

Medical Image Processing : Classification of Diabetic Retinopathy Using Machine Learning

HILLOL GHOSH (11700114033)
APARKADIPTA BISWAS (11700114013)
BIDIT RAY (11700114022)



Department of Computer Science and Engineering of
RCC INSTITUTE OF INFORMATION TECHNOLOGY

Medical Image Processing : Classification of Diabetic Retinopathy Using Machine Learning

Thesis submitted in

May 2018

Session 2017-18

to the department of

Computer Science and Engineering

Of

RCC INSTITUTE OF INFORMATION TECHNOLOGY

in partial fulfillment of the requirements
for the degree of

Bachelor of Technology

in

Computer Science and Engineering

By

HILLOL GHOSH (11700114033)

APARKADIPTA BISWAS (11700114013)

BIDIT RAY (11700114022)

with the supervision of

Prof. Dr. Minakshi Banerjee



Department of Computer Science and Engineering
RCC INSTITUTE OF INFORMATION TECHNOLOGY



Department of Computer Science and
Engineering RCC INSTITUTE OF
INFORMATION TECHNOLOGY
Canal South Road, Beliaghata, Kolkata, West
Bengal PIN-700015

May 16, 2018

This is to certify that the thesis entitled, Classification of DiabeticRetinopathy Using Machine Learning submitted by Hillol Ghosh (11700114033), Arkadipta Biswas (11700114013), Bidit Ray (11700114022), in partial fulfillment of the requirements for the completion of Bachelor of Technology Degree in Computer Science and Engineering at the RCC INSsTITUTE OF INFORMATION TECHNOLOGY, is an authentic work carried out by them under my supervision and guidance .

To the best of my knowledge, Neither this thesis or any part or it has been submitted for any degree or diploma award else-where.

Prof. Dr. Minakshi Banerjee
Professor
Department of Computer Science and
Engineering
RCCIIT

Countersigned:

.....

Head

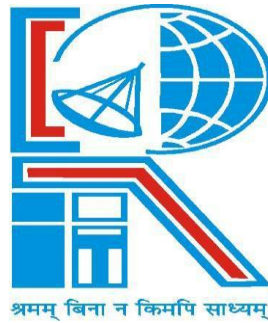
Department of Computer Sc. & Engg,

RCC Institute of Information Technology

Kolkata – 700015.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RCC INSTITUTE OF INFORMATION TECHNOLOGY



CERTIFICATE OF APPROVAL

The foregoing Project is hereby accepted as a credible study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it is submitted.

FINAL EXAMINATION FOR
EVALUATION OF PROJECT

1. _____

2. _____

(Signature of Examiners)

Acknowledgment

With great satisfaction and pride I present my thesis on the project under the Research Project paper during final Year, for partial fulfillment of my Bachelor of Technology degree in Computer Science and Engineering at RCC INSTITUTE OF INFORMATION TECHNOLOGY.

I am thankful to Prof. Dr. Minakshi Banerjee for being the best guide and advisor for this research work in every field I have taken to complete my requirement. Her ideas and inspirations have helped me make this nascent idea of mine into a fully fledged project. Without presence of her I may never had tasted the favor in a research work.

Again I am thankful to my batch-mates to support me in my implementation part sometime. I am also grateful to all the professors of my department for being a constant source of inspiration and motivation during the course of the project.

I would like to dedicate this project to my parents, who always stood by me in each and every point of my life.

So I am thankful again to all who are being a part of my final year research project.

HILLOL GHOSH (11700114033)

APARKADIPTA BISWAS (11700114013)

BIDIT RAY (11700114022)

Abstract :

Diabetic Retinopathy (DR) is the damage caused to the retina of the eye due the complications of

diabetes. There are a large number of people, who are suffering from Diabetic Retinopathy that leads to blurring of vision or even blindness, if not treated at an early stage. Hence it is important to detect DR at an earlier stage and provide treatment otherwise it may lead to blurring of vision or total blindness.

It is an ocular manifestation of diabetes, affects up to 80% of all patients suffering from diabetes for over 10 years.

In medical practice so far, the stages of DR are detected by merely looking at the fundus images. This paper presents a more radical & modern approach to identify the different stages of Diabetic Retinopathy.

In this paper development of an algorithm for the identification of different stages of DR via Machine Learning is discussed.

The proposed algorithm is simulated on Jupyter platform. The algorithm is tested on the images of EyePac Data Set and has yielded an accuracy of 89.61%.

Contents

1. Introduction .
2. Review of Literature
3. Objective of the Project
4. System Design
5. Methodology for implementation
(Formulation/Algorithm)
6. Implementation Details
7. Results/Sample output
8. Conclusion

CHAPTER 1

INTRODUCTION

Diabetes is a group of metabolic diseases in which a person has high blood sugar, either because the body does not produce enough insulin, or because the cells do not respond to the insulin that is produced.

Chronically high blood sugar from diabetes is associated with damage to the tiny blood vessels in the retina, leading to diabetic retinopathy. The retina detects light and converts it to signals sent through the optic nerve to the brain.

Diabetic retinopathy can cause blood vessels in the retina to leak fluid or hemorrhage (bleed), distorting vision.

In its most advanced stage, new abnormal blood vessels proliferate (increase in number) on the surface of the retina, which can lead to scarring and cell loss in the retina.

Diabetic retinopathy is one of the common complications of diabetes. It is a severe and widely spread eye disease. It damages the small blood vessels in the retina resulting in loss of vision.

The risk of the disease increases with age and therefore, middle aged and older diabetics are prone to Diabetic Retinopathy.

The progression from no retinopathy to PDR can take 2 decades or more, and this slow rate enables DR to be identified and treated at an early stage.

Development and progression of DR are related to duration and control of diabetes. DR in its early form is often asymptomatic but amenable to treatment.

How Severe an Issue is Diabetic Retinopathy ?

Diabetes mellitus is becoming a global epidemic and is now one of the top causes of vision loss globally.

In 2014, there were approximately 422 million people (8.5% of the world's adult population) living with diabetes; compared to 108 million in 1980 (2016 WHO Global Report on Diabetes).

Increased urbanisation, consumption of less – nutritious food, more sedentary lifestyles and resulting obesity have all contributed to the dramatic rise in the global prevalence of diabetes, particularly in resource – poor countries.

Low and middle income countries account for approximately 75% of the global diabetes burden yet many are ill equipped to properly identify, treat and manage the complex and varied consequences of this disease.

Currently, South East Asia and the Western Pacific account for more than half of adults with diabetes worldwide.

China, India, Indonesia and Bangladesh alone represent 45% of the global burden.

