A Project report

on

Facial Diagnosis Using Deep Transfer Learning

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

by

S.TASNEEM	**184G1A05A4**
K.UDAYASREE	**184G1A05A6**
S. SAIF SADIQ	**184G1A0582**
P.VASANTHALAKSHMI	**174G1A05A8**

Under the Guidance of

Mr. C. Sudheer Kumar, M. Tech.,(Ph.D)

Assistant Professor

Department of Computer Science & Engineering

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)

(Accredited by NAAC with 'A' Grade &Accredited by NBA(EEE, ECE &CSE))

Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.

2021-2022

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA ,Approved by AICTE, New Delhi, Accredited by NAAC with
'A' Grade& Accredited by NBA(EEE, ECE &CSE)
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu – 515701

Certificate

This is to certify that the project report entitled FACIAL DIAGNOSIS USING DEEP TRANSFER LEARNING is the bonafide work carried out by **S.TASNEEM** bearing Roll Number 184G1A05A4,**K. UDAYASREE** bearing Roll Number184G1A05A6,**S.SAIFSADIQ** bearing Roll Number 184G1A0582 and **P.VASANTHA LAKSHMI** bearing Roll Number 174G1A05A8in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2021-2022.

Signature of the Guide	**Head of the Department**
Mr. C. Sudheer Kumar,M. Tech.,(Ph.D)	Mr. P. Veera Prakash ,M. Tech.,(Ph.D).
Assistant Professor	Assistant Professor & HOD

Date:

Place: Rotarypuram

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that we would like to express my indebted gratitude to my Guide **Mr.C.Sudheer Kumar,M.Tech.,(Ph.D) Assistant Professor, Computer Science & Engineering**, who has guided me a lot and encouraged me in every step of the project work. We thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We are very much thankful to **Mr. P. Veera Prakash, M. Tech, (Ph. D)., Assistant Professor & Head of the Department, Computer Science & Engineering,** for his kind support and for providing necessary facilities to carry out the work.

We wish to convey my special thanks to **Dr. G. Bala Krishna, Ph.D, Principal** of **Srinivasa Ramanujan Institute of Technology** or giving the required information in doing our project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported us in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities**.**

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

Project Associates

DECLARATION

We, Ms. S.Tasneem bearing reg no : 184G1A05A4, Ms. K. UdayaSree bearing reg no : 184G1A05A6, Mr. S.Saif Sadiq bearing reg no : 184G1A0582, Ms.P. Vasantha Lakshmi bearing reg no : 174G1A05A8, students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram , hereby declare that the dissertation entitled "FACIAL DIAGNOSIS USING DEEP TRANSFER LEARNING" embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of Mr. C. Sudheer Kumar,M.Tech.,(Ph.D) Assistant Professor, Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities of Institute for the award of Degree.

S. TASNEEM	Reg no:184G1A05A4
K. UDAYASREE	Reg no:184G1A05A6
S.SAIF SADIQ	Reg no:184G1A0582
P.VASANTHA LAKSHMI	Reg no:174G1A05A8

CONTENTS

Page No

List of Figures

LIST OF ABBREVATIONS

FD	Facial Diagnosis
ICLSR	ImageNet Classification Large Scale Visual Recognition
DS	Down syndrome
DTL	Deep transfer learning
DCNN	Deep Convolutional Neural networks
FC	Fully Connected layer
ReLU	Rectified Linear Unit
UML	Unified Modelling Language
DFD	Data Flow Diagram
E-R	Entity Relationship Diagram
PiP	preferred installer program
SDLC	Software Development Life Cycle
DDS	Design Document Specification

ABSTRACT

The relationship between face and disease has been discussed for thousands years, which leads to the occurrence of facial diagnosis. The objective here is to explore the possibility of identifying diseases from uncontrolled 2D face images by deep learning techniques. We propose using deep transfer learning from face recognition to perform the computer-aided facial diagnosis on various diseases.

We are going to perform the computer-aided facial diagnosis on diseases (beta-thalassemia, hyperthyroidism, Down syndrome, and leprosy) with a dataset. The overall top-1 accuracy by deep transfer learning MobileNet Model which can reach over 90% which outperforms the performance of both traditional machine learning methods and clinicians in the experiments. In practice, collecting disease-specific face images is complex, expensive and time consuming, and imposes ethical limitations due to personal data treatment. Therefore, the datasets of facial diagnosis related research are private. The success of deep transfer learning applications in facial diagnosis with a dataset could provide a low-cost and non-invasive way for disease screening and detection.

Keywords*:* *Deep Transfer Learning,Face Recognition,,Non-Invasive,beta-thalassemia, hyperthyroidism, down syndrome, leprosy.*

CHAPTER-1

INTRODUCTION

1.1 Motivation:

In recent years, deep learning technology improves the state of the art in many areas for its good performances especially in computer vision. Deep learning inspired by the structure of human brains is to use a multiple-layer structure to perform nonlinear information processing and abstraction for feature learning.deep learning has become one of the newest trends in artificial intelligence research.Face recognition refers to the technology of verifying or identifying the identity of subjects from faces in images or videos. It is a hot topic in the field of computer vision. Face verification is the task of comparing a candidate face to another, and verifying whether it is a match or not. It is a oneto-one mapping. Face identification is the task of matching a given face image to one in a database of faces. These two can be implemented by separate algorithm frameworks, or they can be unified into one framework by metric learning. With the development of deep learning in recent years, traditional face recognition technology has gradually been replaced by deep learning methods.

1.2 Problem Definition:

Nowadays, it is still difficult for people to take a medical examination in many rural and underdeveloped areas because of the limited medical resources, which leads to delays in treatment in many cases. Even in metropolises, limitations including the high cost, long queuing time in hospital and the doctor-patient contradiction which leads to medical disputes still exist. Computer-aided facial diagnosis enables us to carry out non-invasive screening and detection of diseases quickly and easily. Therefore, if facial diagnosis can be proved effective with an acceptable error rate, it will be with great potential. With the help of artificial intelligence, we could explore the relationship between face and disease with a quantitative approach.(see Figure 1.1).

The CNN architectures such as MobileNet ,VGG-Face,Resnet get inspired from a ImageNet Classification Large Scale Visual Recognition(ICLSR).. If we train a deep neural network from scratch, it will inevitably lead to overfitting. Apparently face recognition and facial diagnosis are related. Since the labeled data in the area of face recognition is much more, transfer learning technology comes into our view. In traditional learning, we train separate isolated models on specific datasets for different tasks. Transfer learning is to apply the knowledge gained while solving one problem to a different but related problem. According to whether the feature spaces of two domains are same or not, it can be divided into homogeneous transfer learning and heterogeneous transfer learning. In our task, it belongs to homogeneous transfer learning. Deep transfer learning refers to transfer knowledge by deep neural networks. Thus, transfer learning makes it possible that identifying diseases from 2D face images by deep learning technique to provide a non-invasive and convenient way to realize early diagnosis and disease screening. In this paper, the next four diseases introduced and the corresponding health controls are selected to perform the validation.

1.3 History:

Thousands years ago, Huangdi Neijing , the fundamental doctrinal source for Chinese medicine, recorded "Qi and blood in the twelve Channels and three hundred and sixty-five Collaterals all flow to the face and infuse into the Kongqiao (the seven orifices on the face)." It indicates the pathological changes of the internal organs can be reflected in the face of the relevant areas. In China, one experienced doctor can observe the patient’s facial features to know the patient’s whole and local lesions, which is called “facial diagnosis”. Similar theories also existed in ancient India and ancient Greece. Nowadays, facial diagnosis refers to that practitioners perform disease diagnosis by observing facial features. The shortcoming of facial diagnosis is that getting a high accuracy facial diagnosis requires doctors to have a large amount of practical experience. Modern medical research indicates that, indeed, many diseases will express corresponding specific features on human faces.

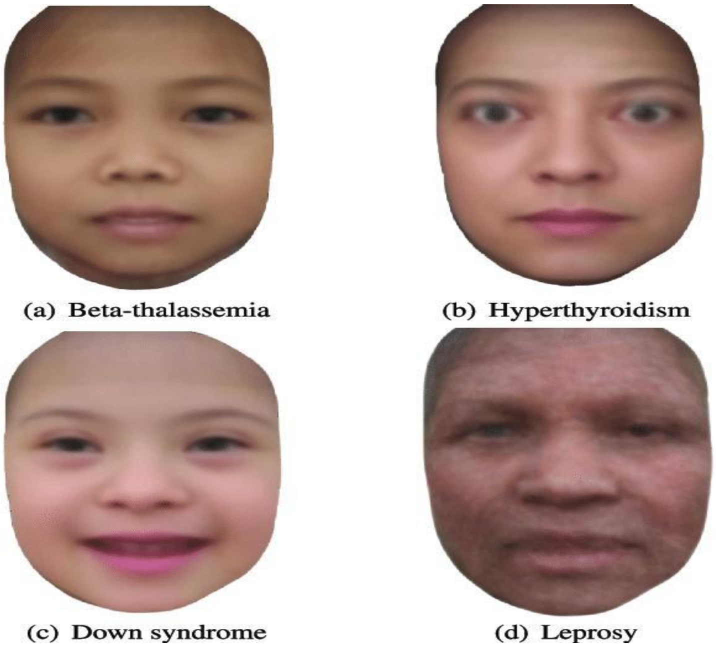
1.4 Details:

Thalassemia is a genetic disorder of blood caused by abnormal hemoglobin production, and it is one of the most common inherited blood disorders in the world. It is particularly common in people of Mediterranean, the Middle East, South Asian, Southeast Asian and Latin America. Since thalassemia can be fatal in early childhood without ongoing treatment, early diagnosis is vital for thalassemia. There are two different types of thalassemia: alpha (α) and beta (β). Beta-thalassemia is caused by mutations in the HBB gene which provides instructions for making a protein named beta globin on chromosome 11, and is inherited in an autosomal recessive fashion. It is estimated that the annual incidence of symptomatic beta-thalassemia individuals worldwide is 1 in 100,000. According to medical research, beta-thalassemia can result in bone deformities, especially in the face. The typical characteristics of beta-thalassemia on the face include small eye openings, epicanthal folds, low nasal bridge, flat midface, short nose, smooth philtrum, thin upper lip and underdeveloped jaw.

Hyperthyroidism is a common endocrine disease caused by excessive amounts of the thyroid hormones T3 and T4 which can regulate the body’s metabolism by various causes. The estimated average prevalence rate is 0.75% and the incidence rate is 51 per 100,000 persons per year by the metaanalysis . If it is not treated early, hyperthyroidism will cause a series of serious complications and even threaten the patient’s life. The typical characteristics of hyperthyroidism on the face include thinning hair, shining and protruding or staring eyes, increased ocular fissure, less blinking, nervousness, consternation and fatigue. The characteristic hyperthyroidism-specific face.

Down syndrome (DS) is a genetic disorder caused by the trisomy of chromosome 21. DS occurs in about one per one thousand newborns each year. The common symptoms include physical growth delays, mild to moderate intellectual disability, and the special face. The typical characteristics of DS on the face include large head compared to the face, upward slant of palpebral fissures, flattened nasal bridge, Brushfield spots, epicanthal fold, low-set, small, folded ears, short, broad nose with depressed root and full tip, a small oral cavity with broadened alveolar ridges and narrow palate, small chin and short neck. The characteristic DS-specific face.

