

DIABETIC RETINOPATHY DETECTION USING IMAGE PROCESSING

A PROJECT REPORT

Submitted by

KATHIRVEL K S [REGISTER NO:211418104114]

NAGAKUMAR B [REGISTER NO:211418104165]

SALAI ARUN MANI J B [REGISTER NO:211418104223]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING



PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MAY 2022

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report “**DIABETIC RETINOPATHY DETECTION USING IMAGE PROCESSING**” is the bonafide work of “**KATHIRVEL K S , NAGAKUMAR B , SALAI ARUN MANI J B**” who carried out the project work under my supervision.

SIGNATURE

**Dr.S. MURUGAVALLI , M.E., Ph.D.,
PROFESSOR
HEAD OF THE DEPARTMENT**

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING,
PANIMALAR ENGINEERING COLLEGE,
CHENNAI - 123.

SIGNATURE

**Dr.S.MURUGAVALLI, M.E., Ph.D.,
SURPERSVISOR
HEAD OF THE DEPARTMENT**

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING,
PANIMALAR ENGINEERING COLLEGE,
CHENNAI - 123.

Certified that the above mentioned students were examined in End Semester Project
Viva-Voce held on.....

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We **KATHIRVEL K S [211418104114]**, **NAGAKUMAR B [211418104165]**, **SALAI ARUN MANI J B [211418104223]** hereby declare that this project report titled “**DIABETIC RETINOPATHY DETECTION USING IMAGE PROCESSING**” under the guidance of **Dr. S.MURUGAVALLI** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

KATHIRVEL K S (211418104114)

NAGAKUMAR B (211418104165)

SALAI ARUN MANI J B (211418104223)

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.**, for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C. VIJAYA RAJESWARI, Dr.C. SAKTHI KUMAR,M.E.,Ph.D.**, and **Dr. SARANYASREE SAKTHI KUMAR B.E., M.B.A., Ph.D.**, for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K. MANI, M.E., Ph.D.**, who facilitated us in completing the project.

We thank the Head of the CSE Department , **Dr. S.MURUGAVALLI , M.E.,Ph.D.**, for the support extended throughout the project.

We would like to thank my Project Guide **Dr. S.MURUGAVALLI , M.E.,Ph.D** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

KATHIRVEL K S

NAGAKUMAR B

SALAI ARUN MANI J B

ABSTRACT

Diabetic retinopathy is a leading problem throughout the world and many people are losing their vision because of this disease. The disease can get severe if it is not treated properly at its early stages. The damage in the retinal blood vessel eventually blocks the light that passes through the optical nerves which makes the patient with Diabetic Retinopathy blind. Therefore, in our research we wanted to find out a way to overcome this problem and thus using the help of **Convolutional Neural Network** (ConvNet), we were able to detect multiple stages of severity for Diabetic Retinopathy. There are other processes present to detect Diabetic Retinopathy and one such process is manual screening, but this requires a skilled ophthalmologist and takes up a huge amount of time. Thus our automatic diabetic retinopathy detection technique can be used to **replace** such **manual processes** and the ophthalmologist can spend more time taking proper care of the patient or at least decrease the severity of this disease. we proposing Deep Learning classification technique using CNN trained model [resnet152] to classify severity levels of DR ranging from 0 (NO DR) to 4 (Proliferative DR).Deep learning looks promising because already various types of image classification tasks has been performed by various CNN's so, we can rely on DL pretrained models or we can modify some layers if we wish to. A GUI based system has been made using Tkinter and used heidiSQL to maintain and store a list of predictions with their username , password and predict value.

TABLE OF CONTENTS

S.NO	TITLE	PAGE NUMBER
	Abstract	i
	List of Figures	iv
	List of Abbreviation	vi
1.	Introduction	1
1.1	Introduction	2
1.1.1	Diabetic Retinopathy	2
1.1.2	Transfer learning	7
1.2	Motivation for problem	11
1.3	Problem Statement	11
2.	Literature Survey	13
3.	System Design	19
3.1	System configuration	20
3.2	Existing System	20
3.3	Proposed System	24
3.4	Use Case Diagram	29
3.5	DFD Diagram	30
4.	System Implementation	32

TABLE OF CONTENTS

S.NO	TITLE	PAGE NUMBER
4.1	Algorithm	33
4.2	Data Collection And Preparation	37
4.3	Image Processing	38
4.4	Modelling	42
4.5	Evaluation	44
4.5.1	Performance measures	45
4.6	Module Description	45
5.	Testing	47
5.1	Test Case	48
6.	Conclusion and Future work	53
6.1	Conclusion	54
6.2	Future work	54
7.	Appendix 1	55
8.	Appendix 2	64
9.	References	69

List of Figures

Figure Number	Figure Name
1.1.2.1	Differences between Traditional Machine learning and Transfer learning
1.1.2.2	Graph Representation for Transfer learning and without Transfer learning
1.3	Image Labelling
3.2.1	Structure of Neural Network for Image Recognition
3.2.2	Confusion Matrix for the Classification of the Network
3.3.1	System Architecture
3.4.1	Use Case Diagram
3.5.1	DFD level 0 Diagram
3.5.2	DFD level 1 Diagram
4.1.1	Various Scaling Methods
4.1.2	Efficient net- b5 Architecture
4.2.1	Block Diagram
4.3.1	Flow Chart of Image Classification
4.3.2	Image of Diabetic Retinopathy
4.3.3	After Pre-processing the images
4.3.4	Original Input Image
4.3.5	After Cropping Images
4.6.1	Sign up Message
4.6.2	Result of the Image

List of Figures

Figure Number	Figure Name
5.1.1	Login Credentials Error
5.1.2	Login Message
5.1.3	Login Username and Password Error
5.1.4	welcome message
5.1.5	Successful Message To Login
5.1.6	Blank Message Error
5.1.7	Successful Message
5.1.8	If Existing User Tries to sign-up
8.1	UI of the System
8.2	LOG-IN page
8.3	Wrong Authentication
8.4	Successful Log-in
8.5	Prediction Report
8.6	Displaying Report
8.7	Database Updation

List of Abbreviation

Abbreviation	Expansion
NPDR	Non-proliferative diabetic retinopathy
PDR	proliferative diabetic retinopathy
DR	Diabetic Retinopathy
CNN	Convolutional Neural Network
DL	Deep Learning
ResNet	Residual Neural Network
DRD	Diabetic Retinopathy Detection
GUI	Graphical User Interface
DFD	Data Flow Diagram

CHAPTER - 1

INTRODUCTION

CHAPTER - 1

INTRODUCTION

1.1Introduction

1.1.1 Diabetic Retinopathy

Diabetic retinopathy is a condition that may occur in people who have diabetes. It causes progressive damage to the retina, the light-sensitive lining at the back of the eye. Diabetic retinopathy is a serious sight-threatening complication of diabetes. Diabetes interferes with the body's ability to use and store sugar (glucose).

The disease is characterized by too much sugar in the blood, which can cause damage throughout the body, including the eyes. Over time, diabetes damages small blood vessels throughout the body, including the retina. Diabetic retinopathy occurs when these tiny blood vessels leak blood and other fluids. This causes the retinal tissue to swell, resulting in cloudy or blurred vision. The condition usually affects both eyes. The longer a person has diabetes, the more likely they will develop diabetic retinopathy. If left untreated, diabetic retinopathy can cause blindness.

Symptoms of diabetic retinopathy include:

- Seeing spots or floaters.
- Blurred vision.
- Having a dark or empty spot in the center of your vision.
- Difficulty seeing well at night.

When people with diabetes experience long periods of high blood sugar, fluid can accumulate in the lens inside the eye that controls focusing. This changes the curvature

of the lens, leading to changes in vision. However, once blood sugar levels are controlled, usually the lens will return to its original shape and vision improves. Patients with diabetes who can better control their blood sugar levels will slow the onset and progression of diabetic retinopathy. According to a 2018 American Eye- Q Survey conducted by the AOA, nearly half of Americans didn't know whether diabetic eye diseases have visible symptoms (often which the early stages of diabetic retinopathy does not). The same survey found that more than one-third of Americans didn't know a comprehensive eye exam is the only way to determine if a person's diabetes will cause blindness, which is why the AOA recommends that everyone with diabetes have a comprehensive dilated eye examination at least once a year. Early detection and treatment can limit the potential for significant vision loss from diabetic retinopathy. Treatment of diabetic retinopathy varies depending on the extent of the disease. People with diabetic retinopathy may need laser surgery to seal leaking blood vessels or to discourage other blood vessels from leaking. Your Doctor of Optometry might need to inject medications into the eye to decrease inflammation or stop the formation of new blood vessels. People with advanced cases of diabetic retinopathy might need a surgical procedure to remove and replace the gel-like fluid in the back of the eye, called the vitreous. Surgery may also be needed to repair a retinal detachment. This is a separation of the light-receiving lining in the back of the eye. If you are diabetic, you can help prevent or slow the development of diabetic retinopathy by:

- Taking your prescribed medication
- Sticking to your diet
- Exercising regularly
- Controlling high blood pressure

What causes diabetic retinopathy?

Diabetic retinopathy results from the damage diabetes causes to the small blood vessels located in the retina. These damaged blood vessels can cause vision loss:

Fluid can leak into the macula, the area of the retina responsible for clear central vision. Although small, the macula is the part of the retina that allows us to see colors and fine detail. The fluid causes the macula to swell, resulting in blurred vision.

In an attempt to improve blood circulation in the retina, new blood vessels may form on its surface. These fragile, abnormal blood vessels can leak blood into the back of the eye and block vision.

Diabetic retinopathy is classified into two types:

Non-proliferative diabetic retinopathy (NPDR) is the early stage of the disease in which symptoms will be mild or nonexistent. In NPDR, the blood vessels in the retina are weakened. Tiny bulges in the blood vessels, called microaneurysms, may leak fluid into the retina. This leakage may lead to swelling of the macula.

Proliferative diabetic retinopathy (PDR) is the more advanced form of the disease. At this stage, circulation problems deprive the retina of oxygen. As a result, new, fragile blood vessels can begin to grow in the retina and into the vitreous, the gel-like fluid that fills the back of the eye. The new blood vessels may leak blood into the vitreous, clouding vision. Other complications of PDR include detachment of the retina due to scar tissue formation and the development of glaucoma. Glaucoma is an eye disease in which there is progressive damage to the optic nerve. In PDR, new blood vessels grow into the area of the eye that drains fluid from the eye. This greatly raises the eye pressure, which damages the optic nerve. If left untreated, PDR can cause severe vision loss and even blindness. Risk factors for diabetic retinopathy include:

Diabetes: People with type 1 or type 2 diabetes are at risk for developing diabetic retinopathy. The longer a person has diabetes, the more likely he or she is to develop diabetic retinopathy, particularly if the diabetes is poorly controlled.

Race: Hispanics and African Americans are at greater risk for developing diabetic retinopathy.

Medical conditions: People with other medical conditions, such as high blood pressure and high cholesterol, are at greater risk.

Pregnancy: Pregnant women face a higher risk for developing diabetes and diabetic retinopathy. If a woman develops gestational diabetes, she has a higher risk of developing diabetes as she ages.

How is diabetic retinopathy diagnosed?

Diabetic retinopathy can be diagnosed through a comprehensive eye examination.

Testing with emphasis on evaluating the retina and macula may include:

Patient history to determine vision difficulties, presence of diabetes, and other general health concerns that may be affecting vision.

- Visual acuity measurements to determine how much central vision has been affected
- Refraction to determine if a new eyeglass prescription is needed.
- Evaluation of the ocular structures, including the evaluation of the retina through a dilated pupil
- Measurement of the pressure within the eyes.

Supplemental testing may include:

- Retinal photography or tomography to document the current status of the retina
- Fluorescent angiography to evaluate abnormal blood vessel growth

How is diabetic retinopathy treated?

Laser treatment (photocoagulation) is used to stop the leakage of blood and fluid into the retina. A laser beam of light can be used to create small burns in areas of the retina with abnormal blood vessels to try to seal the leaks.

Treatment for diabetic retinopathy depends on the stage of the disease. The goal of any treatment is to slow or stop the progression of the disease.

In the early stages of non-proliferative diabetic retinopathy, regular monitoring may be the only treatment. Following your doctor's advice for diet and exercise and controlling blood sugar levels can help control the progression of the disease.

Injections of medication in the eye are aimed at discouraging the formation of abnormal blood vessels and may help slowdown the damaging effects of diabetic retinopathy. If the disease advances, the abnormal blood vessels can leak blood and fluid into the retina, leading to macular edema. Laser treatment (photocoagulation) can stop this leakage. A laser beam of light creates small burns in areas of the retina with abnormal blood vessels to try to seal the leaks. Widespread blood vessel growth in the retina, which occurs in proliferative diabetic retinopathy, can be treated by creating a pattern of scattered laser burns across the retina. This causes abnormal blood vessels to shrink and disappear. With this procedure, some side vision may be lost in order to safeguard central vision.

1.1.2 Transfer learning

Humans have an inherent ability to transfer knowledge across tasks. What we acquire as knowledge while learning about one task, we utilize in the same way to solve related tasks. The more related the tasks, the easier it is for us to transfer, or cross-utilize our knowledge. Some simple examples would be,

- Know how to ride a motorbike -> Learn how to ride a car
- Know how to play classic piano -> Learn how to play jazz piano
- Know math and statistics -> Learn machine learning

The first thing to remember here is that, 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.

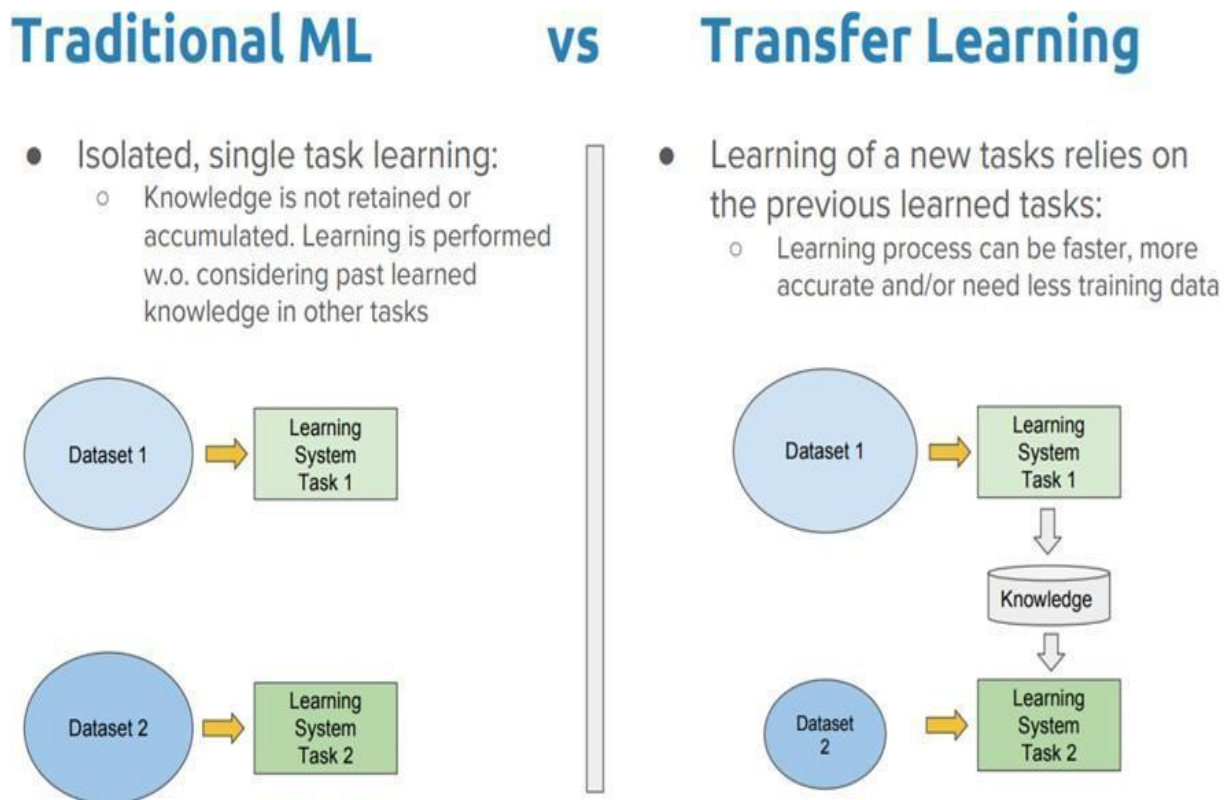


Fig 1.1.2.1 Differences between Traditional Machine Learning and Transfer Learning

Let's understand the preceding explanation with the help of an example. Let's assume our task is to identify objects in images within a restricted domain of a restaurant. Let's mark this task in its defined scope as ***T1***. Given the dataset for this task, we train a model and tune it to perform well (generalize) on unseen data points from the same domain (restaurant). Traditional supervised ML algorithms break down when we do not have sufficient training examples for the required tasks in given domains. Suppose, we now must detect objects from images in a park or a cafe (say, task ***T2***). Ideally, we should be able to apply the model trained for ***T1***, but in reality, we face performance degradation and models that do not generalize well. This happens for a variety of reasons, which we can liberally and collectively term as the model's bias towards training data and domain.