Yet the highest prevalence of diabetes is found in the Eastern Mediterranean, where close to 14% of the population is afflicted.

Efforts to reduce the prevalence of diabetes or to more effectively manage its health consequences are further undermined by the fact that approximately 50% of people with diabetes are currently undiagnosed.

Diabetes increases the risk of a range of eye diseases, but the main cause of blindness associated with diabetes is diabetic retinopathy (DR).

DR damages blood vessels inside the retina at the back of the eye. It commonly affects both eyes and can lead to **vision loss if it is not treated**. Poorly controlled blood sugars, high blood pressure and high cholesterol increase the risk of developing DR.

People with DR whose sight is at risk **can be treated**, most commonly with laser, **to prevent visual impairment and blindness**.

However, there is no treatment that can restore vision that has already been lost.

Because DR is initially asymptomatic many people with diabetes are not aware that their condition, if left unmanaged it may affect their vision and lead to blindness.

The vast majority of patients who develop DR have no symptoms until the very late stages (by which time it may be too late for effective treatment).

Therefore **screening and early intervention is critical.**

What are the several Stages of Diabetic Retinopathy ?

Diabetic retinopathy may progress through five stages:

0. No Diabetic Retinopathy – The eye is perfectly clear of problems caused by Diabetes .

1. Mild Non-proliferative retinopathy - Small areas of balloon-like swelling in the retina's tiny blood vessels, called microaneurysms, occur at this earliest stage of the disease. These microaneurysms may leak fluid into the retina.

2. Moderate Non-proliferative retinopathy - As the disease progresses, blood vessels that nourish the retina may swell and distort. They may also lose their ability to transport blood. Both conditions cause characteristic changes to the appearance of the retina and may contribute to DME.

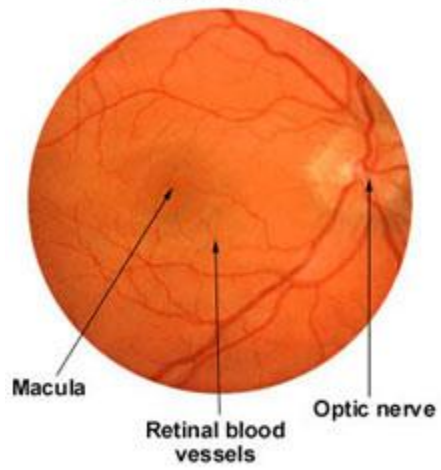
3. Severe Non-proliferative retinopathy - Many more blood vessels are blocked, depriving blood supply to areas of the retina. These areas secrete growth factors that signal the retina to grow new blood vessels.

4. Proliferative diabetic retinopathy (PDR) - At this advanced stage, growth factors secreted by the retina trigger the proliferation of new blood vessels, which grow along the inside surface of the retina and into the vitreous gel, the fluid that fills the eye.

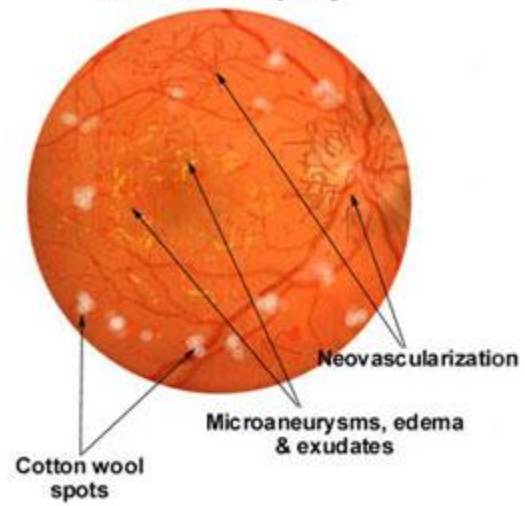
The new blood vessels are fragile, which makes them more likely to leak and bleed. Accompanying scar tissue can contract and cause retinal detachment—the pulling away of the retina from underlying tissue.

Retinal detachment can lead to permanent vision loss.

Normal Retina



Diabetic Retinopathy



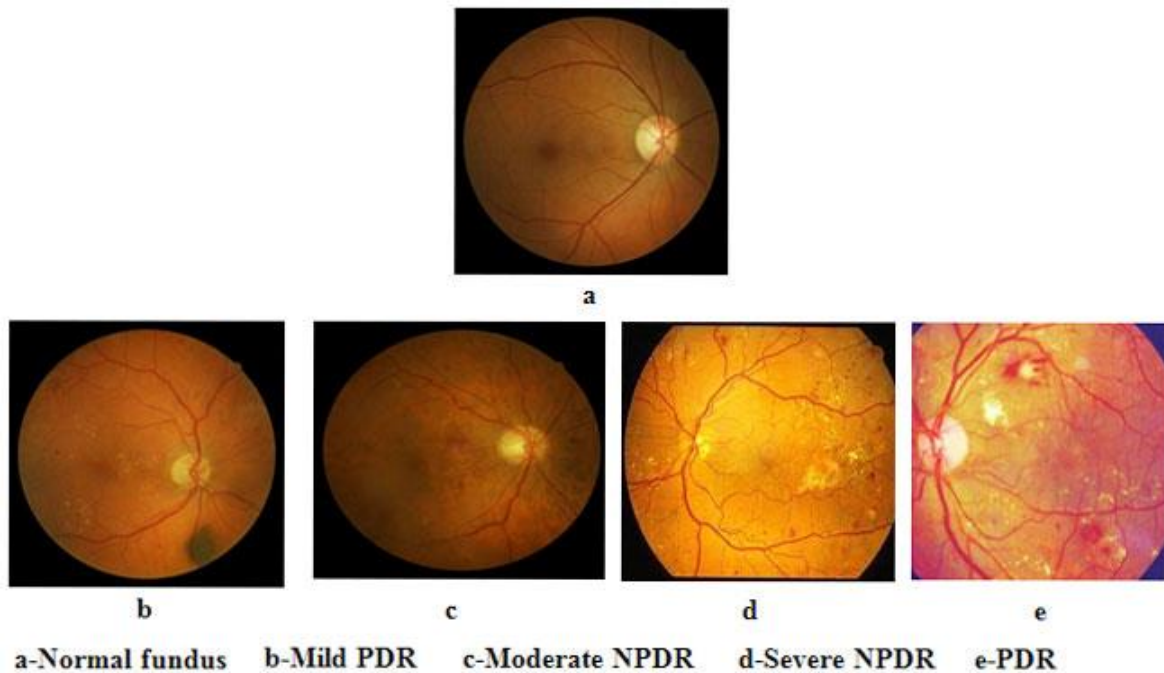


Figure 2: Stages of Diabetic Retinopathy and their symptoms

Issue with Current Recognition technique :

Currently, detecting DR is a time-consuming and manual process that requires a trained clinician to examine and evaluate digital color fundus photographs of the retina.