Leprosy (also known as Hansen's disease) caused by a slow growing type of bacteria called Mycobacterium leprae is an infectious disease. If the leper doesn't accept timely treatment, leprosy will cause losing feelings of pain, weakness and poor eyesight. According to the World Health Organization, there are about 180,000 people infected with leprosy most of which are in Africa and Asia until 2017. The typical characteristics of leprosy [16] on the face include granulomas, hair loss, eye damage, pale areas of skin and facial disfigurement (e.g. loss of nose). The characteristic leprosy specific face.



(a) Beta-thalassemia

(b) Hyperthyroidism

(c) Down syndrome

(d) Leprosy

Fig 1.4 Diseased Faces

Identifying above diseases from uncontrolled 2D face images by deep learning technique has provided a good start for a non-invasive and convenient way to realize early diagnosis and disease screening. In this paper, our contributions are as follows:

(1) We definitely propose using deep transfer learning from face recognition to perform the computer-aided facial diagnosis on various diseases.

(2) We validate deep transfer learning methods for single and multiple diseases identification on a small dataset.

(3) Through comparison, we find some rules for deep transfer learning from face recognition to facial diagnosis.

CHAPTER-2

LITERATURE SURVEY

[1] . Bo Jin, Leandro Cruz, and Nuno Goncalves, "Deep Facial Diagnosis by Deep Transfer Learning ", ***IEEE Access*****, 29 June 2020.**

Training a CNN which is end to end learning from scratch will inevitably lead to over-fitting since that the training data is generally insufficient for the task of facial diagnosis. Transfer learning is applying the knowledge gained while solving one problem to a different but related problem.Deep transfer learning (DTL) is to transfer knowledge by pre-trained deep neural network which originally aims to perform facial verification and recognition in this paper. Thus the source task is face recognition and verification, and the target task is facial diagnosis. In this case, the feature spaces of the source domain and target domain are the same while the source task and the target task are different but related.The pretrained CNN is for end-to-end learning so that it can extract high-level features automatically. Since deep transfer learning is based on the fact that CNN features are more generic in early layers and more original dataset-specific in later layers, operation should be performed on the last layers of DCNN models.

[2]. Haihong Pan, Zaijun Pang, Yaowei Wang, Yijue wang , And Lin Chen , "Image recognition combining Transfer learning and MobileNet model", ***IEEE Access*****, July 9, 2020.**

Deep learning has been successfully applied to image analysis and target recognition. However, the use of deep learning to identify welding defects is time-consuming and less accurate due to the lack of adequate training data samples, which easily cause redundancy into the classifier.In this situation, we proposed a new transfer learning model based on MobileNet as a welding defect feature extractor. By using the

ImageNet dataset (non-welding defect data) to pre-train a MobileNet model, migrate the MobileNet model to the welding defects classification field. This article suggested a new TL-MobileNet structure by adding a new Full Connection layer (FC-128) and a Softmax classifier into a traditional model called MobileNet.

In this Paper, They proposed a new image recognition and classification method for welding defects, which combines the transfer learning algorithm and MobileNet model, namely TL-MobileNet model. This TL-MobileNet model has three advantages. (1) It can solve the problems of low prediction accuracy and time-consuming, which are induced by insufficient welding defects in learning samples. This model combines transfer learning theory with trained MobileNet model form a welding defects feature extractor. (2) It has an enhanced feature extraction capability, since it added a new Fully Connected layer (FC-128) and a Softmax classifier after the MobileNet.

[3]. Maciej Geremek and Krzysztof Szklanny , "Deep Learning-Based Analysis of Face Images" *Sensors 2021, 21, 6595.*

Deep learning models have demonstrated improved efficiency in image classification since the ImageNet Large Scale Visual Recognition in 2010. Classification of images has further augmented in the field of computer vision with the dawn of transfer learning. To train a model on a huge dataset demands huge computational resources and adds a lot of cost to learning. Transfer learning allows to reduce the cost of learning and also help avoid reinventing the wheel.This paper demonstrates image classification using pretrained deep neural network model VGG16 which is trained on images from ImageNet dataset. After obtaining the convolutional base model, a new deep neural network model is built on top of it for image classification based on a fully connected network. This classifier will use features extracted from the convolutional base model.Transfer learning allows to transfer the knowledge gained by previously learned task and apply it to similar another task . With transfer learning the base network which consists of different layers depending upon the architecture is trained on the base dataset and the learned parameters are transferred to another network. Convolutional neural network models have layered architecture where different features are learnt at each layer. Therefore, transfer learning can be easily accomplished with convolutional neural networks, where the lower layer acts as a feature extractor and the final layers are used to extract more specific features.

[4]. Barlian Khasoggi, Ermatita, Samsuryadi, “Efficient mobilenet architecture as image recognition”, Master of Informatics Engineering,Sriwijaya University, Indonesia, Vol. 16, No. 1, October 2019.

The introduction of a modern image recognition that has millions of parameters and requires a lot of training data as well as high computing power that is hungry for energy consumption so it becomes inefficient in everyday use. Machine Learning has changed the computing paradigm, from complex calculations that require high computational power to environmentally friendly technologies that can efficiently meet daily needs. To get the best training model, many studies use large numbers of datasets. However, the complexity of large datasets requires large devices and requires high computing power. Therefore large computational resources do not have high flexibility towards the tendency of human interaction which prioritizes the efficiency and effectiveness of computer vision. This study uses the Convolutional Neural Networks (CNN) method with MobileNet architecture for image recognition on mobile devices and embedded devices with limited resources with ARM-based CPUs and works with a moderate amount of training data (thousands of labeled images). With the level of accuracy and efficiency of the resources used, it is expected that MobileNet's architecture can change the machine learning paradigm so that it has a high degree of flexibility towards the tendency of human interaction that prioritizes the efficiency and effectiveness of computer vision. The MobileNet model is based on depth wise separable convolutions which is a procedure of factorized convolutional which factorizes a regular convolution into a depthwise convolution and a 1 × 1 convolution named a pointwise convolution.. A final average pooling reduces the spatial resolution to 1 before the fully connected layer. Counting depthwise and pointwise convolutions as separate layers, MobileNet has 28 layers.

CHAPTER –3

FEASIBILITY STUDY

3.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ♦ ECONOMICAL FEASIBILITY
- ♦ TECHNICAL FEASIBILITY
- ♦ SOCIAL FEASIBILITY

3.2 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.3 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.4 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER –4

REQUIREMENTS

4.1 Software Requirement Specification:

Requirement Analysis is the first activity in SDLC followed by Functional Specification and so on. Requirement analysis is a vital step in SDLC as it resonates with acceptance testing that is critical for product acceptance by customers. To make sure that all the steps mentioned above are appropriately executed, clear, concise, and correct requirements must be gathered from the customer. The customer should be able to define their requirements properly and the business analyst should be able to collect them in the same way the customer is intending it to convey them.

A System Requirements Specification (SRS) (also known as a Software Requirements Specification) is a document or set of documentation that describes the features and behavior of a system or software application. It includes a variety of elements (see below) that attempts to define the intended functionality required by the customer to satisfy their different users.

4.2 User Requirements:

User requirements are just what the name implies. They are requirements set by the end user. These requirements express how a facility, equipment or process should perform in terms of the product to be manufactured, required throughput, and conditions in which product should be made.

4.3 System Requirements:

System requirements is a statement that identifies the functionality that is needed by a system in order to satisfy the customer's requirements.System requirements are a broad and also narrow subject that could be implemented to many items.

Processor	- I3/Intel Processor
Hard Disk	-160GB
Key Board	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- SVGA
RAM	- 8Gb

4.4 Software Requirements:

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client’s point of view.

• Operating System	: Windows 7/8/10
• Server side Script	: Python, Anaconda
• IDE	: Pycharm
• Libraries Used	: Numpy, IO, OS, Flask, keras,
• Technology	: Python 3.6+

4.5 Requirements :

1. **IDE -** PyCharm
2. **Coding Language -** Python
3. **Libraries**:

a. NumPy:

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. It stands for 'Numerical Python'.

b. **Tensorflow**:

TensorFlow is a free and open-source software library for machine learning and artificial intelligence. TensorFlow provides a collection of workflows to develop and train models using Python or JavaScript, and to easily deploy in the cloud, on-prem, in the browser, or on-device no matter what language you use.

c. **Keras:**

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. Keras acts as an interface for the TensorFlow library. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation.

d. **Pandas**:

pandas is a software library written for the Python programming language for data manipulation and analysis.pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

4. **Dataset**:

The Dataset consists of over hundreds of Diseased Faces.The Face Images with four disease classes such as Beta-Thalassemia, Down Syndrome, Hyper-Thyroidism, Leprosy. The images are labeled and preprocessing techniques are used for predicting with the model performance and accuracy.

5. Model:

MobileNet model is used for facial diagnosis based on deep transfer learning on face recognition.MobileNet is a type of convolutional neural network designed for mobile and embedded vision applications. We train our model by a dataset with images , and predict which type of disease it is.

CHAPTER –5

ANALYSIS

5.1 Architecture of MobileNet:

MobileNet is a type of convolutional neural network designed for mobile and embedded vision applications. They are based on a streamlined architecture that uses depthwise separable convolutions to build lightweight deep neural networks that can have low latency for mobile and embedded devices. MobileNet uses depthwise separable convolutions. It significantly reduces the number of parameters when compared to the network with regular convolutions with the same depth in the nets. This results in lightweight deep neural networks (see Figure 5.1).

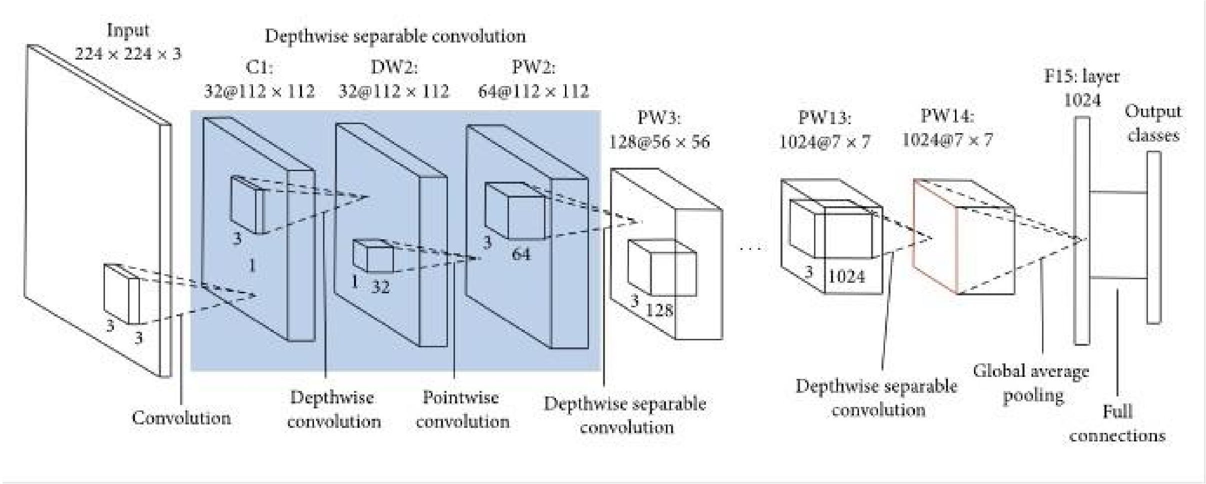


Fig 5.1 : Architecture of MobileNet

A depthwise separable convolution is made from two operations.

