PROJECT REPORT

DEEP LEARNING FUNDUS IMAGE ANALYSIS FOR EARLY DETECTION OF DIABETIC RETINOPATHY

TEAM ID: PNT2022TMID29261

SUBMITTED BY:

BALAJI V (422519104008) [TEAM LEADER]

THAPASHVIN R (422519104048)

SRIRAM K (422519104041)

VIGNESH B (422519104049)

INDEX

1) INTRODUCTION

Diabetic Retinopathy (DB) is a complication of diabetes that affects the eyes. It is usually caused by damage of blood vessels in the tissue at retina (back layer of the eye). Early symptoms may include blurriness, floaters, dark or empty areas in vision and difficulty perceiving colour blindness. It requires constant monitoring and if complications develop, it can reduce life expectancy. It can cause blindness if it is not diagnosed and untreated. Still now there is no cure for the medical medication. While the treatment can stop or slow down the progression of Diabetic Retinopathy. Mild cases may be carefully treated with diabetes management.

1.1Project Overview

Diabetic retinopathy is the most common microvascular complications in diabetes, for the screening of which the retinal imaging is the most widely used method due to its high sensitivity in detecting retinopathy. The evaluation of the severity and degree of retinopathy associated with a person having diabetes, is currently performed by medical experts based on the fundus or retinal image of the patients' eyes. As the number of patients with diabetes is rapidly increasing, the number of retinal images produced by the screening programmes, it will also increase, which in turn introduces a large labour-intensive burden on the medical experts as well as cost to the health care services. This could be alleviated with an automated system either as support for medical experts' work or as full diagnosis tool. There are two recent studies have investigated the use of deep learning system in automated detection of diabetic retinopathy.

1.2Purpose

The Proposed work intends to automate the detection and classification of diabetic retinopathy from retinal fundus image which is very important in ophthalmology. Most of the existing methods use handcrafted features and those are fed to the classifier for detection and classification purpose. Recently convolutional neural network (CNN) is used for this classification problem but the architecture of CNN is manually designed. In this work, a genetic algorithm-based technique is proposed to automatically determine the parameters of CNN and then the network is used for classification of diabetic retinopathy.

The proposed CNN model consists of serious of convolution and pooling layer used for feature extraction. Finally, support vector machine (SVM) is used for classification. Hyper parameters like number of convolutions and pooling layer, number of kernel and kernel size of convolution layer are determined by using the genetic algorithm. The proposed methodology is tested on publicly available messidor dataset. Experimental result shows that proposed auto-tuned CNN performs significantly better than the existing method. Use of CNN takes away the burden of designing the image features and on the other hand genetic algorithm-based methodology automates the design of CNN hyper-parameters.

2) LITERATURE SURVEY

2.1) EXISTING PROBLEM

Diabetic Retinopathy (DR) is a degenerative disease that impacts the eyes and is a consequence of diabetic's mellitus, where high blood glucose levels induce lesions on the eye retina. Diabetic Retinopathy is regarded as the leading cause of blindness for diabetic patients especially the working age population in developing nations. Treatment involves sustaining the patient's current grade of vision since the disease is irreversible. Early detection of diabetic retinopathy is crucial in order to sustain the patient's vision effectively. The main issue involved with DR detection is that manual diagnosis progress is vey time, money, effort consuming and involves an ophthalmologist's examination of eye retinal fundus images. The late also proves to be more difficult, particularly in the early stages of the disease when the disease features are less prominent in accessing retinal fundus images and utilization of deep leaning algorithm has aided the early diagnosis of diabetic retinopathy (DR). This paper reviews and analysis state of the art deep learning methods in supervised, self-supervised and vision transformer setup proposing retinal fundus image classification and detection.

For instances, referable, non-referable and proliferative classifications of diabetic retinopathy are reviewed and summarized. More over the paper discusses the available retinal fundus data sets for diabetic retinopathy that are used for the tasks such as detection, classification and segmentation. The paper also accesses research gaps in the area of diabetic retinopathy detection or classification and addresses various challenges that need further study and investigation.

SURVEY:

[1] The proposed method uses Alex net Convolutional Neural Network (CNN) to detect the diabetes on a fundus image. The dataset used was from the MESSIDOR database and it contains 1200 fundus images and they were filtered for the project into 580 images of normal and exudates images. On the CNN process the dataset has been divided into two which is training and testing dataset. This method gives more than 90% accuracy on 50% of the training dataset and the remaining 50% dataset is used for testing purpose. The testing gives about 85% accuracy.

Advantages: The detection of diabetic retinopathy has good accuracy and the CNN have been used to detect it (which is a widely used method in medical image analysis and classification).

Limitations: The dataset was insufficient to train the neural network and only 580 images were used for both training and testing even though they received a good accuracy. It also found it difficult to detect smaller exudates on the image.

[2] The Proposed system developed a CNN architecture to classify the diabetic retinopathy as 5 classes namely No DR, Mild DR, Moderate DR, Severe DR, Proliferative DR respectively in the fundus imagery. They have analysed the past works done using CNN to detect DR and they have corrected the networks in CNN to make it more accurate and efficient. On the dataset containing 80000 images they have achieved an accuracy of 75%.

Advantages: A larger dataset has been used to train the CNN and overfitting issue is solved. They have used a 5-class problem to classify DR. Healthy eye have been detected correctly.

Limitations: There is some issues in classification to distinguish between the mild, moderate and severe cases of DR.

[3] They have used a Deep Convolutional Neural Network (DCNN) to analyse the fundus image and predict the stage such as No DR, Moderate DR (a combination of mild and moderate Non-Proliferative DR), Severe DR (severe NPDR, and Proliferative DR). They have almost used 3468 fundus images taken in multiple clinics available in Kaggle over a period of time. They have a achieved an accuracy of more than 80 percent.

Advantages: It has achieved a competitive accuracy, sensitivity and specificity when compared to other CNN based techniques.

Disadvantages: Overfitting issue arises when the model is trained with limited dataset and fails when applied on a new dataset.

[4] The proposed model is based on the DenseNet121 architecture. A speciality about this is, each output of a convolution layer (feature map) is concatenated with the subsequent layers of the same block. It classifies DR as 5 classes based on the level of DR which comprises of No DR, Slight DR, Medium DR, Severe DR and PDR. The proposed method has used 2 datasets (Messidor and APTOS) by a cross-testing approach so that the model can acquire the complex features.

Advantages: The model was designed to detect DR at early stages.

Limitations: As they have used cross-testing approach with unbalanced data the accuracy is low compared to the state of art methods. Also, the model had difficulty to classify the Slight NDPR class.