By the time human readers submit their reviews, often a day or two later, the delayed results lead to lost follow up, miscommunication, and delayed treatment.

Unfortunately, there is no effective known cure for diabetic retinopathy and the present treatments available are just management strategies at best.

So its very important to detect the disease in its early stages.

Clinicians can identify DR by the presence of lesions associated with the vascular abnormalities

caused by the disease. While this approach is effective, its resource demands are high.

The expertise and equipment required are often lacking in areas where the rate of diabetes in local populations is high and DR detection is most needed.

As the number of individuals with diabetes continues to grow, the infrastructure needed to prevent blindness due to DR will become even more insufficient.

The need for a comprehensive and automated method of DR screening has long been recognized, and previous efforts have made good progress using image classification, pattern recognition, and machine learning.

With fundus photography as input, the goal of this project is to push an automated detection system to the limit of what is possible – ideally resulting in models with realistic clinical potential.

Given a image of left and right eye of the patient, the main goal is to classify the DR in eye status among one of the following classes 0 - No DR, 1 - Mild, 2 - Moderate, 3 - Severe, 4 - Proliferative DR.

So our task is to create an automated analysis system capable of assigning a score based on the above scale.

Chapter 2

Literature Review:

In Image recognition, a Convolutional Neural Network (CNN) is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of Organic brain pattern , whose individual neurons are arranged in such a way that responds to overlapping regions tiling the visual field.

Currently, CNNs are the most researched machine learning algorithms in medical image analysis.

The reason for this is that CNNs preserve spatial relationships when filtering input image

In deep learning, the convolutional neural network uses a complex architecture composed of stacked layers in which it is particularly well-adapted to classify the images.

For multiclass classification, this architecture is perceptive enough to each feature present in the images. It is robust & Sensitive.

Spatial relationships are of crucial importance in radiology, ex- in how the edge of a bone joins with tissue, or where normal cells interfaces with cancerous cells.

CNN takes an input image of raw pixels, and prepares it for processing it via Convolutional Layers, Rectified Linear Unit (RELU) Layers and Pooling Layers.

This feeds into a final Fully Connected Layer which assigns class scores or probabilities, thus classifying the input into the class with the highest probability that it has been trained in.

Features of CNN :

CONVOLUTION LAYER

A convolution is defined as an operation on two functions. In image analysis, one function consists of input values (e.g. pixel values) at a position in the image, and the second function is a filter (or kernel); each can be represented as array of numbers.

Computing the dot product between the two functions gives an output. The filter is then shifted to the next position in the image as defined by the stride length.

The computation is repeated until the entire image is covered, producing a feature (or activation) map.

This is a map of where the filter is strongly activated and 'sees' a feature such as a straight line, a dot, or a curved edge.

If a photograph of a face was fed into a CNN, initially low-level features such as lines and edges are discovered by the filters...but it is progressive.

These build up to progressively higher features in subsequent layers, such as a nose, eye or ear, as the feature maps become inputs for the next layer in the CNN architecture.

Convolution exploits three ideas intrinsic to perform computationally efficient machine learning: sparse connections, parameter sharing (or weights sharing) and equivariant (or invariant) representation .

Unlike some neural networks where every input neuron is connected to every output neuron in the subsequent layer, CNN neurons have sparse connections, meaning that only some inputs are connected to the next layer.

By having a small, local receptive field (i.e., the area covered by the filter per stride), meaningful features can be gradually learnt, and the number of weights to be calculated can be drastically reduced, increasing the algorithm's efficiency.

In using each filter with its fixed weights across different positions of the entire image, CNNs reduce memory storage requirements. This is known as parameter sharing.

This is in contrast to a fully connected neural network where the weights between layers are more numerous, used once and then discarded. Parameter sharing results in the quality of equivariant representation to arise.

An output (or feature map) $s(t)$ is defined below when input $I(t)$ is convolved with a filter or kernel $K(a)$.

$$s(t) = (I * K)(t). \quad (1)$$

If t can only take integer values, the **discretized convolution** is given by:

$$s(t) = \sum_a I(a) \cdot K(t - a). \quad (2)$$

The above assumes a one-dimensional convolutional operation.

Activation function: A function used to transform the activation level of neuron (weighted sum of inputs) to an output signal

$$y = \Theta \left(\sum_{j=1}^N w_j I_j - T \right) \quad (3)$$

RECTIFIED LINEAR UNIT (RELU) LAYER

The RELU layer is an activation function that sets negative input values to zero. This simplifies and accelerates calculations and training, Relu behaves close to a linear unit.

Relu is like a switch for linearity. If you don't need it, you "switch" it off. If you need it, you "switch" it on. So it gives binary result "Yes" or "No".

The derivative is 1 when it's active or 0 elsewhere. Thus, it's a very simple function. That makes optimisation much easier.

Mathematically it is defined as

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

where x is the input to the neuron.

Sigmoid function

In general, a sigmoid function is real-valued and differentiable, having a non-negative or non-positive first derivative, one local minimum, and one local maximum.

Sigmoid functions are used in artificial neural networks to predict nonlinearity in the working model.

A neural network element computes a linear combination of its input signals, and applies a sigmoid function to the result.

They are more useful for recurrent networks, probabilistic models.

Pooling Layer

The Pooling layer is inserted between the Convolution and RELU layers to reduce the number of parameters to be calculated. Another side effect is the reduction in size. (width and height, but not depth).

Max-pooling is commonly used, other pooling layers include Average, normalization pooling.

Max-pooling simply takes the largest input value within a filter and discards the other values; effectively it summarizes the strongest activations over a neighborhood. It can be said that it takes the highest pixel density among the adjacent pixel values.

Fully Connected Layer

The final layer in a CNN is the Fully Connected Layer, meaning that every neuron in the preceding layer is connected to every neuron in the Fully Connected Layer. Like the convolution, RELU and pooling layers, there can be 1 or more fully connected layers depending on the level of feature abstraction desired.

This layer takes the output from the preceding layer (Convolutional, RELU or Pooling) as its input, and computes a probability score for classification into the different available classes. In essence, this layer looks at the combination of the most strongly activated features that would indicate the image belongs to a particular class.

Chapter 3

Motivation

In some of the previous works done by others differing approaches & hence differing amounts of accuracy were found out.

Wong Li Yun et al. in their paper, six input features Red, Green and Blue layers of perimeter and areas of blood vessels are extracted from the raw images using the image processing techniques and fed to a neural network based classifier for classification.