1. Depthwise convolution.
2. Pointwise convolution.

MobileNet is a class of CNN that was open-sourced by Google, and therefore, this gives us an excellent starting point for training our classifiers that are insanely small and insanely fast.

5.1.1: Depthwise Separable Convolution:

This convolution originated from the idea that a filter's depth and spatial dimension can be separated- thus, the name separable. Let us take the example of Sobel filter, used in image processing to detect edges.

Depthwise separable convolution is a depthwise convolution followed by a pointwise convolution as shown in the Figure 5.1.1.

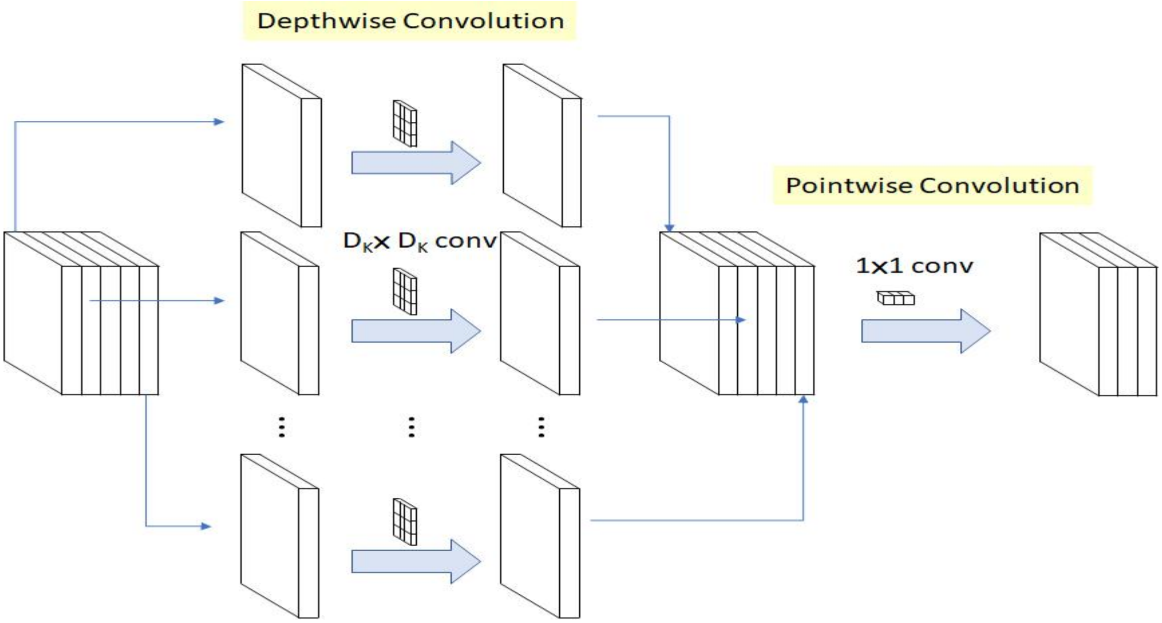


Fig 5.1.1: Depthwise Separable Convolution

1. Depthwise convolution is the channel-wise DK×DK spatial convolution. Suppose in the figure above, and we have five channels; then, we will have 5 DK×DK spatial convolutions.

2. Pointwise convolution is the 1×1 convolution to change the dimension.

It is a map of a single convolution on each input channel separately. Therefore its number of output channels is the same as the number of the input channels. Its computational cost is

$Df^2 * M * Dk^2$.

5.1.2: Pointwise Convolution:

Convolution with a kernel size of 1x1 that simply combines the features created by the depthwise convolution as shown in the figure 5.1.2.

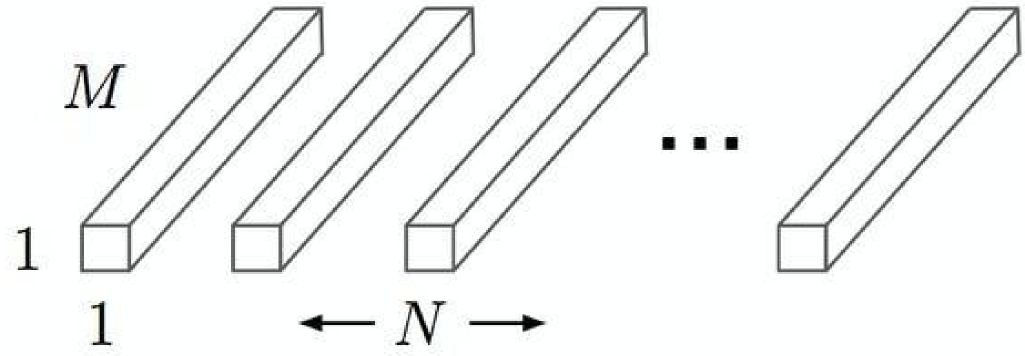


Fig 5.1.2: Pointwise Convolution

5.2: Convolutional Layer and MobileNet:

The main difference between MobileNet architecture and a traditional CNN instead of a single 3x3 convolution layer followed by the batch norm and ReLU. Mobile Nets split the convolution into a 3x3 depth-wise conv and a 1x1 pointwise conv, as shown in the Figure 5.2.

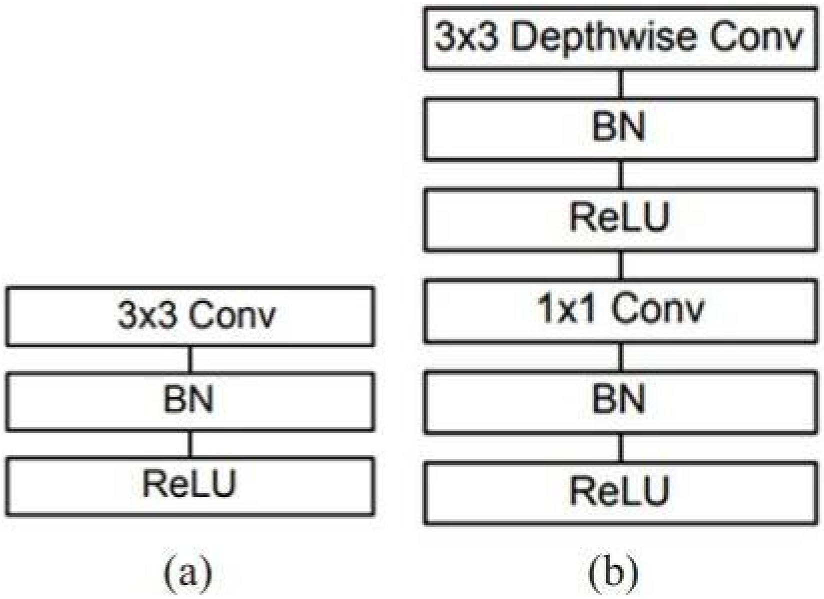


Fig 5.2: Standard convolutional layer and MobileNet

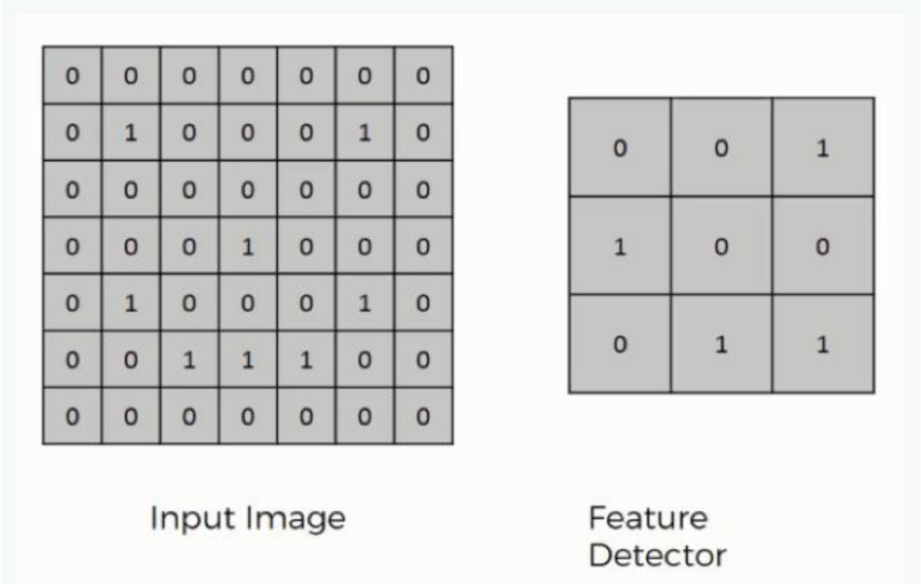
5.3 Algorithm

MobileNet

Step(1a): convolutional operation

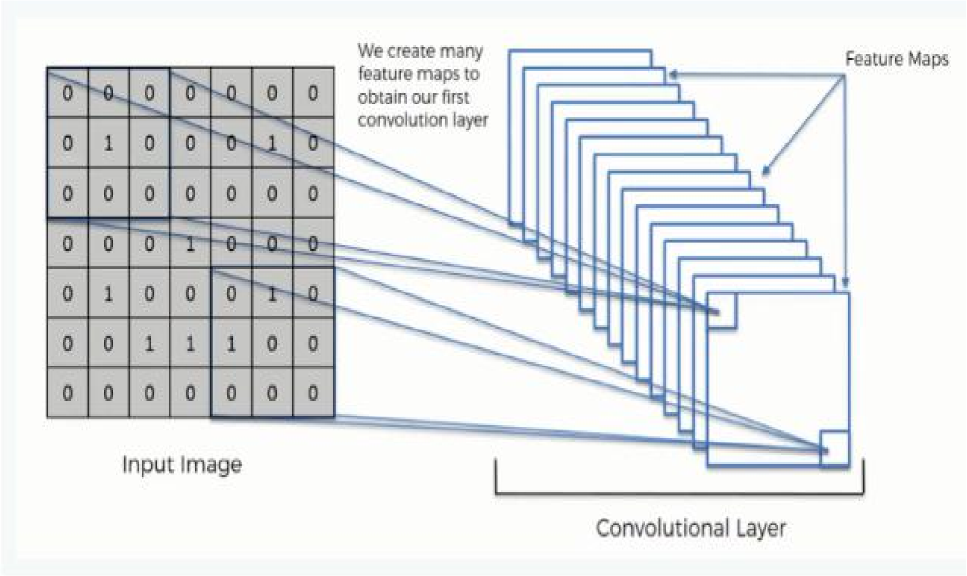
The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out see figure 5.3.

The Convolution Operation



0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

0	0	1
1	0	0
0	1	1



We create many feature maps to obtain our first convolution layer
Feature Maps
Input Image
Convolutional Layer

Fig.5.3 Convolution Operation

Step (1b): Relu Layer

The second part of this step will involve the Rectified Linear Unit or Relook. We will cover Relook layers and explore how linearity functions in the context of Convolutional Neural Networks. Not necessary for understanding CNN's, but there's no harm in a quick lesson to improve your skills.