2.2) Problem Statement

Diabetic Retinopathy (DR) is common complication of diabetes mellitus, which will cause lesions on the retina that affects vision. If it is not detected early, it can lead to blindness. Unfortunately, DR is not a reversible proves, and the given treatment will only give us a sustain vision. DR early detection and treatment can significantly reduce the risk of vision loss

WHAT? In contrast to computer-aided diagnosis systems, the manual / human-based diagnosis process of DR retina fundus images by doctors (ophthalmologists) is time-consuming, labour-intensive, expensive, and prone to error.

WHY? Diabetes-related retinopathy is brought on by high blood sugar levels harming the eye's iris. which could result in a permanent loss of vision.

WHEN? Early on, the DR has no symptoms, but later on, the vessels may start to leak a tiny amount of blood into your retina.

WHERE? Blurred vision, Distorted vision will occur.

WHO? It is common among the Diabetic patients.

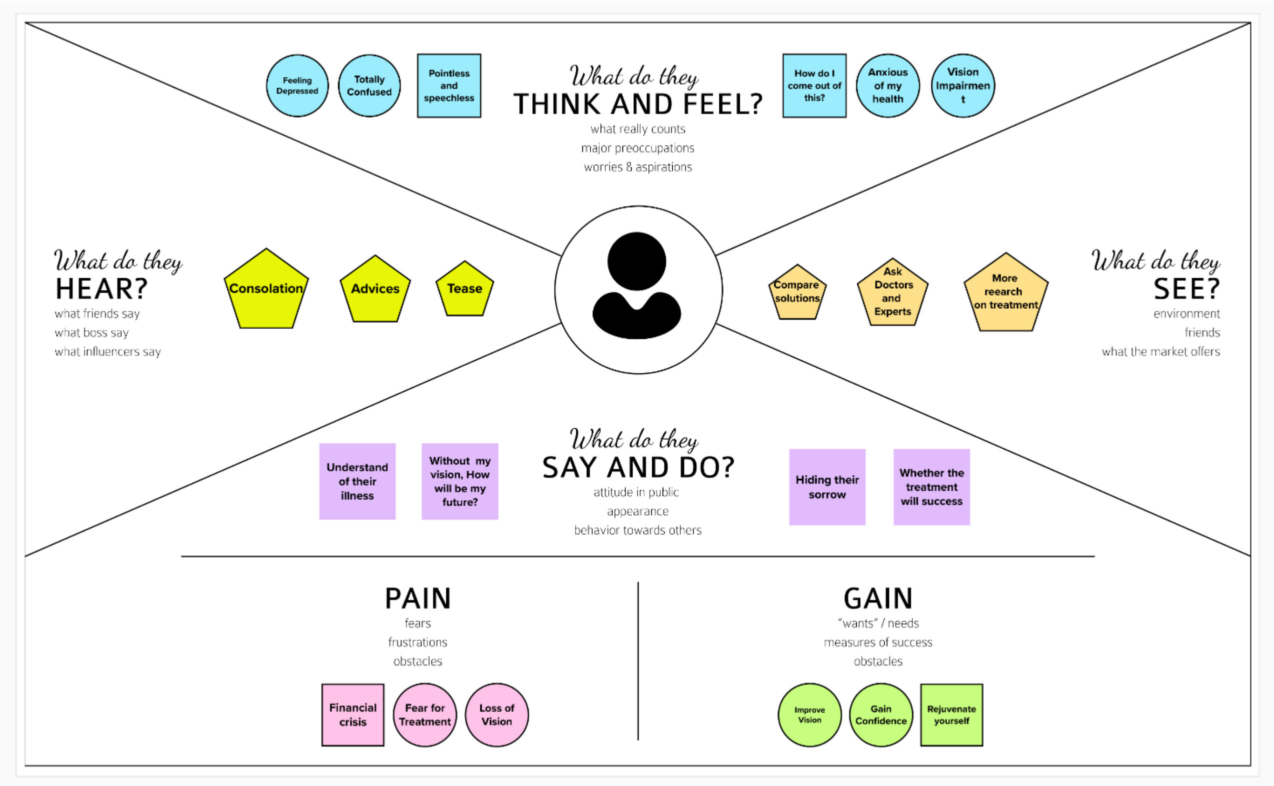
HOW? The manual early detection of this DR is a challenging task.

OBJECTIVES:

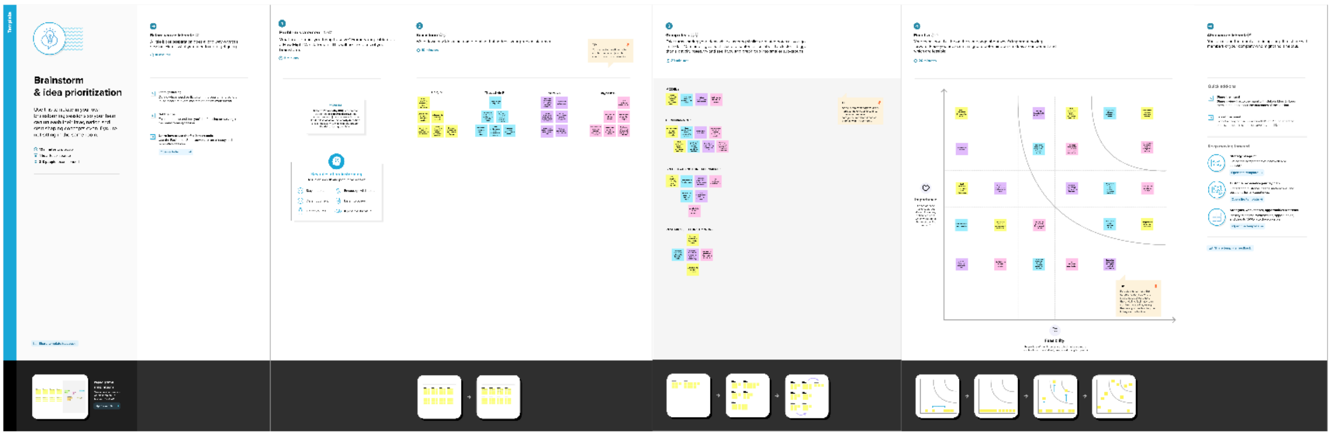
The primary goal is to identify diabetic retinopathy by processing retinal images. Transfer learning has arisen as one of the most popular techniques that has enhanced performance in many areas, notably in the analysis and classification of medical images. We used transfer learning techniques that are more frequently used in medical image analysis and have been extremely effective, including such Inception V3, Resnet50, and Xception V3.

