DEEP LEARNING FOR THYROID CANCER DETECTION USING THERMOGRAPHY

A PROJECT REPORT

Submitted by

KARTHICK S	1807026
PRABAGARAN R	1807041
SUBHASH M	1807050

In partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

COIMBATORE INSTITUTE OF TECHNOLOGY

(Government Aided Autonomous Institution Affiliated to Anna University)
(Accredited by National Board of Accreditation Council)
Coimbatore-641014
Tamil Nadu
ANNA UNIVERSITY, CHENNAI 600025
(APRIL 2021)

COIMBATORE INSTITUTE OF TECHNOLOGY

(A Govt. Aided Autonomous Institution Affiliated to Anna University)
COIMBATORE – 641014

BONAFIDE CERTIFICATE

Certified that this project report titled "**THYROID CANCER DETECTION USING THERMOGRAPHY**" is the bonafide work of **KARTHICK (1807026), PRABAGARAN (1807041)** and **SUBHASH (1807050),** who carried out the project work under my supervision.

Dr.N.K. KARTHIKEYAN Ph.D.,	**Ms.S.RAJASREE,**
HEAD OF THE DEPARTMENT,	**SUPERVISOR,**
Department of Information Technology,	Department of Information Technology,
Coimbatore Institute of Technology,	Coimbatore Institute of Technology,
Coimbatore - 641014.	Coimbatore - 641014.

Certified that the candidates were examined by us in the project work viva-voce examination held on …………………

Internal Examiner **External Examiner**

Place: Coimbatore
Date:

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We express our sincere thanks to our Secretary **Dr.R.Prabhakar** and our Principal **Dr.V.Selladurai** for providing us a greater opportunity to carry out our work. The following words are rather very meager to express our gratitude to them. This work is the outcome of their inspiration and product of their plethora of knowledge and rich experience.

We record the deep sense of gratefulness to **Dr.N.K.Karthikeyan**, Head of the department of Information Technology, for his encouragement during this tenure.

We equally tender my sincere thankfulness to **Ms.S.RAJASREE**, our supervisor, Department of Information Technology, for her valuable suggestions and guidance during this course.

During the entire period of study, the entire staff members of the Information Technology have offered ungrudging help. It is also a great pleasure to acknowledge the unfailing help we have received from our friends.

It is a matter of great pleasure to thank our parents and family members for their constant support and cooperation in the pursuit of this endeavor. We express our sincere thanks to God Almighty for His cherished blessings.

ABSTRACT

ABSTRACT

In this modern world a lot of people suffer from cancer diseases, about 1 in 6 deaths is due to cancer. Among them thyroid cancer is also one of the deadliest diseases. With the help of Convolution Neural Network (CNNs) techniques and the dynamic infrared thermography (IRT) can be used to detect the cancer tissues in the human body. This technique can be used for detecting body disorders based on an abnormal temperature rise. As from the researches made before it has confirmed that thyroid tumors are higher in temperature in comparison to the thyroid gland. In this project IR thermal image of both healthy people and thyroid cancer affected people are taken. The thermal images undergo specific image processing algorithms for the image segmentation and noise reduction. The contrast between image components is increased by a cooling process during thermal imaging. Thermal imaging process indicates a temperature rise of 1-1.5 °C in front of the thyroid cancer tissues when compared to the surrounding healthy tissues. CNN architecture can be used to detect tumors of size less than 0.5 mm .Parameters of the cancerous tumor are determined by the thermal classification of the processed thermal images. Results show that, from the processed thermal images and the parameters obtained we can predict that person is affected or not.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

CHAPTER 1

INTRODUCTION

THYROID CANCER DETECTION

Cancer is the uncontrolled proliferation of cells that may result in the creation of tumors. Cancerous tumors disturb the normal activity of the host tissue. The thyroid gland is an endocrine gland in your neck. It makes two hormones that are secreted into the blood: thyroxine and triiodothyronine. These hormones are necessary for all the cells in your body to work normally. Thyroid disorders are very common and tend mainly to occur in women, although anybody - men, teenagers, children and babies, too - can be affected. About one in 20 people has some kind of thyroid disorder, which may be temporary or permanent. Thyroid cancer is commonly diagnosed at a younger age than most other adult cancers. And women are 3 times more likely to develop thyroid cancer than men. The main reasons for this difference in female than male are polymorphism's role in female oestrogen receptors. Oestrogen also significantly increases the amount of cell proliferation in cancerous thyroid cells compared to the male hormone.

Thyroid cancer was the most rapidly increasing cancer in the world, largely due to increased detection. Much of this rise appears to be the result of the use of more sensitive diagnostic procedures, such as CT or MRI scans, which can detect incidental small thyroid nodules that might not otherwise have been found in the past.

The American Cancer Society's most recent estimates for thyroid cancer in the United States for 2021 are:
About 44,280 new cases of thyroid cancer (12,150 in men and 32,130 in women).
About 2,200 deaths from thyroid cancer (1,050 men and 1,150 women).

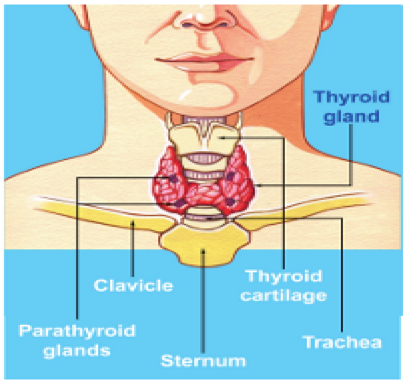


Fig-1.1

The human neck mainly includes the thyroid gland, trachea, common carotid artery (CCA), internal jugular vein (IJV), muscle, subcutaneous fat, and the skin. The CCA brings oxygenated blood to the neck and head, and the IJV collects the blood from the head and face and brings it back to the heart. The thyroid gland is an endocrine gland with a butterfly shape and is located in the front of the neck, under the thyroid cartilage and in front of the trachea, IJV and CCA. This gland makes hormones by using the iodine and controls the metabolism and the energy balance of the body.

Physiologically, thyroid diseases produce some heat around the affected area. By using dynamic thermography and CNN we can find the affected area in the early stages as small as possible. This method gives better accuracy and improves the model training time.

THYROID CANCER DIAGNOSIS

Thyroid cancer diagnosis can take place in different ways. Some of them are:

- Physical examination
- Blood tests
- Ultrasound imaging
- Removing a sample of thyroid tissue
- Other imaging tests
- Genetic testing

In a physical examination doctor will examine your neck to feel for physical changes in your thyroid, such as thyroid nodules. He or she may also ask about your risk factors, such as past exposure to radiation and a family history of thyroid tumors.

Blood tests help determine if the thyroid gland is functioning normally.

Ultrasound uses high-frequency sound waves to create pictures of body structures. To create an image of the thyroid, the ultrasound transducer is placed on your lower neck. The appearance of your thyroid on the ultrasound helps doctors determine whether a thyroid nodule is likely to be noncancerous (benign) or whether there's a risk that it might be cancerous.

During sample removing diagnosis a fine-needle aspiration biopsy, the doctor inserts a long, thin needle through your skin and into the thyroid nodule. Ultrasound imaging is typically used to precisely guide the needle into the nodule. Doctor uses the needle to remove samples of suspicious thyroid tissue. The sample is analyzed in the laboratory to look for cancer cells.

We may have one or more imaging tests to help doctors determine whether cancer has spread beyond the thyroid. Imaging tests may include CT, MRI and nuclear imaging tests that use a radioactive form of iodine.

In Genetic testing people with medullary thyroid cancer may have genetic changes that can be associated with other endocrine cancers. Patient family history may prompt doctors to recommend genetic testing to look for genes that increase your risk of cancer.

In this developing world, artificial intelligence and machine learning are taking their lead role in all fields such as automobile, home automation, etc. This artificial intelligence and machine learning also made a great impact on the medical field. With the help data available today , artificial intelligence and machine learning both are able to predict , analyze, classify various types of diseases.

Generally , they started with the medical records of patients. They initially collect huge amounts of medical records , and process them . From the processed results they build a model to do what the things they want.

At the beginning state they process the textual data from reports. After advancement in ML and AI they also started to work on medical images to predict the diseases. This mainly depends on the image processing algorithms.

With the help of Image processing techniques, we are able to predict the diseases from images of x-rays, ultrasound , MRI images, thermal images, etc ,.

The medical images initially undergo the process of pre-processing. After pre-processing the required features are extracted from the images. With the obtained features of images , we are able to train the model to predict or detect the diseases.

The features extracted are generally passed to artificial neural networks (ANN) or Convolutional neural networks (CNN) to generate weighted images. With the help of weighted images obtained the input image features are compared with the weighted images stored and able to predict the diseases.

LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

Inorder to understand the problems in thyroid cancer detection using thermography and also to know more about the existing models to predict and detect the thyroid cancer we made survey on lot of research pages .The understandings form the survey are formed into some points and we have listed them below:

2.1.1. Thyroid cancer estimation using infrared thermography data (2019)

The infrared (IR) thermal imaging could be utilized as a noninvasive tool for detecting body disorders based on an abnormal temperature rise. In the present study, the IR thermography is employed for the detection of malignant thyroid tumors. Previous studies on the thyroid thermography have confirmed the higher temperature of thyroid tumors in comparison to the thyroid gland, which appears as hot spots and disturbs the symmetry of the thermogram. However, the thyroid thermography has clinical significance if thyroid cancer could be estimated by studying the IR thermal image. The study of the processed thermal images indicates a local temperature rise of 1–1.5 °C in front of the thyroid cancerous tumor compared to the surrounding healthy tissue.

2.1.2. Methodology

In this project they made use of thermal images of 10 healthy and 8 affected people. This IR images are then send to image preprocessing:

i) It undergoes image segmentation.

ii) Then noise reduction by anisotropic diffusion filter.

iii) Then the region of interest is found.

iv) Finally the processed images are sent to ANN module for prediction.

2.1.3. Advantages

This project is very good in localization and edge detection. Hence it can be used to find the accurate size of tumors. And also able to find small sizes of tumors.

2.1.4. Disadvantages

This project has very processing of blur images and it leads to loss of accuracy.

2.2.1. Thermal Imaging and Deep Learning Approaches for Breast Cancer Detection (2020)

In this project ,a breast cancer screening method is used to facilitate early breast cancer detection and treatment. Building a screening method using medical imaging modality that does not cause body tissue damage (non-invasive) and does not involve physical touch is challenging. Thermography, a non-invasive and non-contact cancer screening method, can detect tumors at an early stage even under precancerous conditions by observing temperature distribution in both breasts. The thermograms obtained on thermography can be interpreted using deep learning models such as convolutional neural networks (CNNs). CNNs can automatically classify breast thermograms into categories such as normal and abnormal.

2.2.2. Methodology

This project makes use of CNN to detect breast cancer.The process involves are as follows:

i) Image pre-processing
ii) Convolution Neural Networks
iii) Image Classification
iv) Backpropagation

2.2.3. Advantages

This project helps to reduce the time of processing the image in convolutional neural networks. It also helps to increase the accuracy of results.

2.2.4. Disadvantages

Sometimes leads to false prediction rates.

2.3.1. Automatic cancer tissue detection using multispectral photoacoustic imaging (2019)

In this project, they make use of sonography images and CNN modules to detect the cancer automatically. In the case of multi specimen study to locate cancer regions, such as in thyroidectomy and prostatectomy, significant labor-intensive processing is required at a high cost. Pathology diagnosis is usually done by a pathologist observing tissue-stained glass slides under a microscope.As it is a time consuming process, we move on to automatic cancer detection using photoacoustic imaging.

2.3.2. Methodology

Multispectral photoacoustic (MPA) specimen imaging has proven successful in differentiating photoacoustic (PA) signal characteristics between a histopathology-defined cancer region and normal tissue. This is mainly due to its ability to efficiently map oxyhemoglobin and deoxyhemoglobin contents from MPA images and key features for cancer detection. A fully automated deep learning algorithm is purposed, which learns to detect the presence of malignant tissue in freshly excised ex vivo human thyroid and prostate tissue specimens using the three-dimensional MPA dataset. The proposed automated deep learning model consisted of the convolutional neural network architecture, which extracts spatially colocated features, and a softmax function, which detects thyroid and prostate cancer tissue at once. This is one of the first deep learning models, to the best of our knowledge, to detect the presence of cancer in excised thyroid and prostate tissue of humans at once based on PA imaging.

2.3.3. Advantages

This CNN model showed similar sensitivity and improved specificity in identifying patients with thyroid cancer compared with a group of skilled radiologists.

2.3.4. Disadvantages

This project takes more time for training and testing the images.

2.4.1. Diagnosis of thyroid cancer using deep convolutional neural network models applied to sonographic images (2019)

In this project, the ultrasound images of thyroid is used to detect the cancer.The incidence of thyroid cancer is rising steadily because of overdiagnosis and overtreatment conferred by widespread use of sensitive imaging techniques for screening. This overall incidence growth is especially driven by increased diagnosis of indolent and well-differentiated papillary subtype and early-stage thyroid cancer, whereas the incidence of advanced-stage thyroid cancer has increased marginally. Thyroid ultrasound is frequently used to diagnose thyroid cancer. The aim of this study was to use deep convolutional neural network (DCNN) models to improve the diagnostic accuracy of thyroid cancer by analysing sonographic imaging data from clinical ultrasounds

2.4.2. Methodology

In this project 131 731 ultrasound images from 17 627 patients with thyroid cancer and 180 668 images from 25 325 controls from the thyroid imaging database at Tianjin Cancer Hospital are taken for processing. All individuals with suspected thyroid cancer after clinical examination in the validation sets had pathological examination. We also compared the specificity and sensitivity of the DCNN model with the performance of six skilled thyroid ultrasound radiologists on the three validation sets.

