VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Belagavi, Karnataka, India

Mini Project Report on

"RECOGNITION OF DIABETIC RETINOPATHY USING CNN ALGORITHM"

Submitted in partial fulfillment of the requirement of VI Semester Mini Project (18ECMP68)

Submitted by,

AKSHAT GUPTA	**1RF20EC003**
MOHAMMED NADEEM	**1RF20EC028**
SHIVAM KUMAR RAI	**1RF20EC042**
VIKRANT RANA	**1RF20EC053**

Under the guidance of

Dr. KAVITHA N,

Assistant Professor,

Dept. of ECE, RVITM

DEPARTMENT OF

Electronics & Communication Engineering

RV INSTITUTE OF TECHNOLOGY AND MANAGEMENT®

BANGALORE-560076

2022-23

RV INSTITUTE OF TECHNOLOGY AND MANAGEMENT®

(Affiliated to Visvesvaraya Technological University, Belagavi & Approved by AICTE, NewDelhi)

Bengaluru-560076

DEPARTMENT OF
ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

Certified that the project work titled **"RECOGNITION OF DIABETIC RETINOPATHY USING CNN ALGORITHM"** is carried out by **AKSHAT GUPTA(1RF20EC003),MOHAMMED NADEEM(1RF20EC028), SHIVAM KUMAR RAI(1RF20EC042), and VIKRANT RANA(1RF20EC053),** who are bonafide students of RV Institute of Technology and Management, Bangalore, in partial fulfillment for the award of degree of **Bachelor of Engineering** in Electronics and Communication Engineering of the Visvesvaraya Technological University, Belagavi during the year **2022-2023**. It is certified that all corrections/suggestions indicated for the internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed by the institution for the said degree.

Signature of Guide	**Signature of Head of the Department**	**Signature of Principal**
Dr. Kavitha N	Dr. Prashant P Patavardhan	Dr. Jayapal R

External Viva

	Name of Examiners	**Signature with date**
1		
2		

RV INSTITUTE OF TECHNOLOGY AND MANAGEMENT®

(Affiliated to Visvesvaraya Technological University, Belagavi & Approved by AICTE, NewDelhi)

Bengaluru-560076

DEPARTMENT OF
ELECTRONICS AND COMMUNICATION ENGINEERING

DECLARATION

We, **AKSHAT GUPTA-1RF20EC003, MOHAMMED NADEEM-1RF20EC028, SHIVAM KUMAR RAI-1RF20EC042** and **VIKRANT RANA-1RF1EC053,** the students of sixth semester DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING, hereby declare that the mini project titled **"RECOGNITION OF DIABETIC RETINOPATHY USING CNN ALGORITHM" has been** carried out by us and submitted in partial fulfillment for the award of the degree of Bachelor of Engineering in **ELECTRONICS AND COMMUNICATION ENGINEERING**. We do declare that this work is not carried out by any other students for the award of a degree in any other branch.

Place: Bengaluru

Date:

Name:

Signature:

1. **AKSHAT GUPTA**
2. **MOHAMMED NADEEM**
3. **SHIVAM KUMAR RAI**
4. **VIKRANT RANA**

ACKNOWLEDGEMENT

The successful presentation of the **PROJECT** would be incomplete without the mention of the people who made it possible and whose constant guidance crowned oureffort with success.

We would like to thank our Project Guide, **Dr. Kavitha N**, Assistant Professor, Department of Electronics and Communication Engineering, RV Institute of Technology and Management, Bengaluru, for her constant guidance and inputs.

We thank **Dr. Prashant P Patavardhan,** Professor and Head of the Department of Electronics and Communication Engineering, RV Institute of Technology and Management, Bengaluru, for his encouragement.

We would like to extend our gratitude to **Dr. Jayapal R**, Principal, RV Institute of Technology and Management, Bengaluru, for providing us an opportunity to work on a project in this institution.

We would like to thank all the **Teaching and Non-Teaching Staff** for their cooperation.

We would like to extend our gratitude to the **MANAGEMENT**, RV Institute of Technology and Management, Bengaluru, for providing all the facilities to present the Project.

Finally, we extend our heartfelt gratitude to our family for their encouragement and support without which we wouldn’t have come so far. Moreover, we thank all our friends for their invaluable support and cooperation.

AKSHAT GUPTA-1RF20EC003
MOHAMMED NADEEM-1RF20EC028
SHIVAM KUMAR RAI-1RF20EC042
VIKRANT RANA-1RF20EC053

ABSTRACT

Diabetes is a chronic end organ disease that occurs when the pancreas does not secrete enough insulin or the body is unable to process it properly. Over time, diabetes affects the circulatory system, including that of the retina. Diabetic retinopathy (DR) is a medical condition where the retina is damaged because fluid leaks from blood vessels into the retina. Ophthalmologists recognize diabetic retinopathy based on features, such as blood vessel area, exudes hemorrhages, microaneurysms and texture. Early detection and treatment can limit the potential for significant vision loss from diabetic retinopathy.

CNN algorithms are widely used for the detection of diabetic retinopathy (DR), it helps in analyzing fundus images in diagnosis. The proposed approach includes 3 stages: Preprocessing, Feature extraction and Classification. The proposed method is evaluated on a publicly available dataset. Image pre-processing is the steps taken to format images before they are used by model training and inference. Image preprocessing may also decrease model training time and increase model inference speed. Feature extraction is a part of the dimensionality reduction process, in which, an initial set of the raw data is divided and reduced to more manageable groups. The last step includes a classification algorithm which classifies the images into 5 categories namely: No DR, Mild DR, Moderate DR, Severe DR and Prolific DR.

The manual process of periodic DR diagnosis and detection for necessary treatment, is time consuming and unreliable due to unavailability of resources and expert opinion. Therefore, computerized diagnostic systems which use Convolutional Neural Network (CNN) architectures, are proposed to learn DR patterns from fundus images and identify the severity of the disease. We have successfully been able to achieve a maximum of 95.40% accuracy using the model that we have developed. Thus, we have minimized erroneous outputs and maximized accuracy to provide better result analysis and help medical workers choose the required treatment accordingly.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS AND NOMENCLATURE

Term	**Description**
Accuracy	A metric used to measure model performance in terms of how correctly it predicts data points or image feature against the ground truth.
AI	Artificial Intelligence; reference to the simulation of human intelligence in machines enabled by computer programs
API	Application Programming Interface
APTOS	Asia Pacific Tele-Ophthalmology Society
CNN	Convolutional Neural Network
Confusion Matrix	An N x N matrix used for evaluating the performance of a classification algorithm
DR	Diabetic Retinopathy; An eye condition caused by diabetes complication that the blood vessels of the light-sensitive tissues (retina) which can lead to total blindness
Diagnosis	identification of the nature of the disease through examination of the symptoms
Epoch	A period of time
Kaggle	An online data science environment that is used for practicing of Data Science technologies
Keras	A Deep Learning API is written in python and running on top of the TensorFlow platform.
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative

Term	**Description**
TensorFlow	Is an end-to-end ecosystem for Machine Learning with tools, libraries, and community resources.
Validation Accuracy	The accuracy obtained from using the validation dataset
RGB	Red, Green and Blue
ReLu	Rectified Linear Unit; An activation function which is used to remove any linear component.
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative

CHAPTER 1

INTRODUCTION

This section briefly describes the background, definition of the problem statement, motivation behind choosing the domain. It also consists of the important problem formulations and objectives along with the scope of the project.

Diabetic retinopathy is an eye disease caused by diabetes that can lead to loss of vision or even complete blindness. Diabetic retinopathy accounts for 12% of all new cases of blindness in the United States, and is the leading cause of blindness for people aged 20 to 64 years. If caught early enough, progression to vision impairment can be slowed if not altogether stopped, however, this is often difficult because symptoms may appear too late to provide effective treatment. Diabetic retinopathy (DR) has been estimated to affect about 93 million people globally, though only half are aware of it.

1.1 Problem Statement

Current Limitations: Existing methods for DR detection often rely on manual assessment by trained specialists, resulting in time-consuming and subjective evaluations that hinder scalability and accessibility of diagnosis.

Need for Automated Solutions: There is a critical need for automated and reliable techniques that can efficiently screen and detect DR from retinal images, enabling early diagnosis and intervention.

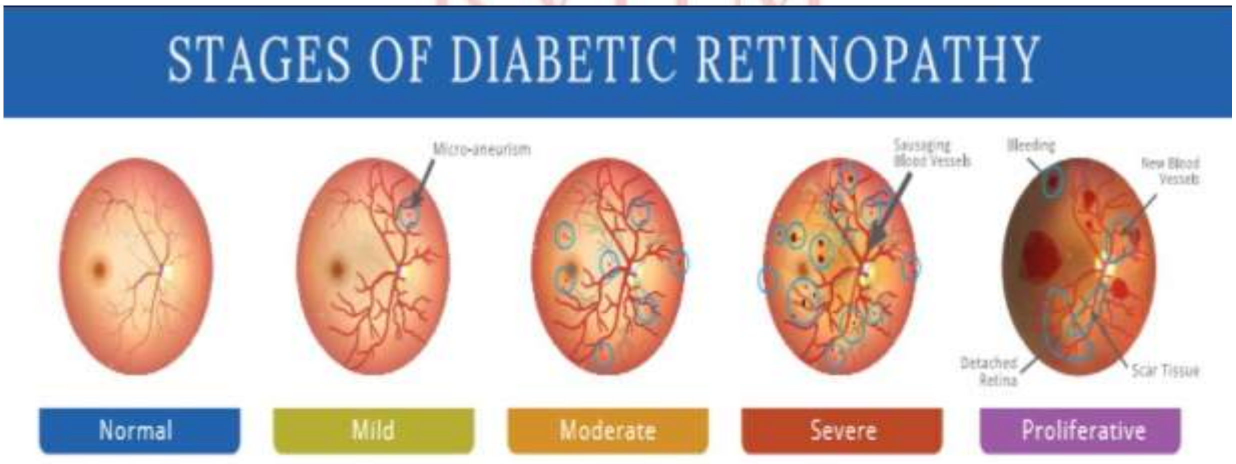


Fig 1.1 Stages of Diabetic Retinopathy

1.2 Literature Survey

This sub-section contains information related to CNN-based approaches:

Diabetic Retinopathy (DR) is caused as a result of Diabetes Mellitus which causes development of various retinal abrasions in the human retina. These lesions cause hindrance in vision and in severe cases, DR can lead to blindness. DR is observed amongst 80% of patients who have been diagnosed from prolonged diabetes for a period of 10–15 years. The manual process of periodic DR diagnosis and detection for necessary treatment, is time consuming and unreliable due to unavailability of resources and expert opinion. Therefore, computerized diagnostic systems which use Deep Learning (DL) Convolutional Neural Network (CNN) architectures, are proposed to learn DR patterns from fundus images and identify the severity of the disease. [1] paper proposes a comprehensive model using 26 state-of-the-art DL networks to assess and evaluate their performance, and which contribute for deep feature extraction and image classification of DR fundus images. In the proposed model, ResNet50 has shown highest overfitting in comparison to Inception V3, which has shown lowest overfitting when trained using the Kaggle's EyePACS fundus image dataset.

CNN, hybrid CNN with ResNet, hybrid CNN with DenseNet are used on a huge dataset with around 3662 train images to automatically detect which stage DR has progressed. Five DR stages, which are 0 (No DR), 1 (Mild DR), 2 (Moderate), 3 (Severe) and 4 (Proliferative DR) are processed in the proposed work. The patient's eye images are fed as input to the model. The proposed deep learning architectures like CNN, hybrid CNN with ResNet, hybrid CNN with DenseNet 2.1 are used to extract the features of the eye for effective classification. The models achieved an accuracy of 96.22%, 93.18% and 75.61% respectively. The [2] paper concludes with a comparative study of the CNN, hybrid CNN with ResNet, hybrid CNN with DenseNet architectures that highlights hybrid CNN with DenseNet as the perfect deep learning classification model for automated DR detection.

To help patients with the early detection of diabetic retinopathy, paper [3],they proposed a computer vision-based technique to analyze and predict diabetes from the retinal input images. Image preprocessing, segmentation, and feature extraction steps are applied. Convolutional neural networks (CNN) and Support Vector Machine (SVM) are trained with diabetic and non-diabetic retinal images. Results show CNN reports
better accuracy in DR compared to SVM.

Ophthalmologists use optical coherence tomography (OCT) and fundus photography for the purpose of assessing the retinal thickness, and structure, in addition to detecting edema, hemorrhage, and scars. Deep learning models are mainly used to analyze OCT or fundus images, extract unique features for each stage of DR and therefore classify images and stage the disease. A deep Convolutional Neural Network (CNN) with 18 convolutional layers and 3 fully connected layers is proposed [4] to analyze fundus images and automatically distinguish between controls (i.e. no DR), moderate DR (i.e. a combination of mild and moderate Non Proliferative DR (NPDR)) and severe DR (i.e. a group of severe NPDR, and Proliferative DR (PDR)) with a validation accuracy of 88%-89%, a sensitivity of 87%-89%, a specificity of 94%-95%, and a Quadratic Weighted Kappa Score of 0.91–0.92 when both 5-fold, and 10-fold cross validation methods were used respectively.

At first, the disease grading is targeted using different classification models. Classification models used in this study are ResNet50, VGG-16, and VGG19. Three datasets generated during pre-processing along with original images were used to train the classification models. [5] These models were pre-trained on the ImageNet dataset having 1000 classes and were finetuned by adding more layers for our disease grading task. The advantage of using a pre-trained model is to mitigate the effect of fewer datasets. shows the additional
fine-tuning layers added to the original architecture of the models.ResNet-50 is a model given by Microsoft and has the quality feed output of some layer directly to the input of some other layer by bypassing layers in This study [6] compared the performance of the revised Resnet model with other common CNNs models (Xception, AlexNet, VggNet-s, VggNet-16 and ResNet-50.

In examining said models, the results alluded to an over-fitting phenomenon, and the outcome of the work demonstrates that the performance of the revised ResNet-50 (Train accuracy: 0.8395 and Test accuracy: 0.7432) is better than other common CNNs (that is, the revised structure of ResNet-50 could avoid the overfitting problem, decease the loss value, and reduce the fluctuation problem).

To solve the classification problems, many different types of ResNets are used, with different numbers of layers: specifically, 18, 34, 50, 101, and 152 layers [7]. The current deep learning framework for detecting and grading DR is ResNet-50. However, the disadvantages of ResNet-50 are overfitting and fluctuations in accuracy, which affect its accuracy in detecting DR. This study proposes three strategies to improve the performance of ResNet-50, as follows:
Adaptive learning rate in ResNet-50. 2) Regularization: Regularization can be employed to minimize the overfitting of the training model. 3) Obtain suitable features from conv5_block1_out and conv5_block2_out in ResNet-50.