3) IDEATION PHASE

3.1) Empathy Map



3.2) Ideation and Brainstorming:

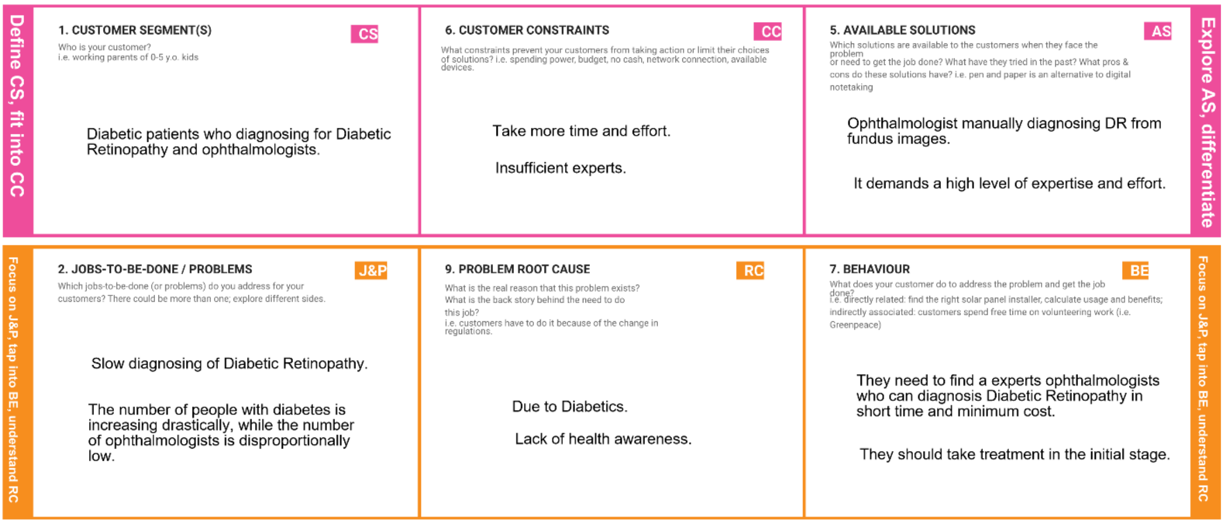


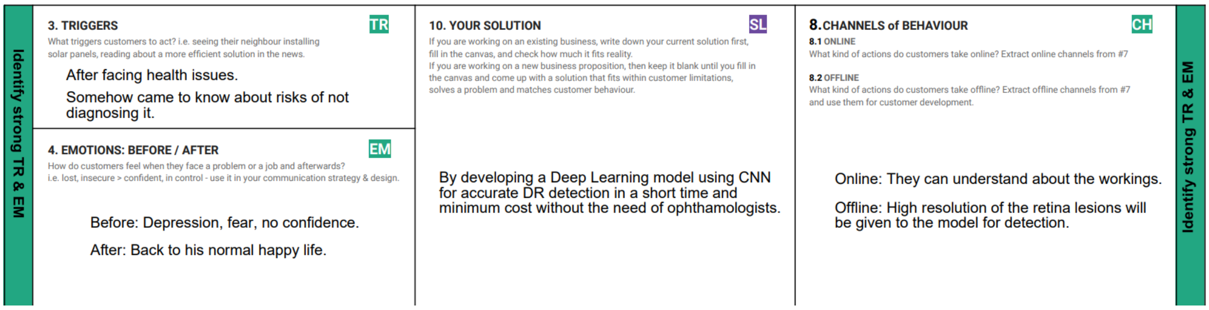
3.3) Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Analysis of fundus image for early detection of Diabetic Retinopathy. • Analyse the level of DR • To detect whether DR is present or not
2.	Idea / Solution description	○ The idea or the solution is to detect the Diabetic Retinopathy from the fundus image dataset as early as possible so that peoples/patients can proceed to their required treatments and prevent vision impairment or permanent vision loss. ○ As there is no complete cure for this DR, we are going to develop a deep learning model (CNN) with good accuracy to detect DR and save peoples in risk of losing their vision.
3.	Novelty / Uniqueness	A class-based classifier is to be given on basis of the level of DR done in analysis. We are also going to try transfer learning approach as a part of the work which can really be effective and achieve better performance.
4.	Social Impact / Customer Satisfaction	This can really save the life of people by regaining their vision. Analysing and Detecting DR at the early stage can help to prevent the patient from losing their vision so it plays a crucial role in social impact.

5.	Business Model (Revenue Model)	By using this model doctors can analyse and detect DR in which it acts as a business model for private hospitals and service model for government hospitals. Even it can serve as a business model by exporting it to other countries who are in need of this.
6.	Scalability of the Solution	There are more and more ways for the scalability of the solution in which the model can be easily integrated & adapted with future technologies.

3.4) Proposed Solution Fit





4) REQUIREMENT ANALYSIS:

4.1) Functional Requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Identify and selecting dataset	It is necessary to select the appropriate dataset to enhance the model's performance.
FR-2	Training	It is required to import the libraries needed for the training of the model.
FR-3	Diagnosis	The training should ensure proper diagnosis and make sure to identify the true and false of the medical condition [Diabetic Retinopathy].
FR-4	Analysis	Based on the training the model should analyse the medical condition [DR] in order to predict/detect the disease accurately.
FR-5	Testing	The trained model is tested with different data to ensure it has trained well to predict/detect the medical condition [DR].
FR-6	Reporting	The result of the experiment gives the medical report of the disease [DR] so that the patient can understand the level of the disease.
FR-7	Treatment	The testing of the model gives us the level of the medical condition so that we can go for the required treatment.

4.2) Non-Functional Requirements

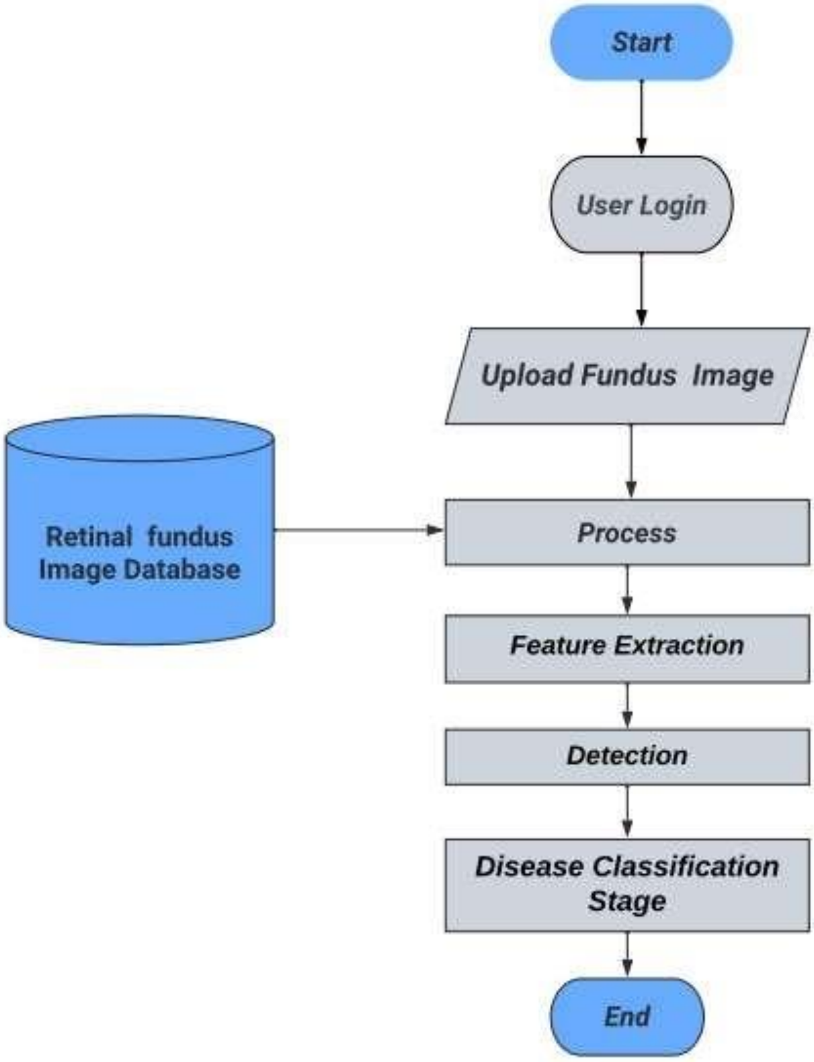
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	User with basic understanding of the medical condition and computer knowledge can operate the system.
NFR-2	**Reliability**	There is a chance of hardware failure or false positives when the testing data is more of different than the training dataset.
NFR-3	**Performance**	The performance of the model is meant to give speedy results for the patients.
NFR-4	**Availability**	The model is made to be available at anytime and anywhere.
NFR-5	**Scalability**	The scalability of the model can be enhanced with future technologies so that the performance of the model can be improved and might affect the reliability when the data given for testing is increased.