2.4.3. Advantages

The DCNN model showed similar sensitivity and improved specificity in identifying patients with thyroid cancer compared with a group of skilled radiologists.

2.4.4. Disadvantages

The sonographic images have less accuracy in detection of tumors.

2.5.1. Breast cancer diagnosis with a microwave thermoacoustic imaging technique (2019)

In this project it makes use of MITAI. Microwave-induced thermoacoustic imaging (MITAI) is an imaging technique with great potential for detecting breast cancer at early stages. Thermoacoustic imaging (TAI) combines the advantages of both microwave and ultrasound imaging techniques. In the current study, a three-dimensional novel numerical simulation of the TAI phenomenon as a multi-physics problem is investigated.

2.5.2. Methodology

Biological breast tissue including three different tissue types along with a tumor is placed in a tank containing castor oil and is irradiated by a 2.45-GHz pulsed microwave source from a rectangular waveguide. The generated heat in the biological tissue due to the electromagnetic wave irradiation and its corresponding pressure gradient in the tissue because of the temperature variations are evaluated. Also, capability of the MITAI process with respect to the tumor location and size is investigated. To identify the required power level needed for producing thermoacoustic signals, different power levels of microwave sources are investigated. The study's results demonstrate a minuscule increase in temperature as a result of the absorption of pulsed microwave energy. This small temperature variation in the tumor produces several kilopascals of pressure variations with a maximum of 0.584016 kPa in the tumor. This pressure variation will produce acoustic signals, which can be detected with an array of transducers and be used for image construction. Results demonstrate that the location of a tumor in the breast plays a vital role in the detecting performance of MITAI.

2.5.3. Advantages

Very small tumors (with the diameter of 0.5 cm) can also be detected using MITAI technique.

2.5.4. Disadvantages

Cost of this process is high.It also makes use of a lot other attributes such as pressure variation,magnitude.

2.6.1. Breast cancer detection using thermography and convolution neural networks (2020)

In this project, the Thermography and Convolution Neural Networks(CNN) are used to predict breast cancer. Thermography is an entirely non-invasive and non-contact imaging technique that is widely used in the medicinal field.

2.6.2. Methodology

A new algorithm is proposed for the extraction of the breast characteristic features based on bio-data, image analysis, and image statistics. These features have been extracted from the thermal images captured by a thermal camera, and will be used to classify the breast images as normal or suspected by using convolutional neural networks (CNNs) optimized by Bayes algorithm. By using this algorithm, a 98.95% accuracy rate was obtained for the thermal images in the dataset belonging to 140 individuals.

2.6.3. Advantages

This method can increase the early diagnosis rate of breast cancer.

2.6.4. Disadvantages

The processing time for training the model is very high.

2.7.1.Infrared Thermal Imaging-Based Crack Detection Using Deep learning (2019)

Vision-based approaches are widely used in steel crack detection. After processing the images taken by the camera, the superficial defects can be detected. Due to the common limitation of the nature of photographic images, internal features of objects cannot be fully discovered. In order to overcome the drawbacks of vision-based methods, this work presents an approach for detecting cracks in infrared thermal imaging steel sheets using Convolutional Neural Networks (CNN).

2.7.2 . Methodology

Thermal imaging cameras convert the energy in the infrared wavelength into a visible light display. All objects above absolute zero emit thermal infrared energy, so thermal cameras can passively see all objects, regardless of ambient light.

2.7.3 Advantages

This has a wide range of uses in the construction, agriculture, security, and surveillance industries.

2.7.4 Disadvantages

Infrared frequencies are affected by hard objects (e.g. walls, doors) , smoke, dust, fog, sunlight etc. Hence it does not work through walls or doors.

2.8.1.Automatic detection of moistures in different construction materials from thermographic images (2019)

Moisture is a pathology that damages all types of construction materials, from materials of building envelopes to materials of bridges. Its presence can negatively affect the users' conditions of indoor comfort. Furthermore, heating and cooling energy demand can be increased by the presence of moist materials. Infrared thermography (IRT) is a common technique in the scientific field to detect moisture areas, because of its non-destructive, non-contact nature.

2.8.2 . Methodology

Infrared thermography is equipment or method, which detects infrared energy emitted from an object, converts it to temperature, and displays an image of temperature distribution. To be accurate, the equipment and the method should be called differently, the equipment to be called as infrared thermography and the method to be called as infrared thermography.

2.8.3 Advantages

Infrared imaging cameras are instrumental in identifying and quantifying heat sources.

2.8.4 Disadvantages

Infrared waves at high power can damage eyes.

2.9.1.Neural Network Based Brain Tumor Detection Using Wireless Infrared Imaging (2019)

This imaging sensor is integrated via Wireless Infrared Imaging Sensor which is produced to transmit the tumor warm data to a specialist clinician to screen the wellbeing condition and for helpful control of ultrasound measurements, especially if there should arise an occurrence of elderly patients living in remote zones.

2.9.2 . Methodology

Back-propagation is the essence of neural net training. It is the method of fine-tuning the weights of a neural net based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and to make the model reliable by increasing its generalization.

2.9.3 Advantages

Backpropagation is fast, simple and easy to program.

It has no parameters to tune apart from the numbers of input.

2.9.4 Disadvantages

The actual performance of backpropagation on a specific problem is dependent on the input data.

2.10.1.Machine Learning in Rehabilitation Assessment for Thermal and Heart Rate Data Processing (2018)

Multimodal signal analysis based on sophisticated noninvasive sensors, efficient communication systems, and machine learning, have a rapidly increasing range of different applications. The present paper is devoted to pattern recognition and the analysis of physiological data acquired by heart rate and thermal camera sensors during rehabilitation.

2.10.2. Methodology:

Neural networks are a series of algorithms that mimic the operations of a human brain to recognize relationships between vast amounts of data. They are used in a variety of applications in financial services, from forecasting and marketing research to fraud detection and risk assessment.

2.10.3 Advantages:

Neural Networks have the ability to learn by themselves and produce the output that is not limited to the input provided to them.

2.10.4 Disadvantages:

ANNs are used for problems having the target function, the output may be discrete-valued, real-valued, or a vector of several real or discrete-valued attributes.

SYSTEM ARCHITECTURE

CHAPTER 3

SYSTEM ARCHITECTURE

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.The System architecture of our project is given below:

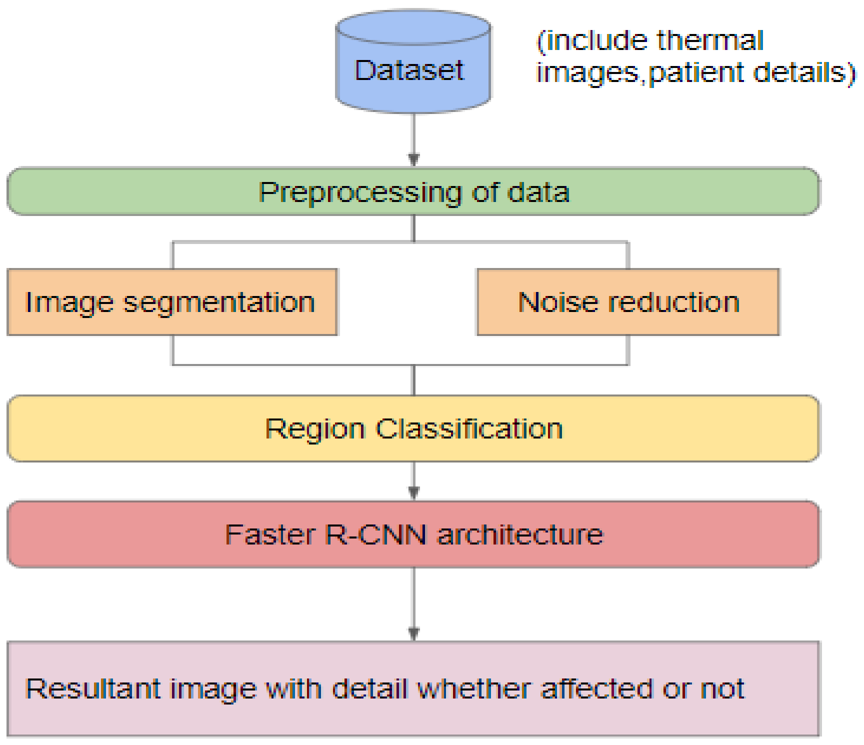

Dataset
(include thermal images,patient details)
Preprocessing of data
Image segmentation
Noise reduction
Region Classification
Faster R-CNN architecture
Resultant image with detail whether affected or not

Fig 3.1- Overall System Architecture

From fig-3.1 , we can infer that the dataset containing thermal images are pre-processed first. The pre-processing of images include image segmentation and noise reduction. After noise reduction region classification takes place.

Then the images are passed to a CNN module for generating weight of images. And we send the predicting images . Resultant we get details like whether affected or not.

FLOW CHART

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task.The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

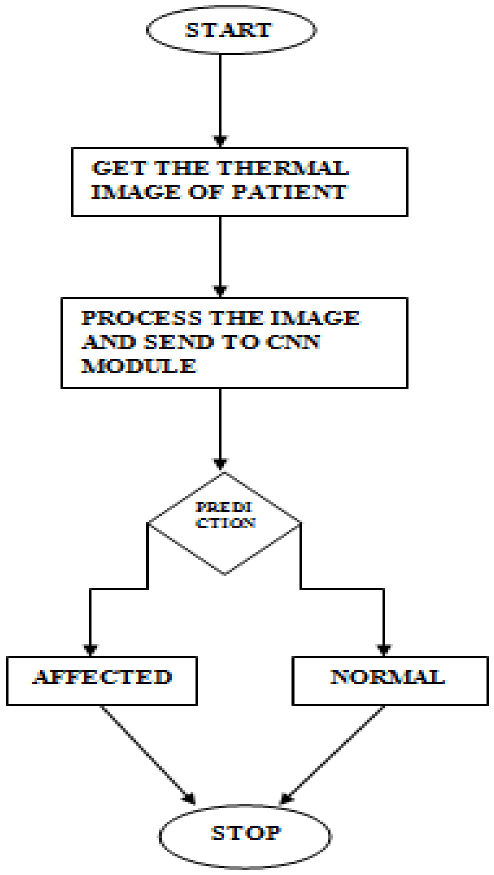

START
GET THE THERMAL IMAGE OF PATIENT
PROCESS THE IMAGE AND SEND TO CNN MODULE
PREDI CTION
AFFECTED
NORMAL
STOP

Fig 3.2-Functional diagram

Fig-3.2 represents the flow chart of our project.This clearly shows the step-by-step process involved in our project

PROJECT FLOW DIAGRAM

Process Flow Diagrams (PFDs) are a graphical way of describing a process, its constituent tasks, and their sequence. A PFD helps with the brainstorming and communication of the process design. The PFMEA process needs a complete list of tasks that comprise the process under analysis.