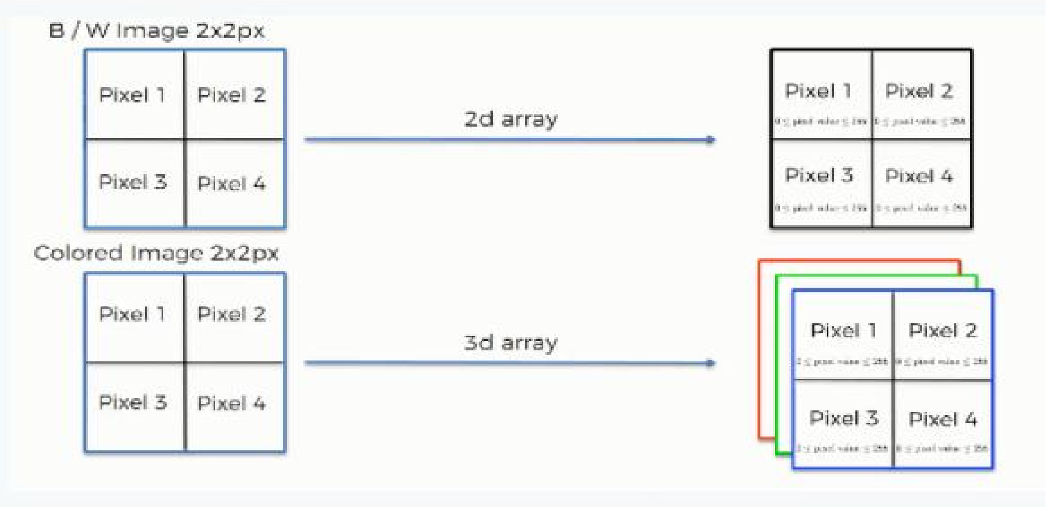


Fig.5.3.1 scanning of Images

Step 2: Pooling Layer

In this part, we'll cover pooling and will get to understand exactly how it generally works. Our nexus here, however, will be a specific type of pooling; max pooling. We'll cover various approaches, though, including mean (or sum) pooling. This part will end with a demonstration made using a visual interactive tool that will definitely sort the whole concept out for you.

Step 3: Flattening

This will be a brief breakdown of the flattening process and how we move from pooled to flattened layers when working with Convolutional Neural Network such as MobileNet.

Step 4: Full Connection

In this part, everything that we covered throughout the section will be merged together. By learning this, you'll get to envision a fuller picture of how MobileNet networks operate and how the "neurons" that are finally produced learn the classification of images.MobileNet uses pointwise convolutional layer to connect all layers from depthwise layer.

5.4 Deep Transfer Learning:

Transfer learning is the idea of overcoming the isolated learning paradigm and utilizing knowledge acquired for one task to solve related ones.Thus, the key motivation, especially considering the context of deep learning is the fact that most models which solve complex problems need a whole lot of data, and getting vast amounts of labeled data for supervised models can be really difficult, considering the time and effort it takes to label data points. A simple example would be the ImageNet dataset, which has millions of images pertaining to different categories.

Transfer learning, is not a new concept which is very specific to deep learning. There is a stark difference between the traditional approach of building and training machine learning models, and using a methodology following transfer learning principles.

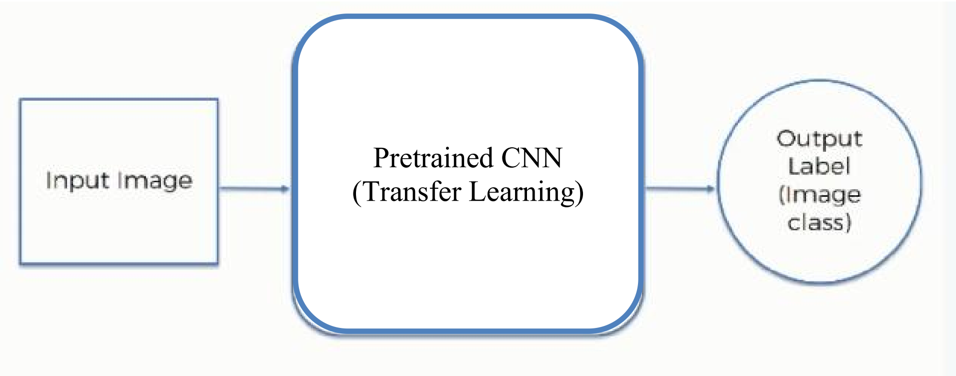


Fig 5.4 Transfer Learning

5.5 Flow Chart of Project:

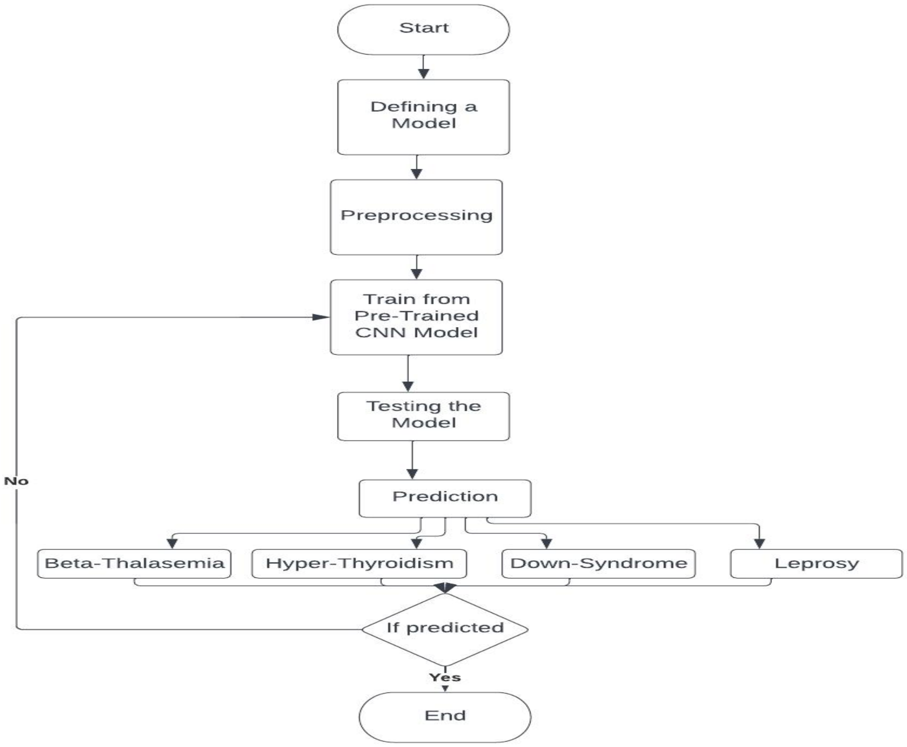


Fig 5.5 Flowchart of Project

CHAPTER –6

DESIGN

6.1 UML Introduction:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

- The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.
- The UML represents a collection of best engineering practices that have proven successful inssss the modeling of large and complex systems.
- The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

6.2 Use Case Diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.(see Figure 6.2).

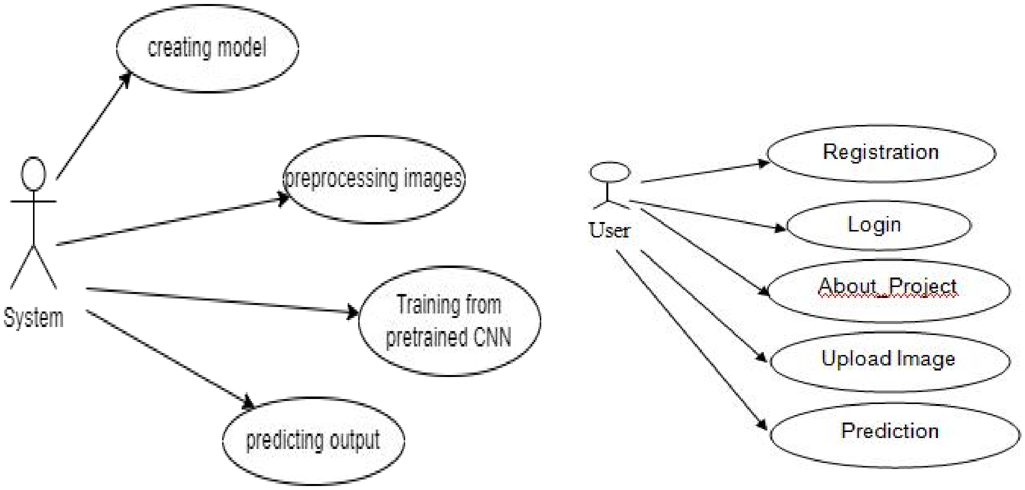


Fig 6.2: Use case diagram for Facial Diagnosis

6.3 Data Flow Diagram:

A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.(see Figure 6.2)

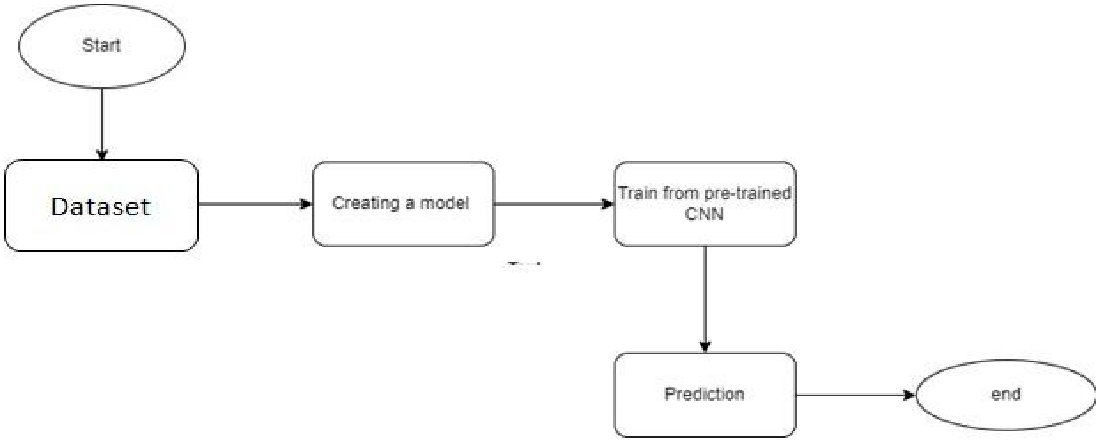


Fig 6.3: Data Flow Diagram for Facial Diagnosis

6.4 : ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.(see Figure 6.4).