Transfer learning should enable us to utilize knowledge from previously learned tasks and apply them to newer, related ones. If we have significantly more data for task ***T1***, we may utilize its learning, and generalize this knowledge (features, weights) for task ***T2*** (which has significantly less data). In the case of problems in the computer vision domain, certain low-level features, such as edges, shapes, corners and intensity, can be shared across tasks, and thus enable knowledge transfer among tasks! Also, as we have depicted in the earlier figure, knowledge from an existing task acts as an additional input when learning a new target task.

How to Use Transfer Learning?

You can use transfer learning on your own predictive modeling problems. Two common approaches are as follows:

- ✓ **Develop Model Approach**
- ✓ **Pre-trained Model Approach**

a). Develop Model Approach

Select Source Task. You must select a related predictive modeling problem with an abundance of data where there is some relationship in the input data, output data, and/or concepts learned during the mapping from input to output data.

Develop Source Model. Next, you must develop a skillful model for this first task. The model must be better than a naive model to ensure that some feature learning has been performed.

Reuse Model. The model fit on the source task can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model, depending on the modeling technique used.

Tune Model. Optionally, the model may need to be adapted or refined on the input-output pair data available for the task of interest.

b). Pre-trained Model Approach

Select Source Model. A pre-trained source model is chosen from available models. Many research institutions release models on large and challenging datasets that may be included in the pool of candidate models from which to choose from.

Reuse Model. The model pre-trained model can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model, depending on the modeling technique used.

Tune Model. Optionally, the model may need to be adapted or refined on the input-

output pair data available for the task of interest.

When to Use Transfer Learning?

Transfer learning is an optimization, a shortcut to saving time or getting better performance. In general, it is not obvious that there will be a benefit to using transfer learning in the domain until after the model has been developed and evaluated.

Lisa Torrey and Jude Shavlik describe three possible benefits to look for when using transfer learning:

Higher start. The initial skill (before refining the model) on the source model is higher than it otherwise would be.

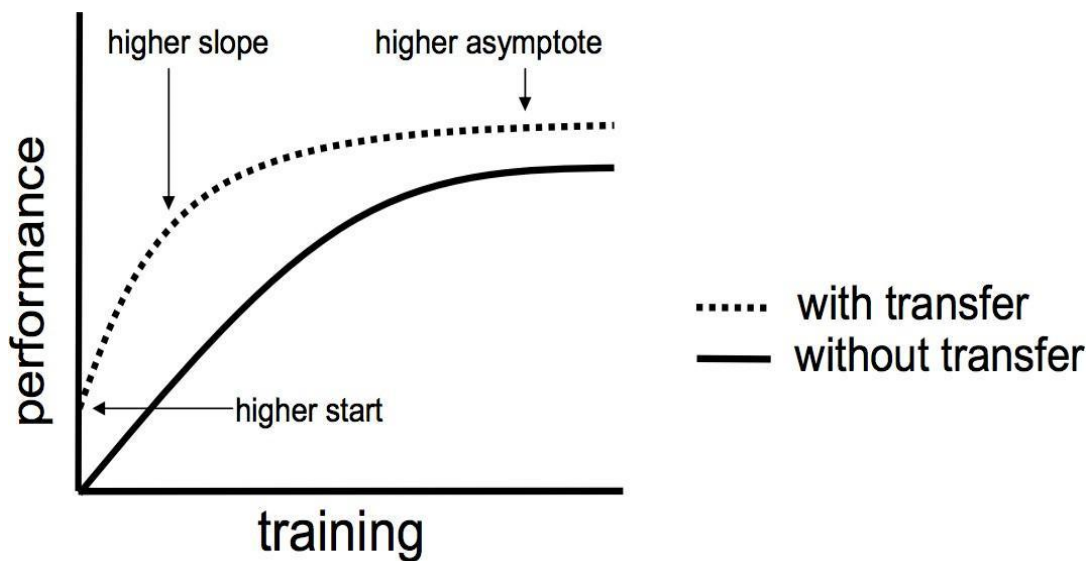


Fig 1.1.2.2 Graph Representation for Transfer learning and without Transfer learning

Higher slope. The rate of improvement of skill during training of the source model is steeper than it otherwise would be.

Higher asymptote. The converged skill of the trained model is better than it otherwise would be.

On some problems where you may not have very much data, transfer learning can enable you to develop skillful models that you simply could not develop in the absence of transfer learning. The choice of source data or source model is an open problem and may require domain expertise and/or intuition developed via experience.

1.2 Motivation for problem

According to WHO (world health organization) 412 million people were living with diabetes mellitus in 2014. In 2010, 33 percent of the people who are suffering from diabetes are detected with diabetic retinopathy and among them one third of the people were affected with loss of vision. It is expected that number of people detected with DR may triple in 2050 particularly in America.

Diagnosis of DR requires expert knowledge and we can detect using deep learning techniques but it requires huge dataset which it lacks in healthcare and takes so much time which we can eliminate in transfer learning.

1.3 Problem statement

Detecting the stage of Diabetic retinopathy using Fundus photograph images with help of transfer learned approach of EfficientNet-B5 model. The main objective of the project is to detect diabetic retinopathy to stop blindness before it is too late. We detect by classifying the images of retina of patient into five labels numbered from 0 to 4 where each label named as Normal, Mild DR, Moderate DR, Severe DR, Prolific DR respectively represents the Complication of the disease using Deep transfer learning and classification techniques. From these 5 stages one stage is observed as an output label for the given input fundus image.

Diabetic Retinopathy is a disease with an increasing prevalence and the main cause of blindness among working-age population. The risk of severe vision loss can be

significantly reduced by timely diagnosis and treatment. Systematic screening for DR has been identified as a cost-effective way to save health services resources. Automatic retinal image analysis is emerging as an important screening tool for early DR detection, which can reduce the workload associated to manual grading as well as save diagnosis costs and time. Many research efforts in the last years have been devoted to developing automated tools to help in the detection and evaluation of DR lesions.

We are interested in automating this perdition using deep learning models.

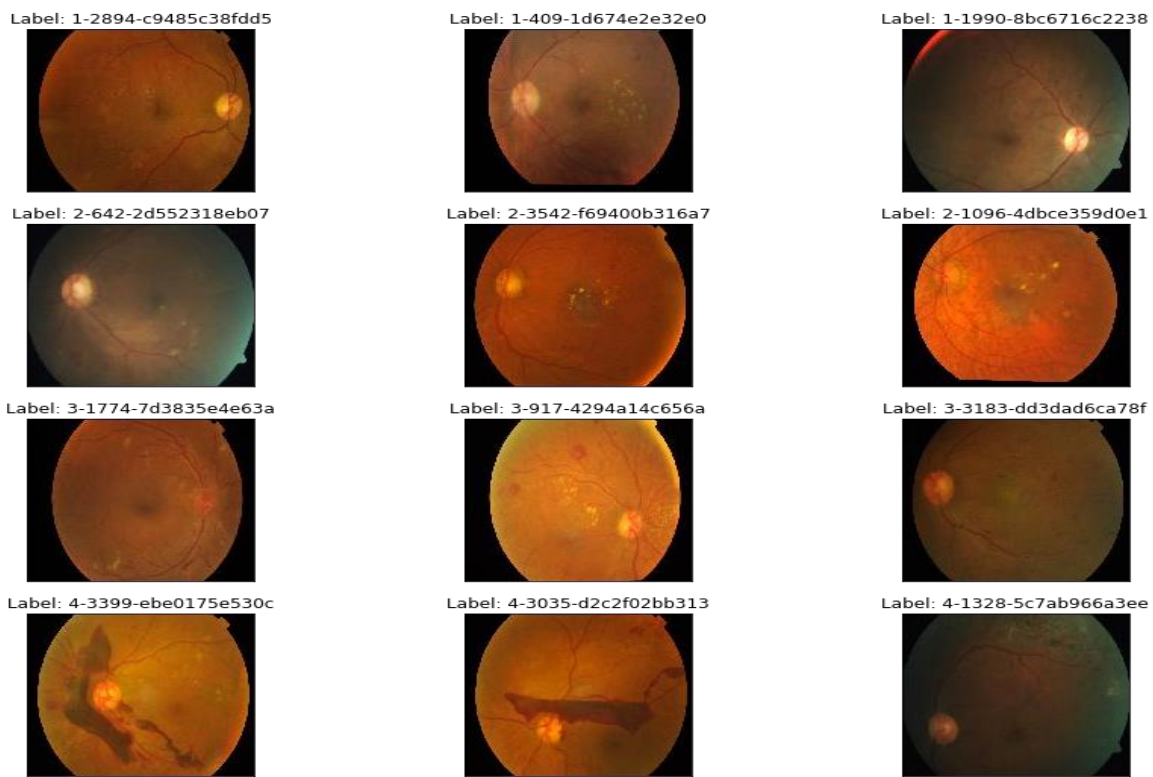


Fig.no:1.3.Image Labeling

Image labeling is the process of identifying and marking various details in an image. Image labeling is useful when automating the process of generating Meta data or making recommendations to users based on details in their images.

CHAPTER-2

LITERATURE SURVEY

CHAPTER - 2

LITERATURE SURVEY

TITLE - 1: Application of higher order spectra for the identification of diabetes retinopathy stages.

AUTHOR: Faust, O., Acharya, R., Ng, E.Y.K., Ng, K.H. and Suri, J.S.

YEAR: 2012

DESCRIPTION:

Feature extraction based classification and DL has been used to classify DR. In Acharya et al. higher order spectra technique was used to extract features from 300 fundus images and fed [1] to a support vector machine classifier; it classified the images into 5 classes with sensitivity of 82% and specificity of 88%. Different algorithms were developed to extract DR lesions such as blood vessels, exudates, and microaneurysms. Exudates have been extracted for DR grading. Support vector machine was used to classify the DIABETDB1 dataset into positive and negative classes using area and number of microaneurysms as features.

TITLE – 2: Rethinking the inception architecture for computer vision.

AUTHOR : Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z

YEAR : 2016

DESCRIPTION:

Feature extraction based classification methods need expert knowledge in order to detect the required features, [10] and they also involve a time consuming process of feature selection, identification and extraction. Furthermore, DL based systems such as CNNs have been seen to outperform feature extraction based methods.

TITLE – 3 : Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs.

AUTHOR : Gulshan, V., Peng, L., Coram, M., Stumpe, M.C., Wu, D., Narayanaswamy, A

YEAR : 2016

DESCRIPTION:

A convolutional neural network (CNN) was trained to classify a dataset of 128,175 fundus images into 2 classes, where the first class contains images with severity levels 0 and 1, and the second class contains levels 2, 3 and 4 . In an operating cut point picked for high sensitivity, [5] had a sensitivity of 97.5% and specificity of 93.4% on the EyePACS-1 dataset which consists of 9963 images; it scored a sensitivity of 96.1% and a specificity of 93.9% on the Messidor-2 dataset; and in an evaluation cut point selected for high specificity, the sensitivity and specificity were 90.3% and 98.1% on the EyePACS-1, while 87% and 98.5% was scored on the Messidor-2, consecutively.

TITLE – 4: Convolutional neural networks for diabetic retinopathy.

AUTHOR : Pratt, H., Coenen, F., Broadbent, D.M., Harding, S.P. and Zheng

YEAR : 2016

DESCRIPTION:

Using a training dataset of over 70,000 fundus images, Pratt et al. trained a CNN using stochastic gradient descent algorithm to classify DR into 5 classes, and it achieved 95% specificity, 75% accuracy and 30% sensitivity. A DL model was trained [9] from scratch on the MESSIDOR-2 dataset for the automatic detection of DR in , and a 96.8% sensitivity and 87% specificity were scored.

TITLE – 5 :Automated identification of diabetic retinopathy using Deep Learning.

AUTHOR : Gargeya, R. and Leng, T.

YEAR : 2017

DESCRIPTION:

A CNN was trained from scratch to classify fundus images from the Kaggle dataset into referable and non-referable classes, and it scored a sensitivity of 96.2% and a specificity of 66.6% . A dataset of 71896 fundus images was used to train a CNN DR classifier and resulted in a sensitivity of 90.5% and specificity of 91.6% . A DL model was designed and trained on a dataset of 75137 fundus images and resulted in a sensitivity and specificity scores of 94% and 98%, respectively [3].

TITLE-6:Comparative Study of Fine-Tuning of Pre-Trained Convolutional Neural Networks for Diabetic Retinopathy Screening

AUTHOR : Mohammadian, S., Karsaz, A. and Roshan, Y.M.

YEAR : 2017

DESCRIPTION:

In order to avoid the time and resource consumed during DL, Mohammadian et al. [8] fine-tuned the Inception-V3 and Exception pre-trained models to classify the Kaggle dataset into two classes. After using data augmentation to balance the dataset, reached at an accuracy score of 87.12% on the Inception-V3, and 74.49% on the Exception model.

TITLE – 7: Deep convolutional neural networks for diabetic retinopathy detection by image classification.

AUTHOR : Wan, S., Liang, Y. and Zhang, Y.

YEAR : 2018

DESCRIPTION:

Wan et al. [11] implemented transfer learning and hyper parameter tuning on the pre-trained models AlexNet, VggNet-s, VggNet-16, VggNet-19, GoogleNet and ResNet using the Kaggle dataset and compared their performances. The highest accuracy score was that of VggNet-s model, which reached 95.68% when training with hyper-parameter tuning . Transfer learning was used to work around the problem of insufficient training dataset in for retinal vessel segmentation. An Inception-V4 [36] model-based DR classification scored higher sensitivity when compared with human expert graders on a 25,326 retinal images of patients with diabetes from Thailand.

