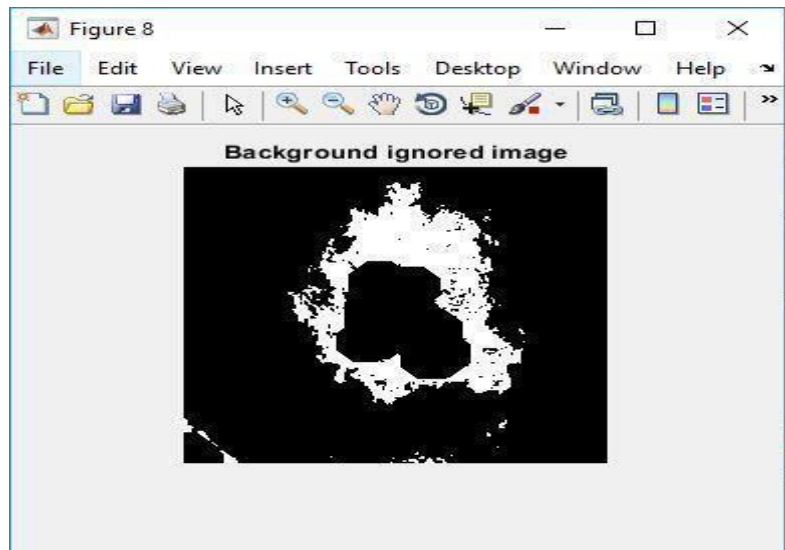


Fig.6.7 background image

Image thresholding is a simple form of image segmentation. It is a way to create a binary image from a single band or multi-band image. The process is typically done in order to separate "object" or foreground pixels from background pixels to aid in image processing.

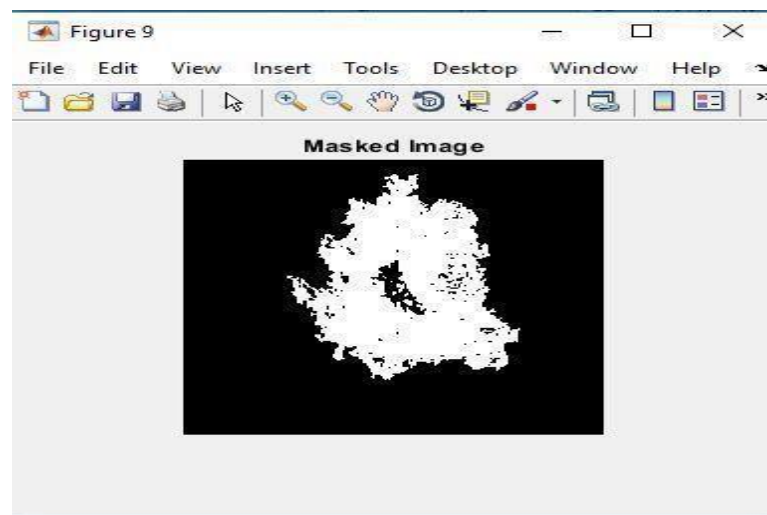


Fig.6.8 Masked image

Masks are used to exclude certain pixels from image processing or when computing image statistics. Masks are used to exclude certain pixels from image processing or when computing image statistics. Masked pixels aren't visible (they are transparent) when displayed. A mask is a binary raster that contains pixel values of 0 and 1,