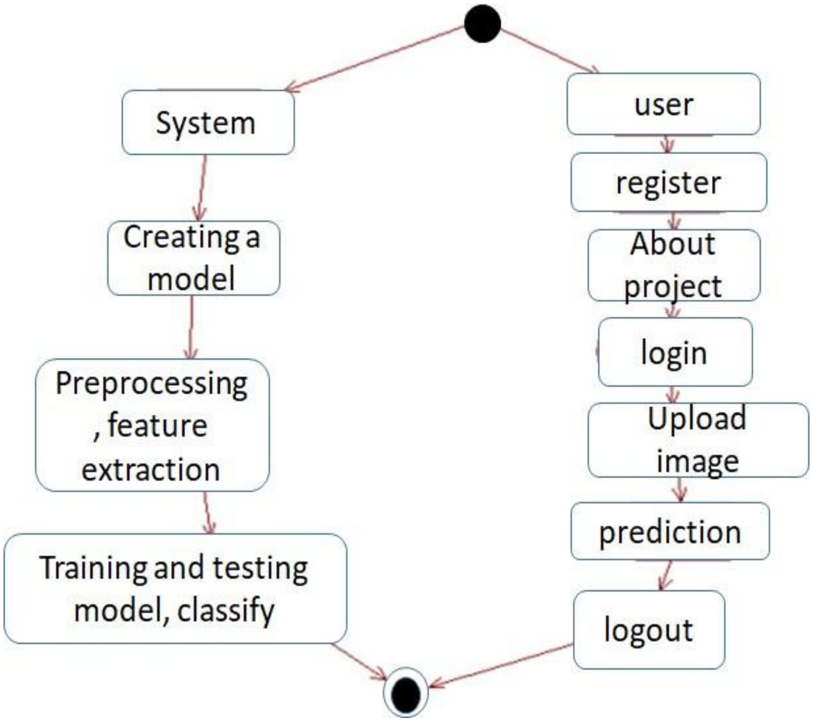


Fig 6.4: Activity Diagram for Facial Diagnosis

6.5 ER Diagram:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.(see Figure 6.5).

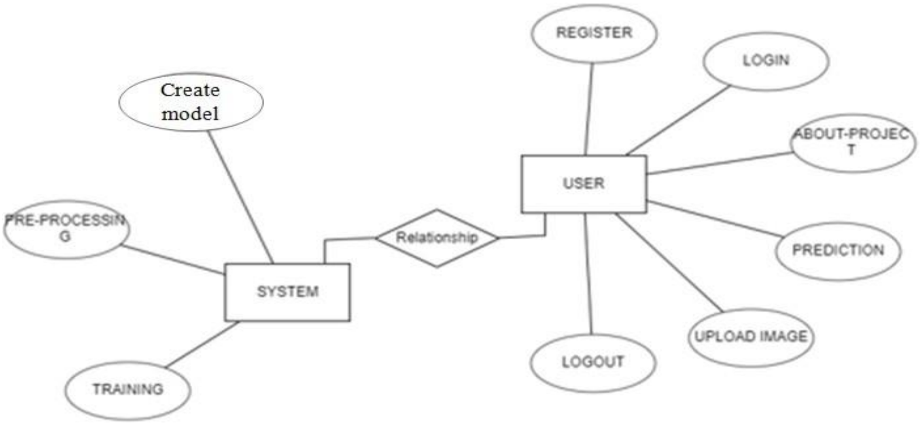


Fig 6.5: E-R Diagram for Facial Diagnosis

CHAPTER- 7

MODEL

7.1 Steps of SDLC :

A typical Software Development Life Cycle consists of the following stages −

Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS(Software Requirement Specification)..

Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

CHAPTER –8

IMPLEMENTATION

Here in our project we are going to use deep transfer learning of face recognition for facial diagnosis. The MobileNet model is used, which gives us 90% of model accuracy. Different types of libraries are used. The steps that are followed are :

8.1 : Libraries Used:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

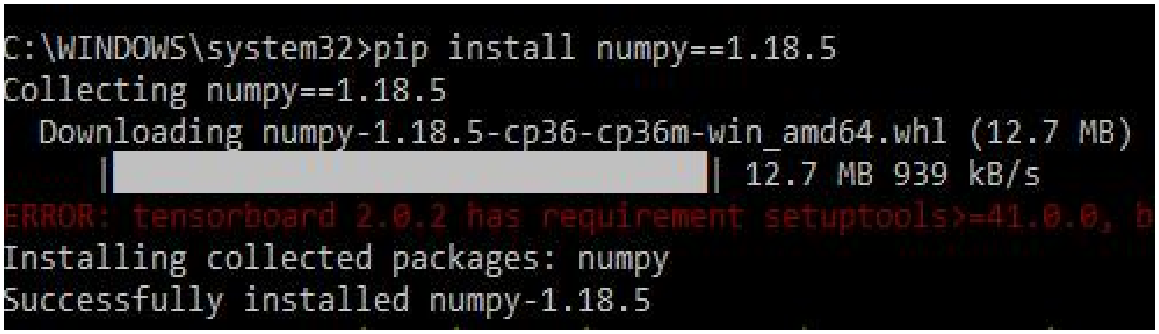
NumPy:

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. It stands for 'Numerical Python'.(see Figure 8.1.1).

Numpy provides the Essential multi-dimensional array-oriented computing functionalities designed for high-level mathematical functions and scientific computation. NumPy can be imported into pyCharm using:

```
Import numpy as np
```

To install numpy:

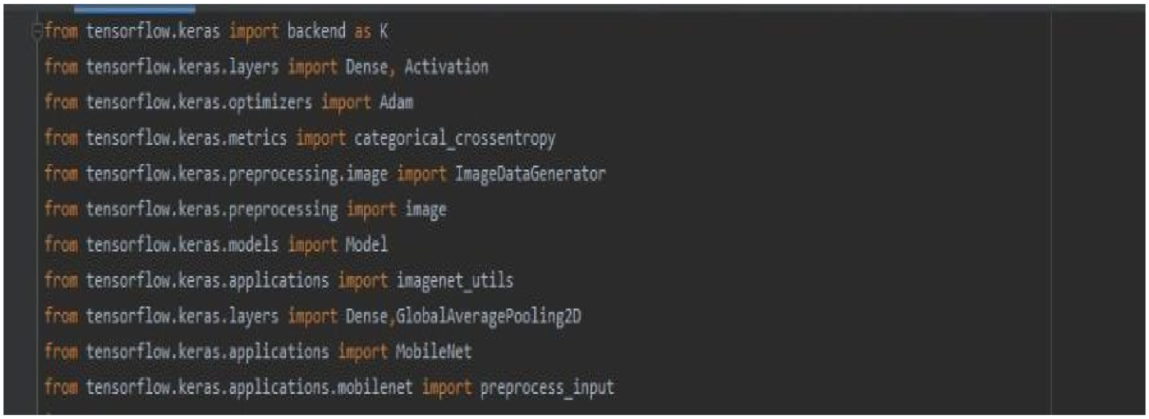


```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
     |████████████████████████████████| 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

Fig 8.1.1: NumPy library

Tensorflow:

TensorFlow is a free and open-source software library for machine learning and artificial intelligence. TensorFlow provides a collection of workflows to develop and train models using Python or JavaScript, and to easily deploy in the cloud, on-prem, in the browser, or on-device no matter what language you use.(see Figure 8.1.2).



```
from tensorflow.keras import backend as K
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import categorical_crossentropy
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import Model
from tensorflow.keras.applications import imagenet_utils
from tensorflow.keras.layers import Dense,GlobalAveragePooling2D
from tensorflow.keras.applications import MobileNet
from tensorflow.keras.applications.mobilenet import preprocess_input
```

Fig 8.1.2 : Tenserflow on Keras

Keras:

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. Keras acts as an interface for the

TensorFlow library. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation.Tenserflow on keras is used in our project to make a optimal solution.

Pandas:

pandas is a software library written for the Python programming language for data manipulation and analysis.pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

```
Import pandas as pd
```

Pip:

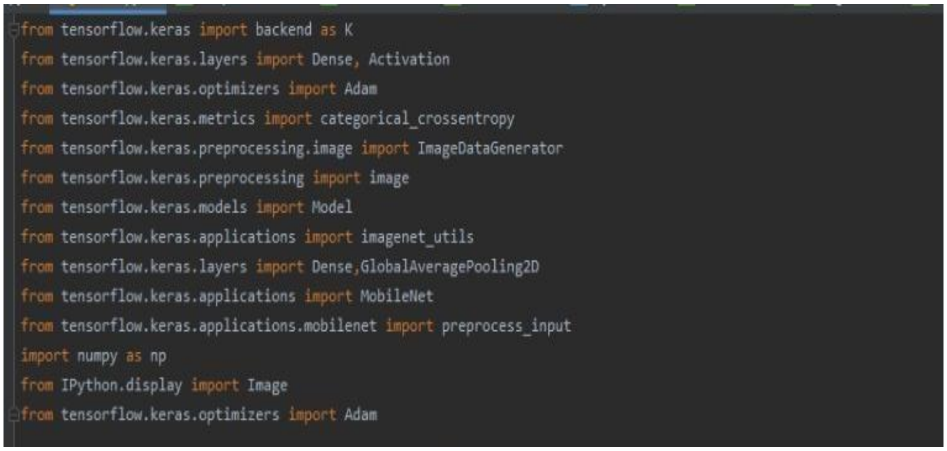
Pip is a recursive acronym that can stand for either "Pip Installs Packages" or "Pip Installs Python". Alternatively, pip stands for "**preferred installer program**". To install any package :

```
Pip install package name
```

8.2 Implementation:

We import libraries in the required environment i.e., PyCharm. And importing image dataset with the help of libraries.(see figure 8.2.1)

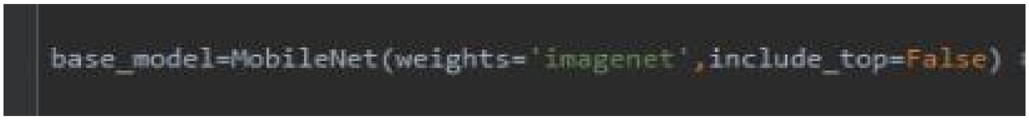
Implementing Libraries:



```
from tensorflow.keras import backend as K
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import categorical_crossentropy
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import Model
from tensorflow.keras.applications import imagenet_utils
from tensorflow.keras.layers import Dense,GlobalAveragePooling2D
from tensorflow.keras.applications import MobileNet
from tensorflow.keras.applications.mobilenet import preprocess_input
import numpy as np
from IPython.display import Image
from tensorflow.keras.optimizers import Adam
```

Fig 8.2: Libraries imported

Input:



```
base_model=MobileNet(weights='imagenet',include_top=False)
```

Fig 8.2.1: Input to Model

Importing the images and freezing the last layers of MobileNet model.

Dataset:

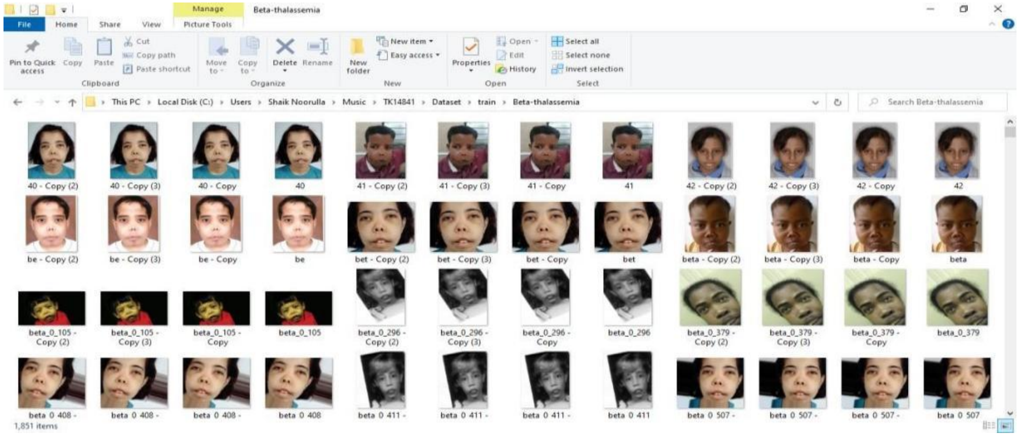


Fig 8.2.2: Augumented Dataset

8.3 Algorithm & WorkFlow:

Steps:

1. Load the pretrained ImageNet weights and set include_top=False to not include the final pooling and fully connected layer in the original model.

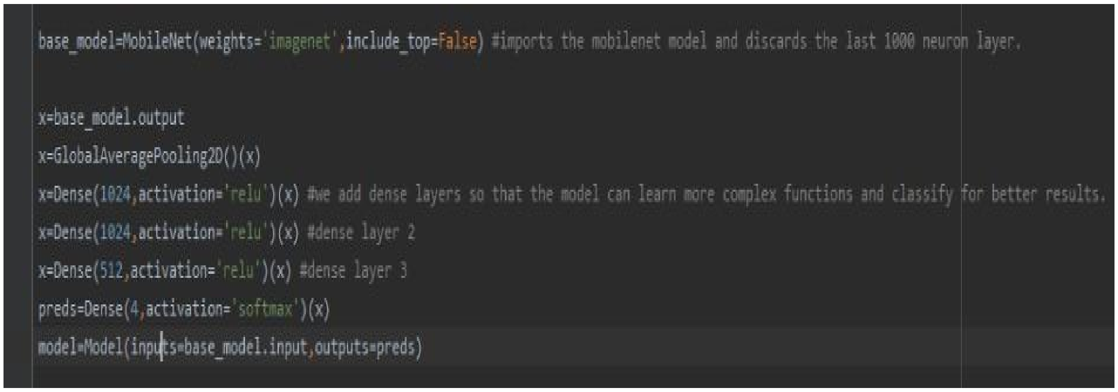
MobileNet(weights='imageNet',include_top=False)

2. Adding a Pooling layer to a model, to reduce the dimensions of the feature maps.

GlobalAveragePooling2D( )

3. Adding dense Activation Layers like Relu(Rectified Linear Unit),(see Figure 8.3)

Code:



```
base_model=MobileNet(weights='imagenet',include_top=False) #imports the mobilenet model and discards the last 1000 neuron layer.

x=base_model.output
x=GlobalAveragePooling2D()(x)
x=Dense(1024,activation='relu')(x) #we add dense layers so that the model can learn more complex functions and classify for better results.
x=Dense(1024,activation='relu')(x) #dense layer 2
x=Dense(512,activation='relu')(x) #dense layer 3
preds=Dense(4,activation='softmax')(x)
model=Model(inputs=base_model.input,outputs=preds)
```

Fig 8.3: Adding Activation layers

4. Splitting the train datset and test dataset and setting first 20 Layers Non-Trainable.

5. Feeding the data to the Model using ImageDataGenerator function:

train_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)

```
train_generator=train_datagen.flow_from_directory('Dataset/train',
                                                  target_size=(224,224),
                                                  color_mode='rgb',
                                                  batch_size=32,
                                                  class_mode='categorical',

                                                  shuffle=True)
```

- The **directory** must be set to the path where your ‘n’ classes of folders are present.
- The **target_size** is the size of your input images, every image will be resized to this size.
- **color_mode:** if the image is either black and white or grayscale set “grayscale” or if the image has three color channels, set “rgb”.
- **batch_size:** No. of images to be yielded from the generator per batch.
- **class_mode:** Set “binary” if you have only two classes to predict, if not set to“categorical”, in case if you’re developing an Autoencoder system, both input and the output would probably be the same image, for this case set to “input”.
- **shuffle:** Set True if you want to shuffle the order of the image that is being yielded, else set False.(see Figure 8.3.1).

6. Testing the data

```
test_set = train_datagen.flow_from_directory('Dataset/test',
                                             target_size=(224,224),
                                              batch_size=32,
                                          class_mode='categorical')
```

7. Compile the Model and Adding Batch Normalization Step (see Figure 8.3.2),

```
model.compile(optimizer='Adam',loss='categorical_crossentropy',
metrics=['accuracy'])
step_size_train=train_generator.n//train_generator.batch_size
```

8. Getting Model Accuracy

```
history=model.fit_generator(generator=train_generator,
                   steps_per_epoch=step_size_train,
                   epochs=5)
```

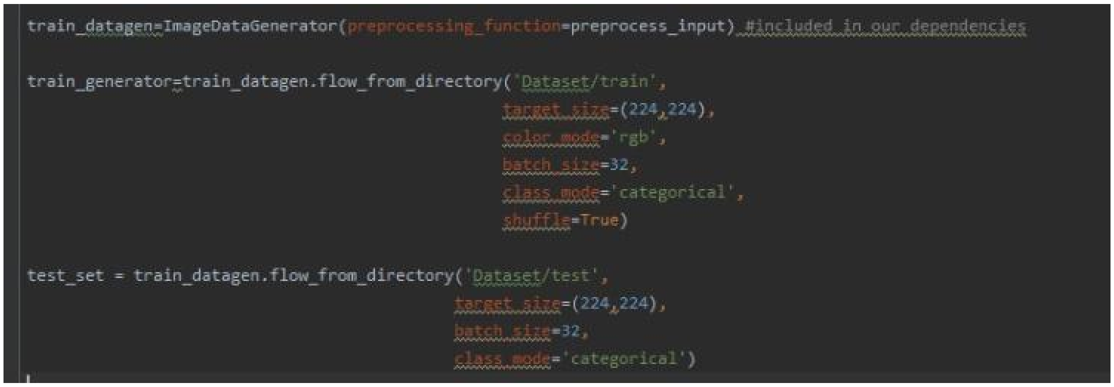
9. Saving the output as .h5 file.

```
model.save("alg/MobileNet.h5")
```

10. importing Matplot Library and plotting graph.

```
    from matplotlib import pyplot as plt
plt.plot(history.history['accuracy'],'r',label='Testing accuracy',color='green')
```

Code:

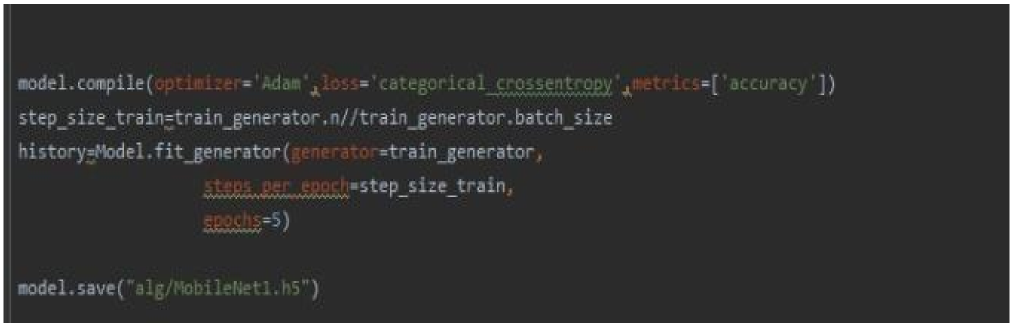


```
train_datagen=ImageDataGenerator(preprocessing_function=preprocess_input) #included in our dependencies

train_generator=train_datagen.flow_from_directory('Dataset/train',
                                                 target_size=(224,224),
                                                 color_mode='rgb',
                                                 batch_size=32,
                                                 class_mode='categorical',
                                                 shuffle=True)

test_set = train_datagen.flow_from_directory('Dataset/test',
                                            target_size=(224,224),
                                            batch_size=32,
                                            class_mode='categorical')
```

Fig 8.3.1: Training and Testing of Data



```
model.compile(optimizer='Adam',loss='categorical_crossentropy',metrics=['accuracy'])
step_size_train=train_generator.n//train_generator.batch_size
history=Model.fit_generator(generator=train_generator,
                   steps_per_epoch=step_size_train,
                   epochs=5)

model.save("alg/MobileNet1.h5")
```

Fig 8.3.2: Compiling and saving file

CHAPTER –9

TESTING

9.1 Types of Testing:

Types of Testing:

Software testing is the act of examining the artifacts and the behavior of the software under test by validation and verification. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation.The Main objective of testing is to uncover a host of errors, systematically and minimum effort and time. Starting formally, Testing is a process of executing a program with the intent of finding an error.

- A successful test is one that uncovers an as yet undiscovered error.

- A good test case is one that has a high probability of finding error,if it exists.

The first approach is what known as Black box testing and the second approach is white box testing. We apply white box testing techniques to as certain the functionalities top-down and then use black box testing techniques to demonstrate that everything runs as expected.

Black-Box Testing:

Black box testing involves testing against a system where the code and paths are invisible. Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.

- Initially, the requirements and specifications of the system are examined.
- Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.

9.2 Unit Testing:

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. This testing methodology is done during the development process by the software developers.

Unit testing is an important step in the development process, because if done correctly, it can help detect early flaws in code which may be more difficult to find in later testing stages. A unit test typically comprises of three stages: plan, cases and scripting and the unit test itself. In the first step, the unit test is prepared and reviewed. The next step is for the test cases and scripts to be made, then the code is tested.

Each test case is tested independently in an isolated environment, as to ensure a lack of dependencies in the code. The software developer should code criteria to verify each test case, and a testing framework can be used to report any failed tests. Developers should not make a test for every line of code, as this may take up too much time. Developers should then create tests focusing on code which could affect the behavior of the software being developed.

9.3 Integration Testing:

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

Unit testing uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules. Once all the components or modules are working independently, then we need to check the data flow between the dependent modules is known as integration testing.

Integration testing is performed using the black box method. This method implies that a testing team interacts with an app and its units via the user interface – by clicking on buttons and links, scrolling, swiping, etc. They don't need to know how code works or consider the backend part of the components.

- Good for testing small systems.
- Allows finding errors very quickly, and thus saves a lot of time.

9.4 Functional Testing:

Functional Testing is a type of software testing that validates the software system against the functional requirements/specifications. The purpose of Functional tests is to test each function of the software application, by providing appropriate input, verifying the output against the Functional requirements.

Functional testing mainly involves black box testing and it is not concerned about the source code of the application. This testing checks User Interface, APIs, Database, Security, Client/Server communication and other functionality of the Application Under Test. The testing can be done either manually or using automation.

- Understand the Functional Requirements

- Identify test input or test data based on requirements
- Compute the expected outcomes with selected test input values
- Execute test cases
- Compare actual and computed expected results

9.5 System Testing:

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In system testing, integration testing passed components are taken as input.

The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested.Performance Testing is a type of software testing that is carried out to test the speed, scalability, stability and reliability of the software product or application.

Two Category of Software Testing

- Black Box Testing
- White Box Testing

System test falls under the black box testing category of software testing.

White box testing is the testing of the internal workings or code of a software application. In contrast, black box or System Testing is the opposite. System test involves the external workings of the software from the user's perspective.

- Testing the fully integrated applications including external peripherals in order to check how components interact with one another and with the system as a whole. This is also called End to End testing scenario.

- Verify thorough testing of every input in the application to check for desired outputs.
- Testing of the user’s experience with the application.

9.6 White Box Testing:

White Box Testing is software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design, usability and security. In white box testing, code is visible to testers so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing and Glass box testing.

It is one of two parts of the Box Testing approach to software testing. Its counterpart, Blackbox testing, involves testing from an external or end-user type perspective. On the other hand, White box testing in software engineering is based on the inner workings of an application and revolves around internal testing.

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code
- Expected output
- The functionality of conditional loops
- Testing of each statement, object, and function on an individual basis.

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

9.7 Black Box Testing:

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.

- Initially, the requirements and specifications of the system are examined.
- Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also, some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

9.8 Test Strategy and approach:

Step 1: We have taken a test dataset directory with diseased face images to test the accuracy of model. The dataset is splitted in to two directories.

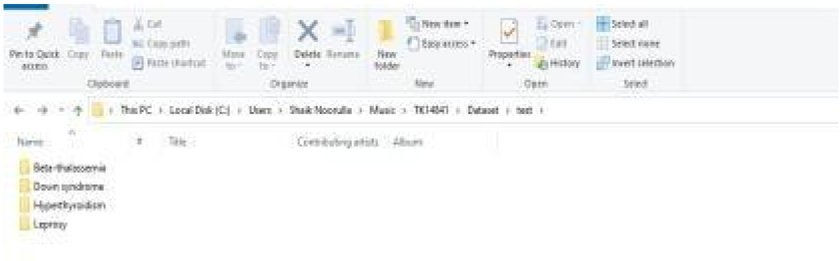
➢ Training set

➢

Train Test

Testing Set

Step 2: Create a test directory with four classes such as HyperThyroidism, Down-Syndrome, Leprosy, Hyper-Thyroidism.(see Figure 9.2).



This PC › Local Disk (C:) › Users › Shaik Noorulla › Music › TK14841 › Dataset › test ›

Beta-thalassemia
Down syndrome
Hyperthyroidism
Leprosy

Fig 9.8.1: Test Dataset

Step 3: Import the images from Test Directory to Model.py using code (see Figure 9.2.1).

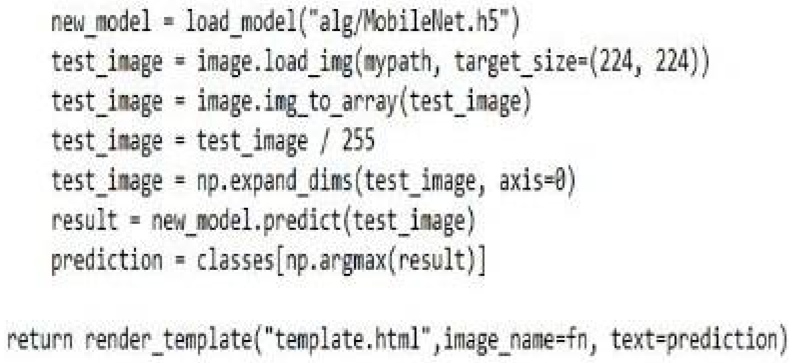
```
test_set = train_datagen.flow_from_directory('Dataset/test',
                                             target_size=(224,224),
                                             batch_size=32,
                                             class_mode='categorical')
```

Code:

```
test_set = train_datagen.flow_from_directory('Dataset/test',
                                             target_size=(224,224),
                                             batch_size=32,
                                             class_mode='categorical')
```

Fig 9.8.2: Test_set code

Step 4: Testing the Image size, Image dimensions.(see Figure 9.2.2).



```
    new_model = load_model("alg/MobileNet.h5")
    test_image = image.load_img(mypath, target_size=(224, 224))
    test_image = image.img_to_array(test_image)
    test_image = test_image / 255
    test_image = np.expand_dims(test_image, axis=0)
    result = new_model.predict(test_image)
    prediction = classes[np.argmax(result)]

return render_template("template.html",image_name=fn, text=prediction)
```

Fig 9.8.3 : Test Image

Step 5: Testing Accuracy as shown in the Figure 9.2.3.

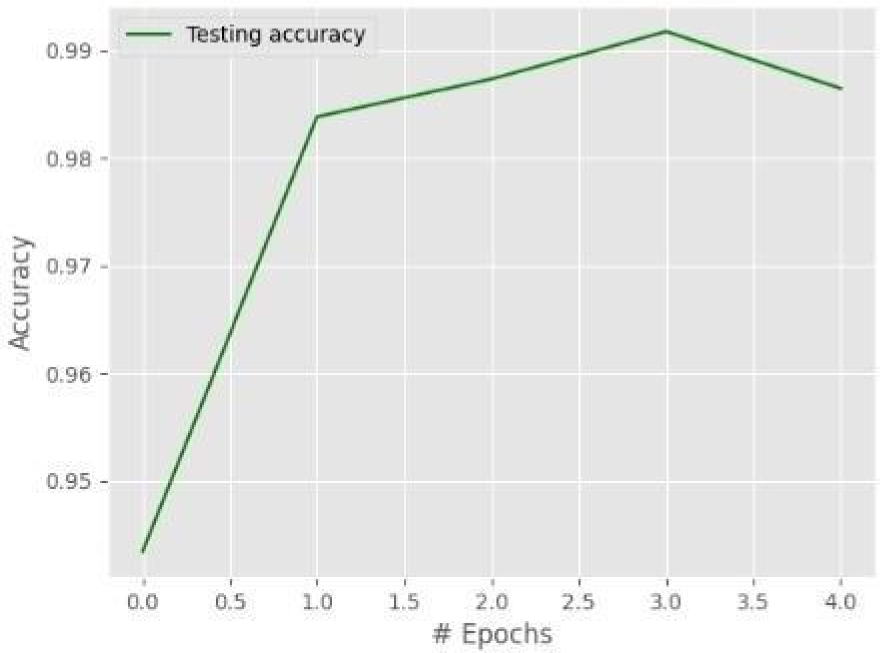


Fig 9.8.4: Test Accuracy

CHAPTER –10

EXECUTION AND RESULTS

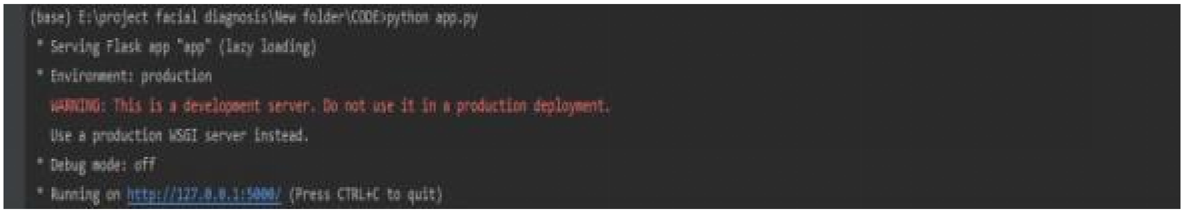
The proposed system is developed by using python. Various datastes like diseased face dataset and crop yield datasets are collected from the different sources like kaggle.com. MobileNet Model is used for predicting Facial diseases such as HyperThyroidism, Beta-Thalasemia, Down-syndrome, Leprosy.The libraries like Tenserflow, Keras, numpy, pandas etc., are used.

10.1 Execution:

To execute the code in pyCharm :

>Python app.py

Run Time Environment:



```
(base) E:\project facial diagnosis\New folder\CODE>python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Fig 10.1 Run Time Environment

10.2 Results:

1.Home Page as shown in the Figure 10.2.1.

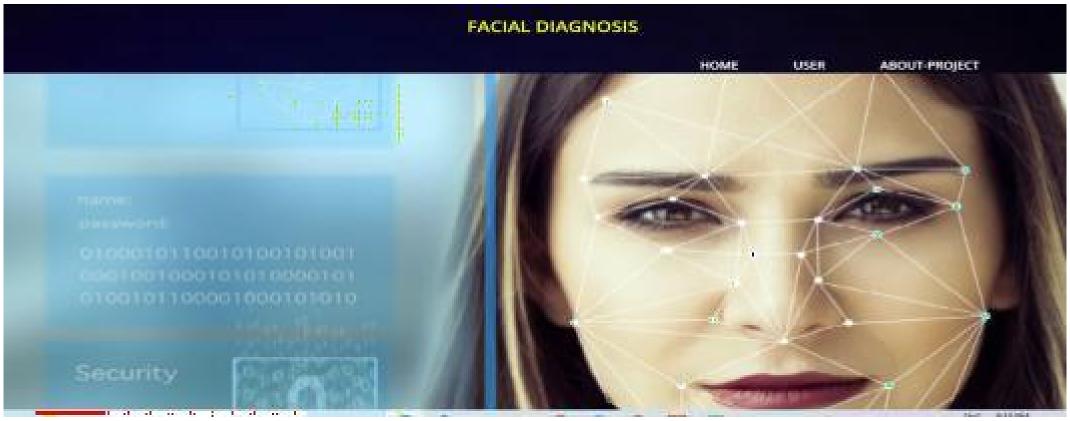


Fig 10.2.1: Home Page

2. User Registration page as shown in the Figure 10.2.2.

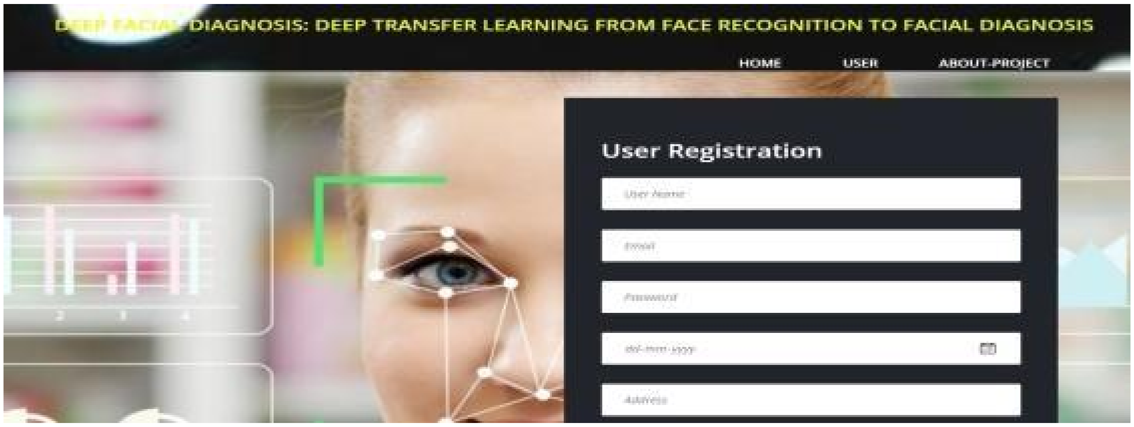

DEEP FACIAL DIAGNOSIS: DEEP TRANSFER LEARNING FROM FACE RECOGNITION TO FACIAL DIAGNOSIS
HOME
USER
ABOUT-PROJECT
User Registration
User Name
Email
Password
dd-mm-yyyy
Address