TITLE--8:Deep-learning-based automatic computer-aided diagnosis system for diabetic retinopathy.

AUTHOR : Mansour, R.F.

YEAR: 2018

DESCRIPTION:

Mansour put to use the Kaggle dataset to train a deep convolutional neural network using transfer learning for feature extraction when building a computer aided diagnosis for DR. In Dutta et al. 2000 fundus images were selected from the Kaggle dataset to train a shallow feed forward neural network, [7]deep neural network and VggNet16 model. On a test dataset of 300 images, the shallow neural network scored an accuracy of 41%, and the deep neural network scored 86.3% whiles the VggNet-16 scored 78.3% accuracy.

TITLE – 9:Diagnosis of Diabetic Retinopathy Using Deep Neural Networks.

AUTHOR : Gao, Z., Li, J., Guo, J., Chen, Y., Yi, Z. and Zhong

YEAR : 2018

DESCRIPTION:

A training dataset of size 4476 was collected and labeled into 4 classes depending on abnormalities and required treatment; they resized input images into 600x600 and cut every image into four 300x300 images, and fed these images into separate pre-trained Inception-V3 models, which they called the Inception. After it was seen that accuracy result of the Inception-4 surpassed the VggNet and ResNet models, it was deployed on a web-based DR classification system[2].

TITLE – 10: Multi-Cell Multi-Task Convolutional Neural Networks for Diabetic Retinopathy Grading.

AUTHOR : Zhou, K., Gu, Z., Liu, W., Luo, W., Cheng, J., Gao, S. and Liu.

YEAR : 2018

DESCRIPTION:

A multi cell, multi task convolutional neural network that uses a combination of cross entropy and mean square error was developed to classify images from the Kaggle dataset into 5 DR degrees . A binary tree based multi-class VggNet classifier was trained on the Kaggle dataset in Adly et al., and it scored an accuracy of 83.2%, sensitivity of 81.8% and specificity of 89.3% on a validation dataset of 6000 fundus images[12].

CHAPTER-3

SYSTEM DESIGN

CHAPTER-3

SYSTEM DESIGN

3.1 System configuration

Software configuration:

Operating System : Windows 10

Language : Python

Database : Heidi SQL

Libraries used : Matplotlib,Numpy,Pillow,Scikit-learn,torch,torch-vision,tkinter

Integrated Development Environment : Visual Studio

Training Platform : Google Colab

Hardware configuration:

Processor : Pentium Dual Core 2.00GHZ

Hard disk : 120 GB

RAM : 2GB (minimum)

Keyboard : 110 keys enhanced

3.2 Existing system

In the paper, they developed a network with CNN architecture and data augmentation which can identify the intricate features involved in the classification task such as micro-aneurysms, exudate and hemorrhages on the retina and consequently provide a diagnosis automatically and without user input. Network was trained using a high-end graphics processor unit (GPU) on the publicly available Kaggle dataset and demonstrate impressive results, particularly for a high-level classification task. On the data set of

80,000 images used our proposed CNN achieves a sensitivity of 95% and an accuracy of 75% on 5,000 validation images.

The structure of our neural network, shown in below was decided after studying the literature for other image recognition tasks.

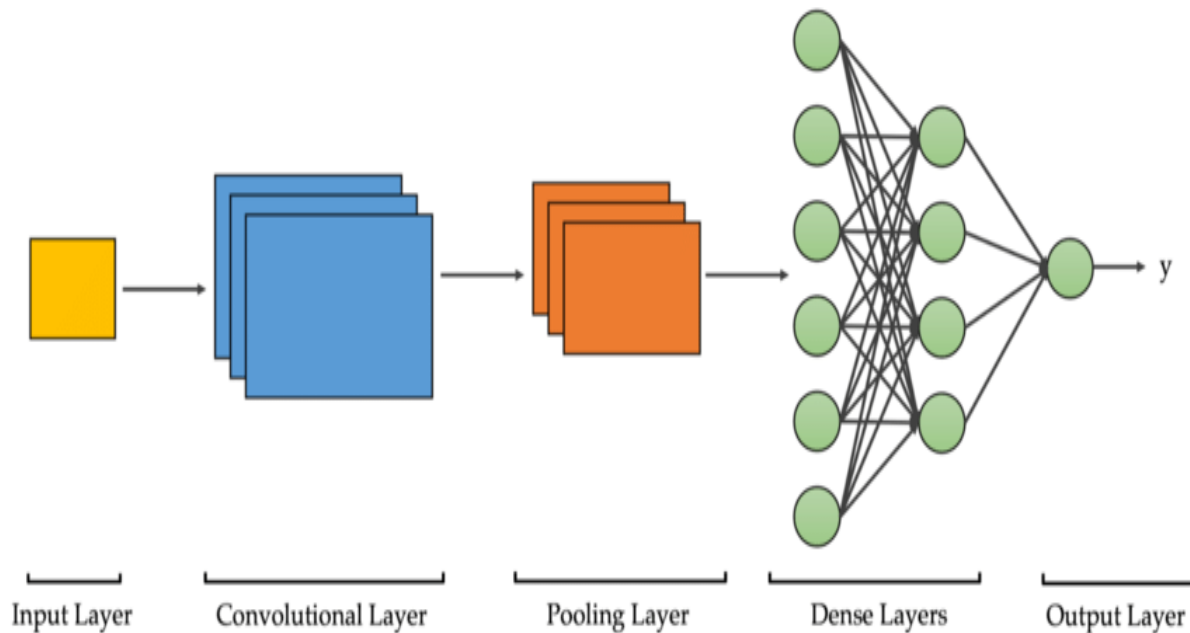


Fig 3.2.1 Structure of neural network for image recognition

Convolutional neural networks are comprised of two very simple elements, namely convolutional_layers and layers. Although simple, there are near-infinite ways to arrange these layers for a given computer vision problem.

Fortunately, there are both common patterns for configuring these layers and architectural innovations that you can use in order to develop very deep convolutional neural networks. Neural networks are broadly used, with applications for financial operations, enterprise planning, trading, business analytics, and product maintenance. Neural networks have also gained widespread adoption in business applications such as forecasting and marketing research solutions, fraud detection, and risk assessment. Also known as a deep learning network, a deep neural

network, at its most basic, is one that involves two or more processing layers – Deep Neural Network. A recurrent neural network is one adapted for analyzing time series data, event history, or temporal ordering – Recurrent Neural Network.

Increased convolution layers are perceived to allow the network to learn deeper features. The first layer learns edges of the deepest layer of the network, the last convolutional layer, should learn the features of classification of DR such as hard exudate. The network starts with convolution blocks with activation and then batch normalization after each convolution layer. As the number of feature maps increases it moves to one batch normalization per block. All max pooling is performed with kernel size 3x3 and 2x2 strides. After the final convolutional block, the network is flattened to one dimension. To avoid overfitting, it uses weighted class weights relative to the number of images in each class. Likewise, we perform dropout on dense layers, to reduce overfitting, until we reach the dense five node classification layer which uses a SoftMax activation function to predict our classification. The leaky rectified linear unit 13 activation function was used, applied with a value of 0.01, to stop over reliance on certain nodes in the network. Similarly, in the convolution layers, L2 regularization was used for weight and biases. The loss function used to optimize was the widely used categorical cross-entropy function.

The dataset used for testing was provided by the Kaggle coding website contains over 80,000 images, of approximately 6M pixels per image and scales of retinopathy. Resizing these images and running CNN on a high-end GPU, the NVIDIA K40c, meant we were able to train on the whole dataset. The NVIDIA K40c contains 2880 CUDA cores and comes with the NVIDIA CUDA Deep Neural Network library (cuDNN) for GPU learning. 5,000 images from the dataset were saved for validation purposes. Running the validation images on the network took 188 seconds. For this five class problem we define specificity as the number of patients correctly identified as not

having DR out of the true total amount not having DR and sensitivity as the number of patients correctly identified as having DR out of the true total amount with DR. We define accuracy as the number of patients with a correct classification. The final trained network achieved, 95% specificity, 75% accuracy and 30% sensitivity.

The classifications in the network were defined numerically as:

- 0 - No DR
- 1- Mild DR
- 2- Moderate DR
- 3- Severe DR
- 4- Proliferative DR

0	3456	0	145	1	34
1	344	0	27	0	1
2	543	0	179	5	40
3	40	0	63	10	15
4	28	0	23	3	43
	0	1	2	3	4
	Predicted Label				

Fig 3.2.2 confusion matrix for the classification of the network

The above table shows the confusion matrix of the result of the classification of five stages in Diabetic Retinopathy. As the number of feature maps increases it moves to one batch normalization per block. All max pooling is performed with kernel size 3x3 and

2x2 strides. After the final convolutional block, the network is flattened to one dimension. To avoid over fitting, it uses weighted class weights relative to the number of images in each class.

3.3 PROPOSED SYSTEM

SYSTEM ARCHITECTURE:

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task. Transfer learning is an optimization that allows rapid progress or improved performance when modeling the second task.



Fig 3.3.1 System architecture

How to use transfer learning?

Basically, the training of a CNN involves finding of the right values on each of the filters so that an input image when passed through the multiple layers, activates certain

neurons of the last layer so as to predict the correct class.

Though training a CNN from scratch is possible for small projects, most applications require the training of very large CNN's and this as you guessed, takes huge amounts of processed data and computational power. And both of these are not found so easily these days. In transfer learning, we take the pre-trained weights of an already trained model (one that has been trained on millions of images belonging to 1000's of classes, on several high-power GPU's for several days) and use these already learned features to predict new classes.

Working of Transfer Learning:

When we train a deep convolutional neural network on a dataset of images, during the training process, the images are passed through the network by applying several filters on the images at each layer. The values of the filter matrices are multiplied with the activations of the image at each layer. The activations coming out of the final layer are used to find out which class the image belongs to. When we train a deep network, our goal is to find the optimum values on each of these filter matrices so that when an image is propagated through the network, the output activations can be used to accurately find the class to which the image belongs. The process used to find these filter matrix values is gradient descent.

When we train a conv net on the ImageNet dataset and then take a look at what the filters on each layer of the conv net has learnt to recognize, or what each filter gets activated by, we are able to see something really interesting.

The filters on the first few layers of the conv net learn to recognize colors and certain horizontal and vertical lines. The next few layers slowly learn to recognize trivial shapes using the lines and colors learnt in the previous layers. Then the next layers learn to recognize textures, then parts of objects like legs, eyes, nose etc. Finally, the filters in

the last layers get activated by whole objects and give the output. By using a pre-trained network to do transfer learning, we are simply adding a few dense layers at the end of the pretrained network and learning what combination of these already learnt features help in recognizing the objects in our new datasets.

EfficientNet pre-trained model:

Convolutional neural networks are commonly developed at a fixed resource cost, and then scaled up in order to achieve better accuracy when more resources are made available. For example, Resnet can be scaled up from ResNet-18 to ResNet-200 by increasing the number of layers, and recently, GPipe achieved 84.3% ImageNet top-1 accuracy by scaling up a baseline CNN by a factor of four. The conventional practice for model scaling is to arbitrarily increase the CNN depth or width, or to use larger input image resolution for training and evaluation. While these methods do improve accuracy, they usually require tedious manual tuning, and still often yield suboptimal performance. We found a more principled method to scale up a CNN to obtain better accuracy and efficiency.

In the ICML 2019 paper, “EfficientNet: A rethinking model for scaling up CNN” a novel scaling method that uses a simple yet highly effective compound coefficient to scale up CNNs in a more structured manner. Unlike conventional approaches that arbitrarily scale network dimensions, such as width, depth and resolution, our method uniformly scales each dimension with a fixed set of scaling coefficients. Powered by this novel scaling method and recent progress on **AutoML** was developed a family of models, called EfficientNets, which super pass state-of-the- art accuracy with up to 10x better efficiency (smaller and faster).

AutoML

CNNs have been widely used in image classification, face recognition, object detection and many other domains. Unfortunately, designing CNNs for mobile devices is challenging because mobile models need to be small and fast, yet still accurate. Although significant effort has been made to design and improve mobile models, such as MobileNet and MobileNetV2, manually creating efficient models remains challenging when there are so many possibilities to consider. Inspired by recent progress in AutoML neural architecture search, we wondered if the design of mobile CNN models could also benefit from an AutoML approach.

In “MnasNet: Platform-Aware Neural Architecture Search for Mobile”, we explore an automated neural architecture search approach for designing mobile models using reinforcement learning. To deal with mobile speed constraints, it explicitly incorporates the speed information into the main reward function of the search algorithm, so that the search can identify a model that achieves a good trade-off between accuracy and speed. In doing so, MnasNet is able to find models that run 1.5x faster than state-of-the-art hand-crafted MobileNetV2 and 2.4x faster than NASNet, while reaching the same ImageNet top 1 accuracy.

Unlike in previous architecture search approaches, where model speed is considered via another proxy our approach directly measures model speed by executing the model on a particular platform, e.g., Pixel phones which were used in this research study. In this way, we can directly measure what is achievable in real- world practice, given that each type of mobile device has its own software and hardware idiosyncrasies and may require different architectures for the best trade-offs between accuracy and speed.

The overall flow of our approach consists mainly of three components: a RNN-based controller for learning and sampling model architectures, a trainer that builds and trains models to obtain the accuracy, and an inference engine for measuring the model speed

on real mobile phones using TensorFlow Lite. We formulate a multi-objective optimization problem that aims to achieve both high accuracy and high speed and utilize a reinforcement learning algorithm with a customized reward function to find Pareto optimal solutions (e.g., models that have the highest accuracy without worsening speed).

In order to strike the right balance between search flexibility and search space size, we propose a novel factorized hierarchical search space, which factorizes a convolutional neural network into a sequence of blocks, and then uses a hierarchical search space to determine the layer architecture for each block. In this way, our approach allows different layers to use different operations and connections; Meanwhile, we force all layers in each block to share the same structure, thus significantly reducing the search space size by orders of magnitude compared to a flat per-layer search space. Our MnasNet network, sampled from the novel factorized hierarchical search space illustrating the layer diversity throughout the network architecture.

We tested the effectiveness of our approach on ImageNet classification and COCO object detection. Our experiments achieve a new state-of-the-art accuracy under typical mobile speed constraints. In particular, the figure below shows the results on ImageNet. With the same accuracy, our MnasNet model runs 1.5x faster than the hand-crafted state-of-the-art MobileNetV2, and 2.4x faster than NASNet, which also used architecture search. After applying the squeeze-and-excitation optimization, our MnasNet and SE models achieve ResNet-50 level top-1 accuracy at 76.1%, with 19x fewer parameters and 10x fewer multiply-adds operations. On COCO object detection, our model family achieves both higher accuracy and higher speeds than Mobile Net and achieves comparable accuracy to the SSD300 model with 35x less computation cost.

3.4 Use case Diagram :

In UML, use-case diagrams model the behavior of a system and help to capture the requirements of the system. Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally. A use case diagram is a standard diagram that shows all interactions between the user, dataset, and algorithm used. It is developed in the early stages of the process.