5) PROJECT DESIGN

5.1) Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2) Solution and technical Architecture

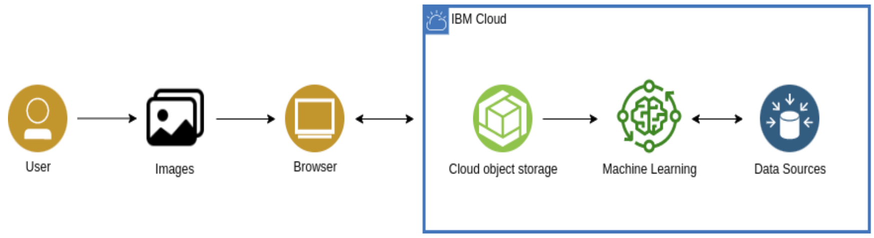


Table1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript etc.
2.	Application Logic-1	Logic for a process in the application	Python, Flask
3.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
4.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
5.	Machine Learning Model	Purpose of Machine Learning Model	Diabetic Retinopathy detection
6.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration:	Cloud.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Flask, TensorFlow. Keras. Numpy, Pandas
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	Built-in protection.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	3-tiers.
4.	Availability	Justify the availability of applications (e.g. use of load balancers, distributed servers etc.)	Load balancer.
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	It depends upon the input images.

5.3) User Stories:

User type	Functional Requirement	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-2
	Login	USN-2	As a user, I can log into the application by entering email & password	I can login the application by email and access the dashboard	High	Sprint-2
	Dashboard	USN-3	As a user, I can navigate through various sections of the application.	I can navigate the sections of the application	High	Sprint-2
	Upload image	USN-4	As a user, I can upload Fundus images.	I can get a result	High	Sprint-2

6) PROJECT PLANNING AND SCHEDULING

6.1) Sprint Planning and Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-2	Registration	USN-1	As a user, I can register for the application by entering my email, and password, and confirming my password.	10	High	THAPASHVIN R SRIRAM K
Sprint-2	Storing credentials in database	Task-1	As a developer, I will store all user credentials safely in the database.	10	High	VIGNESH B BALAJI V
Sprint-2	Login	USN-2	As a user, I can log into the application by entering my email & password	5	High	THAPASHVIN R SRIRAM K
Sprint-2	Upload Images	USN-3	As a user,I should be able to upload the image	10	High	BALAJI V
Sprint-2	Dashboard	USN-4	As a user, based on my requirement I can navigate through the dashboard.	5	Medium	THAPASHVIN R SRIRAM K
Sprint-1	Dataset	Task-2	As a developer, dataset should be preprocessed enough to train the model.	10	High	VIGNESH B

6.2) Sprint Planning and Estimation:

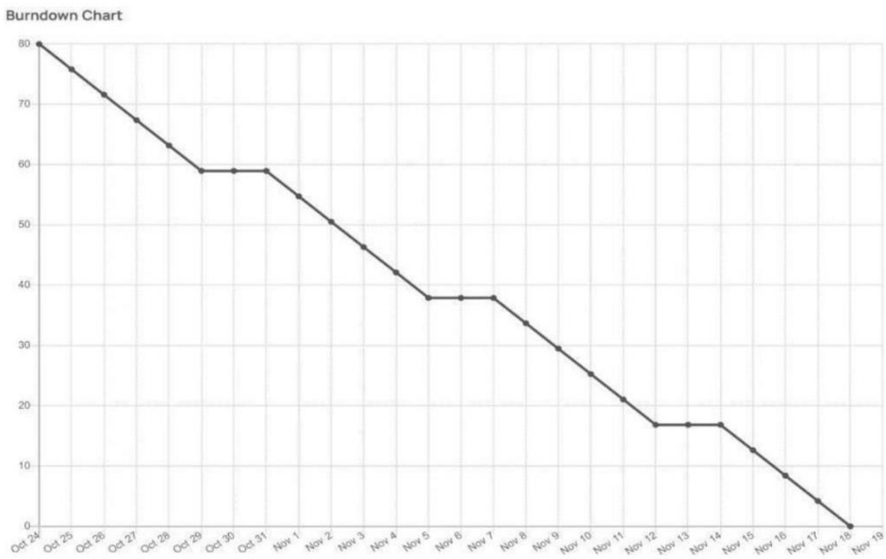
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	7 Days	24 Oct 2022	30 Oct 2022	16	14 Nov 2022
Sprint-2	20	7 Days	31 Oct 2022	06 Nov 2022	16	18 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	16	18 Nov 2022
Sprint-4	20	7 Days	13 Nov 2022	19 Nov 2022	20	19 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint).
Let's calculate the team's average velocity
(AV) per iteration unit (story points per day)