The authors C. Sinthanayothin et al. in their paper have described about an automated screening system to analyse digital colour retinal images for important features of non-proliferative diabetic retinopathy (NPDR). The method employed was high performance preprocessing of the color image.

Previously described automated image analysis systems were used to detect major landmarks of the retinal image (optic disc, blood vessels and fovea).

Image processing, analysis and computer vision techniques are increasing in prominence in all fields of medical science, and are especially pertinent to modern ophthalmology, as it is heavily dependent on visually oriented signs.

The authors David Calvo et.al,[8] have described that the analysis of retinal vessel tree characteristics is an important task in medical diagnosis, specially, in the cases of diseases like vessel occlusion, hypertension or diabetes. In their research a method for detection and classification of retinal vessel tree feature points was presented. The method applied combined imaging techniques such as filters or morphologic operations to obtain an adequate structure for the detection.

Table 1. Comparison of different classification methods from [8].

Authors	No of classes	Method	Accuracy of classification	Sensitivity	Specificity
Wang et al. 2000 [73]	2	Minimum distance discriminant classifier	70%		
Sinthanayothin et al. 2003 [66]	2	Moat operator	Not reported	80%	71%
Usher et al. 2003 [82]	2	Lesions	Not reported	95%	53%
Singalavanija et al. 2005 [81]	2	Blood vessels, exudates, haemorrhages, microaneurysms	Not reported	75%	83%
Lee et al. 2005 [43]	3	Hemorrhages, microaneurysms, hard exudates, cotton wool spots	Max: 88%	Not reported	Not reported
Neubauer et al. 2005 [50]	2	Retinal thickness analyzer	Not reported	93%	100%
Kahai et al. 2006 [36]	2	Decision support system (DSS)	Not reported	100%	63%
Philip et al. 2007 [57]	2	Exudates	Not reported	91%	67%
Estabridis and Figueiredo 2007 [20]	2	Fovea, blood vessel network, optic disk, bright and dark lesions	90%	Not reported	Not reported
Li et al. 2008 [46]	2	Bright lesions, retinal vessel patterns	Not reported	81%	Not reported
Abramoff et al. 2008 [2]	3	Optic disc, retinal vessels, hemorrhages, microaneurysms, vascular, abnormalities, exudates, cotton wool spots, drusen	Not reported	84%	64%
Wong et al. 2008 [75]	4	Area of blood vessel	84%	92%	100%
Nayak et al. 2008 [48]	3	Blood vessels, exudates and texture	94%	90%	100%
Acharya et al. 2008 [3]	5	Higher order spectra	82%	83%	89%
Acharya et al. 2009 [5]	5	Blood vessel, exudates, microaneurysms, haemorrhages	86%	82%	86%
Vujosevic et al. 2009 [71]	2	Single lesions	Not reported	82%	92%

The above table contains some of the analogue techniques used ,some have very high efficiency but require expert & trained professionals to analyze.

These are slow ,cumbersome with its performance depending upon a very specific & narrow parameter ,any variation or difference will mean a wrong result.

Our method focuses on every parameter possible .rather than focusing on any narrow parameter.

Our proposed algorithm performs better than global approaches as it is based on Convolution Neural Network .Using machine learning we can take the human factor out of the equation.

This makes the answers reliable,super fast

As our algorithm utilizes machine learning with already created modules & rather than inventing everything from the ground up , its complexity is low with comparison to global approaches.

It also outperforms the other machine learning codes with an accuracy of 89.61%.

Chapter 4

System Design :

Software Requirements:

Operating system - Linux.

Programming Language -python 3.6.

Library & modules used - NumPy ,Pandas ,Tensorflow ,PIL ,Keras ,os ,Cuda.

Platform used - Jupyter , Tensorboard .

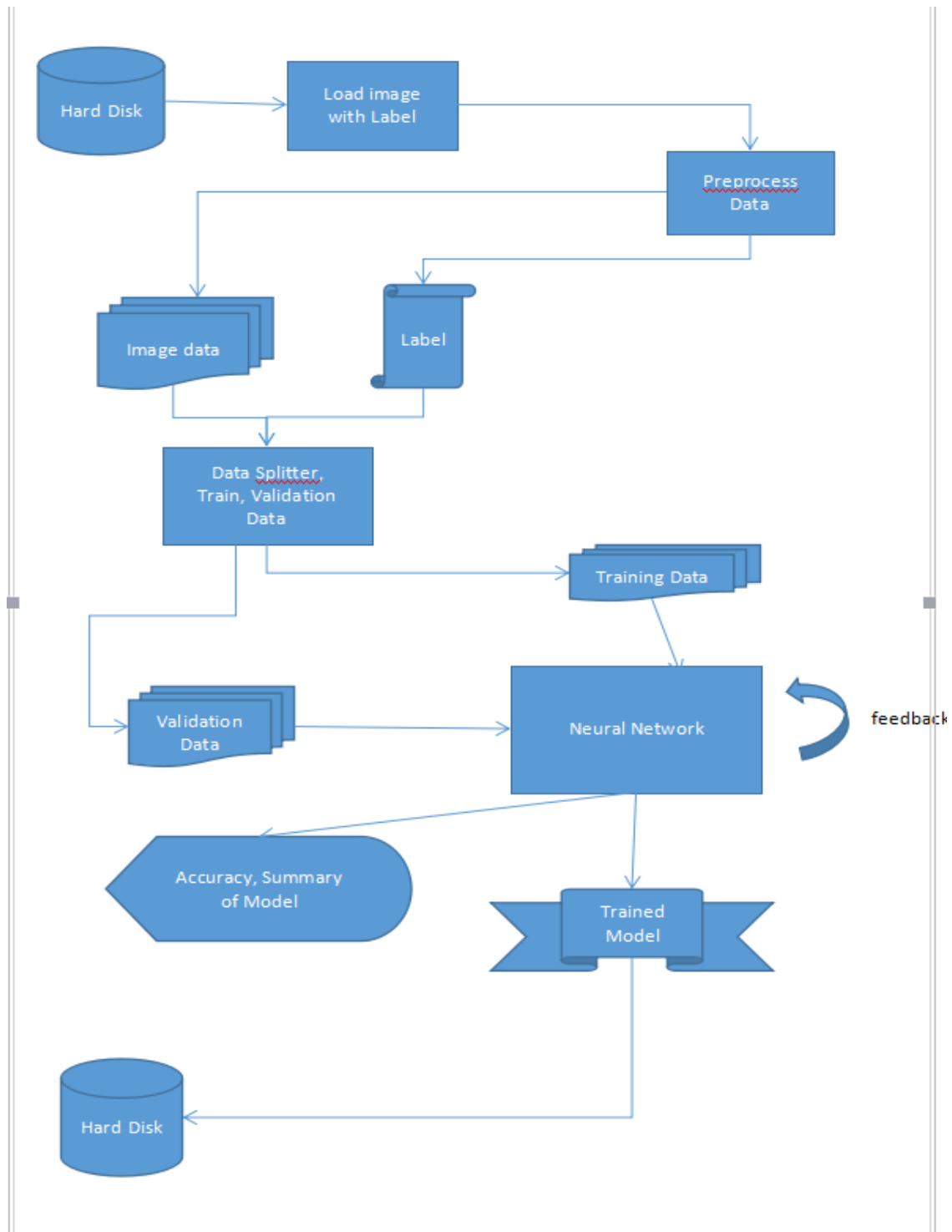
Hardware Requirements.