Elements of a Use Case:

Depending on how in depth and complex you want or need to get, use cases describe a combination of the following elements:

- **Actor** – anyone or anything that performs a behavior (who is using the system)
- **Stakeholder** – someone or something with vested interests in the behavior of the system under discussion (SUD)
- **Primary Actor** – stakeholder who initiates an interaction with the system to achieve a goal

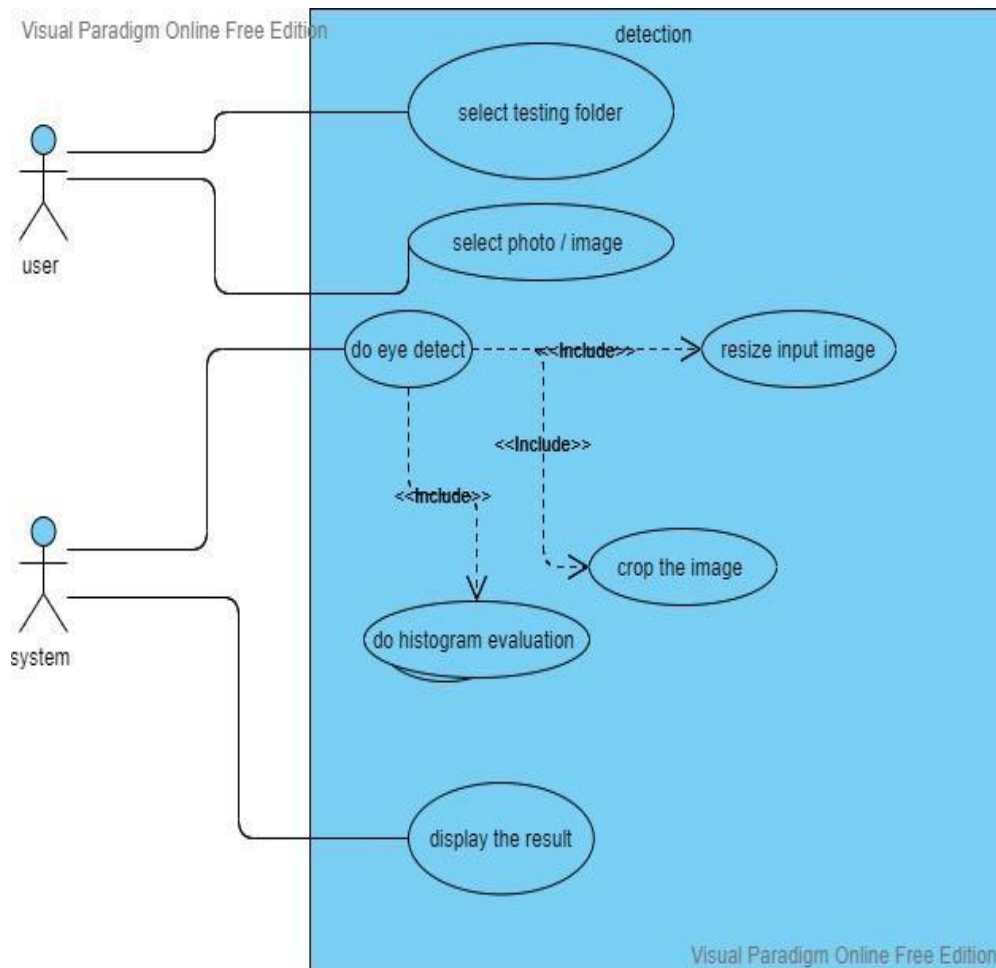


Fig 3.4.1 Use Case diagram

3.5 DFD DIAGRAM:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing

DFD LEVEL 0:

DFD Level 0 is also called a Context Diagram. Its a basic overview of the whole system or process being analyzed or modeled.

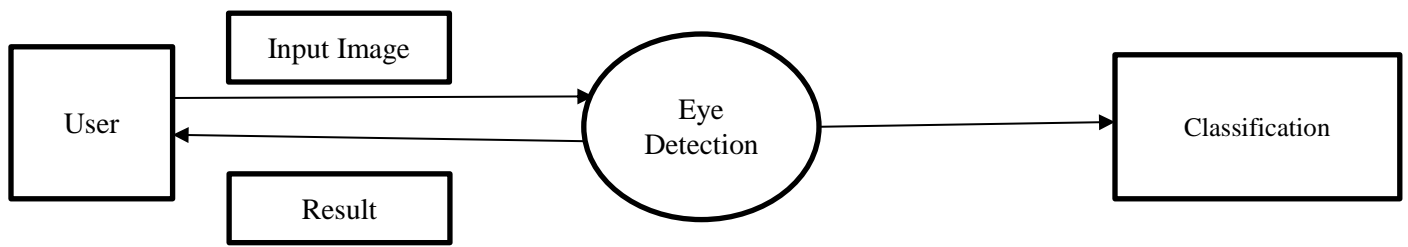


Fig 3.5.1 DFD level 0 diagram

DFD LEVEL 1:

A more detailed breakout of pieces of the Context Level Diagram.

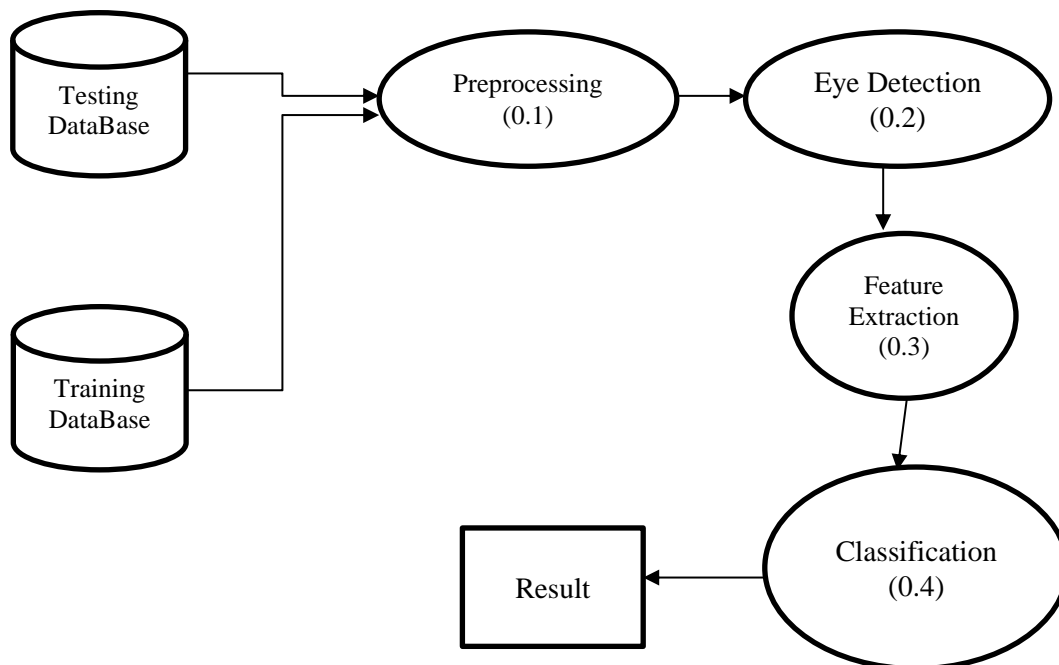


Fig 3.5.2 DFD level 1 diagram

CHAPTER - 4

SYSTEM IMPLEMENTATION

CHAPTER – 4

SYSTEM IMPLEMENTATION

4.1. ALGORITHM.

Compound Model Scaling: A Better Way to Scale Up CNNs

In order to understand the effect of scaling the network, we systematically studied the impact of scaling different dimensions of the model. While scaling individual dimensions improves model performance, we observed that balancing all dimensions of the network—width, depth, and image resolution—against the available resources would best improve overall performance. The first step in the compound scaling method is to perform a grid search to find the relationship between different scaling dimensions of the baseline network under a fixed resource constraint (e.g., 2x more flops). This determines the appropriate scaling coefficient for each of the dimensions mentioned above. We then apply those coefficients to scale up the baseline network to the desired target model size or computational budget.

Input Layers: It's the layer in which we give input to our model. The number of neurons in this layer is equal to the total number of features in our data (number of pixels in the case of an image).

Hidden Layer: The input from the Input layer is then feed into the hidden layer. There can be many hidden layers depending upon our model and data size. Each hidden layer can have different numbers of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of output of the previous layer with learnable weights of that layer and then by the addition of learnable biases followed by activation function which makes the network nonlinear.

Output Layer: The output from the hidden layer is then fed into a logistic function like sigmoid or softmax which converts the output of each class into the probability score of each class.

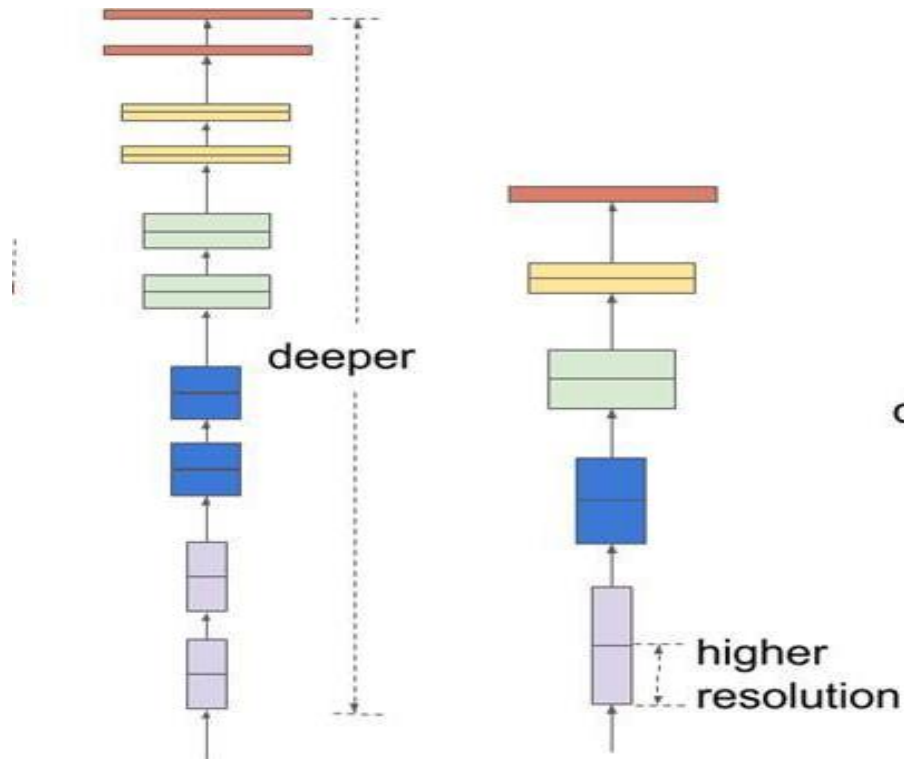


Fig 4.1.1 Various Scaling Methods

From the above diagram we can see that compound scaling which we used in EfficientNet-b5 uniformly scales network width, network depth, resolution scaling at a time. In compound scaling, we use compound coefficient to uniformly scale all dimensions. compound coefficient obtained by applying grid search to find the relationship between different scaling dimensions. Also, the baseline network plays an important role in EfficientNet architecture.

To find the best baseline network, we used neural architectural search using AutoML Mnas network. The resulting architecture is similar to MobileNetV2[18] and then we get EfficientNets by scaling up the resulting architecture.

This compound scaling method consistently improves model accuracy and efficiency for scaling up existing models such as MobileNet (+1.4% ImageNet accuracy), and ResNet (+0.7%), compared to conventional scaling methods.

EfficientNet Architecture:

The effectiveness of model scaling also relies heavily on the baseline network. So, to further improve performance, we have also developed a new baseline network by performing a neural architecture search using the AutoML MNAS framework, which optimizes both accuracy and efficiency (FLOPS). The resulting architecture uses mobile inverted bottleneck convolution (MBConv), similar to MobileNetV2 and MnasNet, but is slightly larger due to an increased FLOP budget. We then scale up the baseline network to obtain a family of models, called *EfficientNets*. The architecture for our baseline network EfficientNet-B0 is simple and clean, making it easier to scale and generalize. Below is the architecture of EfficientNet-b0 architecture. EfficientNet is a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth/width/resolution using a compound coefficient. Unlike conventional practice that arbitrary scales these factors, the EfficientNet scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients. For example, if we want to use α times more computational resources, then we can simply increase the network depth by d^α , width by w^α , and image size by r^α , where d, w, r are constant coefficients determined by a small grid search on the original small model. EfficientNet uses a compound coefficient μ to uniformly scales network width, depth, and resolution in a principled way. The compound scaling method is justified by the intuition that if the input image is bigger, then the network needs more layers to increase the receptive field and more channels to capture more fine-grained patterns on the bigger image.

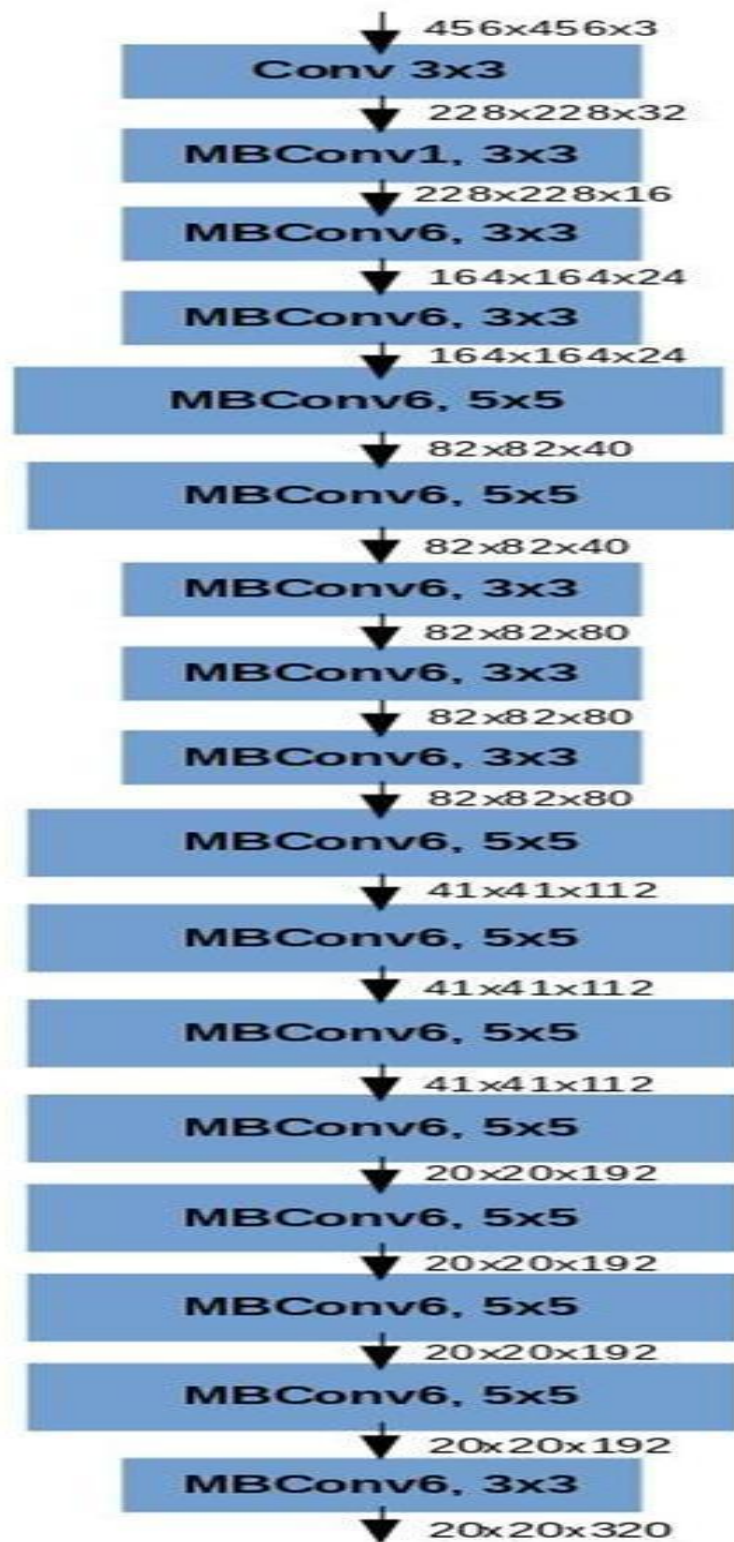


Fig 4.1.2 EfficientNet-B5 Architecture

EfficientNet Performance

We have compared our EfficientNets with other existing CNNs on ImageNet. In general, the EfficientNet models achieve both higher accuracy and better efficiency over existing CNNs, reducing parameter size and FLOPS by an order of magnitude. For example, in the high-accuracy regime, our EfficientNet-B7 reaches state-of-the-art 84.4% top-1 / 97.1% top-5 accuracy on ImageNet, while being 8.4x smaller and 6.1x faster on CPU inference than the previous Gpipe. Compared with the widely used ResNet-50, our EfficientNet-B4 uses similar FLOPS, while improving the top-1 accuracy from 76.3% of ResNet-50 to 82.6% (+6.3%).