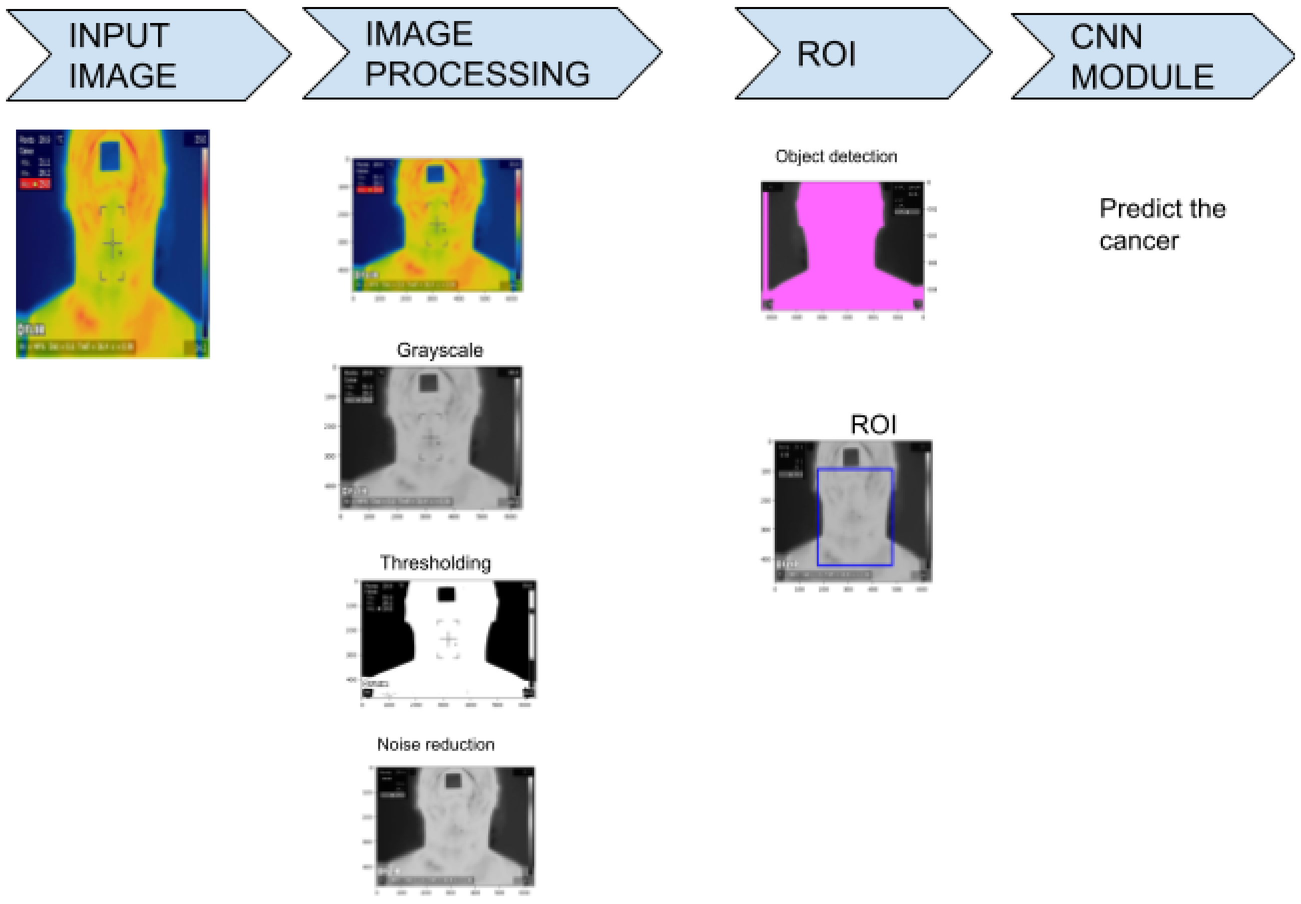


Fig 3.3- Project Flow diagram

Fig-3.3 represents the project flow diagram. From the PFD we infer that this project consists of three modules namely image pre-processing, region of interest identification and CNN architecture.

SYSTEM SPECIFICATIONS

CHAPTER 4

SYSTEM SPECIFICATIONS

The hardware and software for the system is selected by considering the factors such as CPU processing speed, peripheral channel speed, printer speed, seek time, relational delay of hard disk and communication speed etc. The hardware and software specifications are as follows.

HARDWARE REQUIREMENTS

Processor	Intel(R) Core(TM)i5-7200U
RAM	8GB
Hard Disk Drive	80GB
Keyboard	Optical Mouse
Monitor	SVGA/color

Table 4.1- Hardware requirements

Table 4.1- represents the essential hardware requirements for our project.

SOFTWARE REQUIREMENTS

Operating System	Windows 10
IDE used	Google colab
Language Used	Python

Table 4.2- Software requirements

Table 4.2- represents the essential software requirements needed for our project.

SYSTEM DESIGN

CHAPTER 5

SYSTEM DESIGN

Our proposed model makes use of thermography and CNN architecture to detect cancer in the human.This proposed model consist of following modules:

1. Data preprocessing
2. ROI
3. CNN Module

5.1.DATA INPUTS

In this we make use of thermal images obtained from this http://visual.ic.uff.br/thyroid website.We take thermal data of both healthy and affected people for processing. Here we make use of thermal data of 15 affected people and 9 healthy people . We also make use of other parameters like age, gender ,skin color etc. for processing.

5.2.DATA PREPROCESSING

First the obtained thermal images need to be segmented. The image segmentation consist of two process namely:

1. Image segmentation
2. Noise reduction

5.2.1.IMAGE SEGMENTATION

The input thermal image which we have taken may consist of a lot of unwanted background objects. Inorder to get good results the background of images should be removed. This process of removing the unwanted background objects is called Image Segmentation.

Before moving on to the segmentation part we need to know a little bit about images. Generally the colour images which we make use every day consist of 3 channels of colour. Namely RGB(Red, Green,Blue) channel.

To start the process of image segmentation, first we need to convert the thermal image (3 channel colour) to grayscale image (single channel colour).
This grayscale images will be more useful for handling the image processing.

Then we move on to the next step of image thresholding. If the object is lighter than background then thresholding is done . If an object is darker than background then the inverse of it is done.

5.2.2.NOISE REDUCTION

Noise is a great problem in all image processing techniques.

As the thermal data we make use consist a lot of noise. We need to remove that noise for better results.

5.3.REGION OF INTEREST

While processing the whole part of the image , we may lose some features in it. In Order to avoid this problem we make use of only selective regions for processing.

In our project, we need only the neck region for processing. So we need to select the neck region from the whole image. This can be done with the help of ROI (Region Of Interest).
Here we make use of rectangle ROI and we select the neck region.

5.4.CNN MODULE

A CNN architecture is formed by a stack of distinct layers that transform the input volume into an output volume through a differentiable function. A few distinct types of layers are commonly used. These are further discussed below:

Convolutional layer

The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume. During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the filter entries and the input, producing a 2-dimensional activation map of that filter. As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input.

Pooling layer

Another important concept of CNNs is pooling, which is a form of non-linear down-sampling. There are several non-linear functions to implement pooling, where max pooling is the most common. It partitions the input image into a set of rectangles and, for each such sub-region, outputs the maximum.

$$f_{X,Y}(S) = \max_{a,b=0}^{1} S_{2X+a,2Y+b}$$

ReLU layer

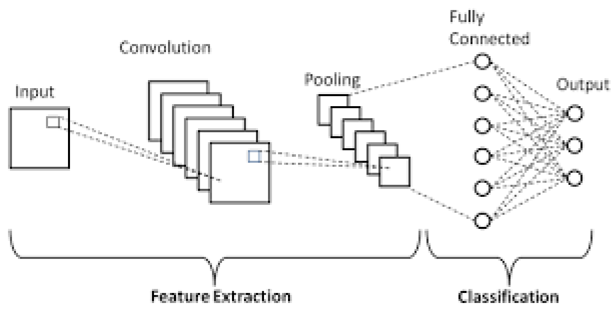
ReLU is the abbreviation of rectified linear unit, which applies the non-saturating activation function . It effectively removes negative values from an activation map by setting them to zero. It introduces nonlinearities to the decision function and in the overall network without affecting the receptive fields of the convolution layers.

$$f(x) = \max(0, x).$$

[70]

Fully connected layer

After several convolutional and max pooling layers, the final classification is done via fully connected layers. Neurons in a fully connected layer have connections to all activations in the previous layer, as seen in regular (non-convolutional) artificial neural networks.



After completion of the above to processes, the obtained results are sent to CNN Architecture for prediction of cancer.

First , the features are extracted from the given input images.

Second, we generate the weights for processing.

At last with the help of obtained weights we predict whether the given input image of a person is affected or not.

5.5. OUTPUT

Computer output is the most important and direct source of information to the user. Efficient and intelligent output design improves the system relationship with the user and helps in decision making. A major form of output is that the person is affected or normal.

IMPLEMENTATION

CHAPTER 6

IMPLEMENTATION

The functioning of all the modules can be well understood by portraying their Pseudo code as follows.

ALGORITHM

- Input (IMG) // IMG – Thermal Image of person
- Output (Whether the person is affected or not)

- Step 1: Collect the image from the dataset.
- Step 2: Pre-process of the images start with Image segmentation.
- Step 3: Image segmentation includes binary thresholding and noise reduction.
- Step 4:From the segmented images the region of interest is extracted.
- Step 5: The extracted image is stored separately.
- Step 6: These images are then passed to CNN Module.
- Step 7: Split the dataset into train and test data .
- Step 8: The CNN module ,generate the weights and train the model for predicting cancer .
- Step 9:For the given test input images , Predict that input images are affected or not.

SYSTEM TESTING

CHAPTER 7

SYSTEM TESTING

This system uses thermal images as an input nearly 25 images of both healthy and affected people. The image should be in a noise free background to attain accuracy. The proposed system is used for comparison of different Machine learning algorithms for higher accuracy. The image is then separated into training and testing dataset. From the data set nearly 70% is used for training and remaining 30% is used for testing. The training and testing are then passed to the CNN module. To improve the accuracy, parameter tuning is done which results in gradual increase of the accuracy. To test the module , real time images are used.

RESULT ANALYSIS

CHAPTER 8

RESULT ANALYSIS

In this project we give thermal images of patients as inputs.Here we make use of 15 unhealthy and 9 healthy patients thermal images for processing.Then after building the model successfully , when we pass the image to predict. It will process it and return the output as patient affected or not along with their image.

Step 1: Read the image from the dataset

Step 2:

The input image will is as follows:

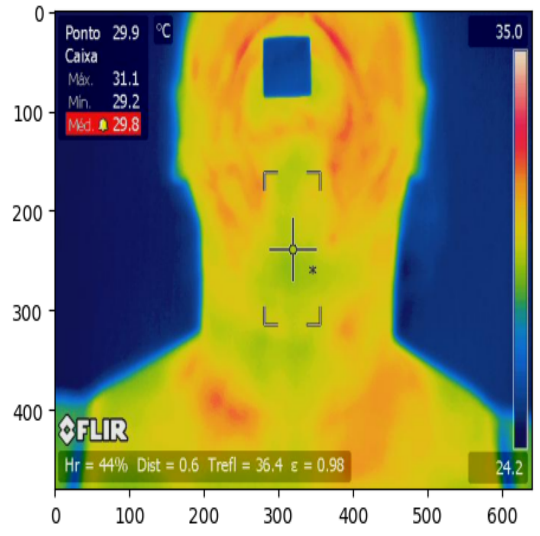


Figure 8.1- Original Image

Fig 8.1 - represent the input thermal image sample which we are using

Step 3:

Now , Convert the thermal image which has 3 channel color (RGB) to a Single Channel (grayscale)image. For an easy way of processing and also to reduce the difficulties in processing.

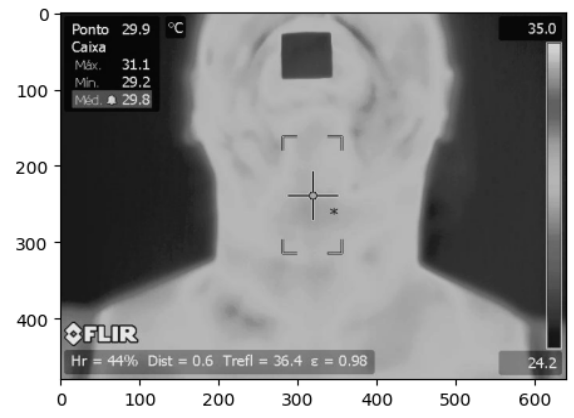


Figure 8.2gray scaled image

Fig 8.2 - represents the grayscale image of thermal data

Step 4:

To remove the background of the image , we use binary thresholding . If the object is lighter than background then thresholding is done . If an object is darker than background then the inverse of it is done.

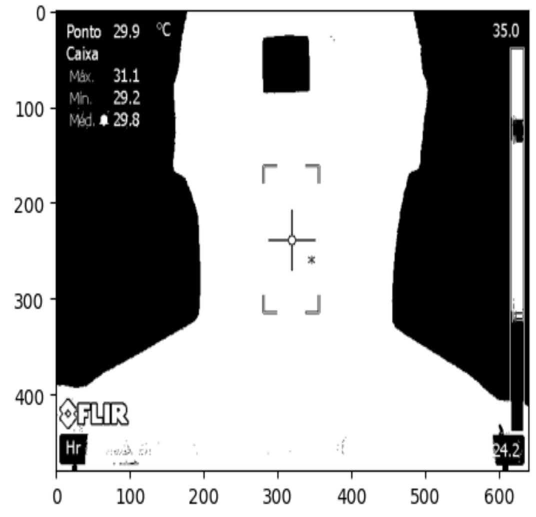


Figure 8.3- Threshold Image

Fig 8.3- represents the thresholded image of thermal data

Step 5:

To remove the noise from the image, in order to increase the accuracy .We make use of gaussian blur to remove the noise.

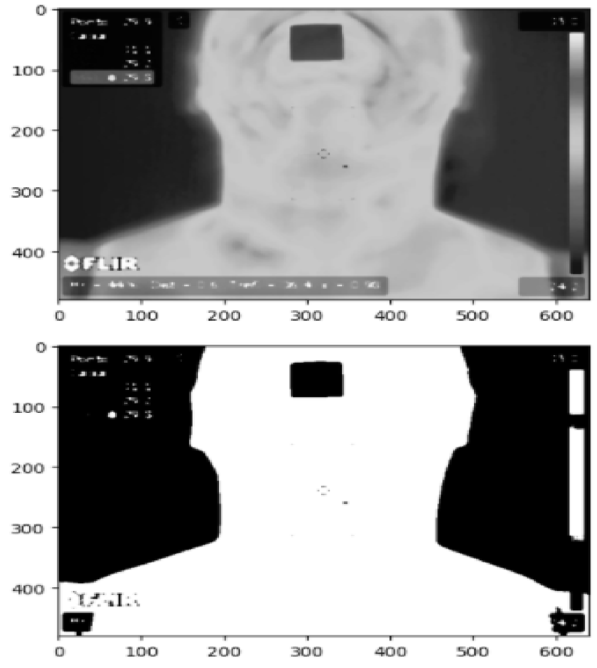


Figure 8.4 - Noise reduced image

Fig 8.4 -represent the noise removed image of thermal data

Step 6:

Make use of counters, and find the required rectangular ROI . Now extract the coordinates of ROI and save the ROI image .