Fig 10.2.2 User Registration

3. User Login Form as shown in the Figure 10.2.3.

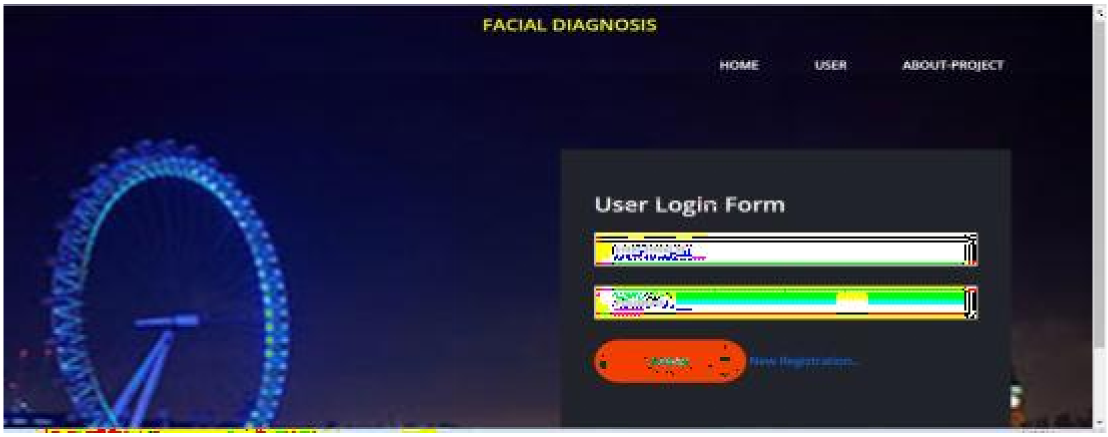


Fig 10.2.3 : User Login Form

4. Upload Image as shown in the Figure 10.2.4.

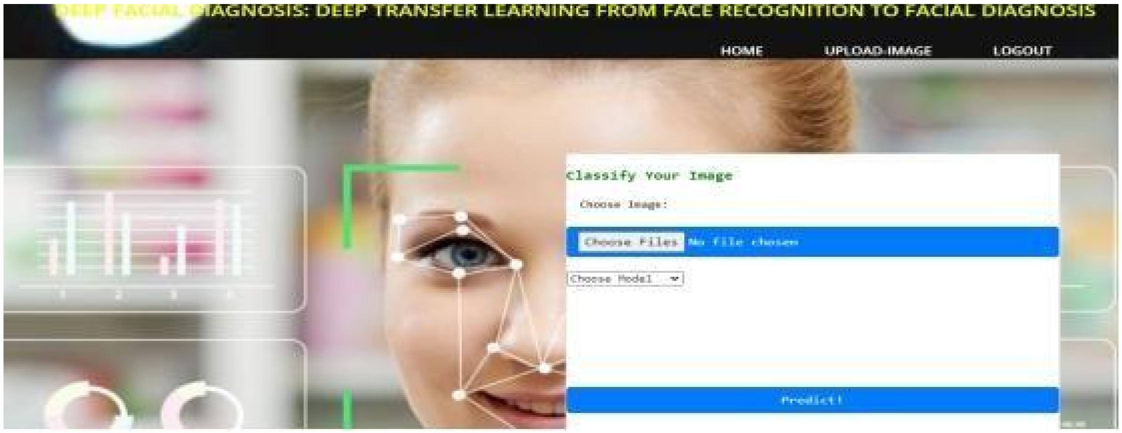
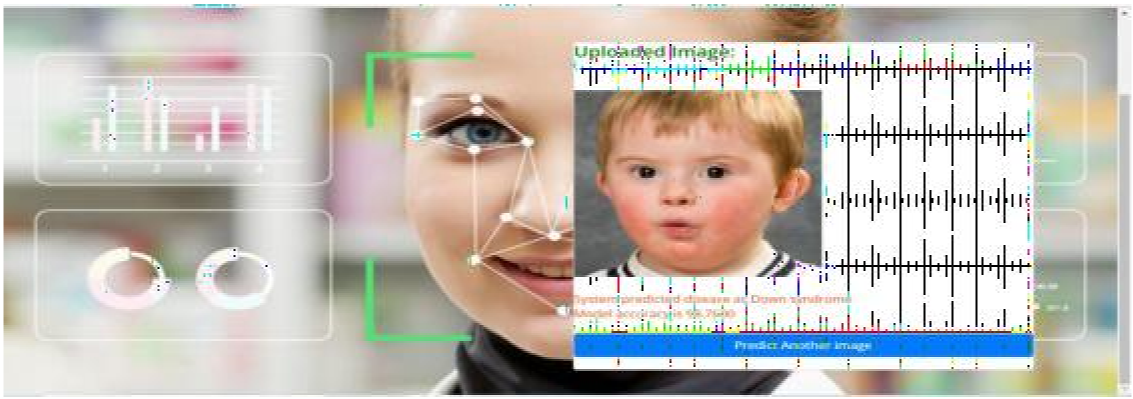


Fig 10.2.4 Upload image Page

5.Predictions

1. Down Syndrome (see Figure 10.2.5)

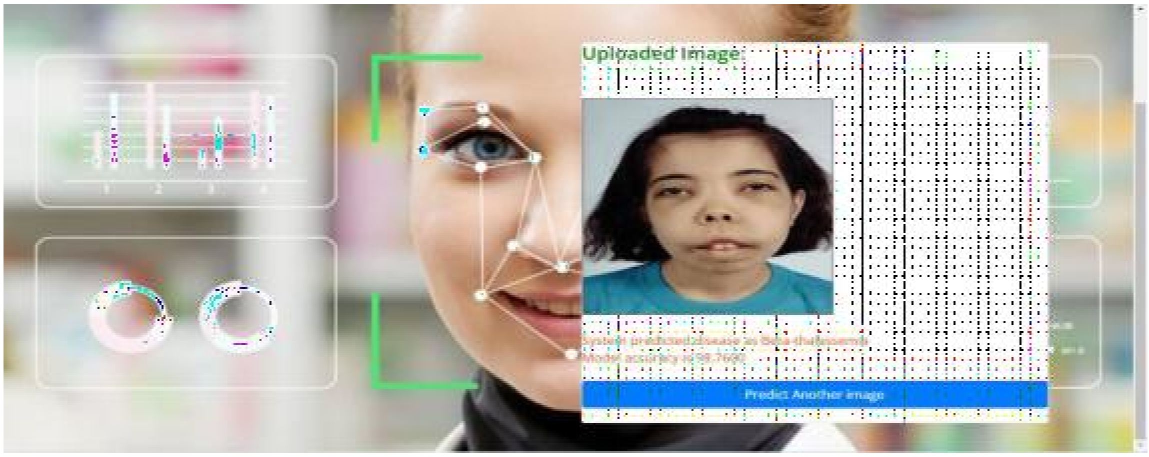


System predicted disease as Down syndrome

Model accuracy is 98.7600

Fig 10.2.5 : Down syndrome Prediction

2.Beta-Thalassemia (see Figure 10.2.6)

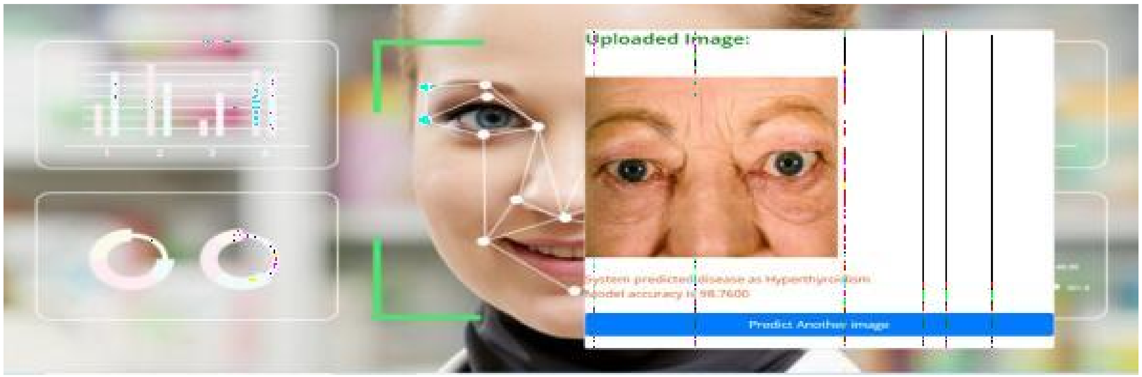


System predicted disease as Beta-thalassemia

Model accuracy is 98.7600

Fig 10.2.6: Beta-Thalassemia Prediction

3. HyperThyroidism (see Figure 10.2.7).



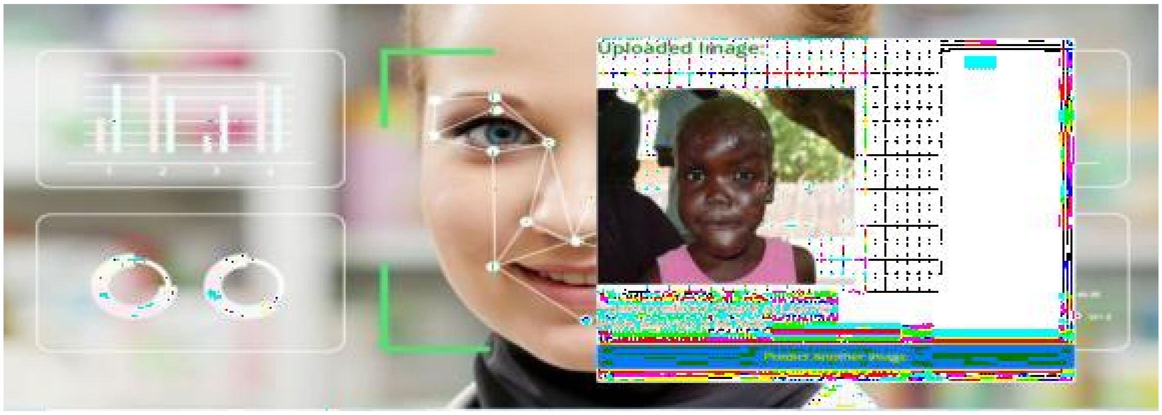
Uploaded Image:

System predicted disease as Hyperthyroidism

Model accuracy is 98.7600

Fig 10.2.7: HyperThyroidism Prediction

4.Leprosy(see Figure 10.2.8).



Uploaded Image:

[illegible]

Fig 10.2.8 :Leprosy Prediction

FUTURE WORK

In Future Work , we would like to collect more features and we have have ability to add more features further. We would like to add cross validation process in future in order to validate our results. We would also like to use better deep learning models and other states of works and compare it with the results obtained.The developed Model can be used in future to Diagnose the remaining faces.

CONCLUSION

Facial diagnosis is one of the most important applications in facial analysis. It canbe traced back thousands of years when traditional medicine practitioners applied their knowledge to diagnose the health status of an individual. A lot has since changed since then and computerized facial diagnosis has since replaced this centuries old practice. The relationship between face and disease has been discussed from thousands years ago, which leads to the occurrence of facial diagnosis. The objective here is to explore the possibility of identifying diseases from uncontrolled 2D face images by deep learning techniques. In practical, collecting disease-specific face images is complex, expensive and time consuming, and imposes ethical limitations due to personal data treatment. Therefore, the datasets of facial diagnosis related researches are private and generally small comparing with the ones of other machine learning application areas. The success of deep transfer learning applications in the facial diagnosis with a small dataset could provide a low-cost and noninvasive way for disease screening and detection.

More and more studies have shown that computer-aided facial diagnosis is a promising way for disease screening and detection. In this paper, we propose deep transfer learning from face recognition methods to realize computer-aided facial diagnosis defifinitely and validate them on single disease and various diseases with the healthy control. The experimental results of above 90% accuracy have proven that CNN as a feature extractor is the mostappropriate deep transfer learning method in the case of the small dataset of facial diagnosis. It can solve the general problem of insuffificient data in the facial diagnosisarea to a certain extent. In future, we will continue to discover deep learning models to perform facial diagnosis effectively with the help of data augmentation methods. We hope that more and more diseases can be detected effificiently by face photographs.

REFERENCES

[1]. Bo Jin, Leandro Cruz, and Nuno Goncalves, "Deep Facial Diagnosis ", *IEEE Access*, 29 June 2020.

[2]. Haihong Pan, Zaijun Pang, Yaowei Wang, Yijue wang , And Lin Chen , "Image recognition combining Transfer learning and MobileNet model", *IEEE Access*, July 9, 2020.

[3]. Barlian Khasoggi, Ermatita, Samsuryadi, "Efficient mobilenet architecture as image recognition", Master of Informatics Engineering,Sriwijaya University, Indonesia, Vol. 16, No. 1, October 2019.

[4]."Image Classification Using Transfer Learning and Deep Learning" International Journal Of Engineering And Computer Science Volume :10 Issue: 9 September 2021.

[5]. Kaun Wang, jeibo luo , "Detecting visually observable disease from faces", vol:9, Mar 2016.