EfficientNet-B0 is the baseline network developed by AutoML MNAS, while Efficient-B1 to B7 are obtained by scaling up the baseline network. In particular, our EfficientNet-B7 achieves state-of-the-art 84.4% top-1 / 96.2% top-5 accuracy, while being 8.4x smaller than the best existing CNN.

Though EfficientNets perform well on ImageNet, to be most useful, they should also transfer to other datasets. To evaluate this, we tested EfficientNets on eight widely used transfer learning datasets. EfficientNets achieved state-of-the-art accuracy in 5 out of the 8 datasets, such as CIFAR-100 (91.7%) and Flowers (98.8%), with an order of magnitude fewer parameters (up to 21x parameter reduction), suggesting that our EfficientNets also transfer well. EfficientNets potentially serve as a new foundation for future computer vision tasks. We have open-sourced all EfficientNet models, which can benefit the larger machine learning community.

4.2 DATA COLLECTION AND PREPARATION

In Pre-processing, there are 4 sub-modules: Data preparation, Exploratory Data Analysis, Metric, Pre-processing.

Data preparation

we collect all the fundus images from APTOS (Asia Pacific Tele-Ophthalmology Society) dataset. In this dataset the fundus images are labelled as 0,1,2,3 and 4 for Normal, Mild DR, Moderate DR, Severe DR, Prolific DR respectively. This dataset provides 4657 fundus images in total. Among these 3662 (stored in Train.csv with image ID and its diagnosis label) and were used for model training and remaining 995 (stored in Test.csv with image ID and its diagnosis label) are used for model testing.

Block Diagram:

We think one should at least examine the label distribution, the images before Pre-processing and the images after Pre-processing

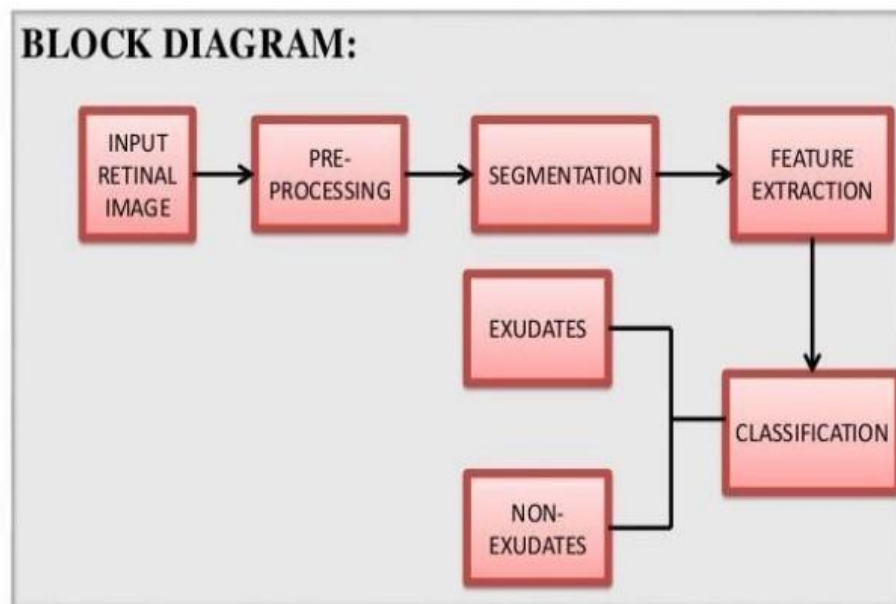


Fig 4.2.1 Block Diagram

4.3 IMAGE PREPROCESSING

One intuitive way to improve the performance of our model is to simply improve the quality of input images. In this kernel, we will share two ideas which we hope may be useful to some of you:

Reducing lighting-condition effects: images come with many different lighting conditions, some images are very dark and difficult to visualize. We can try to convert the image to gray scale, and visualize better.

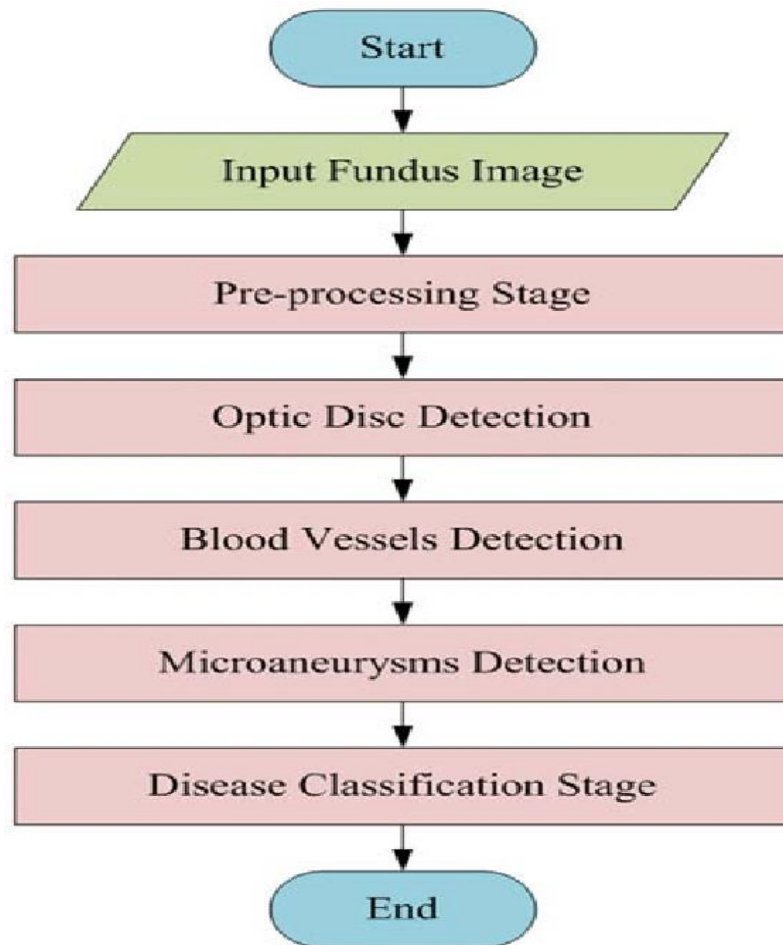


Fig.no.4.3.1 Flow chart of Image Classification

PHASES OF IMAGE PROCESSING:

1.Acquisition – It could be as simple as being given an image which is in digital form.

The main work involves:

- a) Scaling
- b) Color conversion(RGB to Gray or vice-versa)

2.Image enhancement– It is amongst the simplest and most appealing in areas of Image

Processing it is also used to extract some hidden details from an image and is subjective.

3.Segmentation procedure-It includes partitioning an image into its constituent parts or objects. Autonomous segmentation is the most difficult task in Image Processing.

Image to explain diabetic retinopathy:

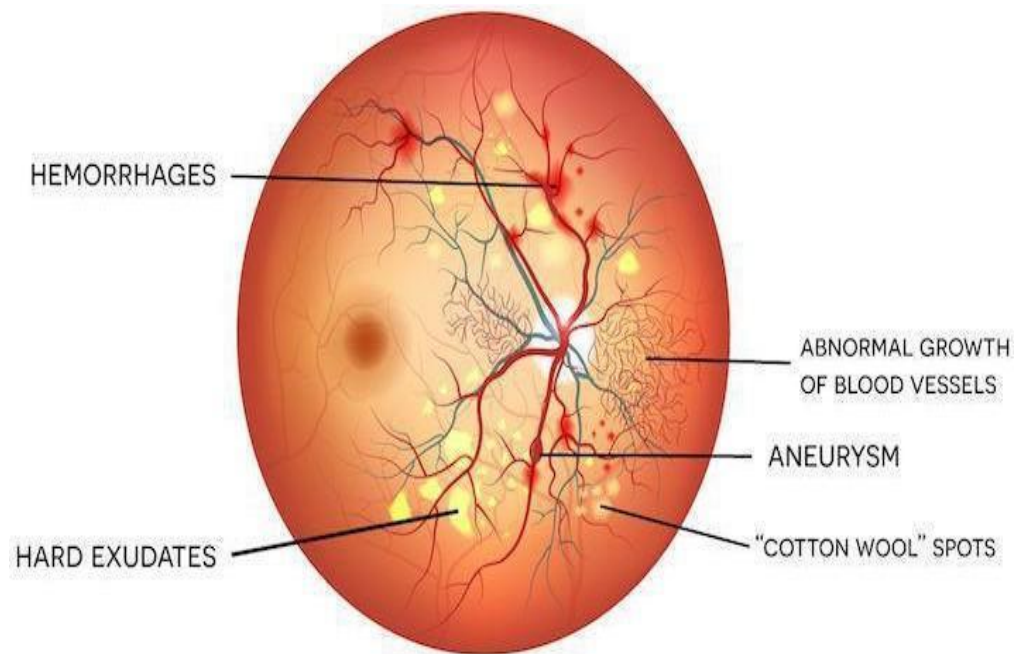


Fig 4.3.2 Image of Diabetic Retinopathy

First, let have a glance of original inputs. Each row depicts each severity level. We can see two problems which make the severity difficult to spot on. First, some images are very dark [pic (0,2) and pic (4,4)] and sometimes different color illumination is confusing [pic (3,3)]. Second, we can get the uninformative dark areas for some pictures [pic (0,1), pic (0,3)]. This is important when we reduce the picture size, as informative areas become too small. So, it is intuitive to crop the uninformative areas out in the second case.

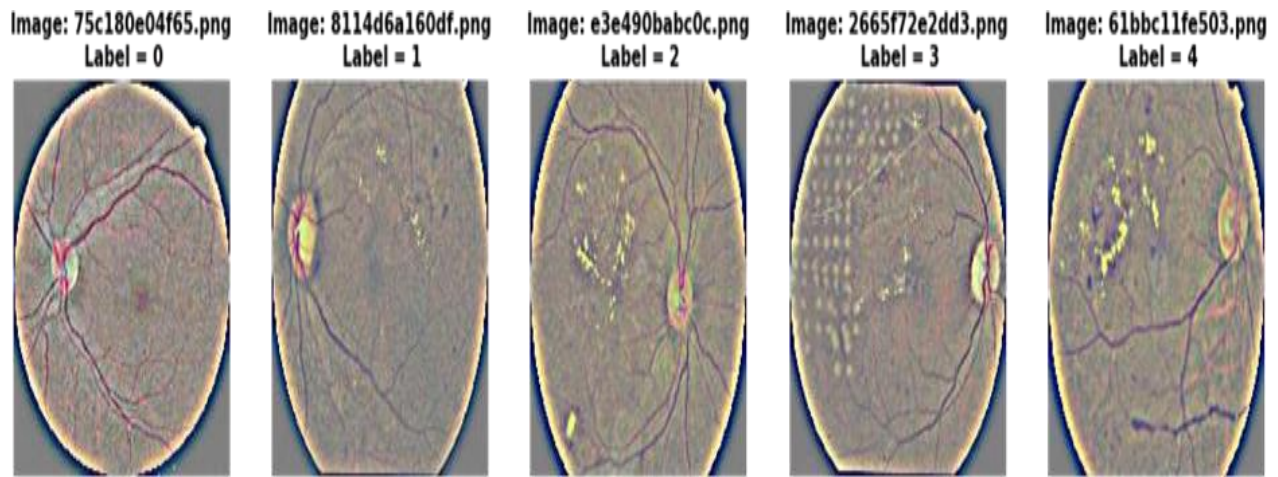


Fig 4.3.3. After Pre-processing the image

Careful management of your diabetes is the best way to prevent vision loss. If you have diabetes, see your eye doctor for a yearly eye exam with dilation — even if your vision seems fine.

Developing diabetes when pregnant (gestational diabetes) or having diabetes before becoming pregnant can increase your risk of diabetic retinopathy. If you're pregnant, your eye doctor might recommend additional eye exams throughout your pregnancy.

Contact your eye doctor right away if your vision changes suddenly or becomes blurry, spotty or hazy.

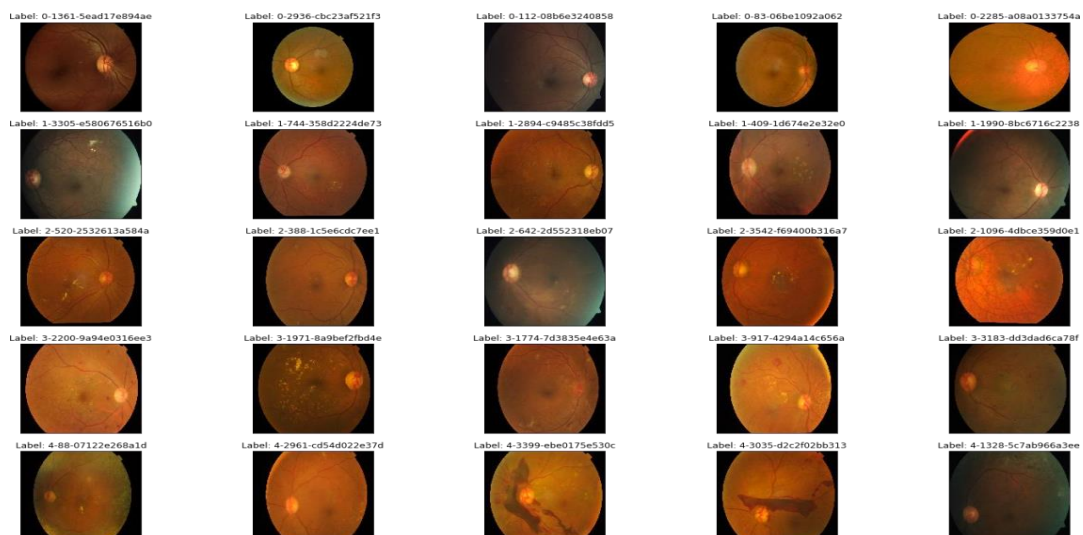


Fig 4.3.4. Original input image

Avoid this color distraction; we convert this original image (BGR format) to RGB format.