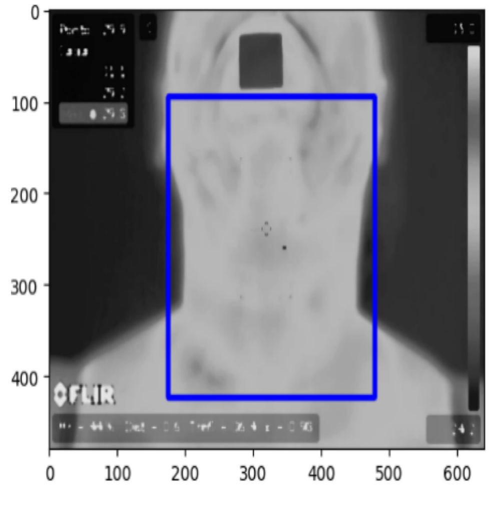
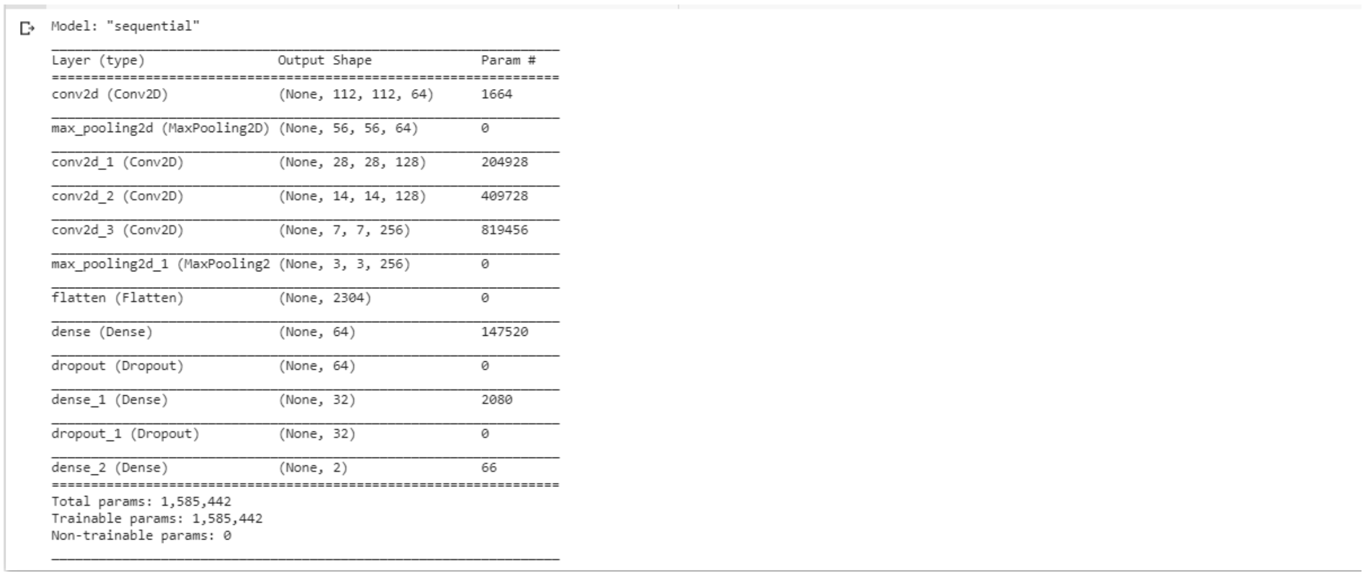


Figure 8.5 - ROI image

Fig 8.5- represents the ROI out of the given thermal image

Step 7:

Build the CNN Module.



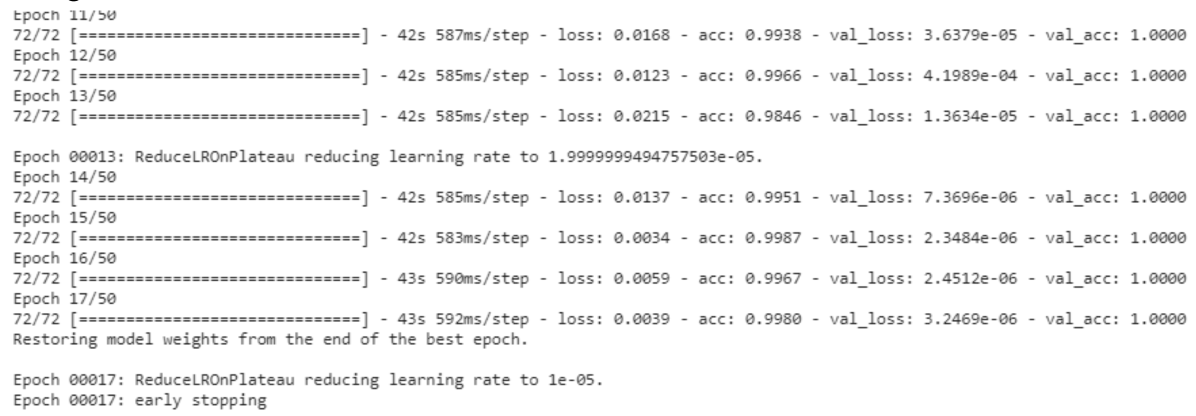
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 112, 112, 64)      1664
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 56, 56, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 28, 28, 128)       204928
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 14, 128)       409728
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 7, 7, 256)         819456
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 3, 3, 256)         0
_________________________________________________________________
flatten (Flatten)            (None, 2304)              0
_________________________________________________________________
dense (Dense)                (None, 64)                147520
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 32)                2080
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 66
=================================================================
Total params: 1,585,442
Trainable params: 1,585,442
Non-trainable params: 0
_________________________________________________________________
```

Figure 8.6 CNN module

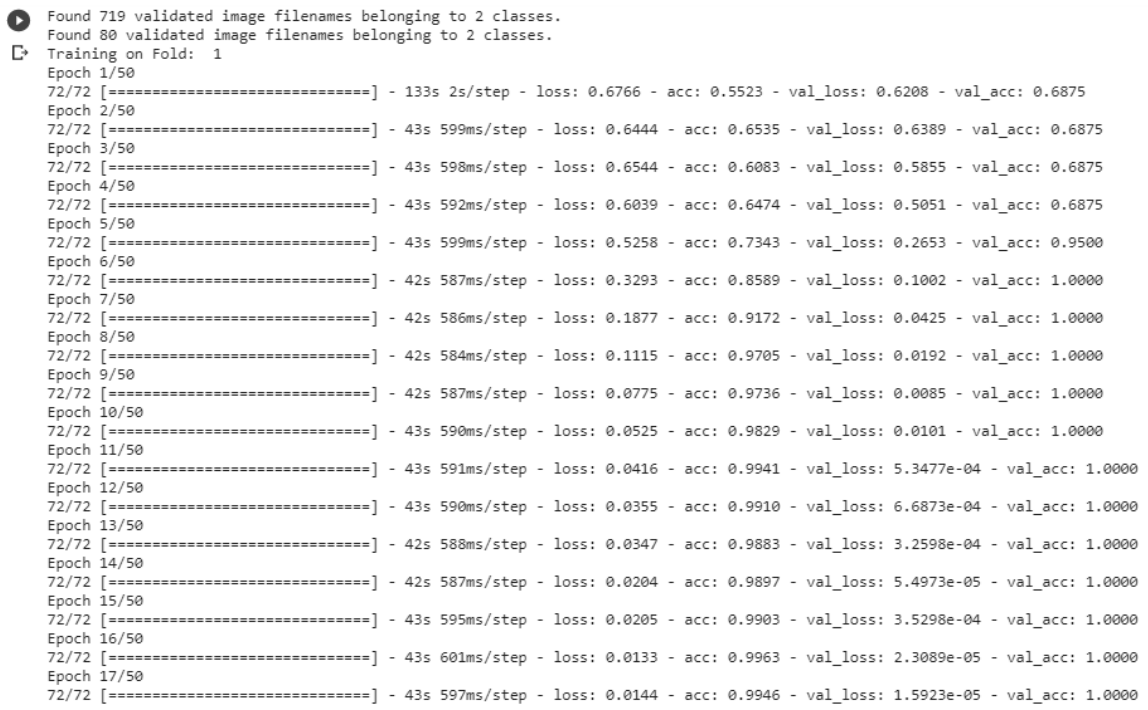
Fig 8.6- represents the output screenshot of creating a CNN module

Step 8:

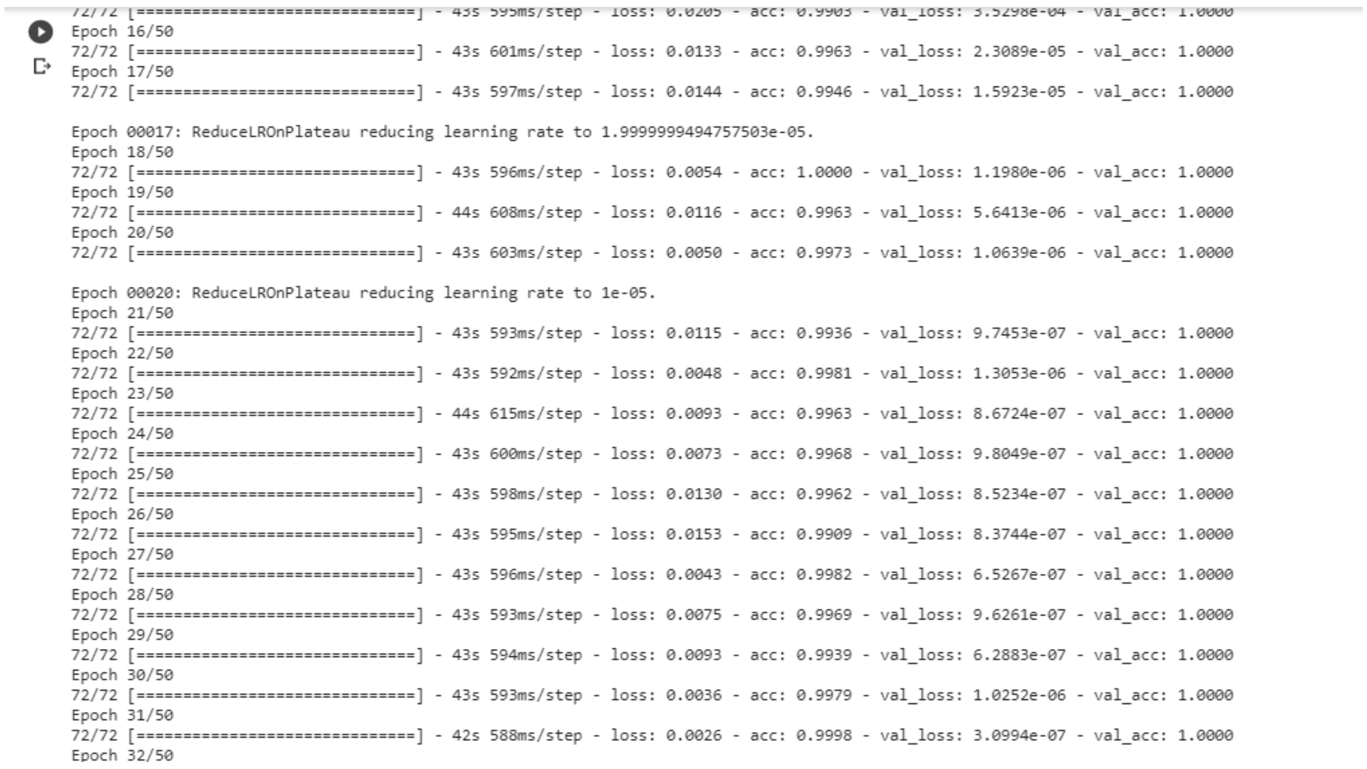
Now with the given input images are passed to extract features and to generate the weight's form the images.This data is now used to train the model.



```
Epoch 11/50
72/72 [==============================] - 42s 587ms/step - loss: 0.0168 - acc: 0.9938 - val_loss: 3.6379e-05 - val_acc: 1.0000
Epoch 12/50
72/72 [==============================] - 42s 585ms/step - loss: 0.0123 - acc: 0.9966 - val_loss: 4.1989e-04 - val_acc: 1.0000
Epoch 13/50
72/72 [==============================] - 42s 585ms/step - loss: 0.0215 - acc: 0.9846 - val_loss: 1.3634e-05 - val_acc: 1.0000

Epoch 00013: ReduceLROnPlateau reducing learning rate to 1.9999999494757503e-05.
Epoch 14/50
72/72 [==============================] - 42s 585ms/step - loss: 0.0137 - acc: 0.9951 - val_loss: 7.3696e-06 - val_acc: 1.0000
Epoch 15/50
72/72 [==============================] - 42s 583ms/step - loss: 0.0034 - acc: 0.9987 - val_loss: 2.3484e-06 - val_acc: 1.0000
Epoch 16/50
72/72 [==============================] - 43s 590ms/step - loss: 0.0059 - acc: 0.9967 - val_loss: 2.4512e-06 - val_acc: 1.0000
Epoch 17/50
72/72 [==============================] - 43s 592ms/step - loss: 0.0039 - acc: 0.9980 - val_loss: 3.2469e-06 - val_acc: 1.0000
Restoring model weights from the end of the best epoch.