The proposed [8] approach uses deep features of ResNet-50 along with Random Forest as a classifier for the detection and grading of diabetic retinopathy. High-level features obtained from the average pooling layer of trained ResNet-50 are fed to a random forest classifier. The depth of the deep network plays a pivotal role in their performance. With the increase in layers, the model gives better performance. However, it has also been observed that the addition of layers may increase the error rate. This is named as an issue of vanishing gradients. The residual neural network, also known as ResNet, was introduced to address this problem. Residual Network uses the skip connection to indiscriminately allow some input to the layer to incorporate the flow of information and also to prevent its loss, hence, addressing the problem of vanishing gradients (which also suppresses the generation of some noise). Suppressing the noise means averaging the models, which keeps a balance between precision and generalization. To achieve higher precision and an estimated level of traversal, the most efficient way is to increase more labeled data. The structure of ResNet speeds up the training of ultra-deep neural networks and increases the model’s accuracy on large training data.

The resolution of the input image has a direct impact on the DR grading performance[9]. Generally, ResNet50 is designed for images of 224×224 input resolution (He et al., 2016). In ResNet-50, a convolution layer with 6 a kernel size of 7×7 and a stride of 2 followed by a max-pooling layer is applied to dramatically down sample the input image first. Therefore, using images with very small input resolution may lose key features for DR grading, such as tiny lesions. In contrast, a network fed with large resolution images can extract more fine-grained and dense features at the cost of a smaller receptive field and a higher computational cost. In this work, a range of resolutions is evaluated to identify the trade-off. Applying data augmentation during training can increase the distribution variability of the input images to improve the generalization capacity and robustness of a model of interest.

After the training of the residual network with 20 epochs, the features can extracted from their average global pooling layer. These features will be detailed and unique as this model averaged out all the activations of the final convolution layer. Due to parameter limitations, the global average pooling does not require optimization. Moreover, owing to spatial translation, it is more robust to the input as it summarizes spatial information. The dropout was set to 0.2 to reduce the overfitting [10].

1.3 MOTIVATION

According to the World Health Organization, diabetic retinopathy is the leading cause of blindness among working-age adults, affecting up to 35% of people with diabetes worldwide.
In 2021, Approximately 537 million adults (20-79 years) are living with diabetes. The total number of people living with diabetes is projected to rise to 643 million by 2030 and 783 million by 2045. 3 in 4 adults with diabetes live in low- and middle-income countries. One out of two people suffering from diabetes has been diagnosed with some stage of DR. Detection of DR symptoms in time can avert the vision impairment in majority of cases, however such revelation is difficult with present tools and methods. There has been a need for comprehensive and automated DR detection tools and methods.

By developing an accurate and reliable method for detecting diabetic retinopathy using deep learning techniques, One can potentially make a significant impact on patient outcomes and improve the quality of life for many people. Generally this is what happens, suppose you have to show your eyes to ophthalmologist , eyes are dilated and then there is consultation with the ophthalmologist and this entire process takes a lot of time for diagnosis and the scarcity of ophthalmologist makes it harder and expensive. By our technique we can simplify the process of detection and make the entire consultation more efficient and cost effective.

1.4 OBJECTIVES

- Develop a CNN algorithm that accurately detects and classifies diabetic retinopathy from retinal images.
- Explore and optimize the performance of the CNN algorithm by experimenting with different network architectures, hyperparameters, and data augmentation techniques.
- Assess the interpretability of the CNN algorithm by analyzing the learned features and generating visual explanations for the algorithm's predictions.
- Develop a user-friendly software or application that integrates the CNN algorithm for easy deployment and utilization by healthcare professionals.

1.5 Organization of the Report

Chapter 1 deals with the introduction to Diabetic retinopathy and briefly describes the background, definition of the problem statement, motivation behind choosing the domain. It also consists of the important problem formulations and objectives of the project.

Chapter 2 deals with the Theory and fundamentals of the area related to the problem statement. it covers the working of existing methods. This chapter discussed the problems that are faced by the people in different places.

Chapter 3 include the methodology implemented in the proposed project. It contains the steps that are followed by the algorithm.

Chapter 4 deals with the implementation of the design model and development of the fronted of the project.

Chapter 5 deals with the results and discussion of the project. It's been discussed about the output that has been achieved using the developed CNN model. It contains detailed graphical representation of the accuracy and losses.

Chapter 6 deals with the conclusion and future scope of the work carried out. The conclusion is a critical section that provides a summary of the key findings and outcomes of the project. the future scope section outlines potential areas for future research or improvement based on the results and findings of the project.

Chapter 2

Theory and Fundamentals

This chapter highlights all the basic fundamentals, theory and principles of the project in brief, starting from the introduction to what Diabetic Retinopathy is to what CNN are.

2.1 Diabetic Retinopathy

2.1.1 Introduction

Diabetic retinopathy is a diabetes complication that affects the eyes. It's caused by damage to the blood vessels of the light-sensitive tissue at the back of the eye (retina).At first, diabetic retinopathy might cause no symptoms or only mild vision problems. But it can lead to blindness. The condition can develop in anyone who has type 1 or type 2 diabetes. The longer you have diabetes and the less controlled your blood sugar is, the more likely you are to develop this eye complication[1].

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[2].

Symptoms:

You might not have symptoms in the early stages of diabetic retinopathy. As the condition progresses, you might develop:

- Spots or dark strings floating in your vision (floaters)
- Blurred vision
- Fluctuating vision
- Dark or empty areas in your vision
- Vision loss

2.1.2 Complications

Diabetic retinopathy involves the growth of abnormal blood vessels in the retina. Complications can lead to serious vision problems such as:

- **Vitreous hemorrhage:** The new blood vessels may bleed into the clear, jelly like substance that fills the center of your eye. If the amount of bleeding is small, you might see only a few dark spots (floaters). In more-severe cases, blood can fill the vitreous cavity and completely block your vision. Vitreous hemorrhage by itself usually doesn't cause permanent vision loss. The blood often clears from the eye within a few weeks or months. Unless your retina is damaged, your vision will likely return to its previous clarity.

- **Retinal detachment:** The abnormal blood vessels associated with diabetic retinopathy stimulate the growth of scar tissue, which can pull the retina away from the back of the eye. This can cause spots floating in your vision, flashes of light or severe vision loss.

- **Glaucoma:** New blood vessels can grow in the front part of your eye (iris) and interfere with the normal flow of fluid out of the eye, causing pressure in the eye to build. This pressure can damage the nerve that carries images from your eye to your brain (optic nerve).

- **Blindness:** Diabetic retinopathy, macular edema, glaucoma or a combination of these conditions can lead to complete vision loss, especially if the conditions are poorly managed. Fig 2.1 shows the comparison between a healthy eye and a diabetic eye.

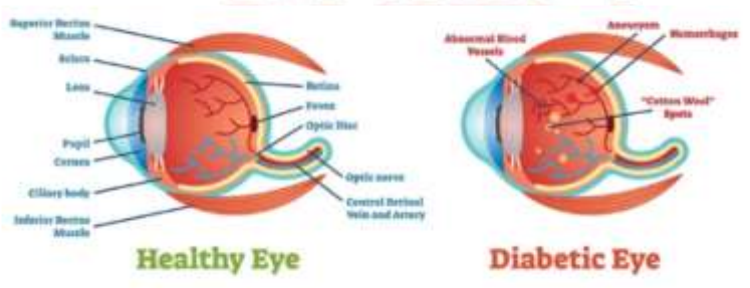


Fig 2.1 Healthy Vs Diabetic Eye

2.1.3 Prevention

You can't always prevent diabetic retinopathy. However, regular eye exams, good control of your blood sugar and blood pressure, and early intervention for vision problems can help prevent severe vision loss.

If you have diabetes, reduce your risk of getting diabetic retinopathy by doing the following:

- Manage your diabetes: Make healthy eating and physical activity part of your daily routine. Try to get at least 150 minutes of moderate aerobic activity, such as walking, each week. Take oral diabetes medications or insulin as directed.

- Monitor your blood sugar level: You might need to check and record your blood sugar level several times a day, or more frequently if you're ill or under stress. Ask your doctor how often you need to test your blood sugar.

- Ask your doctor about a glycosylated hemoglobin test: The glycosylated hemoglobin test, or hemoglobin A1C test, reflects your average blood sugar level for the two to three month period before the test. For most people with diabetes, the A1C goal is to be under 7%.

- Keep your blood pressure and cholesterol under control: Eating healthy foods, exercising regularly and losing excess weight can help. Sometimes medication is needed, too.

- Pay attention to vision changes: Contact your eye doctor right away if your vision suddenly changes or becomes blurry, spotty or hazy.

2.2 Deep Learning - An overview

2.2.1 Introduction to Deep Learning (DL)

Deep learning imitates how humans gain knowledge. Deep learning is part of data science, which covers statistics and predictive modelling. Deep learning helps data scientists acquire, analyze, and interpret massive amounts of data. Deep learning automates predictive analytics. Deep learning algorithms are hierarchical, increasing in complexity and abstraction. Iterate till output is accurate. Data processing layers inspired the name deep.

Deep learning models can be made using several ways. Learning rate decay, transfer learning, starting again, and dropout are approaches.

Deep learning models process information like the brain and can be used for numerous tasks. Most image recognition, NLP, and speech recognition tools use deep learning. Self-driving cars and language translation services are using these tools.

2.2.2 About Convolutional Neural Network (CNN)

In contrast to CNN, the performance of machine learning algorithms like K-nearest neighbor, SVM, etc. reaches saturation after reaching a certain number. Consequently, we intend to use CNN to advance. A convolutional neural network (CNN) is a form of Diabetic retinopathy Detection and Diagnosis using Deep Learning Algorithms artificial neural network that is specifically made to process pixel input and is used in image recognition and processing. Convolution typically means that it takes in an input signal and applies a filter over it, essentially multiplies the input signal with the kernel to get the modified signal. Mathematically, a convolution of two functions f and g is defined as shown in eq.2.1

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f\left(i - j + \frac{m}{2}\right) \qquad eq. 2.1$$

A multilayer perceptron-like system that has been optimized for low processing demands is used by a CNN. An input layer, an output layer, and a hidden layer with several convolutional layers, pooling layers, fully connected layers, and normalizing layers make up a CNN's layers. A system that is significantly more effective and easier to train for image processing and natural language processing is produced by the removal of restrictions and increase in efficiency for image processing[2,13].

Pooling

It is the step of down sampling the image’s features through summing up the information. The operation is carried out through each channel and thus it only affects the dimensions (n_H, n_W)and keeps n_C intact.

Given an image, we slide a filter, with no parameters to learn, following a certain stride, and we apply a function on the selected elements.

The pooling filters used in the model are:

Average pooling: we average on the elements present on the filter.

Max pooling: given all the elements in the filter, we return the maximum, Fig 2.2 gives an illustration of an average pooling

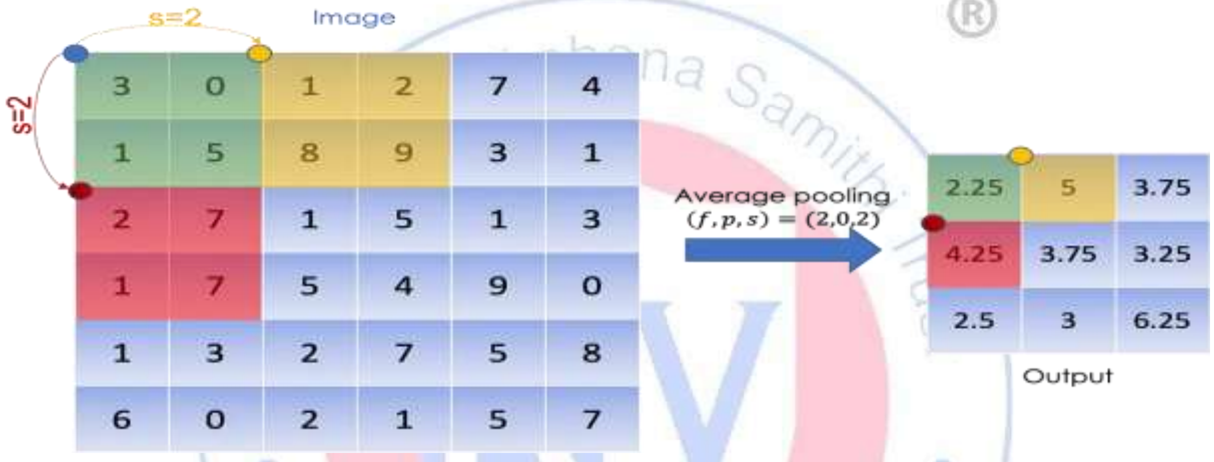


Fig 2.2 Average Pooling

Convolutional layer:

A convolutional layer contains a set of filters whose parameters need to be learned. The height and weight of the filters are smaller than those of the input volume. Each filter is convolved with the input volume to compute an activation map made of neurons. We apply convolutional products, using many filters this time, on the input followed by an activation function ψ. Fig 2.3 illustrates each layer involved in convolution filter.