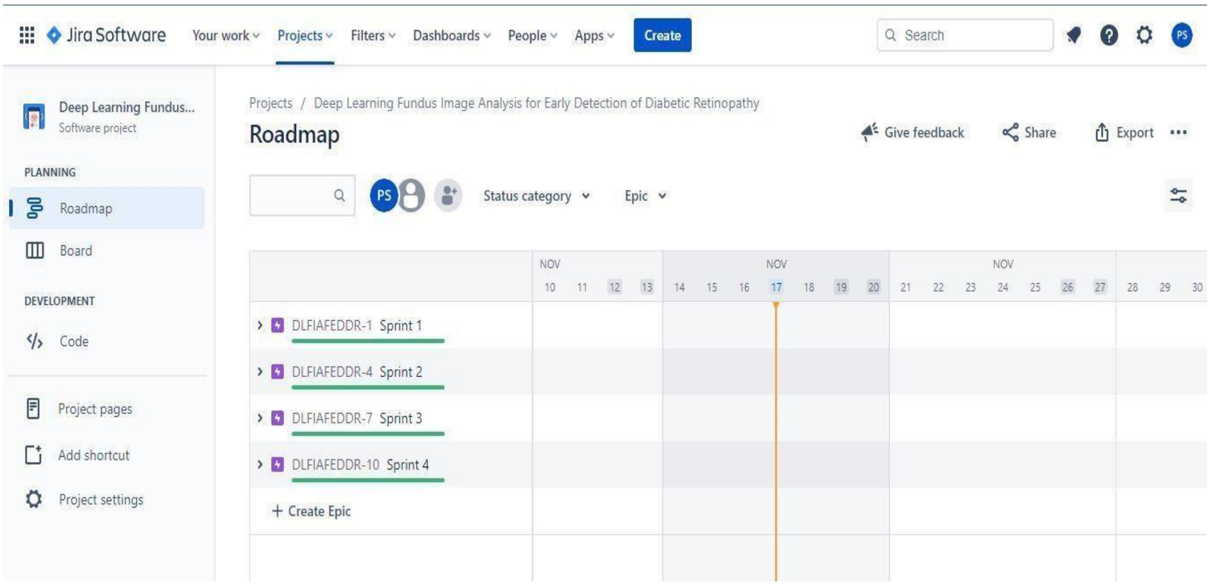
$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

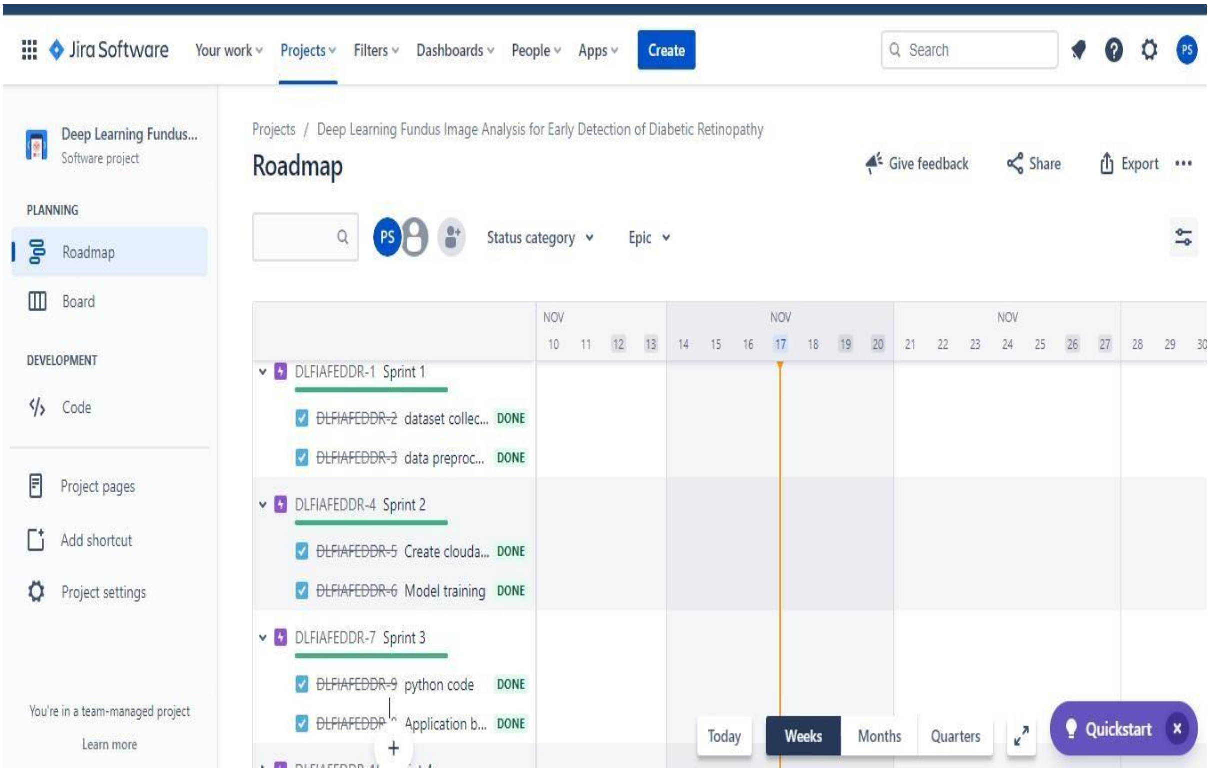
Burndown Chart:



A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

6.3) JIRA SCREENSHOT





(JIRA Folder is created to show the Scrum methodologies and Burn Down chart progress)

7)CODING AND SOLUTIONING

Features 1:

We have developed a website which authenticates users and help them upload and check the seriousness of the diabetics.

Features 2:

We have developed a multilayer deep convolutional neural network that classifies the user image of a eye to which expense has the disease diabetics has been Affected. The model will classify the images into 5 categories of diabetics and report them on asking for prediction. We have also developed a messaging service for receiving message for the type of diabetics.

8) TESTING:

8.1) Test Cases:

Purpose of Document:

This document serves as a quick reference for the Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy project's test coverage and open issues as of the project's release for user acceptance testing.

Defect Analysis:

This shows how many bugs were fixed or closed at each severity level and how they were fixed.

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	9	0	0	9
Client Application	45	0	0	45
Security	2	0	0	2
Out-source Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

8.2) USER ACCEPTANCE TESTING:

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	6	4	2	3	15
Duplicate	1	1	3	0	5
External	4	3	0	1	6
Fixed	9	2	4	15	30
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won’t Fix	0	5	2	1	8
Totals	17	14	13	21	65

9)RESULTS:

9.1) Performance Metrics:

Model Performance Testing:

<table>
<tr><th>S.no</th><th>Parameter</th><th>Values</th><th>Screenshot</th></tr>
<tr><td>1)</td><td>Model Summary</td><td>Total parameters:33,970,989
Trainable params: 13,109,509
Non-trainable params: 20,861,480</td><td><pre>
 block13_sepconv2 (SeparableCon (None, 19, 19, 1024 752024   ['block13_sepconv2_act[0][0]']
 v2D)   )

 block13_sepconv2_bn (BatchNorm (None, 19, 19, 1024 4096   ['block13_sepconv2[0][0]']
 alization)   )

 conv2d_3 (Conv2D)   (None, 10, 10, 1024 745472   ['add_10[0][0]']
)

 block13_pool (MaxPooling2D)   (None, 10, 10, 1024 0   ['block13_sepconv2_bn[0][0]']
)

 batch_normalization_3 (BatchNo (None, 10, 10, 1024 4096   ['conv2d_3[0][0]']
 rmalization)   )

 add_11 (Add)   (None, 10, 10, 1024 0   ['block13_pool[0][0]',
)   'batch_normalization_3[0][0]']

 block14_sepconv1 (SeparableCon (None, 10, 10, 1536 1582080   ['add_11[0][0]']
 v2D)   )

 block14_sepconv1_bn (BatchNorm (None, 10, 10, 1536 6144   ['block14_sepconv1[0][0]']
 alization)   )

 block14_sepconv1_act (Activati (None, 10, 10, 1536 0   ['block14_sepconv1_bn[0][0]']
 on)   )

 block14_sepconv2 (SeparableCon (None, 10, 10, 2048 3159552   ['block14_sepconv1_act[0][0]']
 v2D)   )

 block14_sepconv2_bn (BatchNorm (None, 10, 10, 2048 8192   ['block14_sepconv2[0][0]']
 alization)   )

 block14_sepconv2_act (Activati (None, 10, 10, 2048 0   ['block14_sepconv2_bn[0][0]']
 on)   )

 flatten (Flatten)   (None, 204800)   0   ['block14_sepconv2_act[0][0]']

 dense (Dense)   (None, 64)   13107264   ['flatten[0][0]']

 dense_1 (Dense)   (None, 32)   2080   ['dense[0][0]']

 dense_2 (Dense)   (None, 5)   165   ['dense_1[0][0]']

==
Total params: 33,970,989
Trainable params: 13,109,509
Non-trainable params: 20,861,480
</pre></td></tr>
<tr><td>2)</td><td>Accuracy</td><td>Training Accuracy – 0.7708
Validation Accuracy – 0.6524</td><td><pre>3/3 [==============================] - 35s 11s/step - loss: 0.6524 - accuracy: 0.7708</pre></td></tr>
<tr><td>3)</td><td>Confidence Score</td><td>Class Detected
-
Confidence Score-</td><td>-
-
-
-
-</td></tr>
</table>

10) ADVANTAGES AND DISADVANTAGES:

ADVANTAGES:

There are several advantages of using deep learning for fundus image analysis for early detection of diabetic retinopathy.

1. First, deep learning is well-suited for image analysis tasks. This is because deep learning algorithms can automatically learn features from images, which is essential for accurate image analysis.

2. Second, deep learning is efficient at handling large amounts of data. This is important for medical image analysis, as medical images are often very large.

3. Third, deep learning is scalable. This means that it can be used to train models on very large datasets, which is important for medical image analysis tasks where data is often limited.

4. Fourth, deep learning is able to learn from data with little supervision. This is important for medical image analysis, as often there is limited labelled data available.

5. Finally, deep learning is robust. This means that it is less likely to overfit to the data, which is important for medical image analysis where data is often limited.

DISADVANTAGES:

There are several disadvantages of deep learning for early detection of diabetic retinopathy. One disadvantage is that deep learning requires a large amount of data to train the models. This can be a challenge for researchers who do not have access to a large dataset. Another challenge is that deep learning models can be very complex, which can make them difficult to interpret. Finally, deep learning models can be computationally intensive, which can make them difficult to deploy in resource-limited settings.

11) CONCLUSION:

Diabetic retinopathy (DR) is a leading cause of blindness in the United States. Early detection and treatment of DR is critical to preventing vision loss. However, DR is often asymptomatic in its early stages, making it difficult to detect.

Deep learning (DL) is a type of artificial intelligence that can be used to automatically detect patterns in data. DL has been shown to be effective for detecting DR in images of the retina.

In this study, a DL algorithm was used to automatically detect DR in fundus images. The algorithm was able to accurately detect DR in early stages, before it is symptomatic. This could potentially lead to earlier diagnosis and treatment of DR, which could help to prevent vision loss.

12) FUTURE SCOPE:

There is a great potential for deep learning in fundus image analysis for early detection of diabetic retinopathy. However, there are a few challenges that need to be addressed. First, the current data sets are small and lack diversity.

Second, the images are often low quality and need to be pre-processed before they can be used for deep learning. Third, the ground truth labels for the images are often not available. Finally, the current deep learning models are not able to generalize well to real-world data.

13) APPENDIX

13.1) Source Code

```
  /* Add animation to "page content" */
  .animate-bottom {
    position: relative;
    -webkit-animation-name: animatebottom;
    -webkit-animation-duration: 1s;
    animation-name: animatebottom;
    animation-duration: 1s
  }

  @-webkit-keyframes animatebottom {
    from { bottom:-100px; opacity:0 }
    to { bottom:0px; opacity:1 }
  }

  @keyframes animatebottom {
    from{ bottom:-100px; opacity:0 }
    to{ bottom:0; opacity:1 }
  }

  #myDiv {
    display: none;
    text-align: center;
  }
  </style>
 <title>DR Predcition</title>
</head>

<body onload="myFunction()" >
 <nav class="navbar navbar-expand-lg navbar-light bg-dark">
  <div>
    <a class="navbar-brand" href="#" style="color:aliceblue">Diabetic Retinopathy Classification</a>
  </div>
  <div class="navbar-collapse collapse w-100 order-3 dual-collapse2" id="navbarNav">
```

predict.html

```
        }
    </style>
    <title>DR Predcition</title>
  </head>
  <body>
    <nav class="navbar navbar-expand-lg navbar-light bg-dark">
        <div>
        <a class="navbar-brand" href="#" style="color:aliceblue">Diabetic Retinopathy</a>
        </div>
        <div class="navbar-collapse collapse w-100 order-3 dual-collapse2" id="navbarNav">
          <ul class="navbar-nav mr-auto text-center" id="navbarRight">
            <li class="nav-item active">
              <a class="nav-link" href="index" style="color: aliceblue;">Home </a>
            </li>

            <li class="nav-item">
              <a class="nav-link" href="register"style="color: aliceblue;">Register</a>
            </li>
          </ul>
        </div>
      </nav>
      <br><br>
      <div class="d-flex justify-content-center">
        <div class="row d-flex display-3 justify-content-center">
            Successfully Logged Out!
            <br><br>
            <a href="login" class="btn btn-primary btn-lg">Login for more Information</a>
              </div>
        </div>
  </body>
</html>
```

logout.html

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <!-- CSS only -->
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
      rel="stylesheet"
      integrity="sha384-iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+fzT"
      crossorigin="anonymous"
    />
    <!-- JavaScript Bundle with Popper -->
    <script
      src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"
      integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C8"
      crossorigin="anonymous"
    ></script>
    <style>
        #navbarRight {
            margin-left: auto;
            padding-right:10px;
        }
        .navbar-brand{
            padding-left:15px;
        }
    </style>
    <title>DR Predcition</title>
  </head>
  <body>
    <nav class="navbar navbar-expand-lg navbar-light bg-dark">
        <div>
        <a class="navbar-brand" href="#" style="color:aliceblue">Diabetic Retinopathy Classification</a>
        </div>
        {{msg}}
        <div class="navbar-collapse collapse w-100 order-3 dual-collapse2" id="navbarNav">
          <ul class="navbar-nav mr-auto text-center" id="navbarRight">
```

index.html