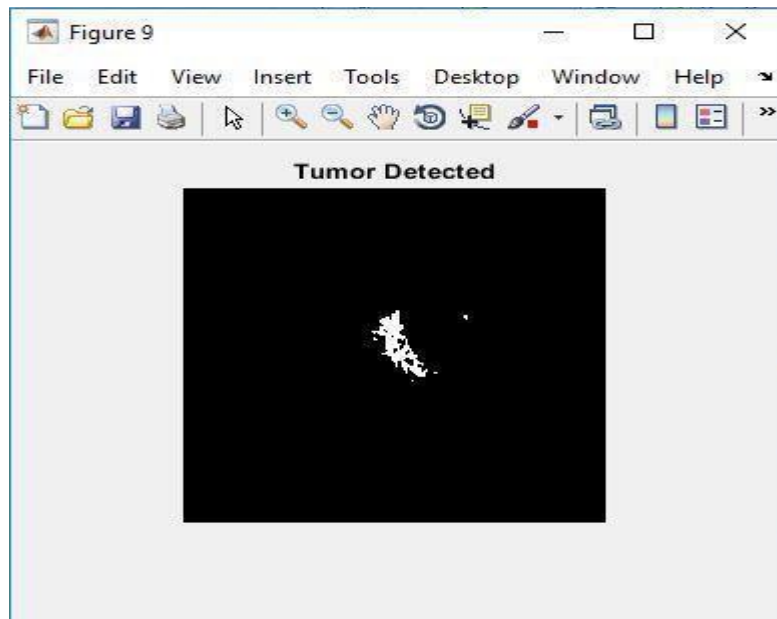
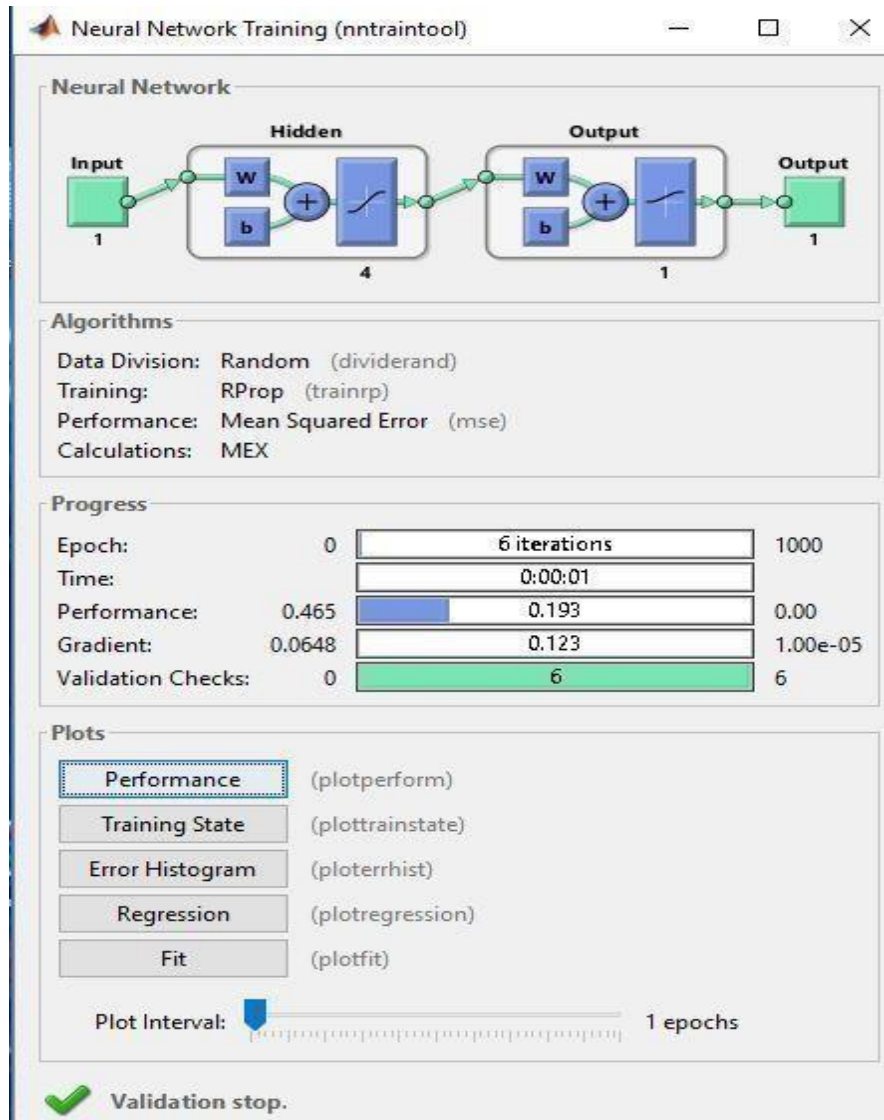


Fig.6.8 Tumor Detection

A threshold level is selected where all pixel values below the threshold are mapped to zero (black) and an upper threshold value is chosen so that all pixel values above this threshold are mapped to 255 (white). The thresholding process can be used to create binary masks for an image. Finally the affected portion of tumor will be displayed.



- The above output window is the Neural network training process which occurs after the successful configuration of the Neural box
- The input words from the customer review act as input to the neural network
- The target is nothing but the good words to be matched and negative words to be matched.