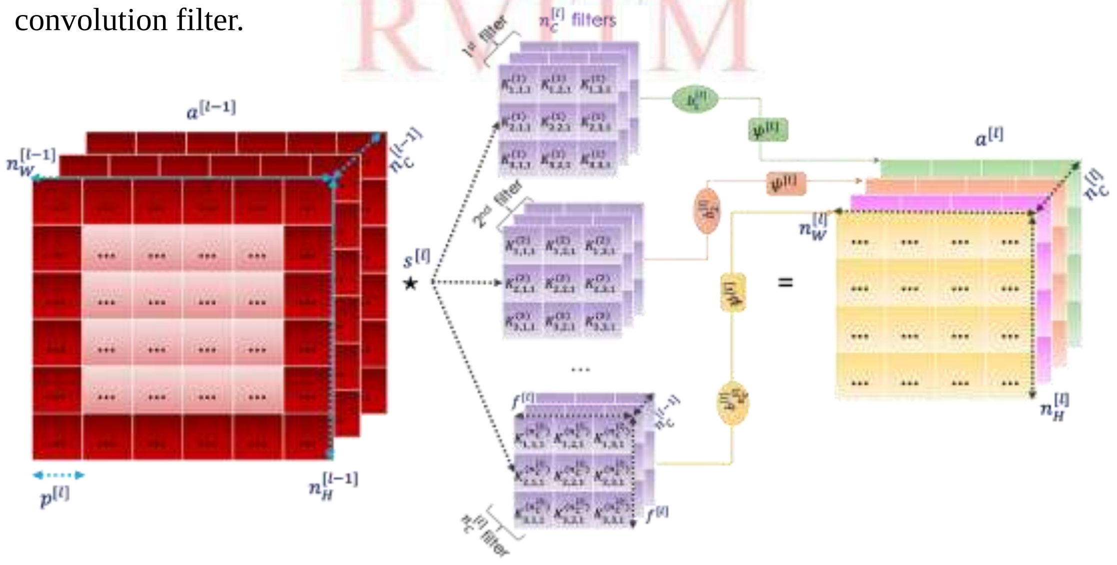


Fig 2.3 Convolution Layers

Padding:

The pixels on the corner of the image (2D matrix) are less used than the pixels in the middle of the picture which means that the information from the edges is thrown away. To solve this problem, we often add padding around the image in order to take the pixels on the edges into account. In convention, we pad with zeros and denote with p the padding parameter which represents the number of elements added on each of the four sides of the image. The following picture illustrates the padding of a grayscale image (2D matrix) where p=1,

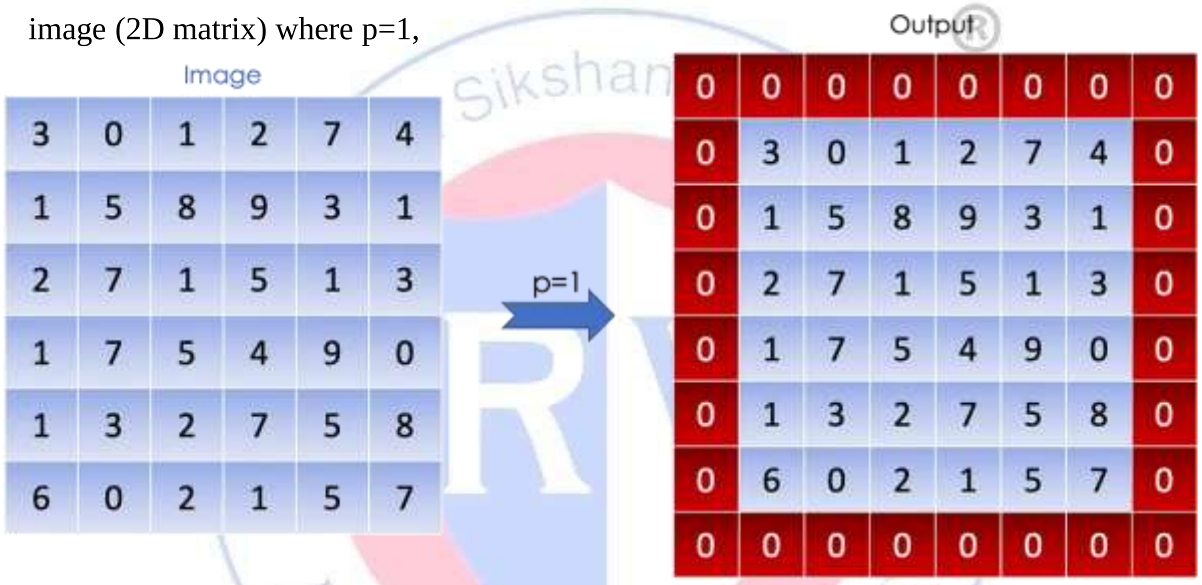


Fig 2.4 Zero Padding

Stride:

The stride is the step taken in the convolutional product. A large stride allows to shrink the size of the output and vice-versa. We denote s the stride parameter.

The Fig 2.5 illustrates a convolutional product (sum of element-wise element per block) with s=1:

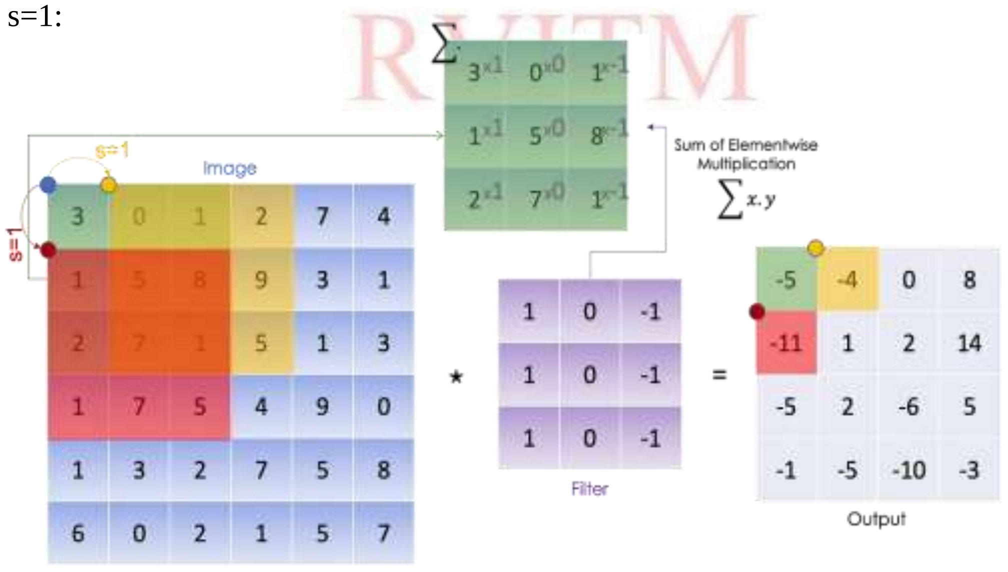


Fig 2.5 Stride

Chapter 3

Methodology and Design Flow

This chapter gives us an insight about the methodology followed in the project along with the details of the basic requirements needed to start working on the project.

3.1 Methodology

Based on the inputs provided, we initially recognize the features of diabetic retinopathy .Later steps involve algorithms from deep learning like Convolutional Neural Networks (CNN) with a combination of Artificial Neural Networks to develop an integrated network which extracts the input features and produces the required output. This will help in classifying the data into i)NO DR ii) MILD DR iii) MODERATE DR iv)SEVERE DR v)PROLIFERATIVE DR based on the stages and making use of the data provided.

3.2 Algorithm

The general algorithm followed for executing the project is:

Step 1: Obtaining appropriate Data sets

Step 2: Pre-processing the Data sets for developing feature matrix

Step 3: Split the Data sets into Train and Test sets

Step 4: Develop multi-layered Neural network with the use of CNN.

Step 5: Feed the training set to the model for feature extraction

Step 6: Define the iterations or epochs accordingly to obtain a good accuracy.

Step 7: Once the model is trained with the training data set, expose it to the test data

Step 8: Obtain validation loss and accuracy, training loss and accuracy graphs.

Step 9: Generate heat and confusion matrix for the obtained results

Step 10: Give any test set input to the network to check the predictions and classifications into five categories into i)NO DR ii) MILD DR iii) MODERATE DR iv)SEVERE DR v)PROLIFERATIVE DR .

3.3 Flow chart/ Block diagram

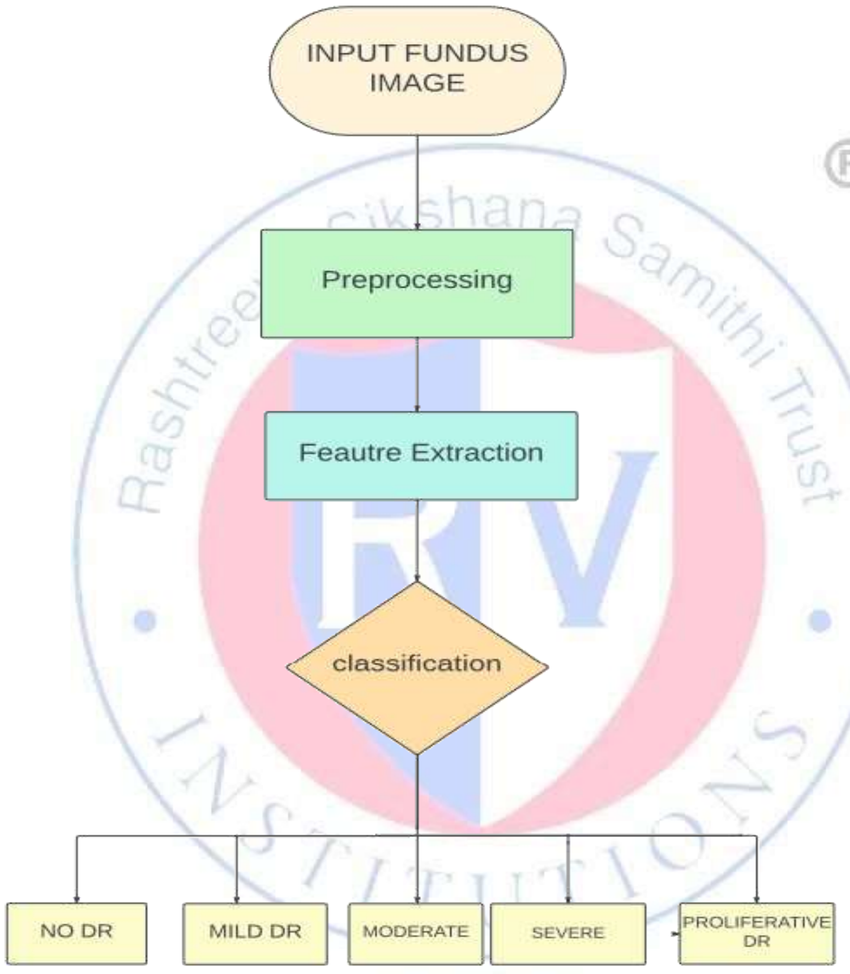


Fig 3.1 Flowchart

Fig. 3.1 represents the design flow of our project in the form of fundus image The dataset comprised of 3662 retinal fundus images. Out of which 2490 images are used for training, 439 images for validation and 733 for testing.

Pre-processing is a crucial process done before putting scanned fundus images into the deep convolutional neural network. The model reads the image files (stored in data folder). After decoding the PNG content it yields RGB grids of channels for the pixels. Convolution and pooling layers are the standard layers in deep convolutional neural networks. Input feature maps are convolved by convolution layers with kernel filters, which then generate output feature maps with an optional non-linear function. The pooling layer only uses one value from the input map, forcing the feature maps' resolution to drop so that the output feature maps can maintain their local deformation invariance. Following the extraction of features from convolutional and pooling layers, fully connected layers combine the output features into a complete feature vector using maximum and average pooling operations.

Additionally, it is necessary to appropriately change the image size before feeding an ultra-scanned image into a separate DCNN. So, we resize all the images in the dataset into 256*256 pixels. These should be transformed into floating-point tensors for neural network input. The extraction of features from scanned fundus images is the basics of image analysis, which contributes to the accurate investigation of breast cancer in patient.

Model fitting is a measure of how well a machine learning model generalizes to similar data to that on which it was trained. A model that is well-fitted produces more accurate outcomes. A model that is over fitted matches the data too closely. A model that is under fitted doesn't match closely enough. After testing the results are analysed in the form of confusion matrix and heat map.

3.4 PLATFORM/SOFTWARE USED

JUPYTER NOTEBOOK

Jupyter notebook lets anyone create and execute arbitrary Python code in the browser. It's ideal for machine learning, data analysis, and education. A notebook is saved with an .jpynb extension.

Python 3.10.0

Python libraries and frameworks offer a reliable environment which reduces software development time significantly. Python is consistent, simple, flexible, platform independent and has wide community which makes it most appropriate for machine learning Python includes a modular machine learning library PyBrain, Tensorflow, Keras, NumPy etc which offers many algorithms for machine learning task. Version

3.10.0 was used for this project.

VS CODE

Visual Studio Code, also commonly referred to as VS Code is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS.
It is used for compiling the HTML code for frontend.

FLASK

Flask is a micro web framework written in python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object relational framework, form validation, upload handling, various open authentication technologies and several common framework related tools.

Python Frontend framework that helps in converting the backend machine learning model to web app that make the classification more interactive and gives a better user experience.

Chapter 4

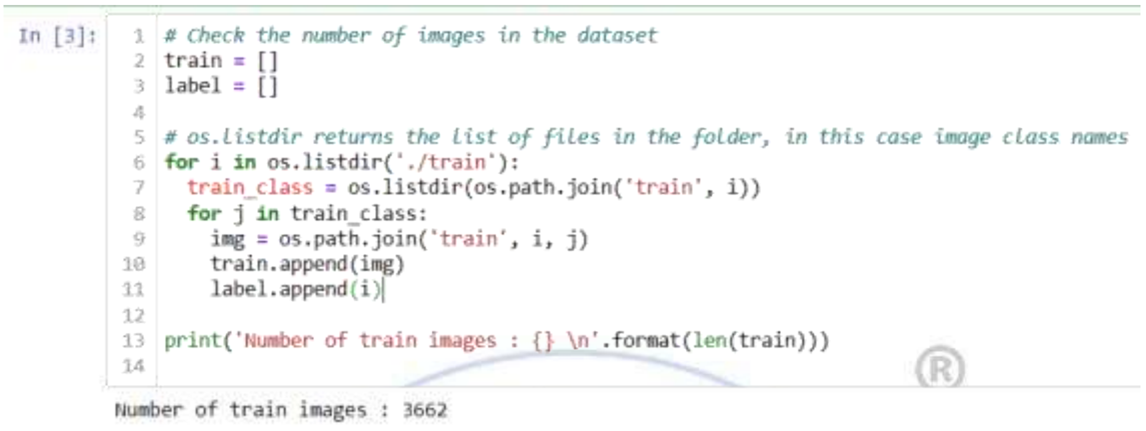
Design and Development

4.1 CODE:

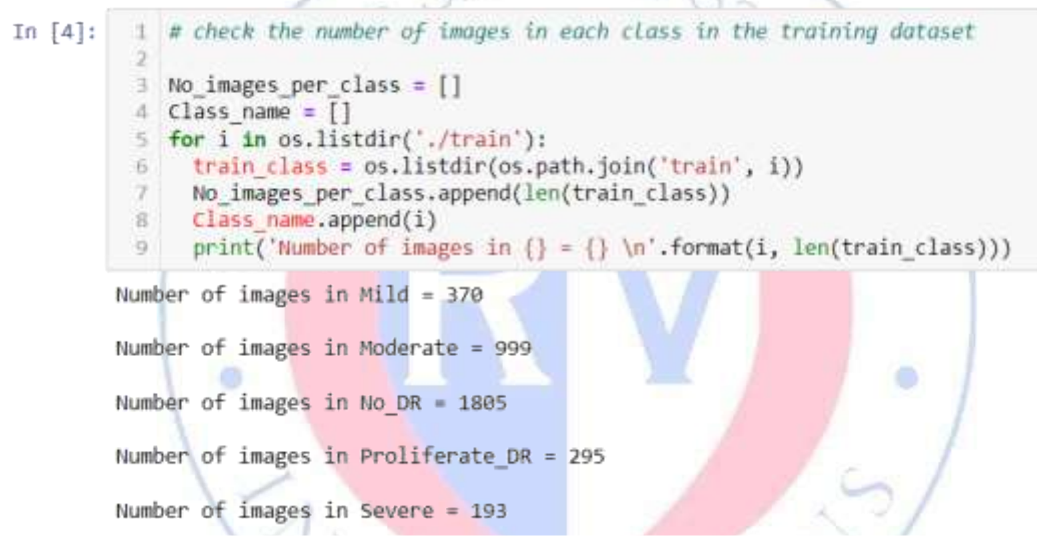
4.1.1 Libraries

```
# Import the necessary packages

import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow import keras
import os
import matplotlib.pyplot as plt
import PIL
import seaborn as sns
import plotly
import plotly.graph_objs as go
from sklearn.model_selection import train_test_split
from sklearn.utils import shuffle
from plotly.offline import iplot, init_notebook_mode
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.applications.inception_resnet_v2 import InceptionResNetV2
from tensorflow.keras.layers import *
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.initializers import glorot_uniform
from tensorflow.keras.utils import plot_model
from IPython.display import display
from tensorflow.keras import backend as K
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping,
        ModelCheckpoint,LearningRateScheduler
```