```python
import numpy as np
import os
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.inception_v3 import preprocess_input
from flask import Flask, request,flash, render_template, redirect,url_for
from cloudant.client import Cloudant

model = load_model(r"Updated-xception-diabetic-retinopathy.h5")
app = Flask(__name__)
app.secret_key="abc"
app.config['UPLOAD_FOLDER'] = "User_Images"
# Authenticate using an IAM API key
client = Cloudant.iam('6333a0a4-d27b-4ca3-aa49-209f62728e28-bluemix',
                        'wehUTzQdRYF0hiw_imbfXGIadZ_Znyiss6C1r9ktkrXe', connect=True)
# Create a database using an initialized client
my_database = client.create_database('retinopathy')
if my_database.exists():
    print("Database '{0}' successfully created.".format('my_db'))
# default home page or route

user = ""

@app.route('/')
def index():
    return render_template('index.html', pred="Login", vis ="visible")

@ app.route('/index')
def home():
    return render_template("index.html", pred="Login", vis ="visible")


# registration page
@ app.route('/register',methods=["GET","POST"])
def register():
    if request.method == "POST":
        name =  request.form.get("name")
        mail = request.form.get("emailid")
        pswd = request.form.get("pass")
        data = {
```

app.py

```html
    <a class="navbar-brand" href="#" style="color:aliceblue">Registration</a>
    </div>
    <div class="navbar-collapse collapse w-100 order-3 dual-collapse2" id="navbarNav">
      <ul class="navbar-nav mr-auto text-center" id="navbarRight">
        <li class="nav-item active">
          <a class="nav-link" href="index" style="color: aliceblue;">Home </a>
        </li>


      </ul>
    </div>
  </nav>
  <br><br>
  <form class="form-inline" method ="POST">
  <div class="container" style="width: 600px; height: 600px;">
    <div class="mb-3 d-flex justify-content-center"><script src="https://cdn.lordicon.com/xdjxvujz.js"></script>
        <lord-icon
        src="https://cdn.lordicon.com/dqxvvqzi.json"
            trigger="hover"
            style="width:200px;height:200px">
        </lord-icon></div>
                    <div class="mb-3">
            <input type="text" class="form-control" id="exampleInputName" name = "name" aria-describedby="nameHelp" placeholder="Enter Name">
          </div>
          <div class="mb-3">
            <input type="email" class="form-control" id="exampleInputEmail1" name="emailid" aria-describedby="emailHelp" placeholder="Enter Mail ID">
          </div>

          <div class="mb-3">
            <input type="password" class="form-control" id="exampleInputPassword1" name="pass" placeholder="Enter Password">
          </div>
          <div class="mb-3">
          <button type="submit form-control" class="btn btn-dark btn-primary" style="width:100%;">Register</button>
        </div>
                    <div class="mb-3 d-flex justify-content-center">
                    <a href="login" class="nav-link"> Already Registered: Login Here</a>
  </div>
  <div class="mb-3 d-flex justify-content-center">
    {{pred}}
```

register.html

13.2) Python Notebook Screenshot

Required libraries

In [10]:
```
import numpy as np
import tensorflow as tf
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.xception import Xception
```

In [1]:
```
! pip install kaggle
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: kaggle in /usr/local/lib/python3.7/dist-packages (1.5.12)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.7/dist-packages (from kaggle) (6.1.2)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.24.3)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.15.0)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.7/dist-packages (from kaggle) (2.8.2)
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from kaggle) (2022.9.24)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from kaggle) (2.23.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from kaggle) (4.64.1)
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.7/dist-packages (from python-slugify->kaggle) (1.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->kaggle) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->kaggle) (2.10)
```

Connecting to kaggle dataset

Uploading the kaggle.json API file

In [2]:
```
from google.colab import files
files.upload()
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving kaggle.json to kaggle.json
```

Out[2]:
```
{'kaggle.json': b'{"username":"vigneshb2001","key":"f0d5bd91ccf3921c165519bd8afb8fd2"}'}
```

Create .kaggle directory and copy the kaggle.json file into it.

```
! mkdir ./.kaggle
! cp kaggle.json ./.kaggle/
```

```
mkdir: cannot create directory '/root/.kaggle': File exists
```

Allocate permission for this kaggle.json file

In [5]:
```
! chmod 600 ./.kaggle/kaggle.json
```

Downloading the dataset

In [6]:
```
! kaggle datasets download arbethi/diabetic-retinopathy-level-detection
```

```
Downloading diabetic-retinopathy-level-detection.zip to /content
100% 9.63G/9.66G [01:18<00:00, 161MB/s]
100% 9.66G/9.66G [01:19<00:00, 131MB/s]
```

Unziping the dataset

In [7]:
```
! unzip diabetic-retinopathy-level-detection.zip
```

```
Archive:  diabetic-retinopathy-level-detection.zip
  inflating: inception-diabetic.h5
  inflating: preprocessed dataset/preprocessed dataset/testing/0/cfb17a7cc8d4.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/cfdbaef73a8b.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/cfed7c1172ec.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/cff262ed8f4c.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/cffc50047828.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d02b79fc3200.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d0926ed2c8e5.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d160ebef4117.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d16e39b9d6f0.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d16e59a2b33a.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d18f6431ebce.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d1a60c3b9fe5.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d1afdb8cf70d.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d1b279cc02ae.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d1ca85af57c9.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d1cf31577a59.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d1f7ea924a01.png
  inflating: preprocessed dataset/preprocessed dataset/testing/0/d1fa0f744620.png
```

```
================================================================================================
Total params: 33,970,989
Trainable params: 13,109,509
Non-trainable params: 20,861,480
________________________________________________________________________________________________
```

Compiling the model

In [22]:
```
model.compile(loss = 'categorical_crossentropy',optimizer = 'adam',metrics = ['accuracy'])
r = model.fit(training_set,validation_data = test_set,epochs = 40,steps_per_epoch = len(training_set)//32,validation_steps = len(test_set)//32)
```