CHAPTER 7

CONCLUSION

We propose an accessible mobile health care solution for melanoma detection, using mobile image analysis. The main characteristics of the proposed system are a K means segmentation scheme suitable for the resource constrained platform a new set of features which efficiently capture the color variation and border irregularity from the smart phone image, and a new mechanism for selecting a compact set of the most discriminative features. In this module the image data are being stored up in an array, and channel values of the images are being compared and stored in temporary arrays. The enhanced values are used to compare the feature extracted images. Finally the resilient neural network is classified whether the given input image is normal or abnormal.

FUTURE WORK

In future we can execute these work on some other deep learning techniques and can also be employed and the various types of melanoma skin disease can be employed.

REFERENCES

- [1] W. Zhou et al., "Towards codebook-free: Scalable cascaded hashing for mobile image search," *IEEE Trans. Multimedia*, vol. 16, no. 3, pp. 601–611, Apr. 2014.
- [2] H. Li, Y. Wang, T. Mei, J. Wang, and S. Li, "Interactive multimodal visual search on mobile device," *IEEE Trans. Multimedia*, vol. 15, no. 3, pp. 594–607, Apr. 2013.
- [3] B. Girod et al., "Mobile visual search," *IEEE Signal Process. Mag.*, vol. 28, no. 4, pp. 61–76, Jul. 2011.
- [4] T. Chen, K. H. Yap, and D. Zhang, "Discriminative soft bag-of-visual phrase for mobile landmark recognition," *IEEE Trans. Multimedia*, vol. 16, no. 3, pp. 612–622, Apr. 2014.
- [5] W. Min, C. Xu, M. Xu, X. Xiao, and B. K. Bao, "Mobile landmark search with 3d models," *IEEE Trans. Multimedia*, vol. 16, no. 3, pp. 623–636, Apr. 2014.
- [6] R. L. Siegel, K. D. Miller, and A. Jemal, "Cancer statistics, 2017," *CA: Cancer J. Clinicians*, vol. 67, no. 1, pp. 7–30, 2017.
- [7] S. Demyanov, R. Chakravorty, M. Abedini, A. Halpern, and R. Garnavi, "Classification of dermoscopy patterns using deep convolutional neural networks," in *Proc. IEEE 13th Int. Symp. Biomed. Imag.*, vol. 124, pp. 983–8, Apr. 2016.

Source code

```
clc;

clear all;

close all;

warning off;

fontSize = 10;

    client_image();

%% Preprocessing

% [fn fp]=uigetfile({'*.jpg;*.jpeg;*.bmp','All video Files';...

%     '*.*.','All Files' }, 'mytitle',...

%     'C:\Work\Twin_A');

% I = imread([fp fn]);

% figure;imshow(I);

% title('Original image');

I = imread('du.jpg');

figure;imshow(I);

title('Original image');

pause(1);

%% Resize image

I1=imresize(I,[211 211],'bilinear');
```

```
figure;

imshow(I1);

title('Resized image');

pause(1);

%% Grayscale conversion

current_image=double(I1);

r = current_image(:, :, 1) + 1;

figure;imshow(uint8(r));title('Original RED Panel');

pause(1);

g = current_image(:, :, 2) + 1;

figure;imshow(uint8(g));title('Original GREEN Panel');

pause(1);

b = current_image(:, :, 3) + 1;

figure;imshow(uint8(b));title('Original BLUE Panel');

pause(1);

[n m]=size(r);

transform=@dct2;

inverse_transform=@idct2;

%% for R color

r_dct1 = transform(r);
```

```
% figure;imshow(r_fft1);title('transform of RED Panel')

a1=r_dct1(1,1);

a2=r_dct1(1,m);

a3=r_dct1(n,1);

a4=r_dct1(n,m);

r_dct1=zeros(n,m);

r_dct1(1,1)=a1;

r_dct1(1,m)=a2;

r_dct1(n,1)=a3;

r_dct1(n,m)=a4;

r_dct = r_dct1;

%% for G color

g_dct1 = transform(g);

% figure;imshow(g_fft1);title('transform of GREEN Panel')

b1=g_dct1(1,1);

b2=g_dct1(1,m);

b3=g_dct1(n,1);

b4=g_dct1(n,m);

g_dct1=zeros(n,m);

g_dct1(1,1)=b1;
```

```

g_dct1(1,m)=b2;

g_dct1(n,1)=b3;

g_dct1(n,m)=b4;

g_dct = (g_dct1);

%% for B color

b_dct1 = transform(b);

% figure;imshow(b_fft1);title('transform of BLUE Panel')

c1=b_dct1(1,1);

c2=b_dct1(1,m);

c3=b_dct1(n,1);

c4=b_dct1(n,m);

b_dct1=zeros(n,m);

b_dct1(1,1)=c1;

b_dct1(1,m)=c2;

b_dct1(n,1)=c3;

b_dct1(n,m)=c4;

b_dct = (b_dct1);

%*****

rgbImage=I1;

% imtool(rgbImage);