Intel Core i7 Octa Core Edition.

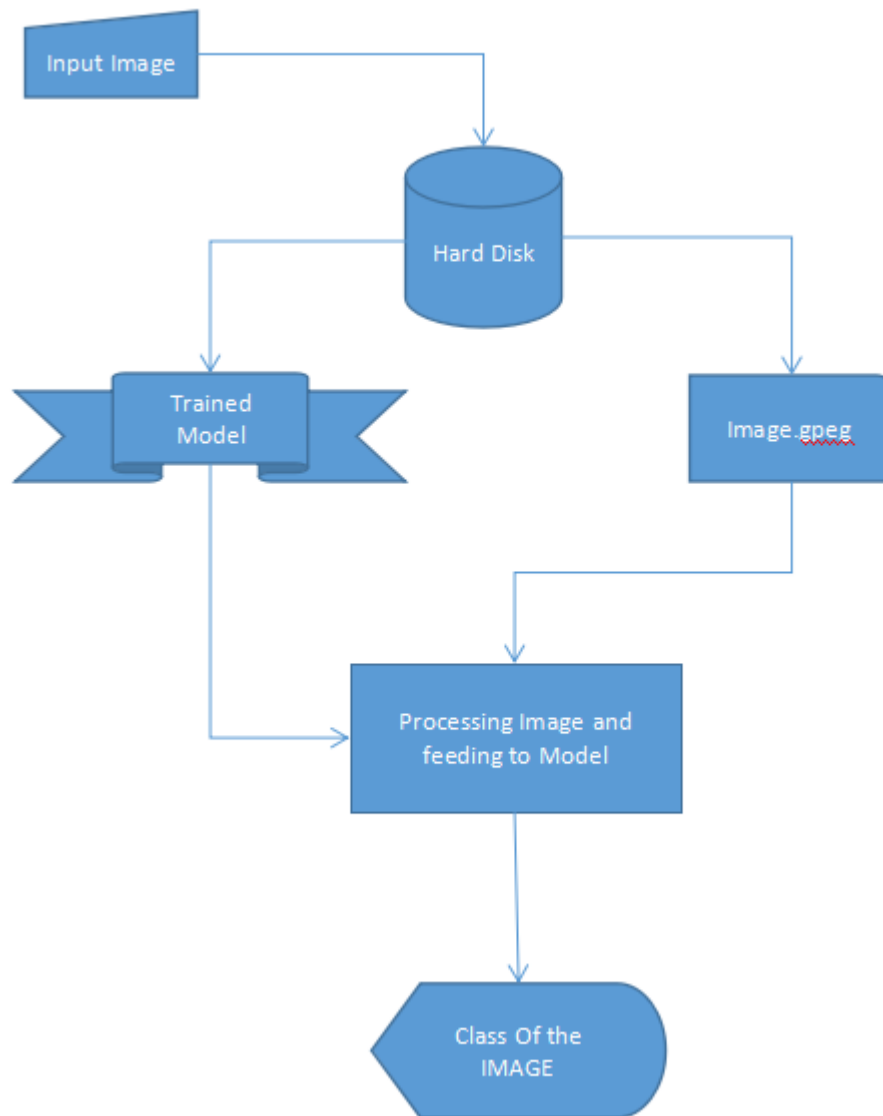
96 GB DDR4 RAM.

Nvidia 1080 Ti

Training Stage:



Testing stage:

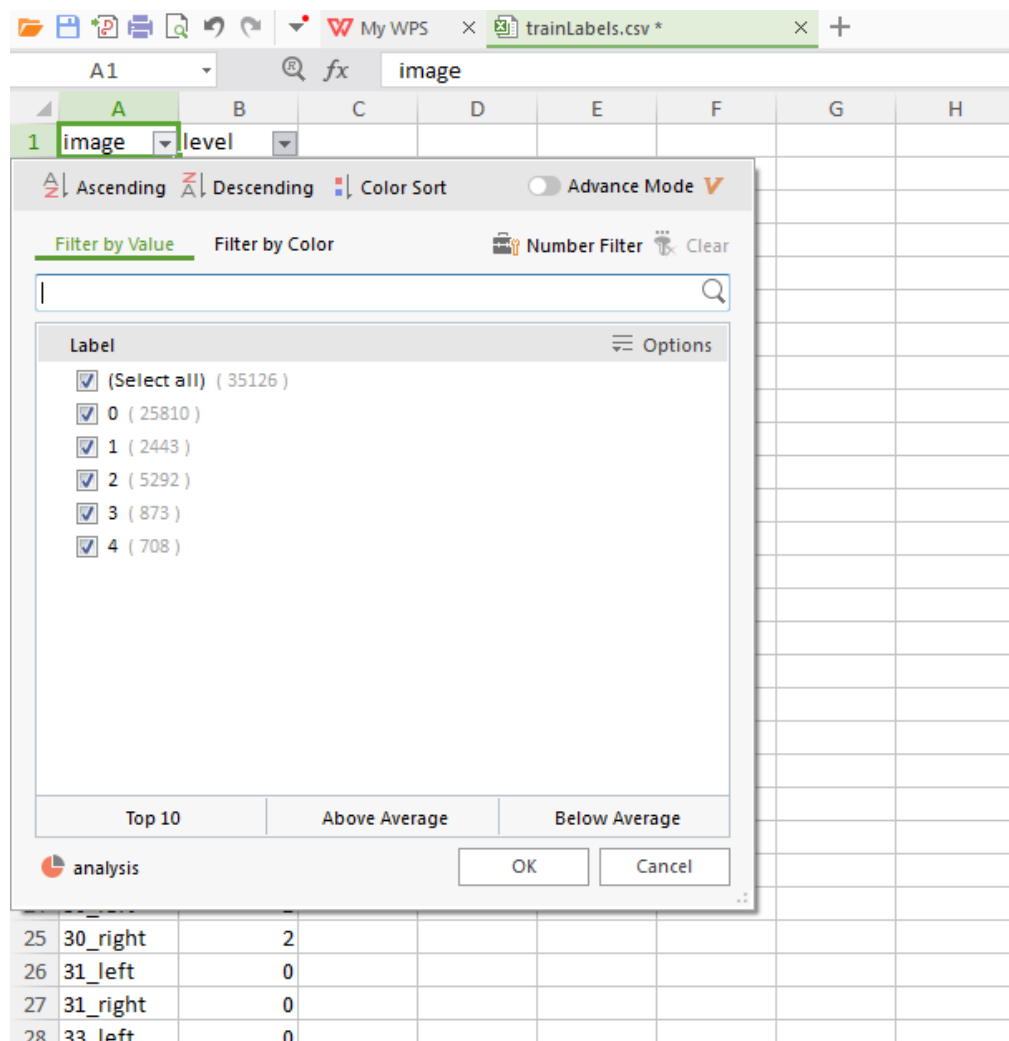


Chapter 5

Methodology & Implementation:

Dataset Distribution:

The Dataset provided by eyePACS has disbalanced labelling. I.e, the 5 classes of data don't have equal no Of dataset.



The screenshot shows a WPS Spreadsheet window with a file named 'trainLabels.csv'. The spreadsheet has columns A through H and rows 1 through 28. The 'image' column (A) contains labels for different eye regions, and the 'level' column (B) contains numerical values. A dialog box titled 'analysis' is open, showing the distribution of values in the 'image' column. The dialog box has tabs for 'Filter by Value', 'Filter by Color', and 'Number Filter'. The 'Filter by Value' tab is selected, and the 'Label' list shows the following distribution:

Label	Count
(Select all)	35126
0	25810
1	2443
2	5292
3	873
4	708

The dialog box also has buttons for 'Top 10', 'Above Average', and 'Below Average', and 'OK' and 'Cancel' buttons at the bottom.

So, we have taken two methodologies to tackle this situation and get our model trained with highest accuracy as possible.

Step 1:

In this step we labelled the data as how it is actually lebeled in the trainLebel.csv file, and taken the least present class data as a standard number say N, and then gathered N number of Image data from all the classes.

```
//just to avoid data imbalance
```

1.Algorithm for data collection:

```
N <- calculate_number_Least(Class0,Class1,Class2,Class3,Class4) //calculating least no of class data
```

```
for i lt N
```

```
do
```

```
if getImage(“train_data_folder”).getLebel() = [0,1,2,3,4]
```

```
then
```

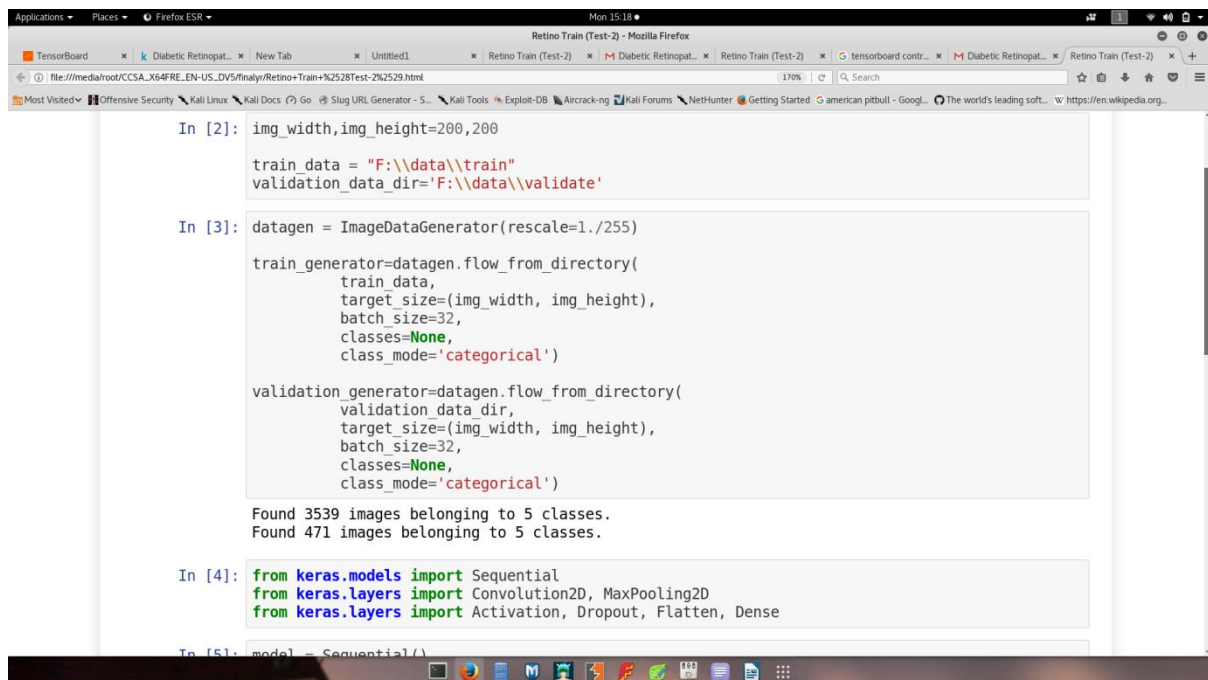
```
    this.lebel <- this.lebel + 1
```

```
    saveImagetoDir(/'+this.Lebel+''+this.Image)
```

```
done
```

this algorithm will put labelled data to their respective labeled folder.

2. Algorithm for Classified Labeling the dataset:

A screenshot of a Jupyter Notebook interface. The top part shows a browser window with multiple tabs, including 'TensorBoard', 'Diabetic Retinopat...', 'New Tab', 'Untitled1', 'Retino Train (Test-2)', 'Diabetic Retinopat...', 'Retino Train (Test-2)', 'tensorboard contr...', 'Diabetic Retinopat...', and 'Retino Train (Test-2)'. The address bar shows a file path: 'file:///media/root/CCSA_X64FRE_EN-US_DV5/finalyr/Retino+Train+%2528Test-2%2529.html'. Below the browser window, the Jupyter Notebook interface is visible, showing a code cell with the following Python code:

```
In [2]: img_width, img_height=200,200
train_data = "F:\\data\\train"
validation_data_dir='F:\\data\\validate'

In [3]: datagen = ImageDataGenerator(rescale=1./255)
train_generator=datagen.flow_from_directory(
    train_data,
    target_size=(img_width, img_height),
    batch_size=32,
    classes=None,
    class_mode='categorical')

validation_generator=datagen.flow_from_directory(
    validation_data_dir,
    target_size=(img_width, img_height),
    batch_size=32,
    classes=None,
    class_mode='categorical')

Found 3539 images belonging to 5 classes.
Found 471 images belonging to 5 classes.

In [4]: from keras.models import Sequential
from keras.layers import Convolution2D, MaxPooling2D
from keras.layers import Activation, Dropout, Flatten, Dense

In [5]: model = Sequential()
```

step 2:

In this step, we labelled the data randomly.