So, we can crop the images for uninformative area. After cropping the images, they are converted into gray scale to resize the images and detect in which stage it is. After preprocessing we have managed to enhance the distinctive features in the images. This will increase performance when we train our Efficient Net model. Jupyter notebook is used for preprocessing. we propose the multistage approach to transfer learning, which makes use of similar datasets with different labeling. The presented method can be used as a screening method for early detection of diabetic retinopathy with sensitivity and specificity of 0.99 and is ranked 54 of 2943 competing methods (quadratic weighted kappa score of 0.925466) on APTOS 2019 Blindness Detection Dataset (13000 images).

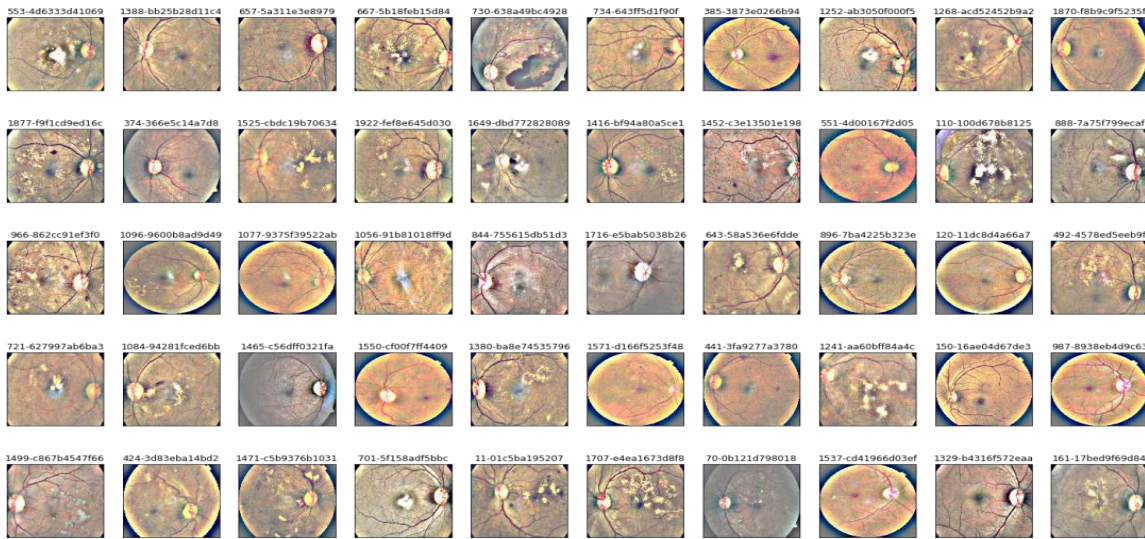


Fig 4.3.5. After cropping the image

4.4. MODELING

Metric (Quadratic Weighted Kappa)

The Quadratic Weighted Kappa is used to calculate the similarity between the actuals

and predictions. A perfect score of 1.0 is granted when both the predictions and actuals are the same. Whereas, the least possible score is -1 which is given when the predictions are furthest away from actuals. In our case, consider all actuals were 0's and all predictions were 4's. The aim is to get as close to 1 as possible. Generally, a score of 0.6+ is considered to be a really good score. This Metric is used to know when to stop the training of the images by the model. The training model be selected based on the best metric value.

This Weighted kappa is calculated as follows:

Step 1: First, an $N \times N$ matrix X is constructed, such that $X(i, j)$ corresponds to the actual ratings i and predicted ratings j . This $N \times N$ matrix is considered as Confusion matrix

Step 2: Construct a weighted matrix W which is calculated based on difference between the actual and predicted rating scores.

Step 3: Create two vectors, one for predictions and another for actuals, which tells us that how many values of each rating exists.

Step 4: Now, Construct Expected Matrix E which is the outer product of two Vectors Prediction vector and the actual vector calculated

Step 5: Normalize both matrices to have same sum. Since, it is easiest to get sum to be '1', we will simply divide each matrix by its sum to normalize the data.

Step 6: Now calculate the weighted kappa as per formulae by substituting the values.

Since we want to optimize the Quadratic Weighted Kappa score, we can formulate this challenge as a regression problem. In this way we are more flexible in our optimization and we can yield higher scores than solely optimizing for accuracy. We will optimize a pre-trained EfficientNetB5 with a few added layers. The metric that we try to optimize is the Mean Squared Error. This is the mean of squared differences between our predictions and labels, as showed in the formula below. By optimizing

this metric, we are also optimizing for Quadratic Weighted Kappa if we round the predictions afterwards. Since we are not provided with that much data (3662 images), we will augment the data to make the model more robust. We will rotate the data on any angle. Also, we will flip the data both horizontally and vertically. Lastly, we will divide the data by 128 for normalization.

We split the training dataset into 85% for training and 15% for validating the model. We resize the images into (image width, image height) of (456,456) so that it is suitable to process in the efficientnet-b5 model. In efficientnet-b5 model batch normalization is applied and batch normalization is unstable for small batch sizes since the non-uniformity of dataset does not change. To solve this problem, we applied the group normalization to each layer of efficientnet-b5 so that it normalizes the features by dividing the channels into groups so that it is used for processing.

4.5 EVALUATION

To evaluate our performance, we test model on validation data and predict values from the validation generator and round them off to the corresponding integer to get valid predictions. To detect the severity level of the diabetic retinopathy of the patient from the values of generator we set threshold value for each stage of the disease During training of the model. Based on the threshold values we get accuracy of the model. we can improve the model Accuracy by optimizing the model performance using optimizers like Radam optimizer we used in this model. We initially set initial values of coefficient values to (0.5,1.5,2.5,3.5). Later we minimize the quadratic weighted kappa Score with respect to coefficient values using nelder- mead method. We evaluate the performance of the model depending on validation accuracy and quadratic weighted kappa score. since quadratic weighted kappa score plays vital role in assessing performance of the model. It further enhances using the optimizers on the quadratic weighted kappa score with respect to coefficients of the threshold values. we get 0.8712

quadratic weighted kappa score and 83% accuracy on validation data. Now train the model till the model does not improve further apply it on test data we get severity of diabetic retinopathy using fundus image.

4.5.1. PERFORMANCE MEASURES:

Metric (Quadratic Weighted Kappa):

The Quadratic Weighted Kappa is used to calculate the similarity between the actuals and predictions. A perfect score of 1.0 is granted when both the predictions and actual values are the same. Whereas, the least possible score is -1 which is given when the predictions are furthest away from actuals. In our case, consider all actuals were 0's and all predictions were 4's. This would lead to a QWKP score of -1. The aim is to get as close to 1 as possible. Generally, a score of 0.6+ is considered to be a really good score. This Metric is used to know when to stop the training of the images by the model. The training model be selected based on the best metric value.

Validation Accuracy:

The percentage of which model which was trained on the training dataset correctly predicts on validation dataset. It is used to evaluate the performance of the model before we deploy the model. feature extractor part of efficientnet-b5 model create the sequential object from keras and pass the feature extractor part of efficientnet-b5 model. Then create the architecture of classifier part. In the classifier part we used the Relu and linear activation function since we get single output as we consider it as regression problem.

4.6. MODULE DESCRIPTION:

Sign-up:

User sign-up page to get the report of the eye. User should sign-up to the system. If the user not having the credentials. If the user having the right credentials then no problem they can login to the system.

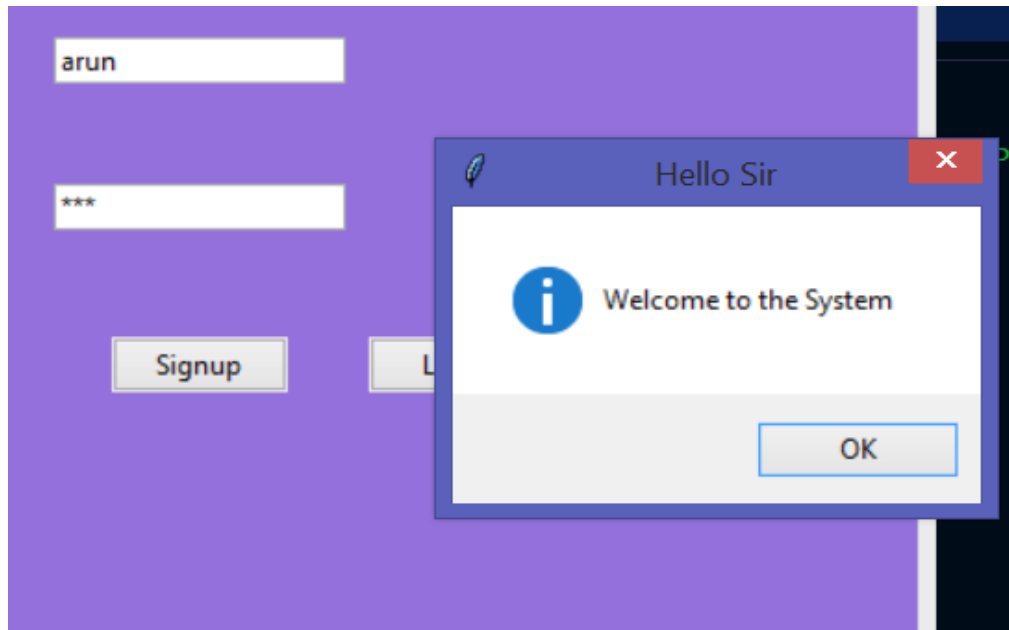


Fig 4.6.1. Sign-up Module

Uploading:

After login upload the image.

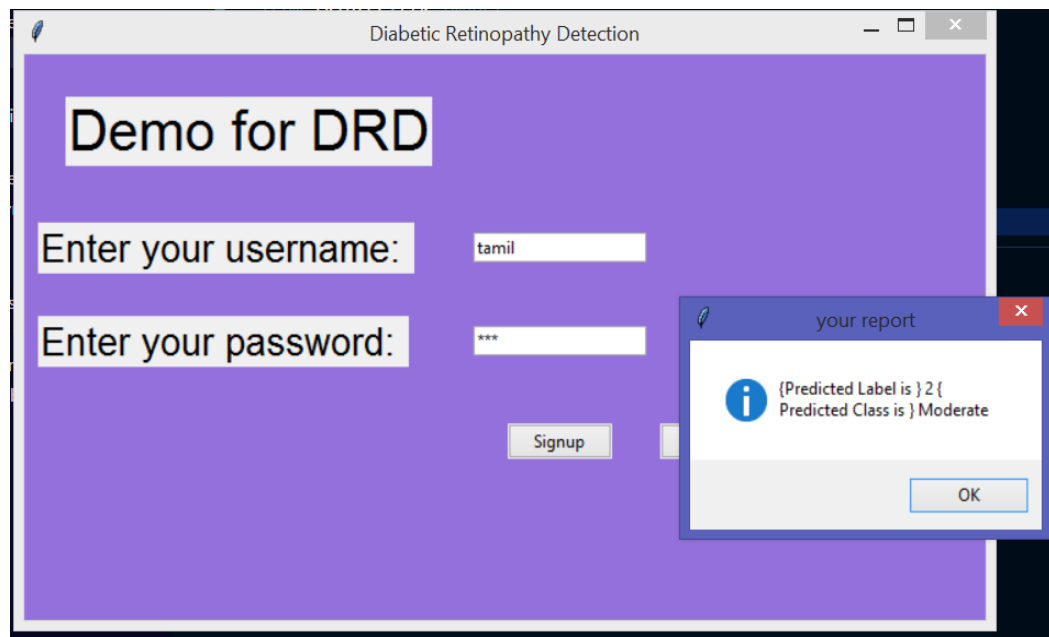


Fig 4.6.2. Uploading Module

CHAPTER-5

TESTING

CHAPTER 5

TESTING

5.1 TEST CASES

The test case is defined as a group of conditions under which a tester determines whether a software application is working as per the customer's requirements or not. Test case designing includes preconditions, case name, input conditions, and expected result. A test case is a first level action and derived from test scenarios

- Test case gives detailed information about testing strategy, testing process, preconditions, and expected output.
- These are executed during the testing process to check whether the software application is performing the task for that it was developed or not.

Test case 1:

Description:

If user click on upload image without the login credentials

Example Inputs:

Username= 'blank' Password='blank'

Error:

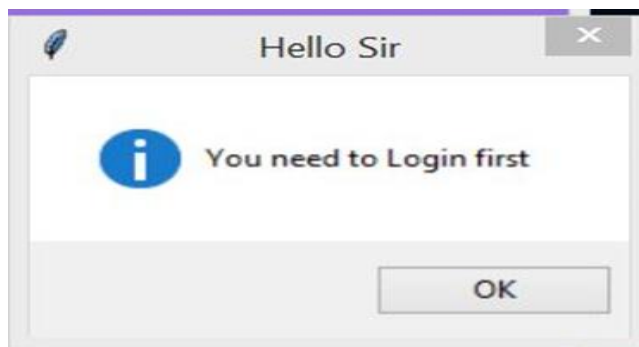


Fig 5.1.1 Login Credentials Error

Solution:

If the user gives the correct username and password he will be able to upload the image

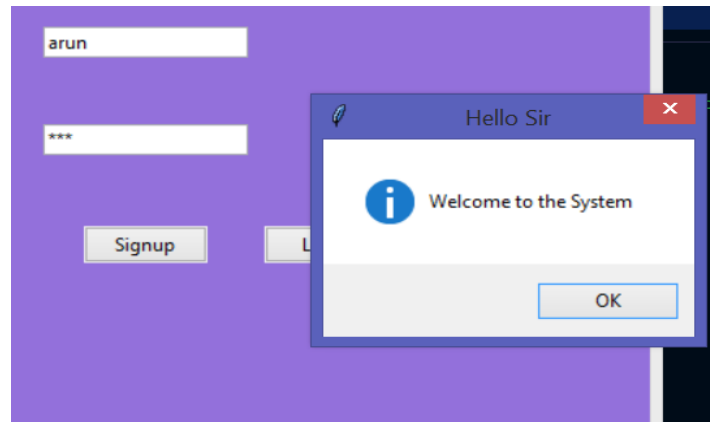


Fig 5.1.2 Login Message

Test case 2:**Description:**

If username is correct and password is wrong or username is wrong and password is correct

Example Inputs:

Correct username: bala Correct Password: 12345

Entered username= abcd Entered password=1234

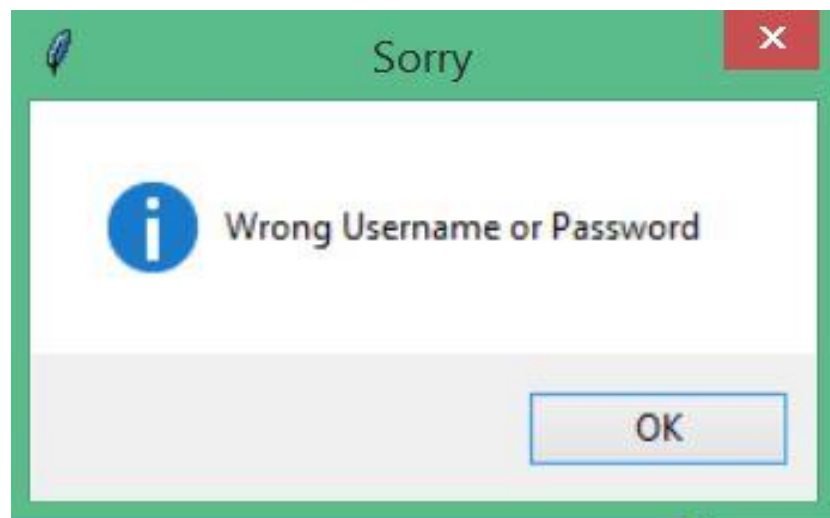
Error:

Fig: 5.1.3 Login Username and Password Error

Solution:

After entering the correct username and password the user able to login to the system.