```

```
% Extract the individual red, green, and blue color channels.
```

```
redChannel = rgbImage(:, :, 1);
```

```
greenChannel = rgbImage(:, :, 2);
```

```
blueChannel = rgbImage(:, :, 3);
```

```
%[lowThresholdR highThreshold lastThresholdedBand] = threshold(83, 255, redChannel)
```

```
% [lowThresholdG highThreshold lastThresholdedBand] = threshold(83, 255, greenChannel)
```

```
% [lowThresholdB highThreshold lastThresholdedBand] = threshold(83, 255,
```

```
blueChannel) binaryImage = redChannel < 173;
```

```
clearbord = imclearborder(binaryImage);
```

```
figure;
```

```
imshow(clearbord);
```

```
title('Boundary Image', 'FontSize', fontSize);
```

```
pause(1);
```

```
%% Fill holes
```

```
HolesFilledImage = imfill(binaryImage, 'holes');
```

```
figure;
```

```
imshow(HolesFilledImage);
```

```
title('Holes Filled image', 'FontSize', fontSize);
```

```
pause(1);
```

```
background = imopen(binaryImage,strel('disk',15));

Ip = imsubtract(binaryImage,background);

figure, imshow(Ip,[])

title('Background ignored image', 'FontSize', fontSize);

pause(1);


skinBinaryImage = bwareaopen(binaryImage, 1000);

% figure;

% subplot(2,2,1);

% imshow(skinBinaryImage);

% title('skin Binary Image', 'FontSize', fontSize)

count=0;

skinImage = redChannel < 60 & greenChannel < 40 & blueChannel < 40;

% subplot(2,2,2);

% imshow(skinImage);

% title('skin Binary Image', 'FontSize', fontSize);

combinedmask = (skinBinaryImage & ~skinImage);

figure;

% subplot(2,2,3);

imshow(combinedmask);
```

```

title('Masked Image', 'FontSize', fontSize);

pause(1);

Tumor_image = redChannel <60 & greenChannel < 40 & blueChannel < 40;

area=sum(sum(Tumor_image));

figure;

% subplot(2,2,4);

imshow(Tumor_image);

title('Tumor Detected', 'FontSize', fontSize);

pause(1);

%% Deep neural network

hiddenLayerSize = 4;

net = patternnet(hiddenLayerSize);

net.inputs{1}.processFcns = {'removeconstantrows','mapminmax'};

net.outputs{2}.processFcns = {'removeconstantrows','mapminmax'};

net.divideFcn = 'dividerand'; % Divide data randomly

net.divideMode = 'sample'; % Divide up every sample

net.divideParam.trainRatio = 70/100;

net.divideParam.valRatio = 15/100;

net.divideParam.testRatio = 15/100;

net.trainFcn = 'traincgb'; % conjugate gradient backPropagation

```

```
net.performFcn = 'mse'; % Mean squared error

net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
'plotregression', 'plotfit'};

inputs=[1 2 3 4];

targets=[0 1 1 0];

% load feature

[net,tr] = trainrp(net,inputs,targets);

outputs = net(inputs);

ktry=net(targets);

%

errors = gsubtract(ktry,outputs);

targets=ktry;

performance = perform(net,targets,outputs)

trainTargets = targets .* tr.trainMask{1};

valTargets = targets .* tr.valMask{1};

testTargets = targets .* tr.testMask{1};

trainPerformance = perform(net,trainTargets,outputs);

valPerformance = perform(net,valTargets,outputs);

testPerformance = perform(net,testTargets,outputs);

% View the Network
```

```

view(net)

if area<10

%   msgbox('Normal');

    test='Normal';

elseif (area >10)& (area<300)

%       msgbox('abnormal-Stage1');

        test='abnormal-Stage1';

elseif (area >300)& (area<500)

%       msgbox('abnormal-Stage2');

        test='abnormal-Stage2';

elseif (area >500)

%       msgbox('abnormal-Stage3');

        test='abnormal-Stage3';

end

out=test;

client(out);

%% Client image

% CLIENT connect to a server and read a message

% Usage - message = client(host, port, number_of_retries)

function message = client_image(host, port)