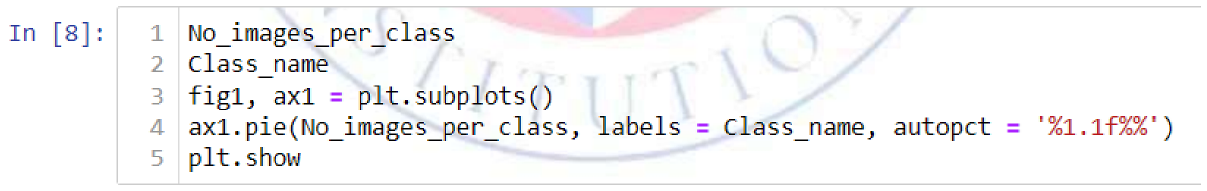
4.1.2 Data Representation And Statistics



```
In [3]:
# Check the number of images in the dataset
train = []
label = []

# os.listdir returns the list of files in the folder, in this case image class names
for i in os.listdir('./train'):
  train_class = os.listdir(os.path.join('train', i))
  for j in train_class:
    img = os.path.join('train', i, j)
    train.append(img)
    label.append(i)

print('Number of train images : {} \n'.format(len(train)))

Number of train images : 3662
```



```
In [4]:
# check the number of images in each class in the training dataset

No_images_per_class = []
Class_name = []
for i in os.listdir('./train'):
  train_class = os.listdir(os.path.join('train', i))
  No_images_per_class.append(len(train_class))
  Class_name.append(i)
  print('Number of images in {} = {} \n'.format(i, len(train_class)))

Number of images in Mild = 370

Number of images in Moderate = 999

Number of images in No_DR = 1805

Number of images in Proliferate_DR = 295

Number of images in Severe = 193
```



```
In [8]:
No_images_per_class
Class_name
fig1, ax1 = plt.subplots()
ax1.pie(No_images_per_class, labels = Class_name, autopct = '%1.1f%%')
plt.show
```

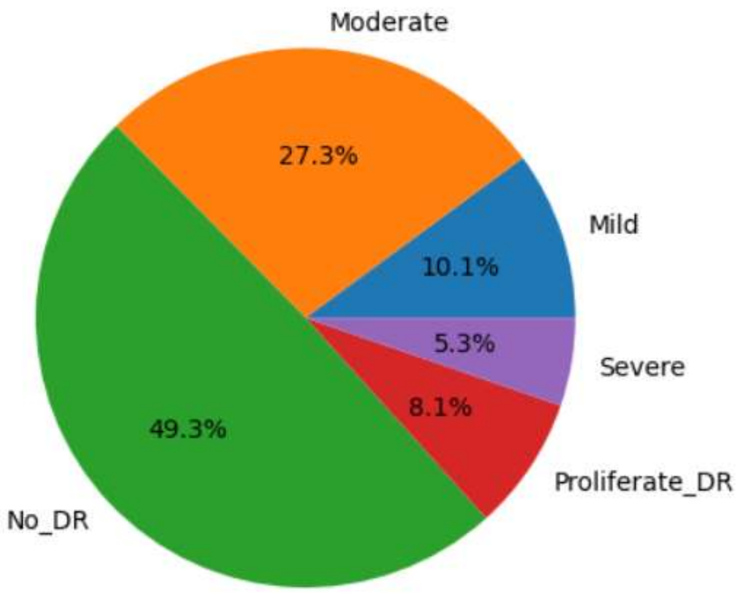
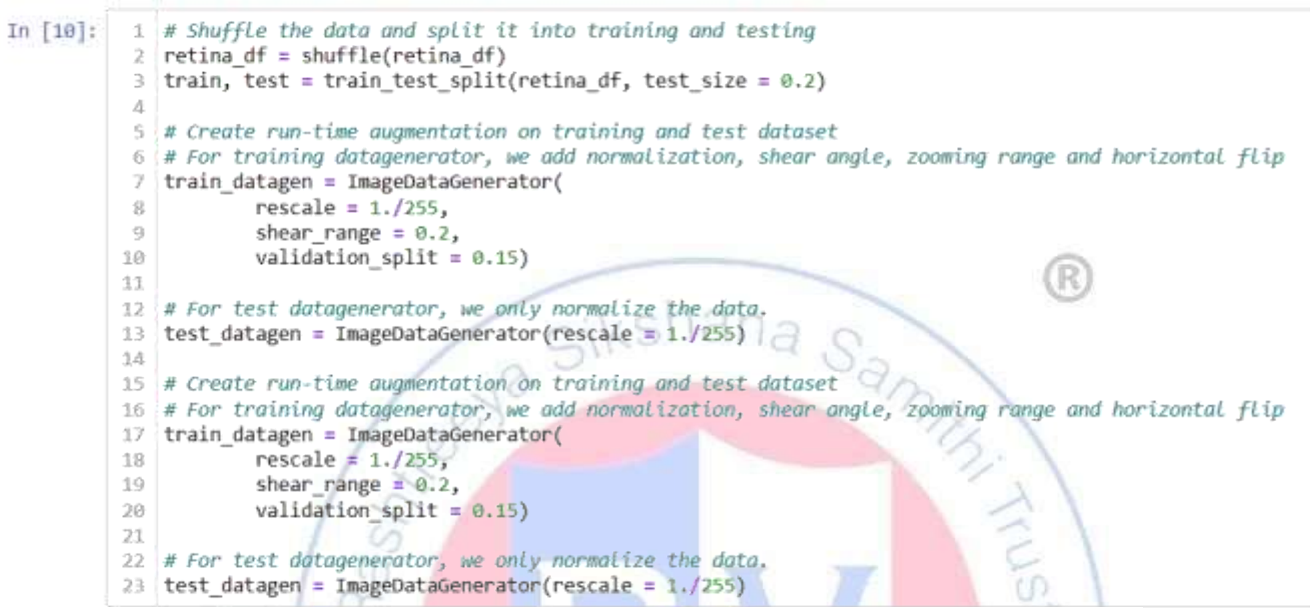
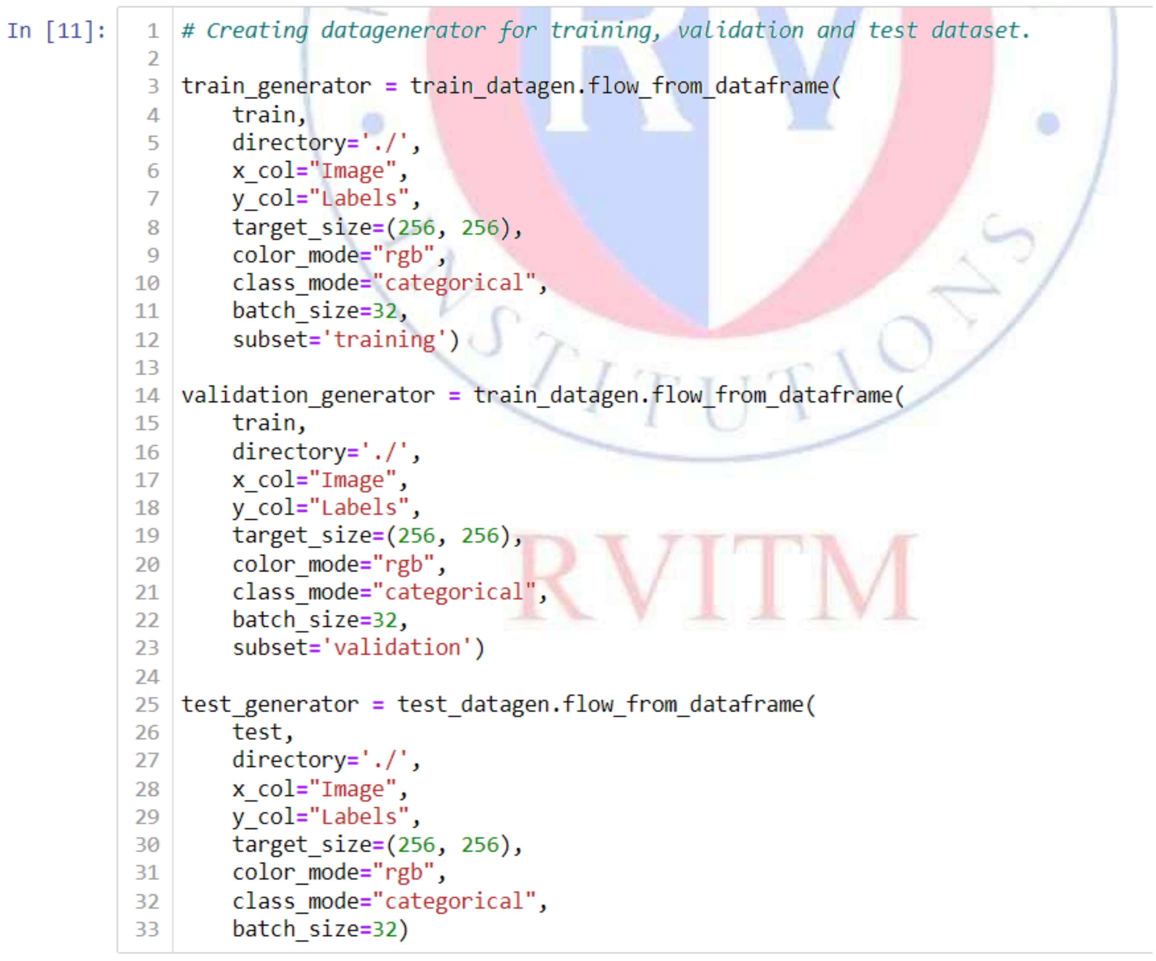


Fig 4.1 Dataset Distribution

4.1.3 Shuffling Data and Data Division



In [10]:
```python
# Shuffle the data and split it into training and testing
retina_df = shuffle(retina_df)
train, test = train_test_split(retina_df, test_size = 0.2)

# Create run-time augmentation on training and test dataset
# For training datagenerator, we add normalization, shear angle, zooming range and horizontal flip
train_datagen = ImageDataGenerator(
        rescale = 1./255,
        shear_range = 0.2,
        validation_split = 0.15)

# For test datagenerator, we only normalize the data.
test_datagen = ImageDataGenerator(rescale = 1./255)

# Create run-time augmentation on training and test dataset
# For training datagenerator, we add normalization, shear angle, zooming range and horizontal flip
train_datagen = ImageDataGenerator(
        rescale = 1./255,
        shear_range = 0.2,
        validation_split = 0.15)

# For test datagenerator, we only normalize the data.
test_datagen = ImageDataGenerator(rescale = 1./255)
```



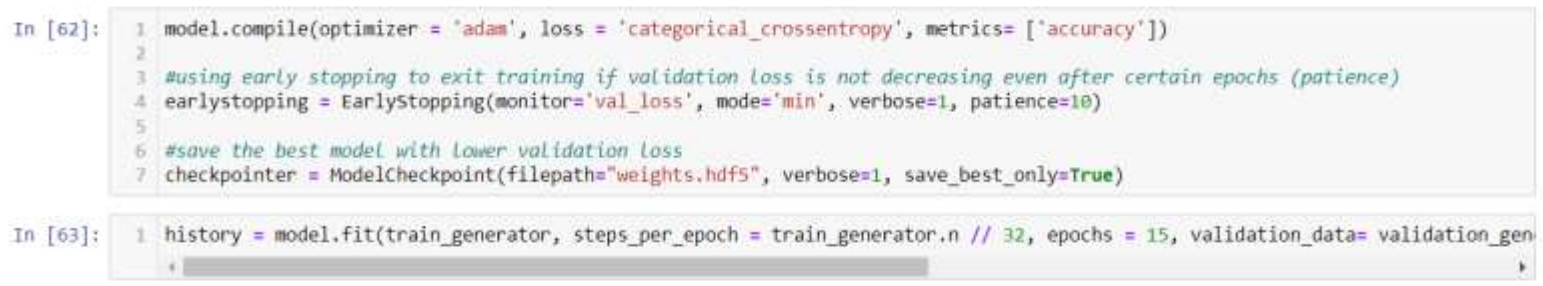
In [11]:
```python
# Creating datagenerator for training, validation and test dataset.

train_generator = train_datagen.flow_from_dataframe(
    train,
    directory='./',
    x_col="Image",
    y_col="Labels",
    target_size=(256, 256),
    color_mode="rgb",
    class_mode="categorical",
    batch_size=32,
    subset='training')

validation_generator = train_datagen.flow_from_dataframe(
    train,
    directory='./',
    x_col="Image",
    y_col="Labels",
    target_size=(256, 256),
    color_mode="rgb",
    class_mode="categorical",
    batch_size=32,
    subset='validation')

test_generator = test_datagen.flow_from_dataframe(
    test,
    directory='./',
    x_col="Image",
    y_col="Labels",
    target_size=(256, 256),
    color_mode="rgb",
    class_mode="categorical",
    batch_size=32)
```