Fig: 5.1.4 welcome message

Test case 3:**Description:**

If user enter correct username and password and clicking on sign up

Output:

Fig; 5.1.5 Successful Message To Login

Test case 4:**Description:**

If user does not enter any username or password

Example inputs:

Username=blank Password=blank

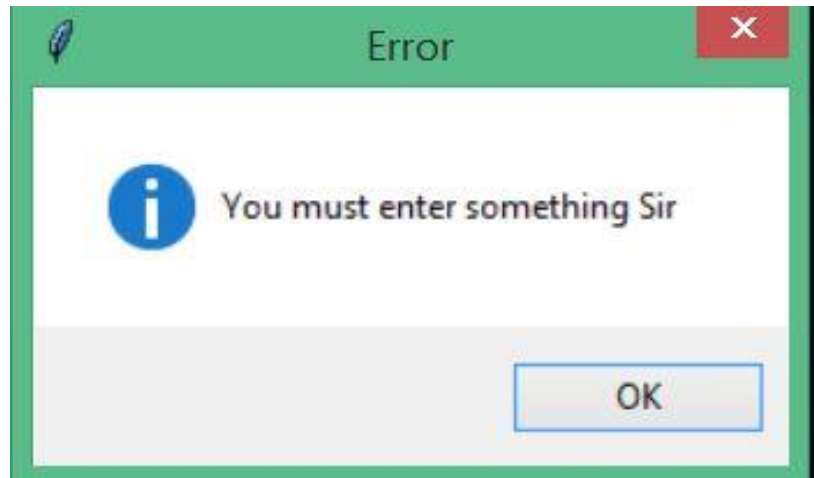
Error:

Fig : 5.1.6 Blank Message Error

Solution:

By entering the correct username and password the user can login into the system

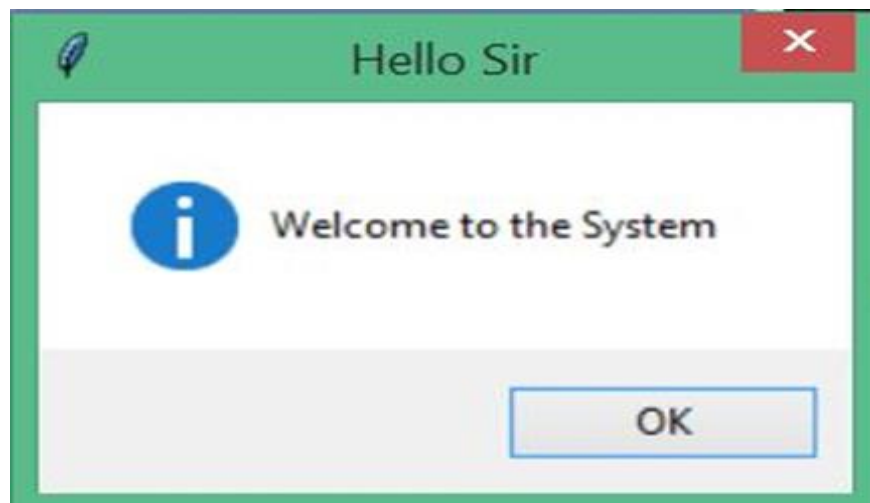


Fig :5.1.7 Successful Message

Test case 5:

Description:

If the username and password is already registered with the system.



Fig: 5.1.8 If Existing User Tries to sign-up

CHAPTER –6

CONCLUSION AND FUTURE WORK

CHAPTER- 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

In this project, transfer learning is implemented to classify DR into 5 classes with a much-reduced training data than other previous DR classification techniques employed. This was done to design a way to train a DL model that performs well on unseen data by efficiently learning from small dataset because training data is limited in healthcare. Our model has reached an accuracy that is higher than other techniques that have used transfer learning on the whole Kaggle DR challenge dataset for multi-class classification. Our model has reached a superior performance on account of the selected training algorithm, which is batch gradient descent with ascending learning rate, and the quadratic weighted kappa loss function. Deep learning techniques that can learn from small dataset to categorize medical images should be utilized to classify DR, as this can be transferred to other medical image classification problems facing the challenge of insufficient training data. Experiments should be done to compare performances of other pre-trained deep convolutional Networks.

6.2 FUTURE WORK

We will the same model on larger dataset compared to the present dataset. We will apply the feature extraction part from pre-trained model and apply to algorithms such as support vector machines and changing the performance measures such as specificity and sensitivity as it gives healthcare more trust to model usage in real time. we will apply the different image pre-processing techniques to the dataset and compare the performances and will compare the different transfer learning techniques and apply the pre-trained models to complex image classification challenges in real world problems.

CHAPTER - 7

APPENDIX 1

CHAPTER – 7

APPENDIX 1

SAMPLE CODE:

blindness.py

```
# Importing all packages
#C:\Users\arun\AppData\Roaming\Microsoft\Windows\Start
Menu\Programs\Python 3.10
from tkinter import *
from tkinter.ttk import *
from tkinter import messagebox
from PIL import Image
import os
import mysql.connector
from tkinter.filedialog import askopenfilename, asksaveasfilename
import mysql.connector as sk
#from models.Model import Model
from model import *
#from send_sms import *
print('GUI SYSTEM STARTED...')
#-----
def LogIn():
    username = box1.get()
    u = 1
    if len(username) == 0:
        u = 0
    messagebox.showinfo("Error", "You must enter something Sir")
    if u:
```

```

password = box2.get()
if len(password):
    query = "SELECT * FROM THEGREAT"
    sql.execute(query)
    data = sql.fetchall()
    g = 0
    b = 0
    for i in data:
        if i[0] == username:
            g = 1
        if i[1] == password:
            b = 1
    if g and b:
        messagebox.showinfo('Hello Sir', 'Welcome to the System')
    else:
        messagebox.showinfo('Sorry', 'Wrong Username or Password')
    global y
    y = True
    else:
        messagebox.showinfo("Error", "You must enter a password Sir!!")
def OpenFile():
    username = box1.get()
    if y:
        try:
            a = askopenfilename()
            print(a)
            value, classes = main(a)
            messagebox.showinfo("your report", ("Predicted Label is ", value, "\nPredicted

```

```

Class is ", classes))
#send(value, classes)
query = 'UPDATE THEGREAT SET PREDICT = "%s" WHERE USERNAME =
"%s"'%(value, username)
sql.execute(query)
#print(query)
connection.commit()
#-----*****Only use when required to send message
#send(value, classes)
#-----*****
image = Image.open(a)
# plotting image
file = image.convert('RGB')
plt.imshow(np.array(file))
plt.title(f'your report is label : {value} class : {classes}')
plt.show()
#print(image)
print("Thanks for using the system !")
#fn, text = os.path.splitext(a) #fn stands for filename
except Exception as error:
print("Pass")
else:
messagebox.showinfo("Hello Sir", "You need to Login first")
x = 0
y = False
def Signup():
username = box1.get()
password = box2.get()

```

```

u = 1
if len(username) == 0 or len(password) == 0:
    u = 0
    messagebox.showinfo("Error", "You must enter something Sir")
if u:
    query1 = "SELECT * FROM THEGREAT"
    sql.execute(query1)
    data = sql.fetchall()
    z = 1
    for i in data:
        if i[0] == username:
            messagebox.showinfo("Sorry Sir", "This username is already registered, try a new
            one")
            z = 0
    if z:
        query = "INSERT INTO THEGREAT (USERNAME, PASSWORD)
        VALUES('%s', '%s')" % (username, password)
        messagebox.showinfo("signed up", ("Hi ",username ,"\n Now you can login with
        your credentials !"))
        sql.execute(query)
        connection.commit()
#-----
connection = sk.connect(
    host="localhost",
    user="root",
    password="tigerarun$2",
    database="batch_db_new")
sql = connection.cursor()

```

```

root = Tk()
root.geometry('700x400')
root.title("Diabetic Retinopathy Detection")
root.configure(bg='mediumpurple')
#pale turquoise
label1 = Label(root, text="Demo for DRD", font=('Arial', 30))
label1.grid(padx=30, pady=30, row=0, column=0, sticky='W')
label2 = Label(root, text="Enter your username: ", font=('Arial', 20))
label2.grid(padx=10, pady=10, row=1, column=0, sticky='W')
label3 = Label(root, text="Enter your password: ", font=('Arial', 20))
label3.grid(padx=10, pady=20, row=2, column=0, sticky='W')
box1 = Entry(root)
box1.grid(row=1, column=1)
box2 = Entry(root, show='*')
box2.grid(row=2, column=1)
button3 = Button(root, text="Signup", command=Signup)
button3.grid(padx=10, pady=20, row=3, column=1)
button1 = Button(root, text="LogIn", command=LogIn)
button1.grid(padx=10, pady=20, row=3, column=2)
button2 = Button(root, text="Upload Image", command=OpenFile)
button2.grid(padx=10, pady=20, row=2, column=3)

```

Model.py

```
from torch import optim
import torchvision
import torch.nn.functional as F
from torchvision import datasets, transforms, models
import torchvision.models as models
from PIL import Image, ImageFile
import json
from torch.optim import lr_scheduler
import random
import os
import sys
print('Imported packages')
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = models.resnet152(pretrained=False)
num_fts = model.fc.in_features
out_fts = 5
model.fc = nn.Sequential(nn.Linear(num_fts,
512),nn.ReLU(),nn.Linear(512,out_fts),nn.LogSoftmax(dim=1))
criterion = nn.NLLLoss()
optimizer = torch.optim.Adam(filter(lambda p:p.requires_grad,model.parameters()), lr = 0.00001)
scheduler = lr_scheduler.StepLR(optimizer, step_size=5, gamma=0.1)
model.to(device);
# to unfreeze more layers
for name,child in model.named_children():
if name in ['layer2','layer3','layer4','fc']:
#print(name + 'is unfrozen')
```

```

for param in child.parameters():
    param.requires_grad = True
else:
    #print(name + 'is frozen')
for param in child.parameters():
    param.requires_grad = False
optimizer = torch.optim.Adam(filter(lambda p:p.requires_grad,model.parameters())
, lr = 0.000001)
scheduler = lr_scheduler.StepLR(optimizer, step_size=5, gamma=0.1)
def load_model(path):
    checkpoint = torch.load(path,map_location='cpu')
    model.load_state_dict(checkpoint['model_state_dict'])
    optimizer.load_state_dict(checkpoint['optimizer_state_dict'])
    return model
def inference(model, file, transform, classes):
    file = Image.open(file).convert('RGB')
    img = transform(file).unsqueeze(0)
    print('Transforming your image...')
    device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
    model.eval()
    with torch.no_grad():
        print('Passing your image to the model....')
        out = model(img.to(device))
        ps = torch.exp(out)
        top_p, top_class = ps.topk(1, dim=1)
        value = top_class.item()
        print("Predicted Severity Value: ", value)
        print("class is: ", classes[value])

```



```

print('Your image is printed:')
return value, classes[value]
#plt.imshow(np.array(file))
#plt.show()
model =
load_model(r'C:\Users\arun\Desktop\project\Retinal_blindness_detection_Pyt
orch-master\classifier.pt')
print("Model loaded Succesfully")
classes = ['No DR', 'Mild', 'Moderate', 'Severe', 'Proliferative DR']
test_transforms = torchvision.transforms.Compose([
torchvision.transforms.Resize((224, 224)),
torchvision.transforms.RandomHorizontalFlip(p=0.5),
torchvision.transforms.ToTensor(),
torchvision.transforms.Normalize(mean=(0.485, 0.456, 0.406), std=(0.229,
0.224, 0.225))])
def main(path):
x, y = inference(model, path, test_transforms, classes)

```

CHAPTER - 08

APPENDIX 2

CHAPTER – 8

APPENDIX 2

SAMPLE OUTPUT:

1. The system looks like this where we can enter the username and password to login the system and upload the image and see our results.