Epoch 00017: ReduceLROnPlateau reducing learning rate to 1e-05.
Epoch 00017: early stopping
```

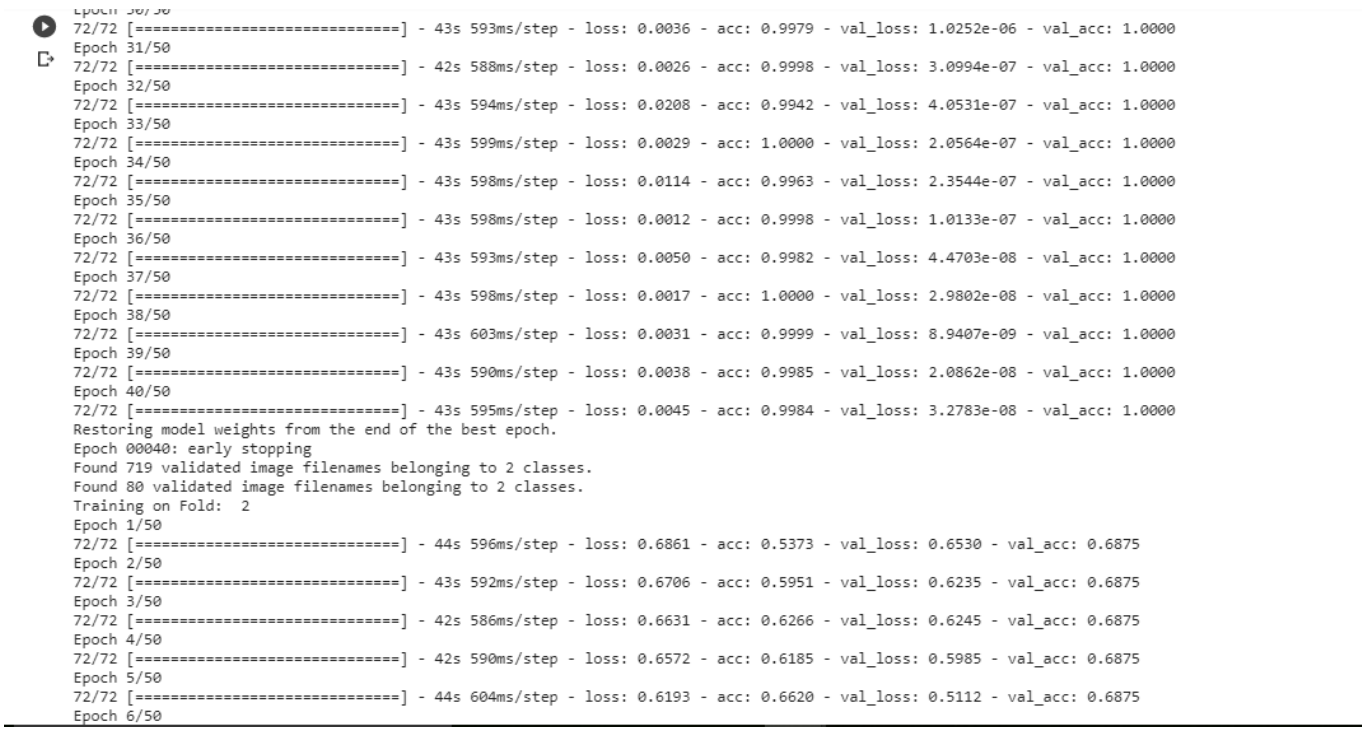


```
Found 719 validated image filenames belonging to 2 classes.
Found 80 validated image filenames belonging to 2 classes.
Training on Fold:  1
Epoch 1/50
72/72 [==============================] - 133s 2s/step - loss: 0.6766 - acc: 0.5523 - val_loss: 0.6208 - val_acc: 0.6875
Epoch 2/50
72/72 [==============================] - 43s 599ms/step - loss: 0.6444 - acc: 0.6535 - val_loss: 0.6389 - val_acc: 0.6875
Epoch 3/50
72/72 [==============================] - 43s 598ms/step - loss: 0.6544 - acc: 0.6083 - val_loss: 0.5855 - val_acc: 0.6875
Epoch 4/50
72/72 [==============================] - 43s 592ms/step - loss: 0.6039 - acc: 0.6474 - val_loss: 0.5051 - val_acc: 0.6875
Epoch 5/50
72/72 [==============================] - 43s 599ms/step - loss: 0.5258 - acc: 0.7343 - val_loss: 0.2653 - val_acc: 0.9500
Epoch 6/50
72/72 [==============================] - 42s 587ms/step - loss: 0.3293 - acc: 0.8589 - val_loss: 0.1002 - val_acc: 1.0000
Epoch 7/50
72/72 [==============================] - 42s 586ms/step - loss: 0.1877 - acc: 0.9172 - val_loss: 0.0425 - val_acc: 1.0000
Epoch 8/50
72/72 [==============================] - 42s 584ms/step - loss: 0.1115 - acc: 0.9705 - val_loss: 0.0192 - val_acc: 1.0000
Epoch 9/50
72/72 [==============================] - 42s 587ms/step - loss: 0.0775 - acc: 0.9736 - val_loss: 0.0085 - val_acc: 1.0000
Epoch 10/50
72/72 [==============================] - 43s 590ms/step - loss: 0.0525 - acc: 0.9829 - val_loss: 0.0101 - val_acc: 1.0000
Epoch 11/50
72/72 [==============================] - 43s 591ms/step - loss: 0.0416 - acc: 0.9941 - val_loss: 5.3477e-04 - val_acc: 1.0000
Epoch 12/50
72/72 [==============================] - 43s 590ms/step - loss: 0.0355 - acc: 0.9910 - val_loss: 6.6873e-04 - val_acc: 1.0000
Epoch 13/50
72/72 [==============================] - 42s 588ms/step - loss: 0.0347 - acc: 0.9883 - val_loss: 3.2598e-04 - val_acc: 1.0000
Epoch 14/50
72/72 [==============================] - 42s 587ms/step - loss: 0.0204 - acc: 0.9897 - val_loss: 5.4973e-05 - val_acc: 1.0000
Epoch 15/50
72/72 [==============================] - 43s 595ms/step - loss: 0.0205 - acc: 0.9903 - val_loss: 3.5298e-04 - val_acc: 1.0000
Epoch 16/50
72/72 [==============================] - 43s 601ms/step - loss: 0.0133 - acc: 0.9963 - val_loss: 2.3089e-05 - val_acc: 1.0000
Epoch 17/50
72/72 [==============================] - 43s 597ms/step - loss: 0.0144 - acc: 0.9946 - val_loss: 1.5923e-05 - val_acc: 1.0000
```

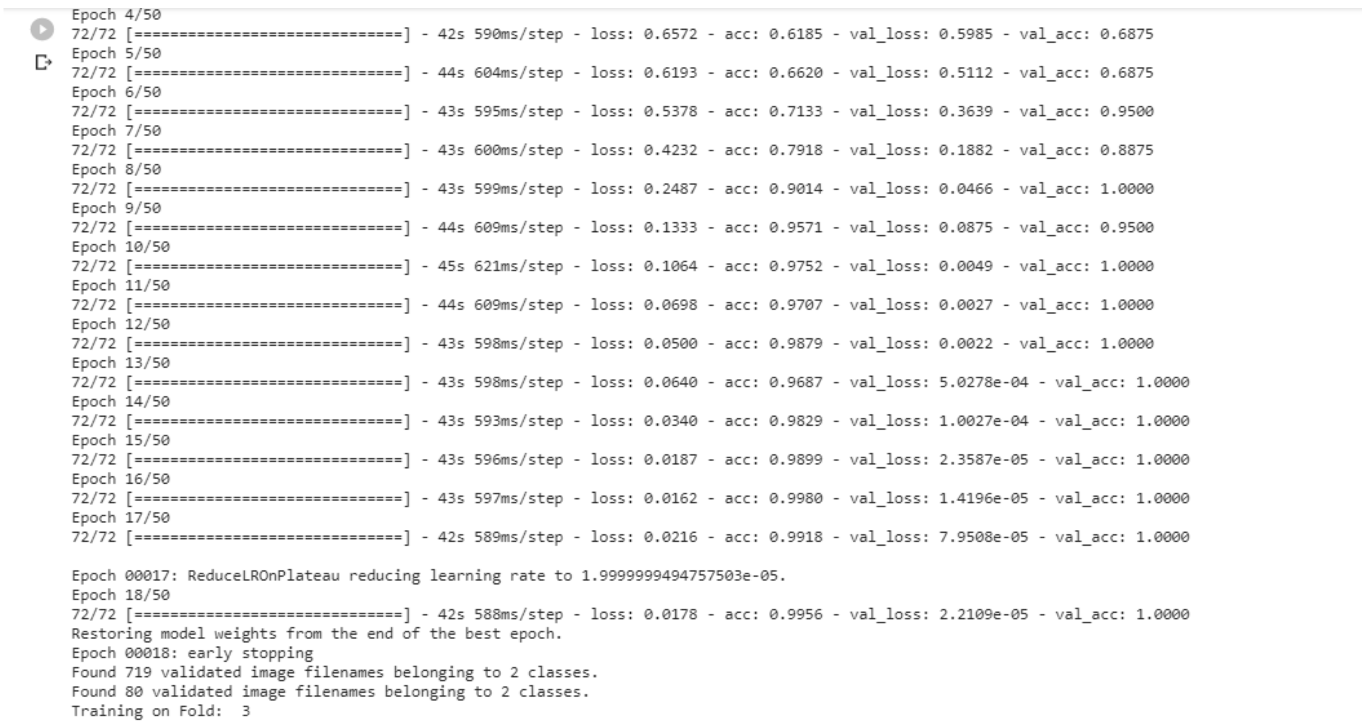


```
72/72 [==============================] - 43s 595ms/step - loss: 0.0205 - acc: 0.9903 - val_loss: 3.5298e-04 - val_acc: 1.0000
Epoch 16/50
72/72 [==============================] - 43s 601ms/step - loss: 0.0133 - acc: 0.9963 - val_loss: 2.3089e-05 - val_acc: 1.0000
Epoch 17/50
72/72 [==============================] - 43s 597ms/step - loss: 0.0144 - acc: 0.9946 - val_loss: 1.5923e-05 - val_acc: 1.0000
```

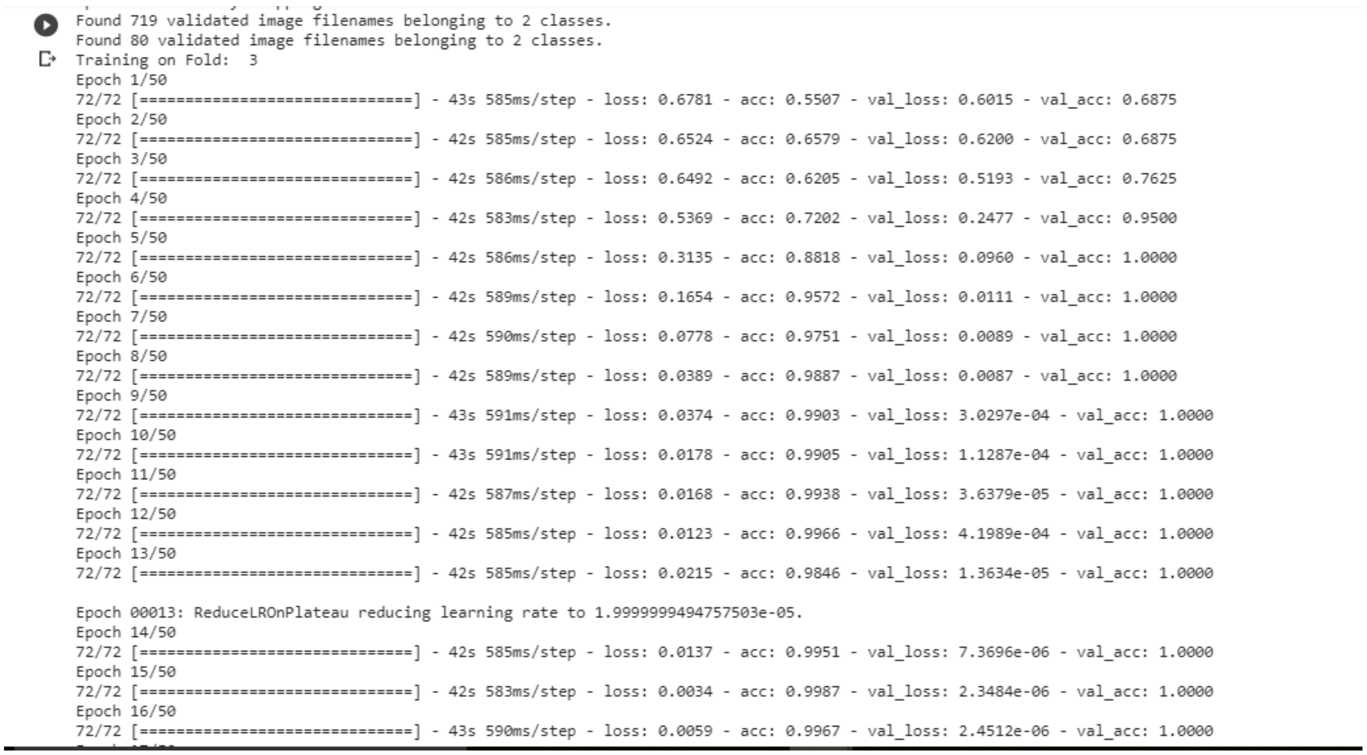
```