4.1.4 CNN Architectural Block

```
def res_block(X, filter, stage):

  # Convolutional_block
  X_copy = X

  f1 , f2, f3 = filter

  # Main Path
  X = Conv2D(f1, (1,1),strides = (1,1), name ='res_'+str(stage)+'_conv_a', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = MaxPool2D((2,2))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_conv_a')(X)
  X = Activation('relu')(X)

  X = Conv2D(f2, kernel_size = (3,3), strides =(1,1), padding = 'same', name ='res_'+str(stage)+'_conv_b', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_conv_b')(X)
  X = Activation('relu')(X)

  X = Conv2D(f3, kernel_size = (1,1), strides =(1,1),name ='res_'+str(stage)+'_conv_c', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_conv_c')(X)


  # Short path
  X_copy = Conv2D(f3, kernel_size = (1,1), strides =(1,1),name ='res_'+str(stage)+'_conv_copy', kernel_initializer= glorot_uniform(seed = 0))(X_copy)
  X_copy = MaxPool2D((2,2))(X_copy)
  X_copy = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_conv_copy')(X_copy)

  # ADD
  X = Add()([X,X_copy])
  X = Activation('relu')(X)

  # Identity Block 1
  X_copy = X


  # Main Path
  X = Conv2D(f1, (1,1),strides = (1,1), name ='res_'+str(stage)+'_identity_1_a', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_identity_1_a')(X)
  X = Activation('relu')(X)
  X = Conv2D(f2, kernel_size = (3,3), strides =(1,1), padding = 'same', name
='res_'+str(stage)+'_identity_1_b', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_identity_1_b')(X)
  X = Activation('relu')(X)

  X = Conv2D(f3, kernel_size = (1,1), strides =(1,1),name
='res_'+str(stage)+'_identity_1_c', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_identity_1_c')(X)

  # ADD
  X = Add()([X,X_copy])
  X = Activation('relu')(X)

  # Identity Block 2
  X_copy = X


  # Main Path
  X = Conv2D(f1, (1,1),strides = (1,1), name ='res_'+str(stage)+'_identity_2_a',
kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_identity_2_a')(X)
  X = Activation('relu')(X)

  X = Conv2D(f2, kernel_size = (3,3), strides =(1,1), padding = 'same', name
='res_'+str(stage)+'_identity_2_b', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_identity_2_b')(X)
  X = Activation('relu')(X)

  X = Conv2D(f3, kernel_size = (1,1), strides =(1,1),name
='res_'+str(stage)+'_identity_2_c', kernel_initializer= glorot_uniform(seed = 0))(X)
  X = BatchNormalization(axis =3, name = 'bn_'+str(stage)+'_identity_2_c')(X)

  # ADD
  X = Add()([X,X_copy])
  X = Activation('relu')(X)

  return X


input_shape = (256,256,3)

#Input tensor shape
X_input = Input(input_shape)

#Zero-padding

X = ZeroPadding2D((3,3))(X_input)

# 1 - stage
```

```
X = Conv2D(64, (7,7), strides= (2,2), name = 'conv1', kernel_initializer= glorot_uniform(seed = 0))(X)
X = BatchNormalization(axis =3, name = 'bn_conv1')(X)
X = Activation('relu')(X)
X = MaxPooling2D((3,3), strides= (2,2))(X)

# 2- stage

X = res_block(X, filter= [64,64,256], stage= 2)

# 3- stage

X = res_block(X, filter= [128,128,512], stage= 3)

# 4- stage

X = res_block(X, filter= [256,256,1024], stage= 4)

# # 5- stage

#X = res_block(X, filter= [512,512,2048], stage= 5)

#Average Pooling

X = AveragePooling2D((2,2), name = 'Averagea_Pooling')(X)

#Final layer

X = Flatten()(X)
X = Dense(5, activation = 'softmax', name = 'Dense_final', kernel_initializer= glorot_uniform(seed=0))(X)


model = Model( inputs= X_input, outputs = X, name = 'Resnet18')

model.summary()
```

4.1.5 Training and Epochs



```
In [62]:
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics= ['accuracy'])

#using early stopping to exit training if validation loss is not decreasing even after certain epochs (patience)
earlystopping = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=10)

#save the best model with lower validation loss
checkpointer = ModelCheckpoint(filepath="weights.hdf5", verbose=1, save_best_only=True)

In [63]:
history = model.fit(train_generator, steps_per_epoch = train_generator.n // 32, epochs = 15, validation_data= validation_gen
```

4.1.6 Graphical Representation

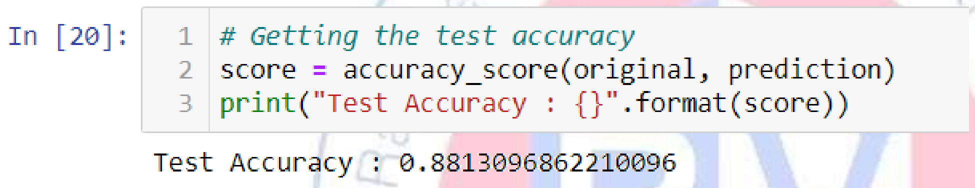
```
#plot of training loss and validation loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Training and Validation loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train_loss','val_loss'], loc = 'upper right')
plt.show()

#plot of training accuracy and validation loss
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Training and Validation Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train_accuracy','val_accuracy'], loc = 'lower right')
plt.show()

#Loading the trained weights
model.load_weights("retina_weights.hdf5")
```

4.1.7 Testing

```
# Assigning label names to the corresponding indexes
labels = {0: 'Mild', 1: 'Moderate', 2: 'No_DR', 3:'Proliferate_DR', 4: 'Severe'}

# Loading images and their predictions

from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
# import cv2

prediction = []
image = []

#Enter filepath
testpic = "./train/Moderate/00e4ddff966a.png"


img = PIL.Image.open(testpic)
plt.imshow(img)
img = img.resize((256,256))
 # converting image to array
img = np.asarray(img, dtype= np.float32)
  # normalizing the image
img = img / 255
```

```
# reshaping the image in to a 4D array
img = img.reshape(-1,256,256,3)
  # making prediction of the model
predict = model.predict(img)
  # getting the index corresponding to the highest value in the prediction
predict = np.argmax(predict)
  # appending the predicted class to the list
prediction.append(labels[predict])
print('Prediction= {}'.format(predict))
predicted_label = labels[predict]
print('Prediction Label:', predicted_label)
```

4.1.8 Accuracy test

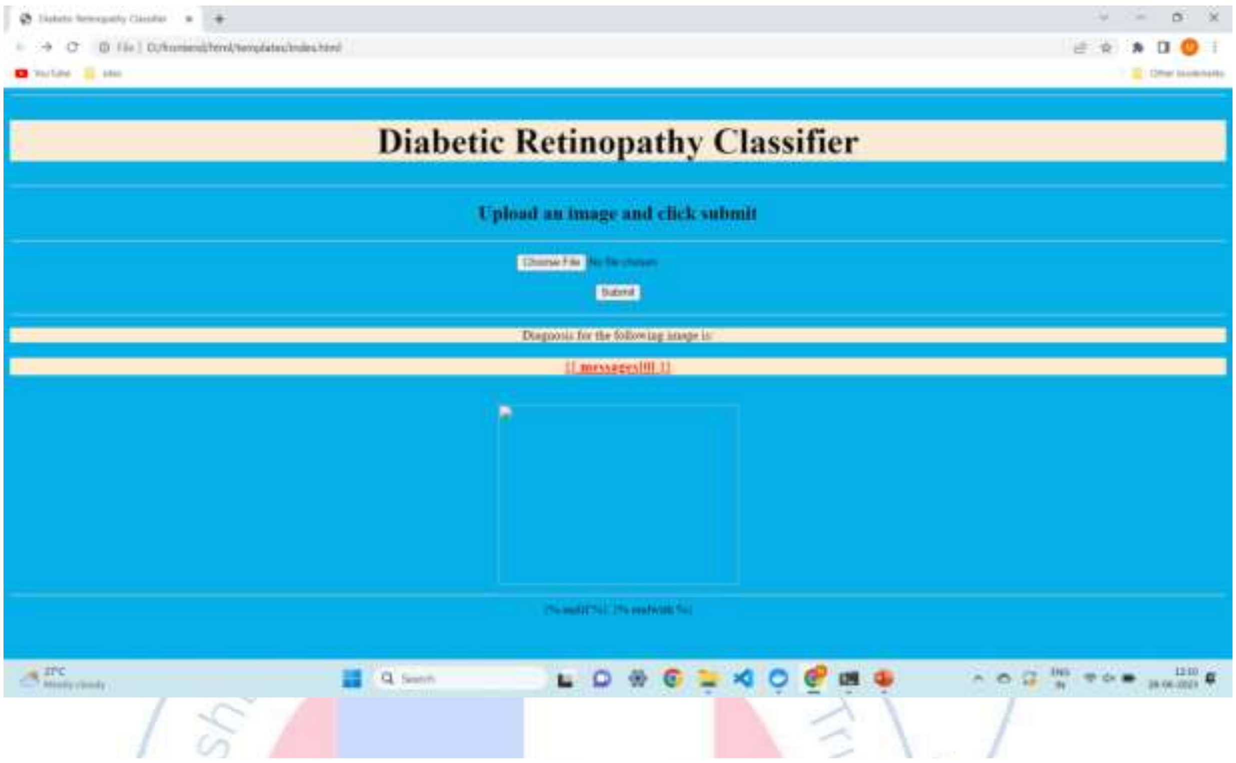


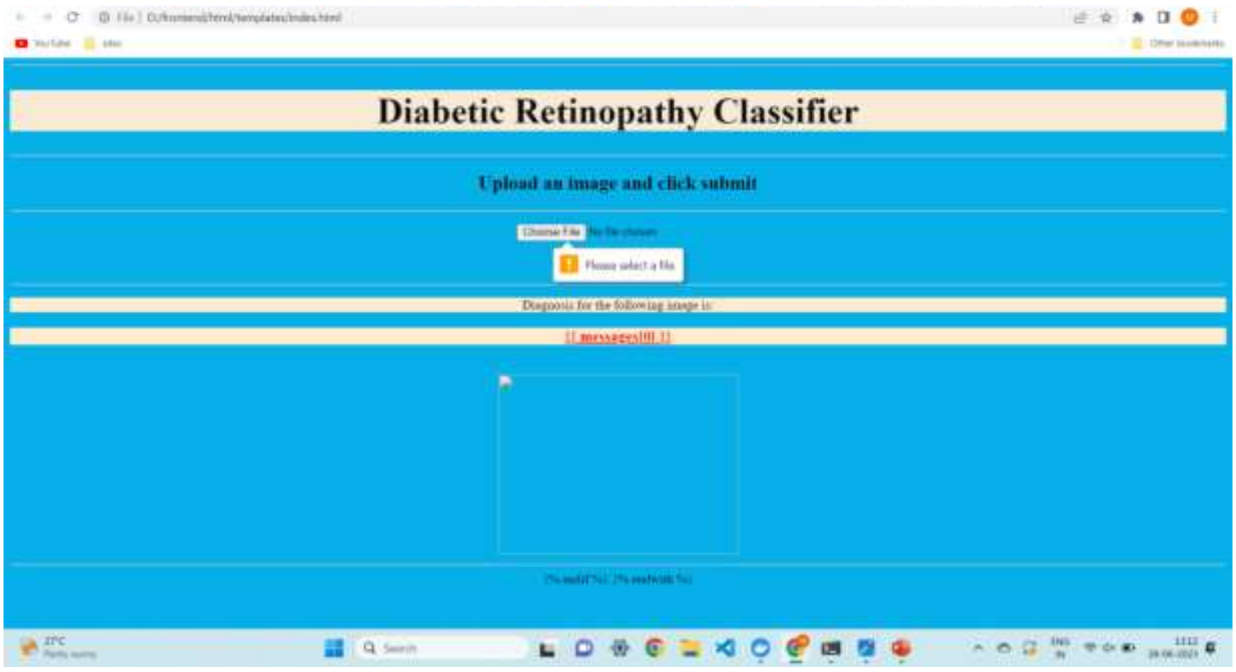
```
In [20]:
# Getting the test accuracy
score = accuracy_score(original, prediction)
print("Test Accuracy : {}".format(score))

Test Accuracy : 0.8813096862210096
```