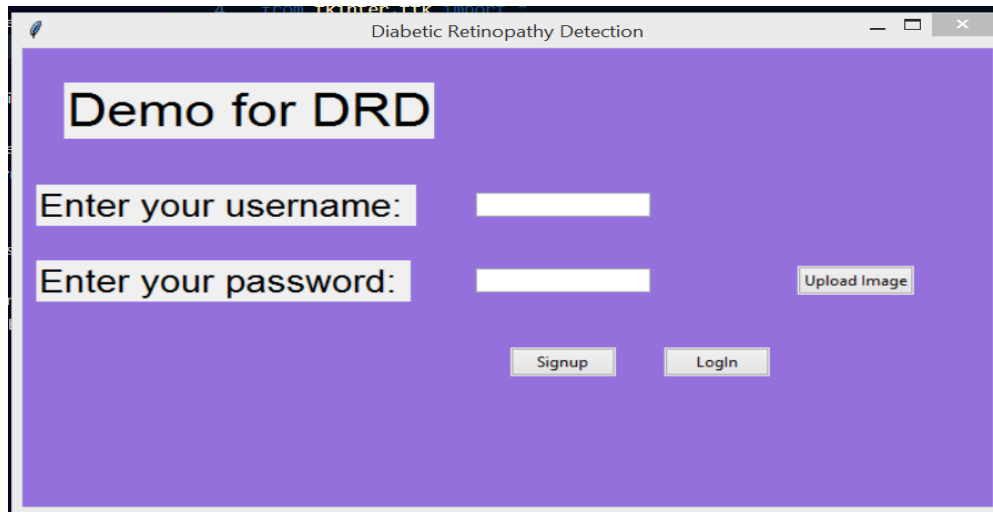


Fig 8.1 UI of the Project

2. If you enter the correct username and password it will display the message like” welcome to the system” like below.

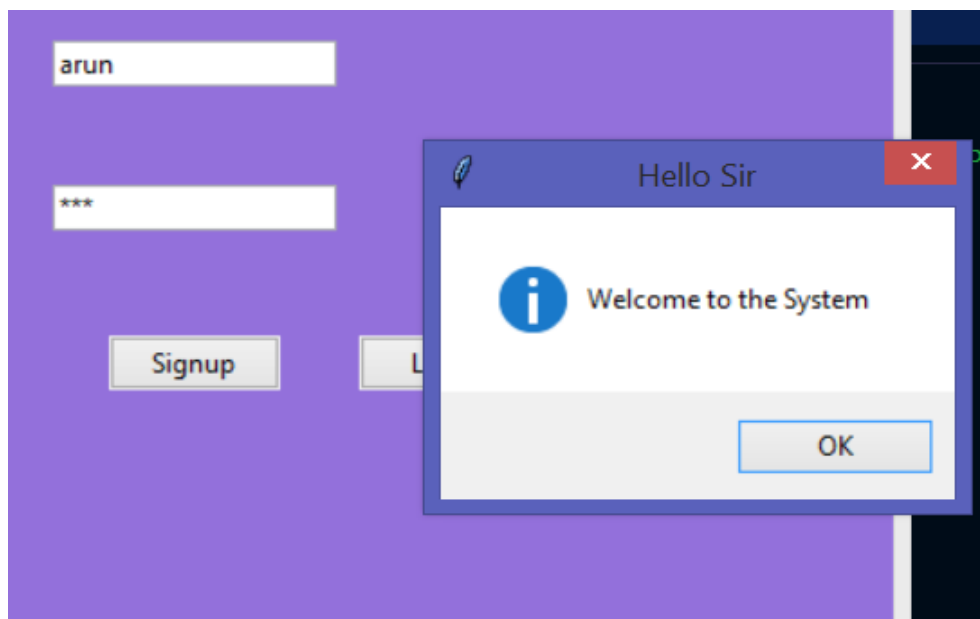


Fig 8.2 Log - In page

3. If the username or password is wrong it will display the message like “wrong username or password”.

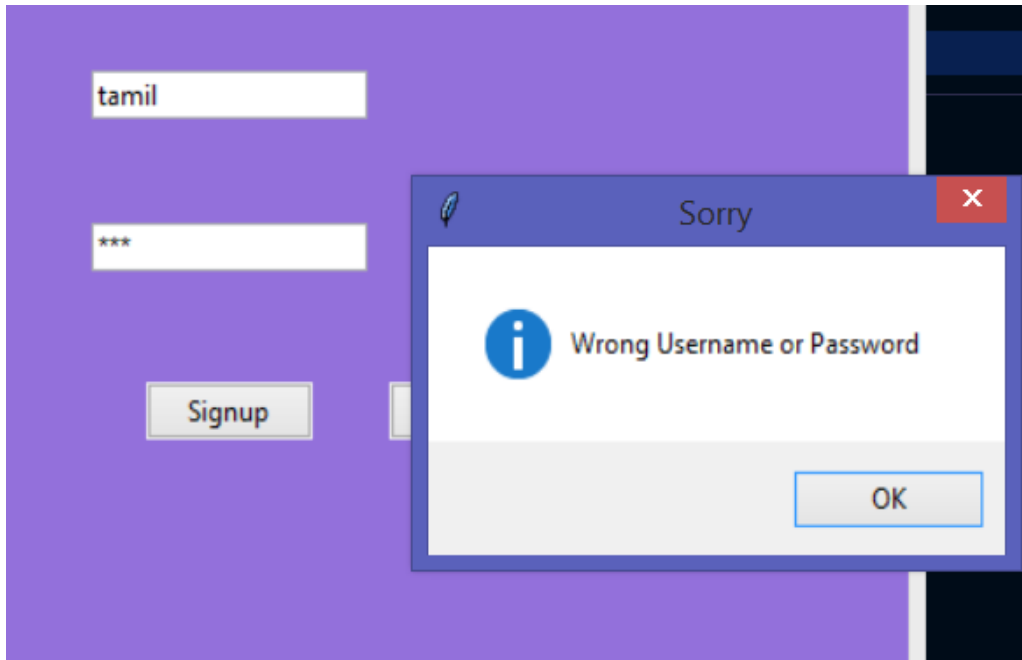


Fig 8.3 Wrong Authentication

4. If you are the new user to the system, you will enter the username and password it will display the message like “now you login with your credentials”.

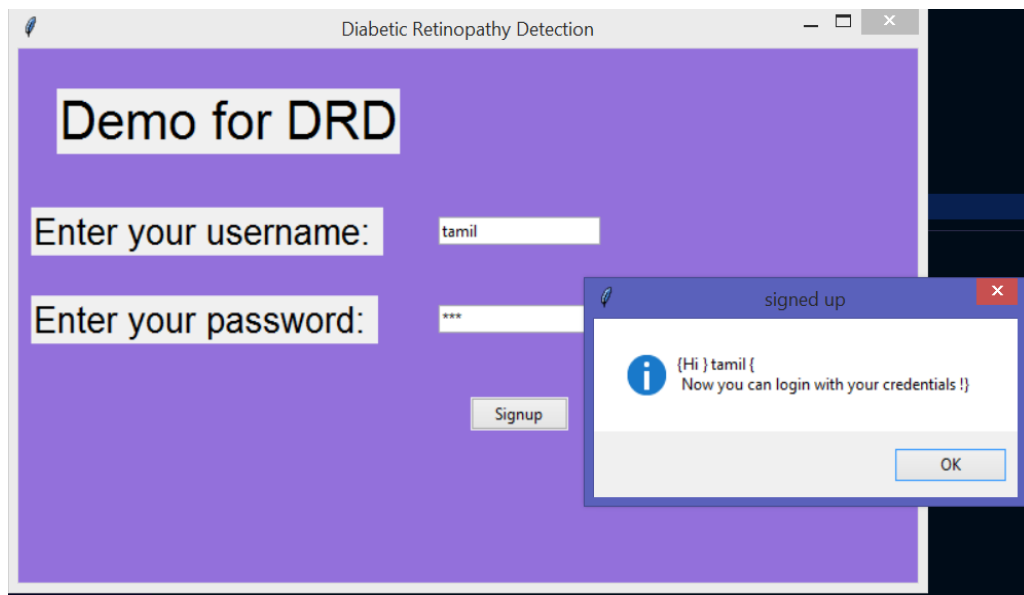


Fig 8.4 Successful Log - in

5 a). After the successful login with your credentials, you can upload the image by clicking on upload image, selecting the image and click upload it will display the output.

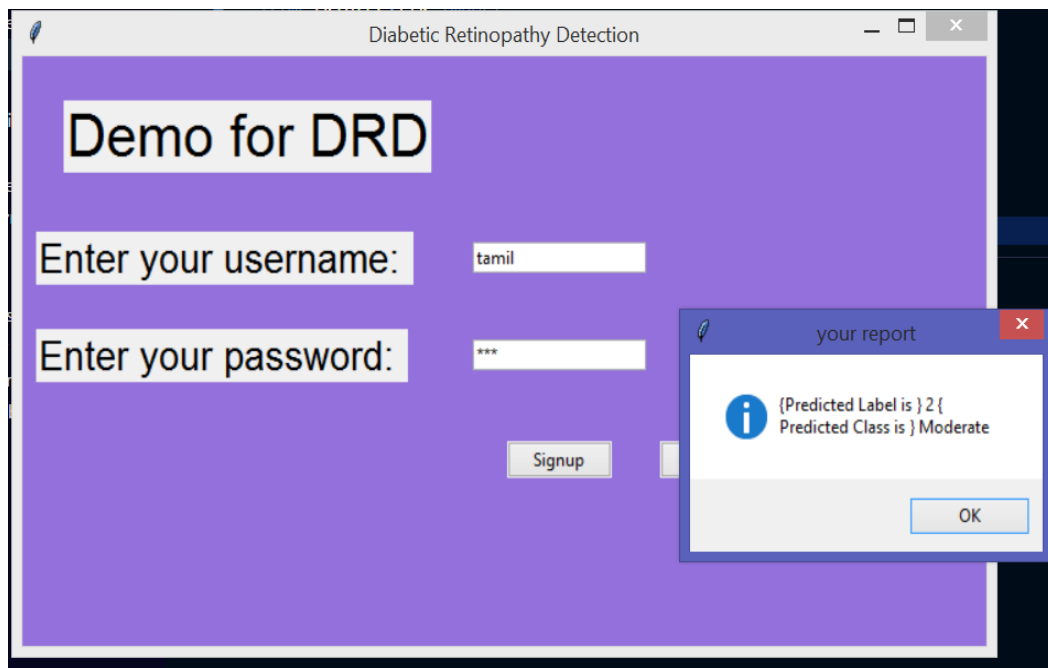


Fig 8.5 Prediction report

5 b). The output is displayed as shown below.

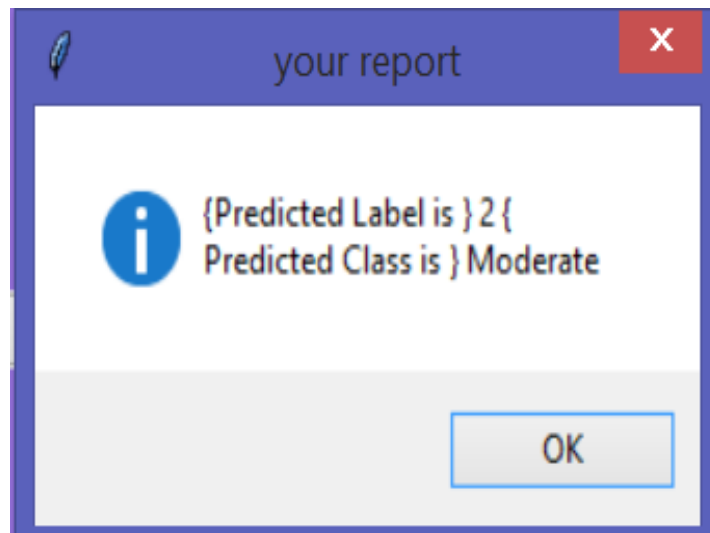


Fig 8.6 Displaying Report

6). The database is updated as shown below.

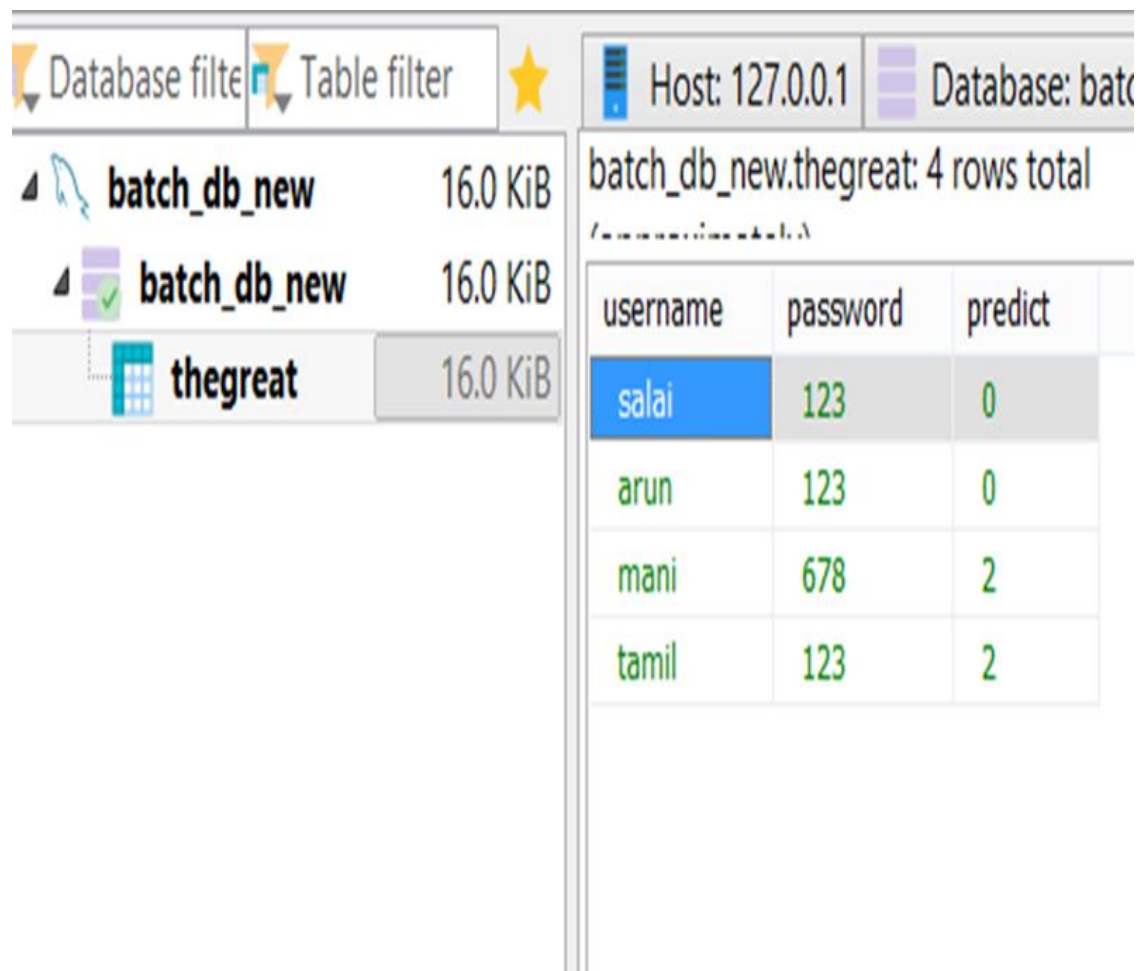
0 - No DR

1- Mild DR

2 - Moderate DR

3 - Severe DR

4 - Proliferative DR



username	password	predict
salai	123	0
arun	123	0
mani	678	2
tamil	123	2

Fig 8.7 Database Updation

CHAPTER-9

REFERENCES

CHAPTER – 9

REFERENCES

1. Faust, O., Acharya, R., Ng, E.Y.K., Ng, K.H. and Suri, J.S., 2012. Algorithms for the automated detection of diabetic retinopathy using digital fundus images: a review. *Journal of medical systems*, 36(1), pp.145-157.
2. Gao, Z., Li, J., Guo, J., Chen, Y., Yi, Z. and Zhong, J., 2018. Diagnosis of Diabetic Retinopathy Using Deep Neural Networks. *IEEE Access*, 7, pp.3360- 3370
3. Gargeya, R. and Leng, T., 2017. Automated identification of diabetic retinopathy using deep learning. *Ophthalmology*, 124(7), pp.962-969
4. Gondal, W.M., Köhler, J.M., Grzeszick, R., Fink, G.A. and Hirsch, M., 2017, September. Weakly-supervised localization of diabetic retinopathy lesions in retinal fundus images. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 2069-2073). IEEE.
5. Gulshan, V., Peng, L., Coram, M., Stumpe, M.C., Wu, D., Narayanaswamy, A., Venugopalan, S., Widner, K., Madams, T., Cuadros, J. and Kim, R., 2016. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *Jama*, 316(22), pp.2402-2410
6. Kumar, S. and Kumar, B., 2018, February. Diabetic Retinopathy Detection by Extracting Area and Number of Microaneurysm from Colour Fundus Image. In *2018 5th International Conference on Signal Processing and Integrated Networks (SPIN)* (pp. 359-364). IEEE
7. Mansour, R.F., 2018. Deep-learning-based automatic computer-aided diagnosis system for diabetic retinopathy. *Biomedical engineering letters*, 8(1), pp.41-57.
8. Mohammadian, S., Karsaz, A. and Roshan, Y.M., 2017, November. Comparative Study of Fine-Tuning of Pre-Trained Convolutional Neural Networks for Diabetic

Retinopathy Screening. In 2017 24th National and 2nd International Iranian Conference on Biomedical Engineering (ICBME) (pp. 1-6). IEEE

9. Pratt, H., Coenen, F., Broadbent, D.M., Harding, S.P. and Zheng, Y., 2016. Convolutional neural networks for diabetic retinopathy. *Procedia Computer Science*, 90, pp.200-205

10. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818-2826).

11. Wan, S., Liang, Y. and Zhang, Y., 2018. Deep convolutional neural networks for diabetic retinopathy detection by image classification. *Computers & Electrical Engineering*, 72, pp.274-282

12. Zhou, K., Gu, Z., Liu, W., Luo, W., Cheng, J., Gao, S. and Liu, J., 2018, July. MultiCell Multi-Task Convolutional Neural Networks for Diabetic Retinopathy Grading. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (pp. 2724-2727). IEEE