Epoch 00017: ReduceLROnPlateau reducing learning rate to 1.9999999494757503e-05.
Epoch 18/50
72/72 [==============================] - 43s 596ms/step - loss: 0.0054 - acc: 1.0000 - val_loss: 1.1980e-06 - val_acc: 1.0000
Epoch 19/50
72/72 [==============================] - 44s 608ms/step - loss: 0.0116 - acc: 0.9963 - val_loss: 5.6413e-06 - val_acc: 1.0000
Epoch 20/50
72/72 [==============================] - 43s 603ms/step - loss: 0.0050 - acc: 0.9973 - val_loss: 1.0639e-06 - val_acc: 1.0000

Epoch 00020: ReduceLROnPlateau reducing learning rate to 1e-05.
Epoch 21/50
72/72 [==============================] - 43s 593ms/step - loss: 0.0115 - acc: 0.9936 - val_loss: 9.7453e-07 - val_acc: 1.0000
Epoch 22/50
72/72 [==============================] - 43s 592ms/step - loss: 0.0048 - acc: 0.9981 - val_loss: 1.3053e-06 - val_acc: 1.0000
Epoch 23/50
72/72 [==============================] - 44s 615ms/step - loss: 0.0093 - acc: 0.9963 - val_loss: 8.6724e-07 - val_acc: 1.0000
Epoch 24/50
72/72 [==============================] - 43s 600ms/step - loss: 0.0073 - acc: 0.9968 - val_loss: 9.8049e-07 - val_acc: 1.0000
Epoch 25/50
72/72 [==============================] - 43s 598ms/step - loss: 0.0130 - acc: 0.9962 - val_loss: 8.5234e-07 - val_acc: 1.0000
Epoch 26/50
72/72 [==============================] - 43s 595ms/step - loss: 0.0153 - acc: 0.9909 - val_loss: 8.3744e-07 - val_acc: 1.0000
Epoch 27/50
72/72 [==============================] - 43s 596ms/step - loss: 0.0043 - acc: 0.9982 - val_loss: 6.5267e-07 - val_acc: 1.0000
Epoch 28/50
72/72 [==============================] - 43s 593ms/step - loss: 0.0075 - acc: 0.9969 - val_loss: 9.6261e-07 - val_acc: 1.0000
Epoch 29/50
72/72 [==============================] - 43s 594ms/step - loss: 0.0093 - acc: 0.9939 - val_loss: 6.2883e-07 - val_acc: 1.0000
Epoch 30/50
72/72 [==============================] - 43s 593ms/step - loss: 0.0036 - acc: 0.9979 - val_loss: 1.0252e-06 - val_acc: 1.0000
Epoch 31/50
72/72 [==============================] - 42s 588ms/step - loss: 0.0026 - acc: 0.9998 - val_loss: 3.0994e-07 - val_acc: 1.0000
Epoch 32/50
```



```
Epoch 30/50
72/72 [==============================] - 43s 593ms/step - loss: 0.0036 - acc: 0.9979 - val_loss: 1.0252e-06 - val_acc: 1.0000
Epoch 31/50
72/72 [==============================] - 42s 588ms/step - loss: 0.0026 - acc: 0.9998 - val_loss: 3.0994e-07 - val_acc: 1.0000
Epoch 32/50
72/72 [==============================] - 43s 594ms/step - loss: 0.0208 - acc: 0.9942 - val_loss: 4.0531e-07 - val_acc: 1.0000
Epoch 33/50
72/72 [==============================] - 43s 599ms/step - loss: 0.0029 - acc: 1.0000 - val_loss: 2.0564e-07 - val_acc: 1.0000
Epoch 34/50
72/72 [==============================] - 43s 598ms/step - loss: 0.0114 - acc: 0.9963 - val_loss: 2.3544e-07 - val_acc: 1.0000
Epoch 35/50
72/72 [==============================] - 43s 598ms/step - loss: 0.0012 - acc: 0.9998 - val_loss: 1.0133e-07 - val_acc: 1.0000
Epoch 36/50
72/72 [==============================] - 43s 593ms/step - loss: 0.0050 - acc: 0.9982 - val_loss: 4.4703e-08 - val_acc: 1.0000
Epoch 37/50
72/72 [==============================] - 43s 598ms/step - loss: 0.0017 - acc: 1.0000 - val_loss: 2.9802e-08 - val_acc: 1.0000
Epoch 38/50
72/72 [==============================] - 43s 603ms/step - loss: 0.0031 - acc: 0.9999 - val_loss: 8.9407e-09 - val_acc: 1.0000
Epoch 39/50
72/72 [==============================] - 43s 590ms/step - loss: 0.0038 - acc: 0.9985 - val_loss: 2.0862e-08 - val_acc: 1.0000
Epoch 40/50
72/72 [==============================] - 43s 595ms/step - loss: 0.0045 - acc: 0.9984 - val_loss: 3.2783e-08 - val_acc: 1.0000
Restoring model weights from the end of the best epoch.
Epoch 00040: early stopping
Found 719 validated image filenames belonging to 2 classes.
Found 80 validated image filenames belonging to 2 classes.
Training on Fold:  2
Epoch 1/50
72/72 [==============================] - 44s 596ms/step - loss: 0.6861 - acc: 0.5373 - val_loss: 0.6530 - val_acc: 0.6875
Epoch 2/50
72/72 [==============================] - 43s 592ms/step - loss: 0.6706 - acc: 0.5951 - val_loss: 0.6235 - val_acc: 0.6875
Epoch 3/50
72/72 [==============================] - 42s 586ms/step - loss: 0.6631 - acc: 0.6266 - val_loss: 0.6245 - val_acc: 0.6875
Epoch 4/50
72/72 [==============================] - 42s 590ms/step - loss: 0.6572 - acc: 0.6185 - val_loss: 0.5985 - val_acc: 0.6875
Epoch 5/50
72/72 [==============================] - 44s 604ms/step - loss: 0.6193 - acc: 0.6620 - val_loss: 0.5112 - val_acc: 0.6875
Epoch 6/50
```



```
Epoch 4/50
72/72 [==============================] - 42s 590ms/step - loss: 0.6572 - acc: 0.6185 - val_loss: 0.5985 - val_acc: 0.6875
Epoch 5/50
72/72 [==============================] - 44s 604ms/step - loss: 0.6193 - acc: 0.6620 - val_loss: 0.5112 - val_acc: 0.6875
Epoch 6/50
72/72 [==============================] - 43s 595ms/step - loss: 0.5378 - acc: 0.7133 - val_loss: 0.3639 - val_acc: 0.9500
Epoch 7/50
72/72 [==============================] - 43s 600ms/step - loss: 0.4232 - acc: 0.7918 - val_loss: 0.1882 - val_acc: 0.8875
Epoch 8/50
72/72 [==============================] - 43s 599ms/step - loss: 0.2487 - acc: 0.9014 - val_loss: 0.0466 - val_acc: 1.0000
Epoch 9/50
72/72 [==============================] - 44s 609ms/step - loss: 0.1333 - acc: 0.9571 - val_loss: 0.0875 - val_acc: 0.9500
Epoch 10/50
72/72 [==============================] - 45s 621ms/step - loss: 0.1064 - acc: 0.9752 - val_loss: 0.0049 - val_acc: 1.0000
Epoch 11/50
72/72 [==============================] - 44s 609ms/step - loss: 0.0698 - acc: 0.9707 - val_loss: 0.0027 - val_acc: 1.0000
Epoch 12/50
72/72 [==============================] - 43s 598ms/step - loss: 0.0500 - acc: 0.9879 - val_loss: 0.0022 - val_acc: 1.0000
Epoch 13/50
72/72 [==============================] - 43s 598ms/step - loss: 0.0640 - acc: 0.9687 - val_loss: 5.0278e-04 - val_acc: 1.0000
Epoch 14/50
72/72 [==============================] - 43s 593ms/step - loss: 0.0340 - acc: 0.9829 - val_loss: 1.0027e-04 - val_acc: 1.0000
Epoch 15/50
72/72 [==============================] - 43s 596ms/step - loss: 0.0187 - acc: 0.9899 - val_loss: 2.3587e-05 - val_acc: 1.0000
Epoch 16/50
72/72 [==============================] - 43s 597ms/step - loss: 0.0162 - acc: 0.9980 - val_loss: 1.4196e-05 - val_acc: 1.0000
Epoch 17/50
72/72 [==============================] - 42s 589ms/step - loss: 0.0216 - acc: 0.9918 - val_loss: 7.9508e-05 - val_acc: 1.0000

Epoch 00017: ReduceLROnPlateau reducing learning rate to 1.9999999494757503e-05.
Epoch 18/50
72/72 [==============================] - 42s 588ms/step - loss: 0.0178 - acc: 0.9956 - val_loss: 2.2109e-05 - val_acc: 1.0000
Restoring model weights from the end of the best epoch.
Epoch 00018: early stopping
Found 719 validated image filenames belonging to 2 classes.
Found 80 validated image filenames belonging to 2 classes.
Training on Fold:  3
```



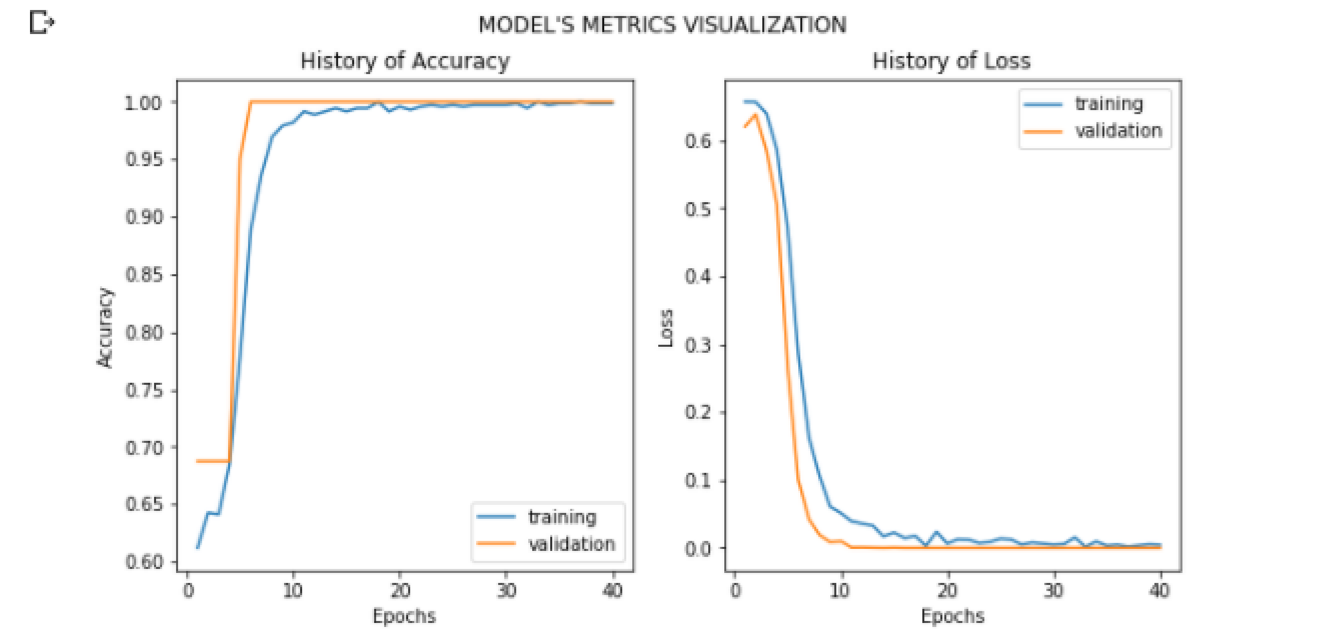
```
Found 719 validated image filenames belonging to 2 classes.
Found 80 validated image filenames belonging to 2 classes.
Training on Fold:  3
Epoch 1/50
72/72 [==============================] - 43s 585ms/step - loss: 0.6781 - acc: 0.5507 - val_loss: 0.6015 - val_acc: 0.6875
Epoch 2/50
72/72 [==============================] - 42s 585ms/step - loss: 0.6524 - acc: 0.6579 - val_loss: 0.6200 - val_acc: 0.6875
Epoch 3/50
72/72 [==============================] - 42s 586ms/step - loss: 0.6492 - acc: 0.6205 - val_loss: 0.5193 - val_acc: 0.7625
Epoch 4/50
72/72 [==============================] - 42s 583ms/step - loss: 0.5369 - acc: 0.7202 - val_loss: 0.2477 - val_acc: 0.9500
Epoch 5/50
72/72 [==============================] - 42s 586ms/step - loss: 0.3135 - acc: 0.8818 - val_loss: 0.0960 - val_acc: 1.0000
Epoch 6/50
72/72 [==============================] - 42s 589ms/step - loss: 0.1654 - acc: 0.9572 - val_loss: 0.0111 - val_acc: 1.0000
Epoch 7/50
72/72 [==============================] - 42s 590ms/step - loss: 0.0778 - acc: 0.9751 - val_loss: 0.0089 - val_acc: 1.0000
Epoch 8/50
72/72 [==============================] - 42s 589ms/step - loss: 0.0389 - acc: 0.9887 - val_loss: 0.0087 - val_acc: 1.0000
Epoch 9/50
72/72 [==============================] - 43s 591ms/step - loss: 0.0374 - acc: 0.9903 - val_loss: 3.0297e-04 - val_acc: 1.0000
Epoch 10/50
72/72 [==============================] - 43s 591ms/step - loss: 0.0178 - acc: 0.9905 - val_loss: 1.1287e-04 - val_acc: 1.0000
Epoch 11/50
72/72 [==============================] - 42s 587ms/step - loss: 0.0168 - acc: 0.9938 - val_loss: 3.6379e-05 - val_acc: 1.0000
Epoch 12/50
72/72 [==============================] - 42s 585ms/step - loss: 0.0123 - acc: 0.9966 - val_loss: 4.1989e-04 - val_acc: 1.0000
Epoch 13/50
72/72 [==============================] - 42s 585ms/step - loss: 0.0215 - acc: 0.9846 - val_loss: 1.3634e-05 - val_acc: 1.0000