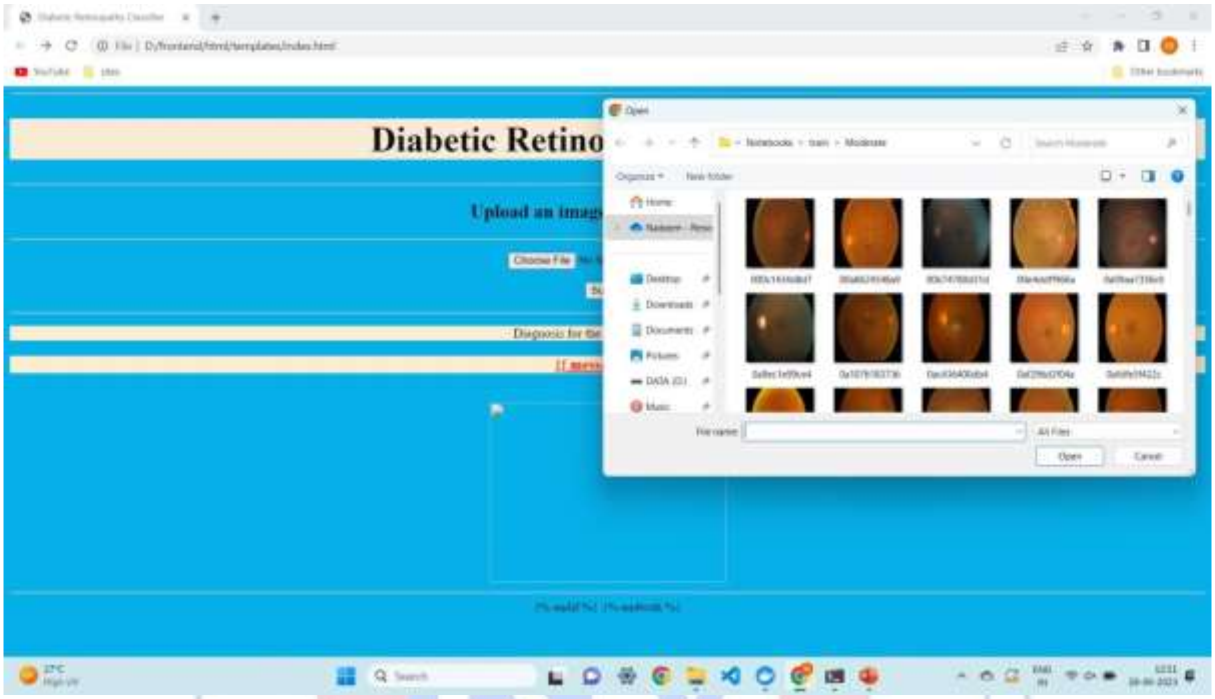
4.1.9 Visualization

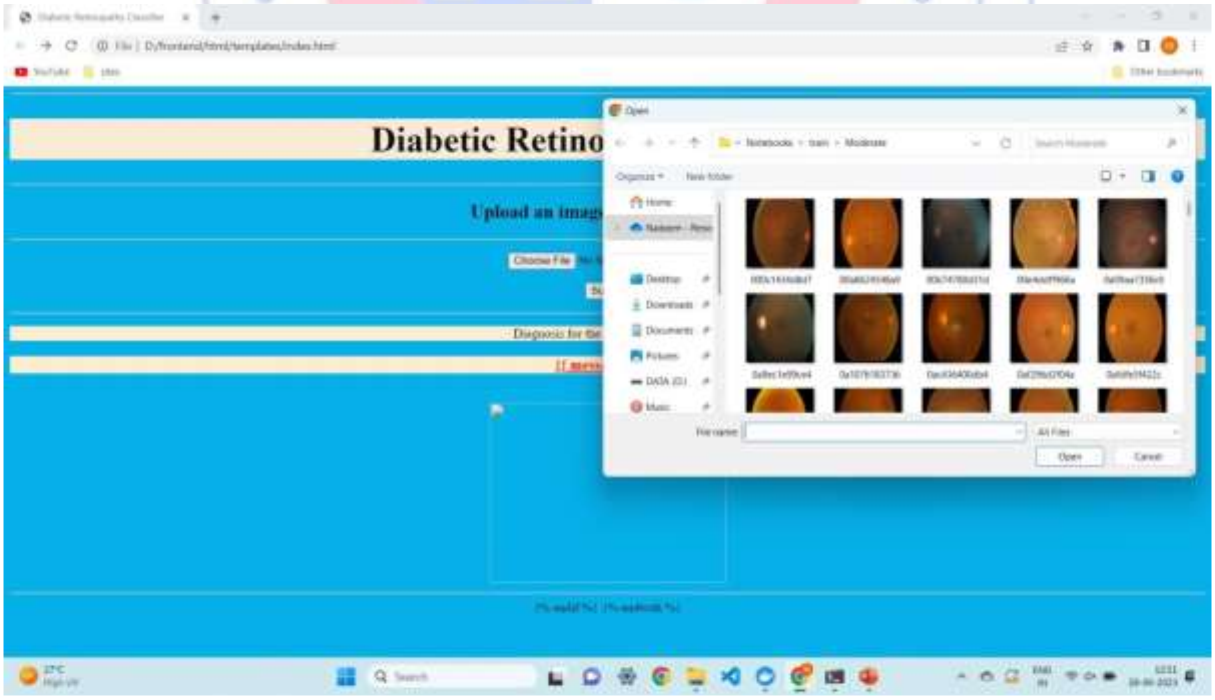
```
# Visualizing the results
import random
fig=plt.figure(figsize = (100,100))
for i in range(30):
    j = random.randint(0,len(image))
    fig.add_subplot(30, 1, i+1)
    plt.xlabel("Prediction: " + prediction[j] +"   Original: " + original[j])
    plt.imshow(image[j])
fig.tight_layout()
plt.show()
```

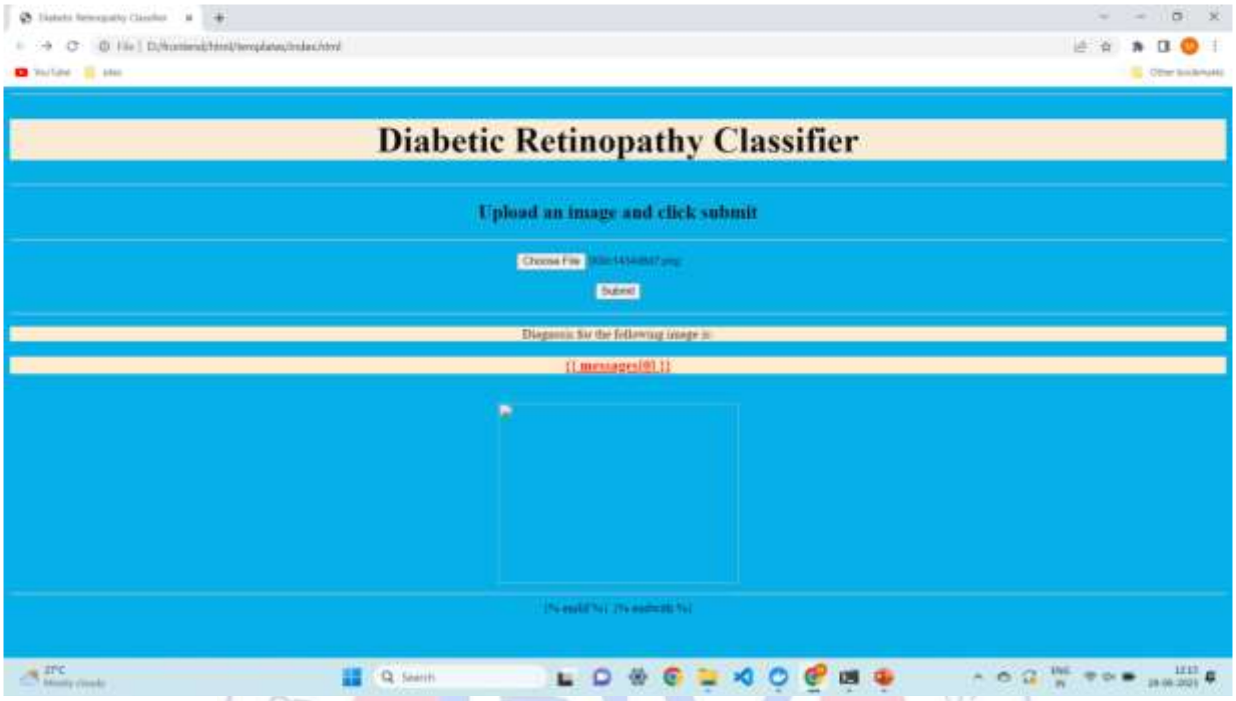
4.2 Frontend HTML page


Diabetic Retinopathy Classifier
Upload an image and click submit
Choose File No file chosen
Submit
Diagnosis for the following image is:
{{ messages[0] }}
{% endif %} {% endwith %}


Diabetic Retinopathy Classifier
Upload an image and click submit
Choose File No file chosen
Please select a file
Diagnosis for the following image is:
{{ messages[0] }}
{% endif %} {% endwith %}


Diabetic Retino
Upload an imag
Choose File
Diagnosis for the
Open
Organize
New folder
Home
Desktop
Downloads
Documents
Pictures
DATA (D:)
Music
File name
All Files
Open
Cancel


Diabetic Retino
Upload an imag
Choose File
Diagnosis for the
Open
Organize
New folder
Home
Desktop
Downloads
Documents
Pictures
DATA (D:)
Music
File name
All Files
Open
Cancel



Diabetic Retinopathy Classifier

Upload an image and click submit

Diagnosis for the following image is:

{{ messages[0] }}

{% endif %} {% endwith %}

Chapter 5

Results and Discussions

5.1 Fitting the Convolutional Neural Network and training

Once the neural network is deployed and appropriately designed using activation functions(Relu, softmax), weights, kernels and biases. The input retinal fundus images which is split into train and test sets using the library specified below from sklearn. `from sklearn.model_selection import train_test_split`

The dataset comprised of 3662 retinal fundus images.Out of which 2490 images are used for training ,439 images for validation and 733 for testing. An epoch means training the neural network with all the training data for one cycle. In an epoch, we use all of the data exactly once. A forward pass and a backward pass together are counted as one pass: An epoch is made up of one or more batches, where we use a part of the dataset to train the neural network.

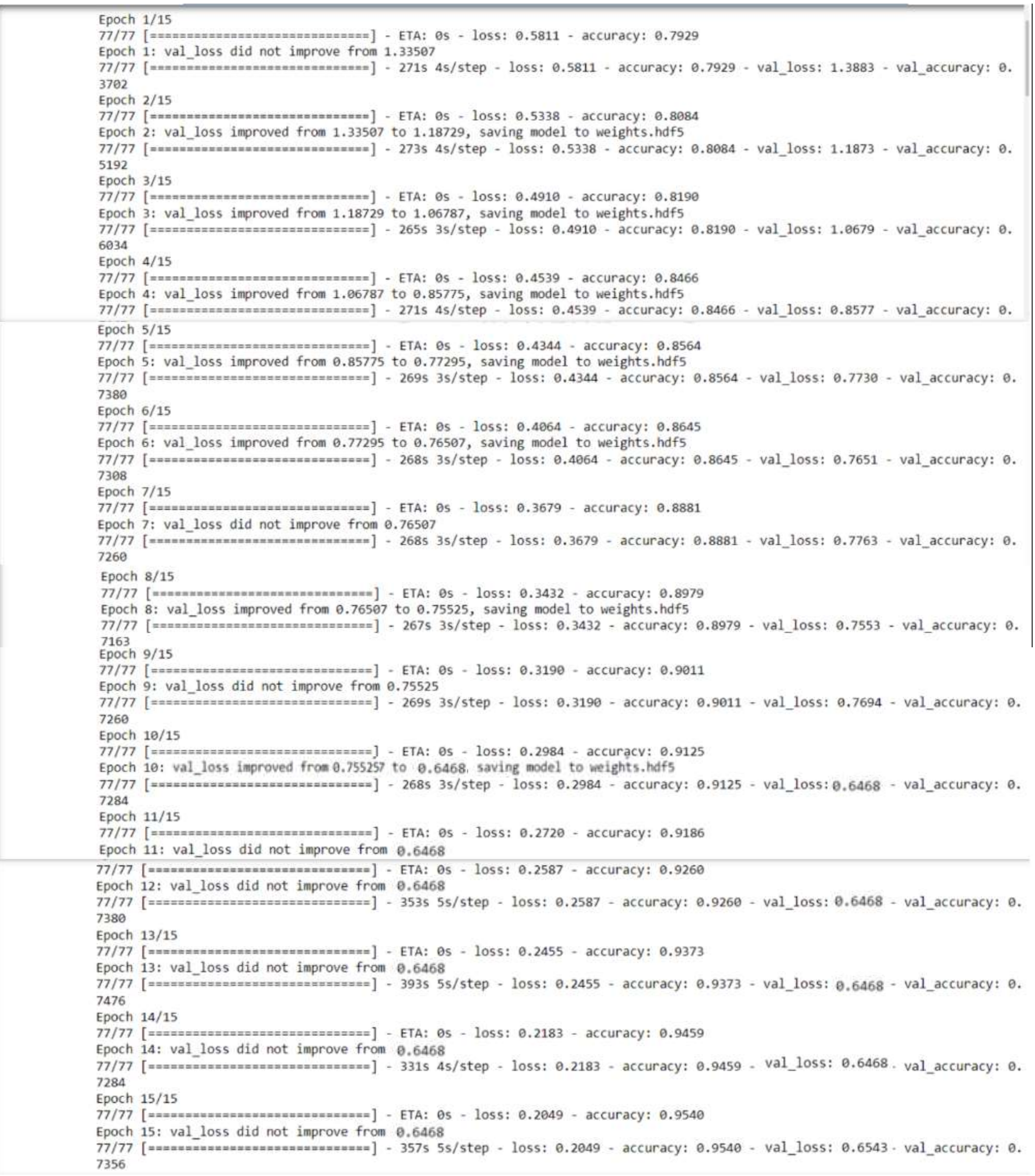
The Fig. 5.1 represents the number of iterations or Epochs which the training data was passed through. A total of 32 batches of data were trained with 15 epochs running for each batch and obtaining the accuracy of prediction and model loss/accuracy.

At the end of the compilation of all the epoch cycles, the values for the model accuracy, model loss, validation accuracy and validation loss efficiencies are obtained as depicted in the Table 5.1.

Table 5.1. Results of accuracy and loss obtained after all the epochs

Sl. No.	Attributes	Efficiency
1	Training Accuracy	95.40%
2	Training Loss	0.2049
3	Validation Accuracy	88.13%
4	Validation Loss	0.6468

The last Epoch shows us the results at the end of 15 iterations. As we can see that the result obtained is about 88.13%.



```
Epoch 1/15
77/77 [==============================] - ETA: 0s - loss: 0.5811 - accuracy: 0.7929
Epoch 1: val_loss did not improve from 1.33507
77/77 [==============================] - 271s 4s/step - loss: 0.5811 - accuracy: 0.7929 - val_loss: 1.3883 - val_accuracy: 0.
3702
Epoch 2/15
77/77 [==============================] - ETA: 0s - loss: 0.5338 - accuracy: 0.8084
Epoch 2: val_loss improved from 1.33507 to 1.18729, saving model to weights.hdf5
77/77 [==============================] - 273s 4s/step - loss: 0.5338 - accuracy: 0.8084 - val_loss: 1.1873 - val_accuracy: 0.
5192
Epoch 3/15
77/77 [==============================] - ETA: 0s - loss: 0.4910 - accuracy: 0.8190
Epoch 3: val_loss improved from 1.18729 to 1.06787, saving model to weights.hdf5
77/77 [==============================] - 265s 3s/step - loss: 0.4910 - accuracy: 0.8190 - val_loss: 1.0679 - val_accuracy: 0.
6034
Epoch 4/15
77/77 [==============================] - ETA: 0s - loss: 0.4539 - accuracy: 0.8466
Epoch 4: val_loss improved from 1.06787 to 0.85775, saving model to weights.hdf5
77/77 [==============================] - 271s 4s/step - loss: 0.4539 - accuracy: 0.8466 - val_loss: 0.8577 - val_accuracy: 0.
Epoch 5/15
77/77 [==============================] - ETA: 0s - loss: 0.4344 - accuracy: 0.8564
Epoch 5: val_loss improved from 0.85775 to 0.77295, saving model to weights.hdf5
77/77 [==============================] - 269s 3s/step - loss: 0.4344 - accuracy: 0.8564 - val_loss: 0.7730 - val_accuracy: 0.
7380
Epoch 6/15
77/77 [==============================] - ETA: 0s - loss: 0.4064 - accuracy: 0.8645
Epoch 6: val_loss improved from 0.77295 to 0.76507, saving model to weights.hdf5
77/77 [==============================] - 268s 3s/step - loss: 0.4064 - accuracy: 0.8645 - val_loss: 0.7651 - val_accuracy: 0.
7308
Epoch 7/15
77/77 [==============================] - ETA: 0s - loss: 0.3679 - accuracy: 0.8881
Epoch 7: val_loss did not improve from 0.76507
77/77 [==============================] - 268s 3s/step - loss: 0.3679 - accuracy: 0.8881 - val_loss: 0.7763 - val_accuracy: 0.
7260
Epoch 8/15
77/77 [==============================] - ETA: 0s - loss: 0.3432 - accuracy: 0.8979
Epoch 8: val_loss improved from 0.76507 to 0.75525, saving model to weights.hdf5
77/77 [==============================] - 267s 3s/step - loss: 0.3432 - accuracy: 0.8979 - val_loss: 0.7553 - val_accuracy: 0.
7163
Epoch 9/15
77/77 [==============================] - ETA: 0s - loss: 0.3190 - accuracy: 0.9011
Epoch 9: val_loss did not improve from 0.75525
77/77 [==============================] - 269s 3s/step - loss: 0.3190 - accuracy: 0.9011 - val_loss: 0.7694 - val_accuracy: 0.
7260
Epoch 10/15
77/77 [==============================] - ETA: 0s - loss: 0.2984 - accuracv: 0.9125
Epoch 10: val_loss improved from 0.755257 to 0.6468, saving model to weights.hdf5
77/77 [==============================] - 268s 3s/step - loss: 0.2984 - accuracy: 0.9125 - val_loss: 0.6468 - val_accuracy: 0.
7284
Epoch 11/15
77/77 [==============================] - ETA: 0s - loss: 0.2720 - accuracy: 0.9186
Epoch 11: val_loss did not improve from 0.6468
77/77 [==============================] - ETA: 0s - loss: 0.2587 - accuracy: 0.9260
Epoch 12: val_loss did not improve from 0.6468
77/77 [==============================] - 353s 5s/step - loss: 0.2587 - accuracy: 0.9260 - val_loss: 0.6468 - val_accuracy: 0.
7380
Epoch 13/15
77/77 [==============================] - ETA: 0s - loss: 0.2455 - accuracy: 0.9373
Epoch 13: val_loss did not improve from 0.6468
77/77 [==============================] - 393s 5s/step - loss: 0.2455 - accuracy: 0.9373 - val_loss: 0.6468 - val_accuracy: 0.
7476
Epoch 14/15
77/77 [==============================] - ETA: 0s - loss: 0.2183 - accuracy: 0.9459
Epoch 14: val_loss did not improve from 0.6468
77/77 [==============================] - 331s 4s/step - loss: 0.2183 - accuracy: 0.9459 - val_loss: 0.6468 - val_accuracy: 0.
7284
Epoch 15/15
77/77 [==============================] - ETA: 0s - loss: 0.2049 - accuracy: 0.9540
Epoch 15: val_loss did not improve from 0.6468
77/77 [==============================] - 357s 5s/step - loss: 0.2049 - accuracy: 0.9540 - val_loss: 0.6543 - val_accuracy: 0.
7356
```