```

```

import java.net.Socket

import java.io.*

host='192.168.1.2';

port=8899;

if (nargin < 3)

    number_of_retries = 200; % set to -1 for infinite

end

retry      = 0;

input_socket = [];

message     = 0;

while retry<50

    retry = retry + 1;

    if ((number_of_retries > 0) && (retry > number_of_retries))

        fprintf(1, 'Too many retries\n');

        break;

    end

    try

```

```

fprintf(1, 'Retry %d connecting to %s:%d\n', ...

        retry, host, port);

input_socket = Socket(host, port);

output_stream = input_socket.getOutputStream();

input_stream  = input_socket.getInputStream();

d_output_stream = DataOutputStream(output_stream);

fprintf(1, 'Connected to server\n');

d_output_stream.writeBytes(char('IMAGE'));

d_output_stream.flush()

pause(.5);

fprintf(1, 'Reading bytes\n');

message=input_stream.read();

fileID = fopen('du.jpg','w');

fprintf(1, 'Reading bytes\n');

fprintf(1, 'Reading bytes\n');

while message ~= -1

    fwrite(fileID, message);

%       fprintf(1,'Chk');

message=input_stream.read();

```

```

        end

        fprintf(1, 'Reading bytes\n');

        fclose(fileID);

        fprintf(1, 'finish file\n');

        input_socket.close;

            fprintf(1, 'Client Close\n');

        break;

catch me

            fprintf(1, 'ERROR %s \n',me.message);

            if ~isempty(input_socket)

                input_socket.close;

            end

        pause(1);

    end

end

end

end

% CLIENT connect to a server and read a message

% Usage - message = client(host, port, number_of_retries)

function message = client(test)

```

```
import java.net.Socket

import java.io.*

host='192.168.1.2';

port=8899;


if (nargin < 3)

    number_of_retries = 200; % set to -1 for infinite

end


retry      = 0;

input_socket = [];

message     = [];


while retry<50


    retry = retry + 1;

    if ((number_of_retries > 0) && (retry > number_of_retries))

        fprintf(1, 'Too many retries\n');

        break;

    end

end
```

```

try

    fprintf(1, 'Retry %d connecting to %s:%d\n', ...

        retry, host, port);

% throws if unable to connect

input_socket = Socket(host, port);

% get a buffered data input stream from the socket

    output_stream = input_socket.getOutputStream;

%     input_stream  = input_socket.getInputStream;

    d_output_stream =

        DataOutputStream(output_stream);

%     d_input_stream = DataInputStream(input_stream);

fprintf(1, 'Connected to server\n');

%     pause(.5);

    d_output_stream.writeBytes(char('DATA'));

    d_output_stream.writeBytes(test);

%     output_stream.writeBytes(char('DATA'));

%     output_stream.write('TEST');

%     output_stream.writeBytes(char('DATA'));

%     output_stream.write('TEST');

```

```
%      output_stream.writeBytes(char('DATA'));
```

```
%      output_stream.write('TEST');

%      output_stream.writeBytes(char('DATA'));

%      output_stream.write('TEST');

% read data from the socket - wait a short time first

pause(1);

%      bytes_available = input_stream.available;

%      fprintf(1, 'Reading %d bytes\n', bytes_available);

%

%      message = zeros(1, bytes_available, 'uint8');

%      for i = 1:bytes_available

%          message(i) = d_input_stream.readByte;

%      end

%

%      message = char(message);


% cleanup

input_socket.close;

      fprintf(1, 'Client Close\n');

break;

catch me
```

```
fprintf(1, 'ERROR %s \n',me.message);

    if ~isempty(input_socket)

        input_socket.close;

    end

    % pause before retrying

    pause(1);

end

end

end
```