Epoch 00013: ReduceLROnPlateau reducing learning rate to 1.9999999494757503e-05.
Epoch 14/50
72/72 [==============================] - 42s 585ms/step - loss: 0.0137 - acc: 0.9951 - val_loss: 7.3696e-06 - val_acc: 1.0000
Epoch 15/50
72/72 [==============================] - 42s 583ms/step - loss: 0.0034 - acc: 0.9987 - val_loss: 2.3484e-06 - val_acc: 1.0000
Epoch 16/50
72/72 [==============================] - 43s 590ms/step - loss: 0.0059 - acc: 0.9967 - val_loss: 2.4512e-06 - val_acc: 1.0000
```

Figure 8.7 -Model training

Fig -8.7 - represent the output screenshot of model training

Step 9:

Finding the accuracy of trained models.A good accuracy rate results in very good results.



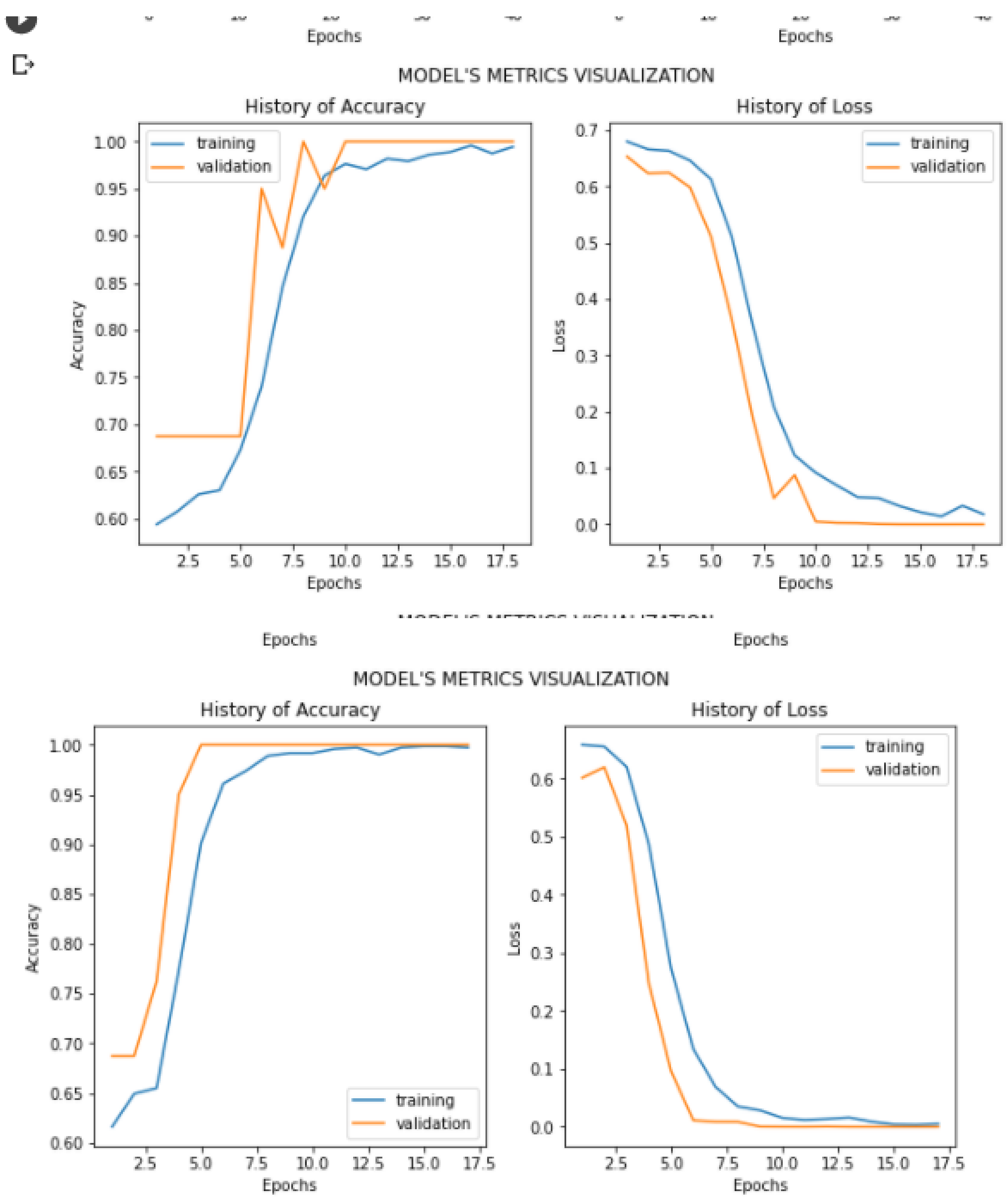


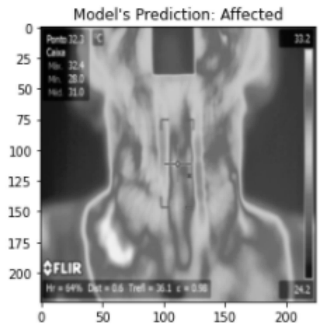
Figure 8.8 - Accuracy and loss graph

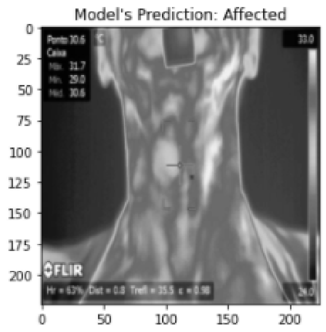
Fig 8.8 - represents the accuracy and loss graph of given input images.

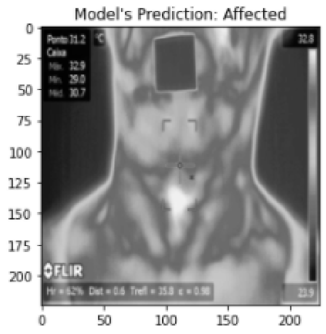
Step 10:

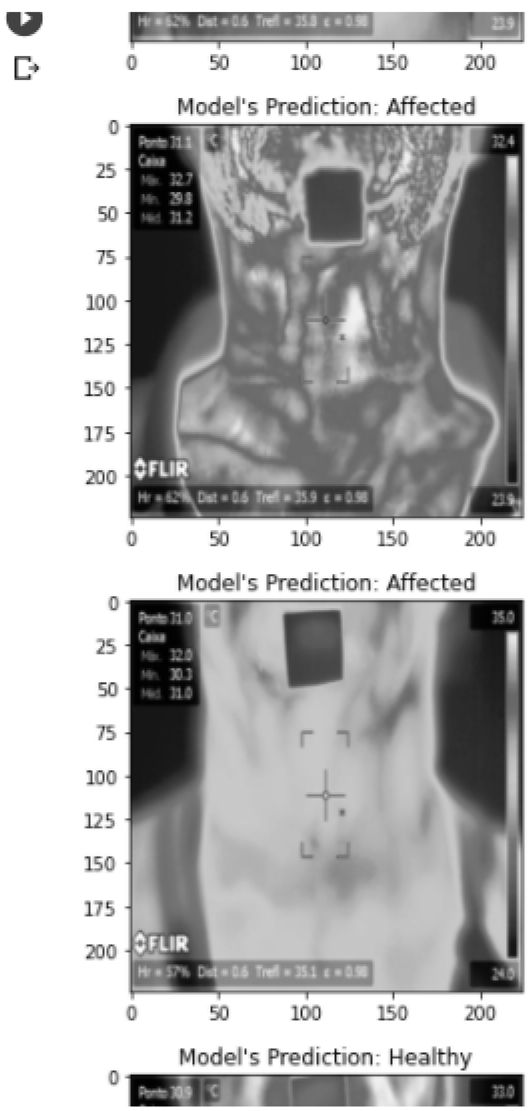
Predict the output of given input images.

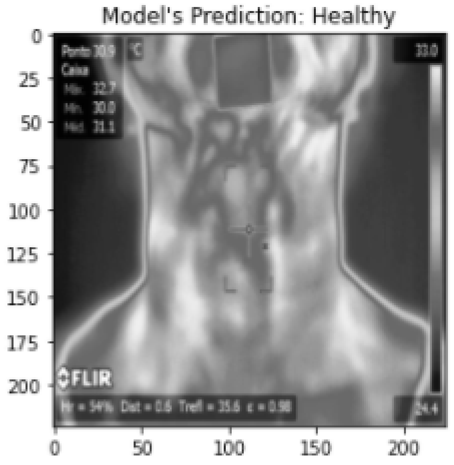
```
def pre_visualization(data, predictions):

    for image,pred in final_pred:
        plt.imshow(image.reshape(224,224), cmap = 'gray')
        plt.title("Model's Prediction: " + str(pred))
        plt.show()