Fig 5.1 Results of accuracy and loss obtained after each epoch

5.2 Graphical Outputs

i. **Model Accuracy** : Machine learning model accuracy is the measurement used to determine which model is best at identifying relationships and pattern between variables in a dataset based on the input, or training, data. The better a model can generalize to ‘unseen’ data, the better predictions and insights it can produce, which in turn deliver more business value.

ii. **Model loss:** The loss function in a neural network quantifies the difference between the expected outcome and the outcome produced by the machine learning model. From the loss function, we can derive the gradients which are used to update the weights. The average over all losses constitutes the cost.

iii. **Validation Accuracy:** It is the sum of errors made for each example in training or validation sets. Loss value implies how poorly or well a model behaves after each iteration of optimization. An accuracy metric is used to measure the algorithm's performance in an interpretable way.

iv. **Validation Loss:** "Validation loss" is the loss calculated on the validation set, when the data is split to train / validation / test sets using cross-validation.

When comparing test accuracy to training accuracy, test accuracy refers to how well the trained model recognises independent images that were not utilised in training. A

Accuracy = Number of correct predictions/ Total number predictions …Eq 5.1

Loss is measured with the help of the loss function.

The difference between the outcome produced by the machine learning model and the intended outcome is measured by the loss function in a neural network.

5.2.1 Model Accuracy

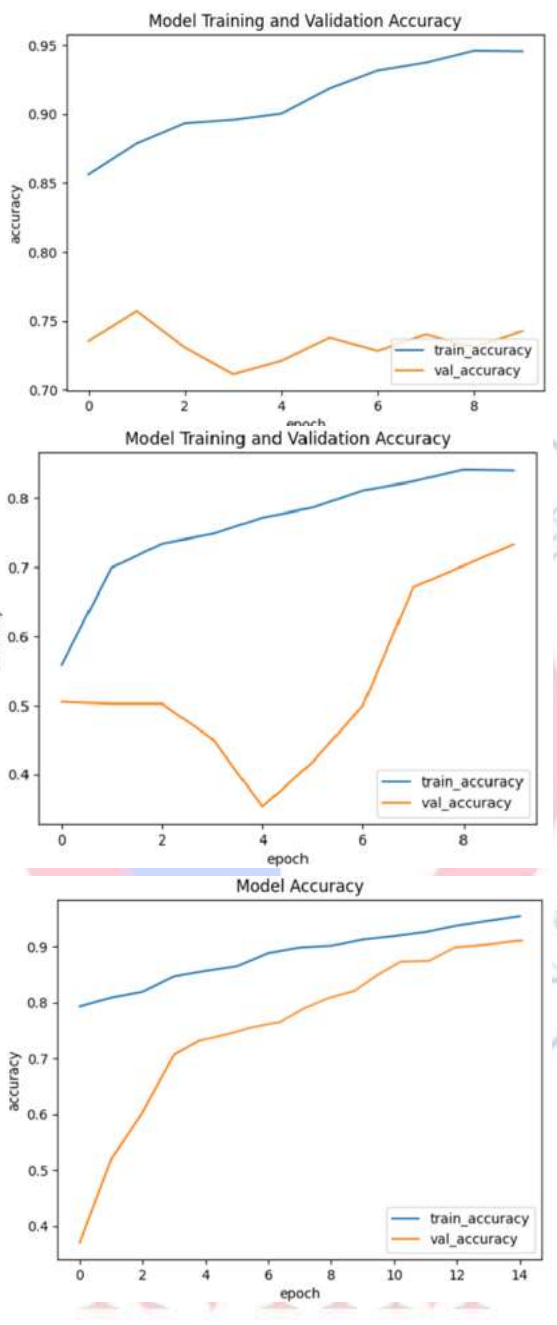


Fig 5.2 Graph for Model accuracy of the ML model for different iterations

The model accuracy as depicted, has 2 main features, curve 1 which is for the training set represented in blue and curve 2 which is the test set represented in yellow. As it can be seen that both the curves increase exponentially which denotes that the accuracy of our model increases periodically with every passing epoch. By comparing the model predictions with the actual values in terms of a percentage, it determines how well our model predicts. The graphs in Fig 5.2 shows the plot of training and validation accuracy when the patience parameter was set to 5, 15, 25 respectively.

5.2.2 Model Loss

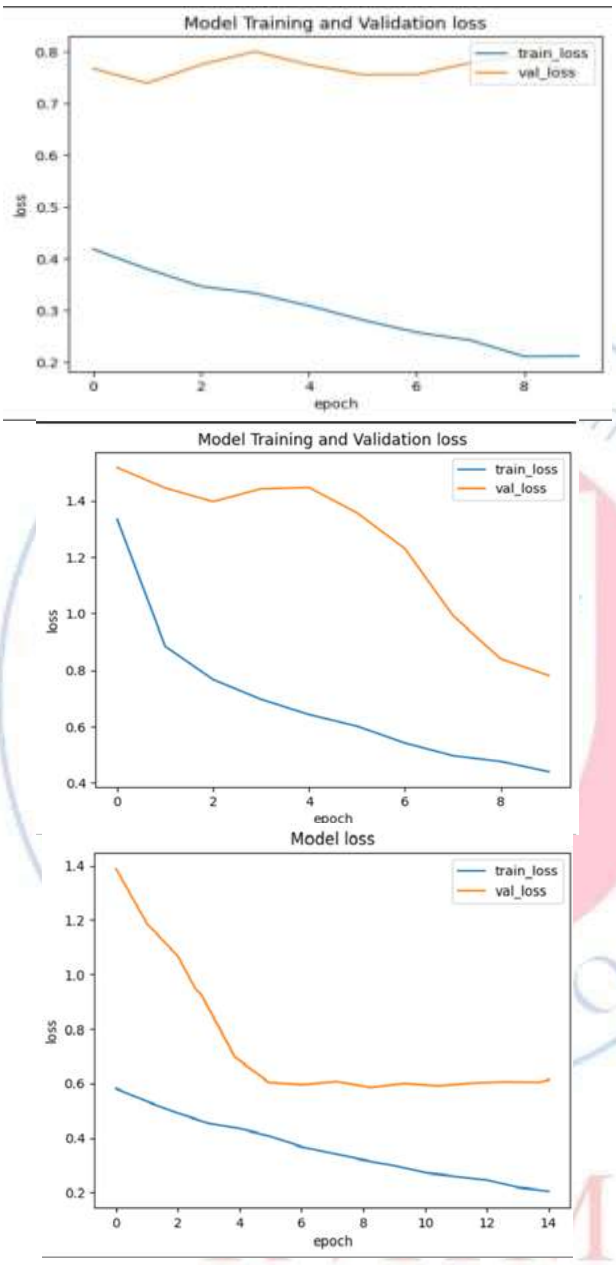


Fig 5.3 Graph for Model loss of ML model for different iterations

The model loss as depicted, has 2 main features, curve 1 which is for the training set represented in blue and curve 2 which is the test set represented in yellow. As it can be seen in Fig. 5.3 both the curves decrease with every passing epoch. This denotes that the model and validation loss of our model decreases from time to time and hence reduces erroneous outputs. The graphs in Fig 5.3 shows the plot of training and validation loss when the patience parameter was set to 5, 15, 25 respectively.

5.3 Confusion Matrix

By displaying the true and false predictions for each class, the confusion matrix goes beyond classification accuracy. A confusion matrix in the context of a binary classification job is a 2x2 matrix. It is a 5x5 matrix if there are five separate classes, and so on. We won't receive the TP, TN, FP, and FN values immediately in the multi-class classification problem as we would in the binary classification problem. For each class, a calculation is required.

FN: A class's False-negative value is the sum of the values in the relevant rows, excluding the TP value. FP: A class's False-positive value is the total of all of the values in the relevant column, excluding the TP value. TN: The total of all columns and rows, excluding those for the class for which we are computing the values, will represent the True Negative value for a given class. TP: When the actual value and the anticipated value are the same, that is a true positive value.

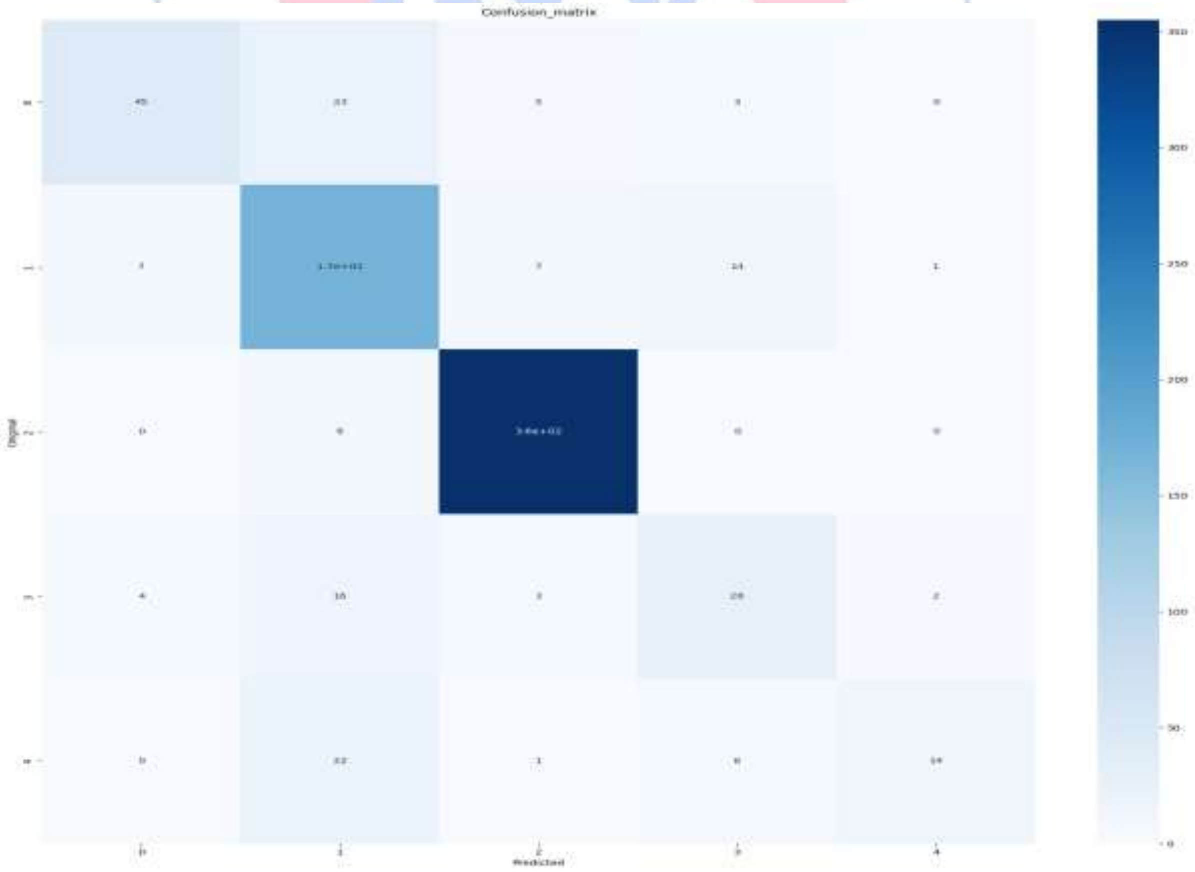
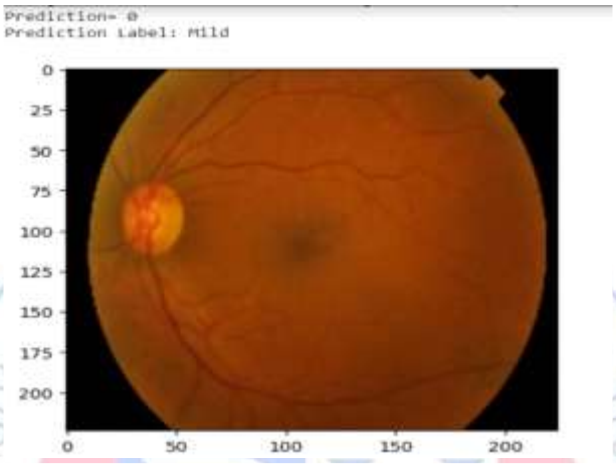


Fig 5.4 Confusion matrix of the model

5.4 Prediction of stage of diabetic retinopathy

The figures shown in this section depicts the expected outputs of the test data set for the five cases:0)mild DR 1)moderate DR 2)no DR 3) proliferative DR 4)severe DR.