```
Epoch 1/40
3/3 [==============================] - 62s 19s/step - loss: 7.0694 - accuracy: 0.2708
Epoch 2/40
3/3 [==============================] - 51s 15s/step - loss: 5.4516 - accuracy: 0.3958
Epoch 3/40
3/3 [==============================] - 50s 14s/step - loss: 5.4221 - accuracy: 0.6771
Epoch 4/40
3/3 [==============================] - 40s 15s/step - loss: 5.0895 - accuracy: 0.5897
Epoch 5/40
3/3 [==============================] - 46s 14s/step - loss: 3.5532 - accuracy: 0.6771
Epoch 6/40
3/3 [==============================] - 47s 14s/step - loss: 2.4607 - accuracy: 0.5625
Epoch 7/40
3/3 [==============================] - 49s 14s/step - loss: 2.0138 - accuracy: 0.5833
Epoch 8/40
3/3 [==============================] - 50s 15s/step - loss: 1.7375 - accuracy: 0.6667
Epoch 9/40
3/3 [==============================] - 48s 14s/step - loss: 1.8682 - accuracy: 0.5729
Epoch 10/40
3/3 [==============================] - 48s 14s/step - loss: 1.2700 - accuracy: 0.7604
Epoch 11/40
3/3 [==============================] - 47s 14s/step - loss: 2.0210 - accuracy: 0.7188
Epoch 12/40
3/3 [==============================] - 52s 16s/step - loss: 2.2501 - accuracy: 0.6042
Epoch 13/40
3/3 [==============================] - 47s 14s/step - loss: 1.1787 - accuracy: 0.7604
Epoch 14/40
3/3 [==============================] - 49s 15s/step - loss: 1.4312 - accuracy: 0.7917
Epoch 15/40
3/3 [==============================] - 51s 16s/step - loss: 1.3212 - accuracy: 0.7500
Epoch 16/40
3/3 [==============================] - 50s 15s/step - loss: 1.7987 - accuracy: 0.6354
Epoch 17/40
3/3 [==============================] - 48s 14s/step - loss: 1.3420 - accuracy: 0.6875
```

```
inflating: preprocessed dataset/preprocessed dataset/training/4/fce55caa4758.png
inflating: preprocessed dataset/preprocessed dataset/training/4/fdd534271f3d.png
inflating: preprocessed dataset/preprocessed dataset/training/4/ff8a0b45c789.png
```

Splitting dataset into train and test dataset

In [8]:

```
imageSize = [299,299]
trainPath = '/content/preprocessed dataset/preprocessed dataset/training'
testPath = '/content/preprocessed dataset/preprocessed dataset/testing'
```

In [11]:

```
train_datagen = ImageDataGenerator(rescale = 1/255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, vertical_flip = True)
test_datagen = ImageDataGenerator(rescale = 1/255)
```

In [14]:

```
training_set = train_datagen.flow_from_directory(trainPath, target_size = (299, 299), batch_size = 32, class_mode = 'categorical')

test_set = train_datagen.flow_from_directory(testPath, target_size = (299, 299), batch_size = 32, class_mode = 'categorical')
```

```
Found 3662 images belonging to 5 classes.
Found 734 images belonging to 5 classes.
```

Model Building

In [15]:

```
xception = Xception(input_shape = imageSize + [3],weights = 'imagenet' ,include_top = False)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/xception/xception_weights_tf_dim_ordering_tf_kernels_notop.h5
83683744/83683744 [==============================] - 1s 0us/step
```

In [16]:

```
for layer in xception.layers:
  layer.trainable = False
```

In [17]:

```
xception.summary()
```

```
Model: "xception"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_1 (InputLayer)           [(None, 299, 299, 3  0           []
                                )]
```

```
Epoch 26/40
3/3 [==============================] - 47s 14s/step - loss: 1.2918 - accuracy: 0.6771
Epoch 27/40
3/3 [==============================] - 47s 14s/step - loss: 1.1866 - accuracy: 0.6667
Epoch 28/40
3/3 [==============================] - 47s 14s/step - loss: 1.0595 - accuracy: 0.7812
Epoch 29/40
3/3 [==============================] - 38s 15s/step - loss: 1.3182 - accuracy: 0.6923
Epoch 30/40
3/3 [==============================] - 48s 14s/step - loss: 0.6461 - accuracy: 0.7604
Epoch 31/40
3/3 [==============================] - 48s 14s/step - loss: 1.2289 - accuracy: 0.6771
Epoch 32/40
3/3 [==============================] - 54s 17s/step - loss: 0.9484 - accuracy: 0.7188
Epoch 33/40
3/3 [==============================] - 48s 15s/step - loss: 1.0351 - accuracy: 0.6979
Epoch 34/40
3/3 [==============================] - 48s 14s/step - loss: 1.4485 - accuracy: 0.7500
Epoch 35/40
3/3 [==============================] - 47s 14s/step - loss: 1.5413 - accuracy: 0.6979
Epoch 36/40
3/3 [==============================] - 47s 15s/step - loss: 1.2884 - accuracy: 0.6875
Epoch 37/40
3/3 [==============================] - 48s 14s/step - loss: 1.4528 - accuracy: 0.6875
Epoch 38/40
3/3 [==============================] - 48s 14s/step - loss: 1.2701 - accuracy: 0.7604
Epoch 39/40
3/3 [==============================] - 46s 14s/step - loss: 1.9731 - accuracy: 0.5521
Epoch 40/40
3/3 [==============================] - 47s 14s/step - loss: 1.0961 - accuracy: 0.7812
```

Saving the model

In [24]:

```
model.save('Xception-diabetic-retinopathy.h5')
```

REFERENCES:

[1] Johari, Mohamad & Hassan, Hiron & Yassin, Ahmad & Tahir, Noorita & Zabidi, Azlee & Rizman, Zairi & Baharom, R. & Wahab, N.. (2018). Early Detection of Diabetic Retinopathy by Using Deep Learning Neural Network. International Journal of Engineering and Technology(UAE). 7. 198-201. 10.14419/ijet.v7i4.11.20804.

[2] Harry Pratta,∗ , Frans Coenenb, Deborah M Broadbentc , Simon P Hardinga,c, Yalin Zhenga,c(2016).Convolutional Neural Networks for Diabetic Retinopathy https://doi.org/10.1016/j.procs.2016.07.014.

[3] Shaban M, Ogur Z, Mahmoud A, Switala A, Shalaby A, Abu Khalifeh H, et al. (2020) A convolutional neural network for the screening and staging of diabetic retinopathy. PLoS ONE 15(6): e0233514. https://doi.org/10.1371/journal. pone.0233514

[4] Ayala, A.; Ortiz Figueroa, T.; Fernandes, B.; Cruz, F. Diabetic Retinopathy Improved Detection Using Deep Learning. Appl. Sci. 2021, 11, 11970. https://doi.org/10.3390/ app112411970

GITHUB LINK: *https://github.com/IBM-EPBL/IBM-Project-13348-1659516930*

DEMO VIDEO: https://github.com/IBM-EPBL/IBM-Project-13348-1659516930/tree/main/Final%20Deliverables/Demonstration%20Video