pre_visualization(images10,prediction10)
```

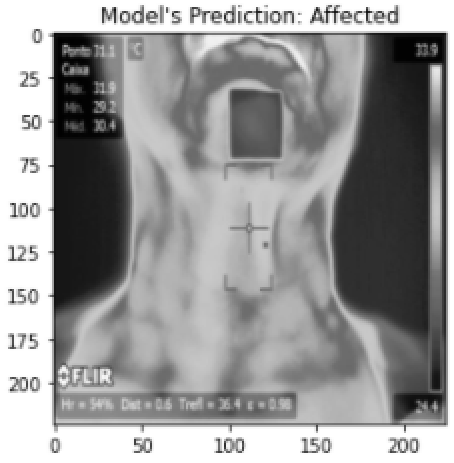







Model's Prediction: Affected
Model's Prediction: Affected
Model's Prediction: Healthy
FLIR


Model's Prediction: Healthy
FLIR


Model's Prediction: Affected
FLIR

Model's Prediction: Affected

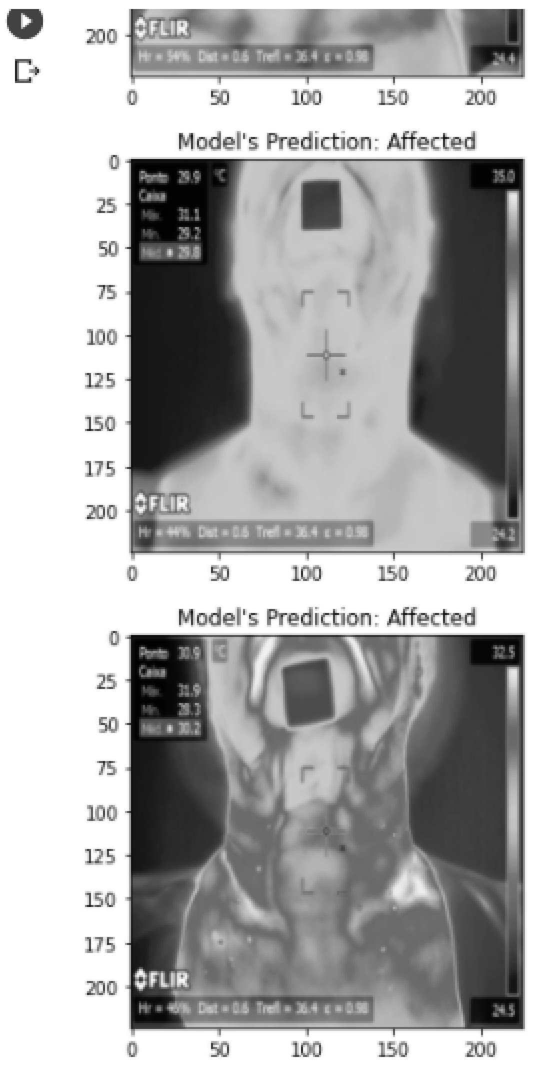


Figure 8.9 - Output images

Fig 8.9- represent the output predicted of given input images.

CONCLUSION AND FUTURE ENHANCEMENTS

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENTS

We proposed a thermography based approach for thyroid cancer detection with several steps including image pre-processing, noise reduction, region of interest, feature extraction, cancer detection, etc. The strength of this approach includes its simplicity, time management, accuracy, ease of implementation, and it does not require any significant amount of training or post processing, it provides us with a higher recognition rate with minimum computation time. The weakness of this method is that we define certain parameters and threshold values experimentally since it does not follow any systematic approach for cancer detection, and maximum parameters taken in this approach are based on assumptions made after testing the number of images. The future work includes identifying the size and location of tumors. And also 3D model construction.

REFERENCES

REFERENCES

[1] Farshad Bahramian,Afsaneh Mojra “Thyroid cancer estimation using infrared thermography data”,2019,ELSEVIER Infrared Physics & Technology Volume 104
[2] Kamal Jnawali, Bhargava Chinni, Vikram Dogra & Navalgund Rao “Automatic cancer tissue detection using multispectral photoacoustic imaging” 2019,Springer
[3]Prof Xiang chun Li, PhD ,Prof Sheng Zhang, MD , Qiang Zhang, MD Xi Wei, MD “Diagnosis of thyroid cancer using deep convolutional neural network models applied to sonographic images”,2019,ELSEVIER
[4] Hailiang Li, Jian Weng, Yujian Shi,Wanrong Gu “An improved deep learning approach for detection of thyroid papillary cancer in ultrasound images”,2018
[5] Roslidar Roslidar ,Aulia Rahman ,Rusdha Muharar ,Muhammad Rizky Syahputra “Thermal Imaging and Deep Learning Approaches for Breast Cancer Detection” ,2020,IEEE
[6]Olivier Janssens ,Rik Van de Walle , Mia Loccufier ,Sofie Van Hoecke “Deep Learning for Infrared Thermal Image Based Machine Health Monitoring”,2019,IEEE
[7] Jose-Luis,Gonzalez-Hernandez,Alyssa,N.Recinella,Satish G.Kandlikar, ,Donnette Dabydeen “Technology, application and potential of dynamic breast thermography for the detection of breast cancer”,2019,ELSEVIER
[8]M. Soltani, Reza Rahpeima & Farshad Moradi Kashkooli “ Breast cancer diagnosis with a microwave thermoacoustic imaging technique”,2019,Springer
[9]Deepika Singh, Ashutosh Kumar Singh, “Role of image thermography in early breast cancer detection”,2020,ELSEVIER
[10] Sami Ekici, ,HushangJawzal “Breast cancer diagnosis using thermography and convolutional neural networks”,2020,ELSEVIER
[11] Jun Yang ,Wei Wang ,Guang Lin,Qing Li ,Yeqing Sun Yixuan Sun ,”Infrared Thermal Imaging-Based Crack Detection Using Deep learning “,2019,IEEE
[12] Paramasivam Alagumariappan,Mohamed Shuaib Y,Sonya A,”Identification of Electrical Faults in Underground Cables Using Machine Learning Algorithm”2019,The 6th International Electronic Conference on Sensors and Applications
[13] S. Lagüela, S. Sfarra, F. J. Madruga &P. Arias,”Automatic detection of moistures in different construction materials from thermographic images”,2019,Springer
[14]P. Mohamed Shakeel Tarek E. El. Tobely ,Haytham Al-Feel ,Gunasekaran Manogaran,” Neural Network Based Brain Tumor Detection Using Wireless Infrared Imaging Sensor”,2019,IEEE

[15] Ahmet,Haydar Örnek ,·Murat Ceylan·SaimErvural,"Health status detection of neonates using infrared thermography and deep convolutional neural networks",2019,ELSEVIER

APPENDIX

APPENDIX

SOURCE CODE:

```
#importing libraries
import cv2
import os
import glob
import shutil
import random
import pandas as pd
from PIL import Image
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

#getting data from dataset

path1 = '/content/drive/MyDrive/Dataset/thyroid_cancer/no/*.jpg'
cnt=0
for filename in glob.glob(path1):
        #reading the thermal images
   image = cv2.imread(filename)
   image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
   image = ~image    # inverting image

   #converting image to grayscale
   heatmap_gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
   heatmap = cv2.applyColorMap(heatmap_gray, cv2.COLORMAP_HOT)
   heatmap = cv2.cvtColor(heatmap, cv2.COLOR_BGR2RGB)
   heatmap_gray = cv2.cvtColor(heatmap, cv2.COLOR_RGB2GRAY)

        #binary thresholding
        ret, binary_thresh = cv2.threshold(heatmap_gray, 200, 255, cv2.THRESH_BINARY)
   kernel = np.ones((5, 5), np.uint8)


        #removing noise in image
   image_erosion = cv2.erode(binary_thresh, kernel, iterations=1)
   image_opening = cv2.dilate(image_erosion, kernel, iterations=1)
```

```
       #generating counters
   contours, _ = cv2.findContours(image_opening, 1, 2)

       #finding the region of interest
   image_with_rectangles = np.copy(heatmap)
   ROI_number = 0
   for contour in contours:
      # rectangle over each contour
      x=175
      y=95
      h=330
      w=305
    image_with_rectangles = cv2.rectangle(image_with_rectangles, (x, y), (x+w, y+h), (36, 255, 12), 2)
      ROI = image[y:y+h, x:x+w]


        #savingROI          cv2.imwrite('/content/drive/MyDrive/Dataset/result/no/no_{}.jpg'.format(cnt),
        ROI)
      cnt += 1



import os
import glob
import shutil
import random
import pandas as pd
from PIL import Image
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


def importing_data(path):
   sample = []
   for filename in glob.glob(path):
      #img = Image.open(filename,'r')
      #IMG = np.array(img)
      sample.append(filename)
   return sample
```

```
path1 = '/content/drive/MyDrive/Dataset/result/no/*.jpg'
path2 = '/content/drive/MyDrive/Dataset/result/yes/*.jpg'
path3 = '/content/drive/MyDrive/Dataset/thyroid_cancer/pred/*.jpg'

train_n = importing_data(path1)
train_y = importing_data(path2)
test = importing_data(path3)

#%% CREATION OF DATASETS

df_train_n = pd.DataFrame({'image':train_n, 'label': 'Healthy'})
df_train_y = pd.DataFrame({'image':train_y, 'label': 'Affected'})
df_test = pd.DataFrame({'image':test})
train_data = pd.concat([df_train_n, df_train_y])
train_data.head()




from sklearn.model_selection import train_test_split

X_train, X_val = train_test_split(train_data,
                                  test_size = 0.1,
                                  shuffle = True,
                                  random_state = 42)

import keras
from keras.metrics import AUC, Recall, Precision
from keras.models import Sequential
from keras.layers import Dense, GlobalAveragePooling2D, Dropout, Conv2D , MaxPooling2D, Flatten
from keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
from keras.optimizers import RMSprop

def build_model():

    '''Sequential Model creation'''
    Cnn = Sequential()

    Cnn.add(Conv2D(64,(5,5), activation = 'relu', padding = 'same',
                   strides=(2,2), input_shape = [224,224,1]))
    Cnn.add(MaxPooling2D(2))
    Cnn.add(Conv2D(128,(5,5), activation = 'relu', padding = 'same', strides=(2,2)))
    Cnn.add(Conv2D(128,(5,5), activation = 'relu', padding = 'same', strides=(2,2)))
    Cnn.add(Conv2D(256,(5,5), activation = 'relu', padding = 'same', strides=(2,2)))
    Cnn.add(MaxPooling2D(2))
    #Cnn.add(GlobalAveragePooling2D())
    Cnn.add(Flatten())
    Cnn.add(Dense(64, activation = 'relu'))
    Cnn.add(Dropout(0.4))
    Cnn.add(Dense(32, activation = 'relu'))
    Cnn.add(Dropout(0.4))
    Cnn.add(Dense(2, activation = 'softmax'))

    return Cnn

keras_model = build_model()
keras_model.summary()


def Model_fit(train_data, val_data):

    keras_model = None

    keras_model = build_model()

    '''Compiling the model'''

    keras_model.compile(optimizer = RMSprop(learning_rate = 1e-4),
                  loss='sparse_categorical_crossentropy',
                  metrics =['acc'])

    es = EarlyStopping(monitor='val_loss', mode='min',
                 patience=2,
                 restore_best_weights=True,
                 verbose=1)


    checkpoint_cb = ModelCheckpoint("thyroid_model_best.h5",
                           save_best_only=True)

    reduce_lr = ReduceLROnPlateau(monitor = 'val_loss',
                              factor = 0.2,
                              patience = 3,
                              min_lr = 1e-5,
                              mode = 'min',
                              verbose=1)


    history = keras_model.fit(train_data,
                              validation_data = val_data,
                              epochs= 50,
                              batch_size = 10,
                              callbacks=[es, checkpoint_cb, reduce_lr])



    return history


from keras.preprocessing.image import ImageDataGenerator

k_fold = 3
IMG_SIZE = 224
size = (IMG_SIZE,IMG_SIZE)
n_CLASS = 2

def CV_training(train_data, val_data):

    cv_histories = []

    for i in range(0,k_fold):

        datagen = ImageDataGenerator(rescale = 1./255)

        train_set = datagen.flow_from_dataframe(train_data,
                                                directory = '/content/drive/MyDrive/Dataset/result/*.jpg',
                                                x_col = 'image',
                                                y_col = 'label',
                                                target_size = size,
                                                color_mode = 'grayscale',
                                                class_mode = 'sparse',
                                                batch_size = 10,
                                        shuffle = True,
                                        interpolation = 'bilinear')

        val_set = datagen.flow_from_dataframe(val_data,
                                       directory = '/content/drive/MyDrive/Dataset/result/*.jpg',
                                       x_col = 'image',
                                       y_col = 'label',
                                       target_size = size,
                                       color_mode = 'grayscale',
                                       class_mode = 'sparse',
                                       batch_size = 10,
                                       shuffle = True,
                                       interpolation = 'bilinear')
        print("Training on Fold: ",i+1)

        cv_histories.append(Model_fit(train_set, val_set))

    return cv_histories

cv_results = CV_training(X_train,X_val)


def acc_results(results):
    i = 0
    for fold in cv_results:
        print('Val_Acc Folder '+ str(i) + ' =', max(fold.history['val_acc']))
        i += 1

acc_results(cv_results)

def Acc_Loss_Plot(results):

    for fold in results:

        acc = fold.history['acc']
        val_acc = fold.history['val_acc']
        loss = fold.history['loss']
        val_loss = fold.history['val_loss']

        fig, (ax1, ax2) = plt.subplots(1,2, figsize= (10,5))
        fig.suptitle(" MODEL'S METRICS VISUALIZATION ")
        ax1.plot(range(1, len(acc) + 1), acc)
        ax1.plot(range(1, len(val_acc) + 1), val_acc)
        ax1.set_title('History of Accuracy')
        ax1.set_xlabel('Epochs')
        ax1.set_ylabel('Accuracy')
        ax1.legend(['training', 'validation'])


        ax2.plot(range(1, len(loss) + 1), loss)
        ax2.plot(range(1, len(val_loss) + 1), val_loss)
        ax2.set_title('History of Loss')
        ax2.set_xlabel('Epochs')
        ax2.set_ylabel('Loss')
        ax2.legend(['training', 'validation'])
        plt.show()

Acc_Loss_Plot(cv_results)


import keras

keras_model = keras.models.load_model('thyroid_model_best.h5')
keras_model.compile(optimizer = RMSprop(learning_rate = 1e-4),
            loss='sparse_categorical_crossentropy', metrics =[ 'acc'])

# Predictions on the test set

datagen = ImageDataGenerator(rescale = 1./255)

test_set = datagen.flow_from_dataframe(df_test,
                    directory = '/content/drive/MyDrive/Dataset/thyroid_cancer/predr/*.jpg',
                    x_col = 'image',
                    y_col = None,
                    target_size = size,
                    color_mode = 'grayscale',
                    class_mode = None,
                    batch_size = 10,
                    shuffle = False,
                    interpolation = 'bilinear')
```

```
predictions = keras_model.predict(test_set)
predictions = predictions.argmax(axis=-1)
print("Where 0 = 'Affected'")
print("Where 1 = 'Healthy'")
print(predictions)

pred = []
[pred.append('Healthy') if i == 1 else pred.append('Affected') for i in predictions]
print(pred)


images10 = [test_set[0][0],test_set[0][1],test_set[0][2],test_set[0][3],test_set[0][4],
        test_set[0][5],test_set[0][6],test_set[0][7],test_set[0][8],test_set[0][9]]
prediction10 = pred[0:9]
final_pred = zip(images10,prediction10)

def pre_visualization(data, predictions):

    for image,pred in final_pred:
        plt.imshow(image.reshape(224,224), cmap = 'gray')
        plt.title("Model's Prediction: " + str(pred))
        plt.show()

pre_visualization(images10,prediction10)
```