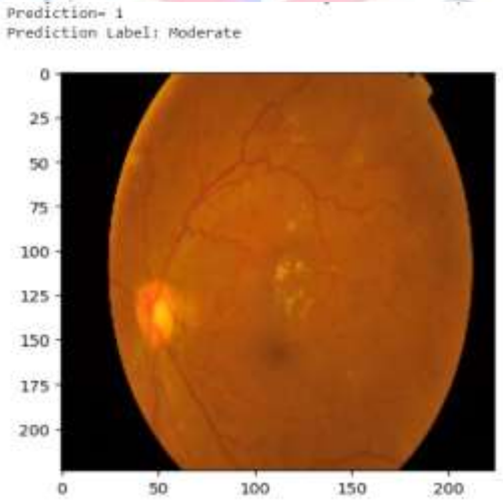


```
Prediction= 0
Prediction Label: Mild
```

Prediction: mild original: mild

Fig. 5.5 Output image for mild DR prediction vs original

The Fig. 5.5 shows the output for the test case image which is rightly predicted as mild DR. Once the input image of the test set is passed to the output variable, the image is processed through all the convolutional network layers and gives the correct prediction at the end of the process. It also mentions the class 0 to which it belongs.



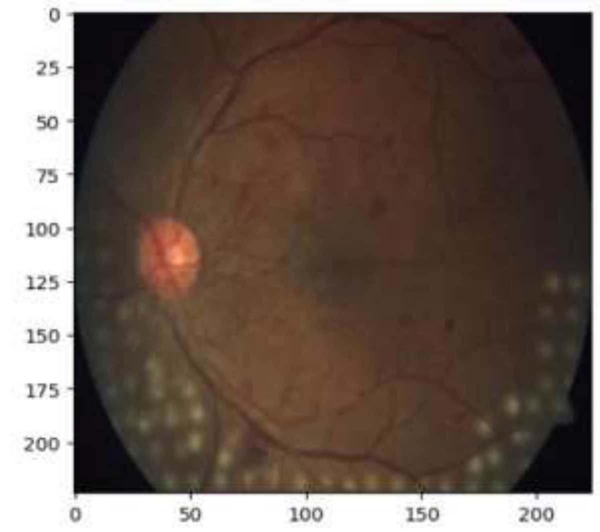
```
Prediction= 1
Prediction Label: Moderate
```

Prediction: moderate original: moderate

Fig. 5.6 Output image for moderate DR prediction vs original

The Fig. 5.6 shows the output for the test case image which is rightly predicted as moderate DR. Once the input image of the test set is passed to the output variable, the image is processed through all the convolutional network layers and gives the correct prediction at the end of the process. It also mentions the class 1 to which it belongs.

```
Prediction= 2
Prediction Label: No_DR
```

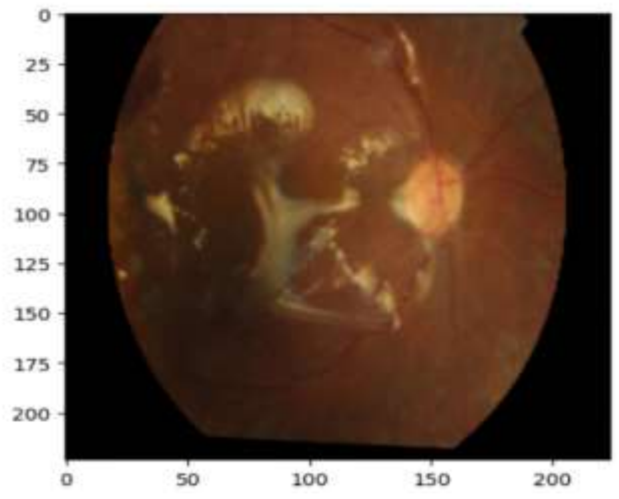


Prediction: No DR original: No DR

Fig. 5.7 Output image for no DR prediction vs original

The Fig 5.7 shows the output for the test case image which is rightly prediction as no DR. The class for the image is 3.

```
Prediction= 3
Prediction Label: Proliferate_DR
```

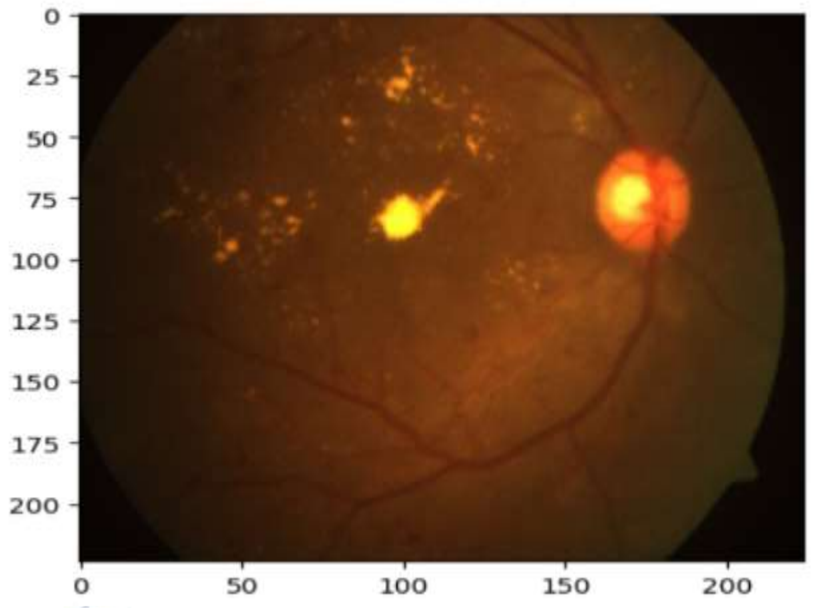


Prediction: proliferative original: proliferative

Fig. 5.8 Output image for proliferative DR prediction vs original

The Fig 5.8 shows the output for the test case image which is rightly predicted as proliferative DR. The class for the image is 4.

```
Prediction= 4
Prediction Label: Severe
```



Prediction: severe   original: severe

Fig. 5.9 Output image for severe DR prediction vs original

The Fig 5.9 shows the output for the test case image which is rightly predicted as severe DR. The class for the image is 5.

Chapter 6

CONCLUSION AND FUTURE SCOPE

Our project aimed at giving a suitable solution to aid the process of predicting the correct stage of Diabetic Retinopathy by obtaining inputs from the fundus images. Hence, we have come up with a user-friendly approach in which we have trained our model in different supervised classification algorithms to improve its accuracy.

We have been able to increase our accuracy to 95.40% which is a very efficient number to rely on. We have also optimized the number of epochs and training cycles for our input fundus images which reduces the error and improves the model’s overall performance. Our main objective for taking up this project has been achieved which was of making a user-friendly interface where the user gives the input fundus images and our model predicts the stage of the DR.

With the advent of India’s own electronic health record system (EHR). EHR will facilitate automatic screening and diagnosis of diabetic retinopathy. This can assist medical professionals in identifying at-risk individuals and initiating quick actions that improve patient outcomes.

India's very own telemedicine program eSanjeevani will play a major role in our project as. The created project may be connected with telemedicine platforms, allowing for remote diabetic retinopathy screening and diagnosis. For patients in rural and underdeveloped areas in particular, this may boost access to affordable healthcare. By enhancing the model, it is possible to diagnose diabetic retinopathy in real-time, which would increase screening effectiveness and reduce the burden on healthcare providers. The project can be further extended to be able to directly be applied to a fundus camera where after taking an image of the retina of the patient , the algorithm can easily detect to which stage of DR it is. Thereby reducing the time spent on diagnosis of a patient and result in more efficiency and accuracy.

REFERENCES

[1] Dolly Das, Saroj Kumar Biswas, and Sivaji Bandyopadhyay " Detection of Diabetic Retinopathy using Convolutional Neural Networks for Feature Extraction and Classification (DRFEC), " *IEEE Trans Geosci Remote Sens,* 2022 Nov 29.

[2] Yasashvini R.,Vergin Raja Sarobin M. ,Rukmani Panjanathan, Graceline Jasmine S.,Jani Anbarasi L.ORCID " Diabetic Retinopathy Classification Using CNN and Hybrid Deep Convolutional Neural Networks ," *42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC),*16 September 2022.

[3] Jaichandran R, Vithiavathi Sivasubramanian, Jaya Prakash ,Varshni M "Detection of Diabetic Retinopathy Using Convolutional Neural Networks," *2022 IEEE International Conference on Information Reuse and Integration (IRI). IEEE,* 18 may 2022.

[4] Mohamed Shaban,Zeliha Ogur,Ali Mahmoud,Andrew Switala,Ahmed Shalaby,Hadil Abu Khalifeh,Mohammed Ghazal,Luay Fraiwan,Guruprasad Giridharan,Harpal Sandhu,Ayman S. El-Baz "A convolutional neural network for the screening and staging of diabetic retinopathy," *39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE* June 22, 2020.

[5] Saleh Albahli, Ghulam Nabi Ahmad Hassan Yar " Detection of Diabetic Retinopathy Using Custom CNN to Segment the Lesions," *Int. J. Eng. Adv. Technol., vol. 8, no. 6, 25 November 2021.*

[6] Chun-Ling Lin, Kun-Chi Wu "Development of revised ResNet-50 for diabetic retinopathy detection," *PLoS One, vol. 14, no. 6, pp. 1–11*, 2023 Apr 19.

[7] Chun-Ling Lin ,Kun-Chi Wu "Development of Revised ResNet-50 for Diabetic Retinopathy Detection," *Comput. Methods Programs Biomed., vol. 153* ,January 4th, 2023.

[8] Muhammad Kashif Yaqoob ,Syed Farooq Ali,Muhammad Bilal ,Muhammad Shehzad Hanif ,Ubaid M. Al-Saggaf " ResNet Based Deep Features and Random Forest Classifier for Diabetic Retinopathy Detection, " *International Conference on Computational Intelligence 36, 1480–1492*, 4 June 2021.

[9] Yijin Huanga, Li Lina,c, Pujin Chenga , Junyan Lyua,d, Roger Tamb, Xiaoying Tanga " Identifying the key components in ResNet-50 for diabetic retinopathy grading from fundus images: a systematic investigation, " *International Conference on Expert System Application,*17 Oct 2022.

[10] Rajesh K, Santhanam A, Sridhar M, Dr. J. Mohan " Diabetic Retinopathy Detection using Convolutional Neural Network, " *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) Volume 2,3,* May 2022.

[11] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," *in Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 246–252, IEEE, Collins, Colorado,* June 2021

[12] O. Perdomo, S. Otalora, F. Rodríguez, J. Arevalo, and F. A. González, “A novel machine learning model based on exudate localization to detect diabetic macularedema,” *In Proceedings of the Ophthalmic Medical Image Analysis Third InternationalWorkshop, Athens, Greece*, October 2021

[13] Yashal Shakti Kanungo, Bhargav Srinivasan, Dr. Savita Choudhary, “Detecting Diabetic Retinopathy using Deep Learning” *International Conference on Expert System Application*,2021.

[14] Mahmut Karakaya* and Recep E. Hacisoftaoglu, “Comparison of smartphone-based retinal imaging systems for diabetic retinopathy detection using deep learning” *24th National and 2nd International Iranian Conference on Biomedical Engineering (ICBME) (pp. 1-6).IEEE,* 2020.

[15] R.Bhargavi (316126510107), R. Geetha Nalini (316126510106), K.Poorna Subhash (316126510089), K. Sainadh, “Diagnosis Of Diabetic Retinopathy Using Transfer Learning” *IEEE Access, 7,pp.3360-3370,*2020.

[16] D. Y. Carson Lam, “Automated detection of diabetic retinopathy using deep learning,” *AMIA Summits on Translational Science Proceedings, vol. 2018, 147 pages,* 2020

[17] F. Zabihollahy, A. Lochbihler, and E. Ukwatta, “Deep learning based approach for fully automated detection and segmentation of hard exudate from retinal images,” *Proceedings of the Medical Imaging 2019: Biomedical Applications in Molecular, Structural, and Functional Imaging vol.2020, Springer, San Diego, CA, USA,* January 2020

[18] Kumar, S. and Kumar, B., 2020, February. Diabetic Retinopathy Detection by Extracting Area and Number of Microaneurysm from Colour Fundus Image. *In 2020 5th International Conference on Signal Processing and Integrated Networks (SPIN) (pp. 359-364). IEEE.*

[19] Asiri, N., Hussain, M. and Abualsamh, H.A., 2022. Deep Learning based Computer-Aided Diagnosis Systems for Diabetic Retinopathy: *A Survey. arXiv preprintarXiv:1811.01238.*

[20] Gulshan, V., Peng, L., Coram, M., Stumpe, M.C., Wu, D., Narayanaswamy, A., Venugopalan, S., Widner, K., Madams, T., Cuadros, J. and Kim, R., 2022. Development and validation of a deep learning algorithm for detection of diabetic retinopathy *in retinal fundus photographs. Jama, 316(22),pp.2402-2410.*

[21] Mohammadian, S., Karsaz, A. and Roshan, Y.M., 2020, November. Comparative Study of Fine-Tuning of Pre-Trained Convolutional Neural Networks for Diabetic Retinopathy Screening. *In 2020 24th National and 2nd International Iranian Conference on Biomedical Engineering (ICBME) (pp. 1-6).IEEE.*

[22] 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.*

[23] Mansour, R.F., 2018. Deep-learning-based automatic computer-aided diagnosis system for diabetic retinopathy. *Biomedical engineering letters*, *8(1), pp.41-57*.

[24] Dutta, S., Manideep, B.C., Basha, S.M., Caytiles, R.D. and Iyengar, N.C.S., 2021. Classification of Diabetic Retinopathy Images by Using Deep Learning Models. *International Journal of Grid and Distributed Computing, 11(1), pp.89-106.*

[25] Gao, Z., Li, J., Guo, J., Chen, Y., Yi, Z. and Zhong, J., 2020. Diagnosis of Diabetic Retinopathy Using Deep Neural Networks. *IEEE Access*, *7,pp.3360-3370*.

[26] B. Harangi and A. Hajdu, "Detection of exudates in fundus images using a Markovian segmentation model," *in 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 130–133, IEEE, Chicago, IL, USA,* November 2022.

[27] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking*," in Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 246–252, IEEE, Collins, Colorado,* June 2021.