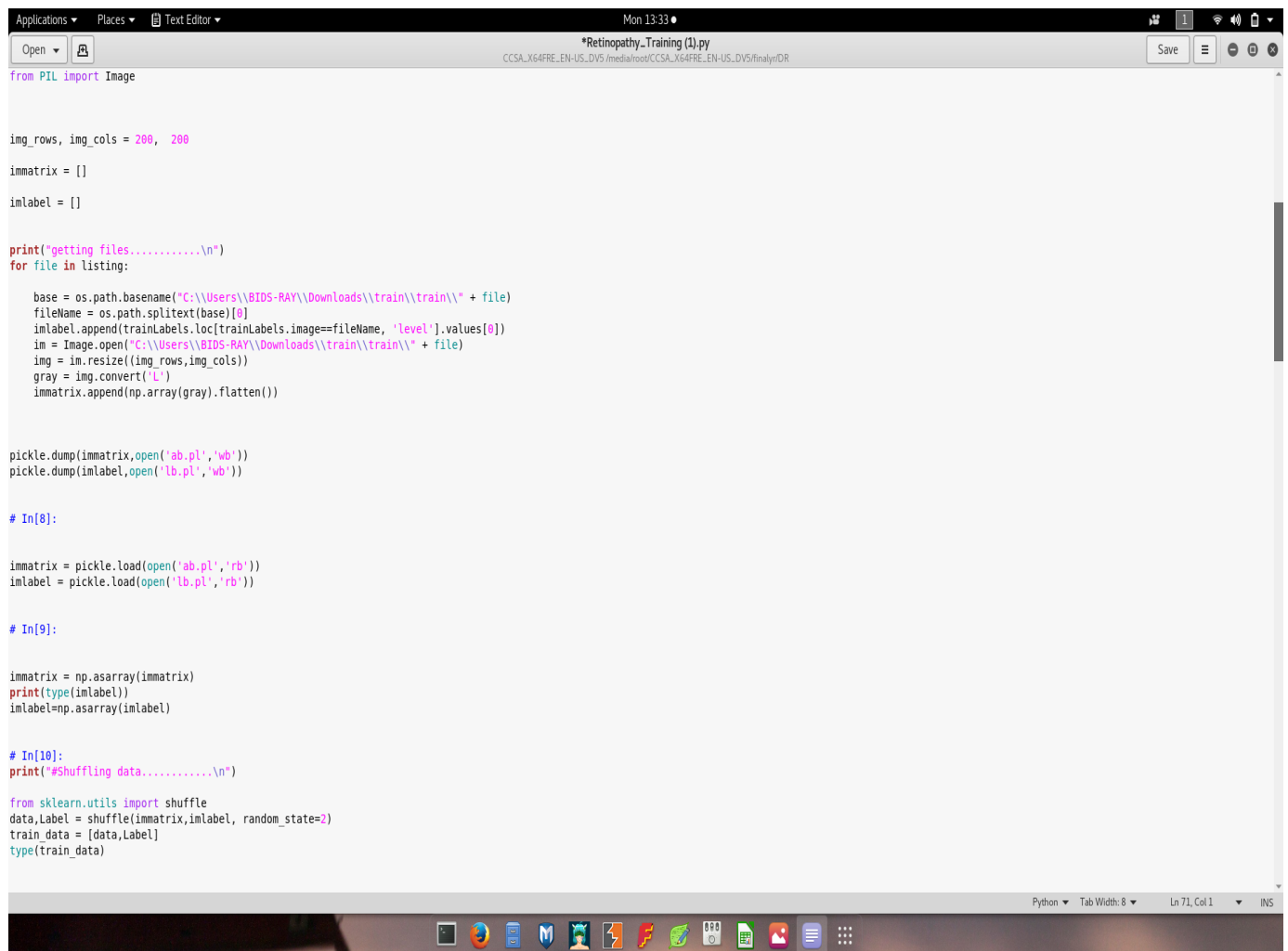
Mission of this step: To train and balance the weight, biases of the neural network towards perfection such that the learning rate increase and provide better and greater accuracy.

1. Algorithm for Random Labelling:

```
for i in image.getImage(fromDirectory):
```

```
    i.setLabel(random[0,1,2,3,4])
```

```
save(i,i.label.toCSV())
```



The screenshot shows a Windows Text Editor window titled '*Retinopathy_Training (1).py'. The window contains a Python script for processing retinopathy training data. The script imports PIL Image and uses os.path to handle file paths. It defines image dimensions (200x200) and initializes an immatrix and imlabel list. The script then iterates over files in a directory, loading each image, converting it to grayscale, and flattening it into a matrix. The matrices are then saved using pickle. Finally, the data is shuffled using sklearn.utils.shuffle and split into training data.

```
from PIL import Image

img_rows, img_cols = 200, 200

immatrix = []
imlabel = []

print("getting files.....\n")
for file in listing:

    base = os.path.basename("C:\\Users\\BIDS-RAY\\Downloads\\train\\train\\" + file)
    fileName = os.path.splitext(base)[0]
    imlabel.append(trainLabels.loc[trainLabels.image==fileName, 'level'].values[0])
    im = Image.open("C:\\Users\\BIDS-RAY\\Downloads\\train\\train\\" + file)
    img = im.resize((img_rows, img_cols))
    gray = img.convert('L')
    immatrix.append(np.array(gray).flatten())

pickle.dump(immatrix, open('ab.pl', 'wb'))
pickle.dump(imlabel, open('lb.pl', 'wb'))

# In[8]:

immatrix = pickle.load(open('ab.pl', 'rb'))
imlabel = pickle.load(open('lb.pl', 'rb'))

# In[9]:

immatrix = np.asarray(immatrix)
print(type(imlabel))
imlabel=np.asarray(imlabel)

# In[10]:
print("#Shuffling data.....\n")

from sklearn.utils import shuffle
data, Label = shuffle(immatrix, imlabel, random_state=2)
train_data = [data, Label]
type(train_data)
```

Algorithm to avoid bottleneck and maintaining flow in the network:

ImageDatagenerator:

```
validationdatagenerator <-ImageDataGenerator()  
traindatagenerator<-ImageDataGenerator(width_shift_range<-0.1,height_shift_range<-  
0.1,rotation_range<-15,zoom_range<-0.1 )  
  
batchsize<-15  
train_generator<-traindatagenerator.flow(X_train, Y_train, batch_size<-batchsize)  
validation_generator<-validationdatagenerator.flow(X_test, Y_test,batch_size<-batchsize)
```

Convolution Neural Network:

//algorithm for generating deep neural network

```
model <-Sequential()
```

//layer1

```
model.add(Convolution2D(nb_filters, nb_conv, nb_conv, border_mode<-'valid',input_shape<-(  
img_cols, img_rows, 1)))  
convout1 <-Activation('sigmoid')  
model.add(convout1)  
model.add(MaxPooling2D(pool_size<-(nb_pool, nb_pool)))
```

```
//Layer2
model.add(Convolution2D(nb_filters, nb_conv-1, nb_conv-1))
convout2 <-Activation('sigmoid')
model.add(convout2)
model.add(MaxPooling2D(pool_size<-(nb_pool, nb_pool)))
model.add(Dropout(0.2))
```

```
//Layer3
model.add(Convolution2D(nb_filters, nb_conv, nb_conv))
convout3 <-Activation('sigmoid')
model.add(convout3)
model.add(MaxPooling2D(pool_size<-(nb_pool, nb_pool)))
model.add(Dropout(0.2))
```

Connected Neural Network:

```
model.add(Flatten())

model.add(Dense(128,activation<-'sigmoid'))

model.add(Dense(5,activation<-'softmax'))
```

Compiling the Model:

```
model.compile(optimizer<-'adam', loss<-'binary_crossentropy', metrics<-['accuracy'])
```

Fitting the model:

```
model.fit_generator(train_generator,shuffle<-True, steps_per_epoch<-int(len(X_train)/batchsize),  
epochs<-10, validation_data<-validation_generator, validation_steps<-  
int(len(X_test)/batchsize),callbacks<-[tensorboard],verbose<-1)
```

Evaluation of model:

```
score <- model.evaluate(X_test, Y_test, verbose<-1)
```

CHAPTER 6

IMPLEMENTATION DETAILS :

The data has been provided by Eyepacs along with their labels.

Observation :

Visualiation :

My WPS × trainLabels.csv *

A1fximage

	A	B	C	D	E	F	G	H
1	image	level						

AscendingDescendingColor SortAdvance Mode

Filter by ValueFilter by ColorNumber FilterClear

LabelOptions

☒ (Select all) (35126)

☒ 0 (25810)

☒ 1 (2443)

☒ 2 (5292)

☒ 3 (873)

☒ 4 (708)

Top 10

Above Average

Below Average

analysis

OKCancel

25	30_right	2						
26	31_left	0						
27	31_right	0						
28	33_left	0						

We found that the data given by Eyepacs has disbalancing on them i.e. the class distribution of data is random.

Now to rectify this problem we follow 2 processes

- 1.Classified Labelling
- 2.Random Labelling

In classified labelling we acquire the least no. Of images in anyone label i.e. Level 4 here(708 images) & acquire the same no. From all the other classes too.

This has an accuracy model of 74% ,so its rather inefficient.

In Random Labelling we use two processes with a large no.of image number difference in between them.

In 1st test case we use 15,000 images & in latter we use 35,000 images.

In Random labelling we get an accuracy model of 89.65% accuracy.

Now our operational process has been outlined below ,this is given in general form ,

First images are fed to the CNN in batch sizes of 10 ,then strided filters of weight 3x3 are used .This is used for feature extraction .It helps in establishing as much parameters as can be deduced

This gives an activation map .

We then use Max Pooling to reduce the size of the image & also reduce the number of parameters to get a more efficient result.

Next we again sent the results to convolution layer.

This coitnues to happen with altered parameters & weights with different batches until full feature extraction has taken place.

Then dense i.e. fully connected layer occurs & all the data gained to this point is pipelined to the final nodes.

The convolution data is next sent to Normal Neural Network.

This neural network establishes prediction which is simply not possible in CNN. The neural network continuously changes the weights & bias .It tries toget as much accurate result as possible.

Using Relu function training is done with corrections being done at every layer.

After all the batches are iterated & epochs are completed ,Sigmoid function are used to establish Nonlinearity & probabilistic model to model the classes.

CHAPTER 7

Sample Output :

We have done comprehensive research work step by step to achieve greater Accuracy in our model.

Step 1:

Method: "Classified lebel based Approach"

Training Data: 3500

Validation [Data:470](#)

Activation Function: All Sigmoid

Output:

```
CA\Windows\System32\cmd.exe - python "Retino+Train+%28Test-2%29.py"
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\BIDS-RAY\Desktop\finalyr\step2>python "Retino+Train+%28Test-2%29.py"
File "Retino+Train+%28Test-2%29.py", line 96
    model.fit_generator(train_generator.shuffle=True, shuffle=True, steps_per_epoch=int(nb_train_samples/10), epochs=3, validation_data=validation_generator, validation_steps=int(nb_validation_samples/10), verbose=1, callbacks=[tensorboards])
                                         ^
SyntaxError: keyword argument repeated

C:\Users\BIDS-RAY\Desktop\finalyr\step2>python "Retino+Train+%28Test-2%29.py"
Using TensorFlow backend.
Found 3539 images belonging to 5 classes.
Found 471 images belonging to 5 classes.
Retino+Train+%28Test-2%29.py:57: UserWarning: Update your 'Conv2D' call to the Keras 2 API: 'Conv2D(2, (3, 3))'
    model.add(Convolution2D(32,3,3))
Retino+Train+%28Test-2%29.py:62: UserWarning: Update your 'Conv2D' call to the Keras 2 API: 'Conv2D(2, (3, 3))'
    model.add(Convolution2D(32,3,3))
WARNING:tensorflow:From F:\Anaconda\lib\site-packages\tensorflow\contrib\learn\python\learn\datasets\base.py:198: retry (from tensorflow.contrib.learn.python.learn.datasets.base) is deprecated and will be removed in a future version.
Instructions for updating:
Use the retry module or similar alternatives.
2018-05-13 22:50:07.986191: I tensorflow/core/platform/cpu_feature_guard.cc:140] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX
Epoch 1/3
13/353 [>.....] - ETA: 1:14:09 - loss: 0.6266 - acc: 0.7421_
```

Observation of Step 1:

So, by this `method our training model got an accuracy of 75% approximately,

Step 2:

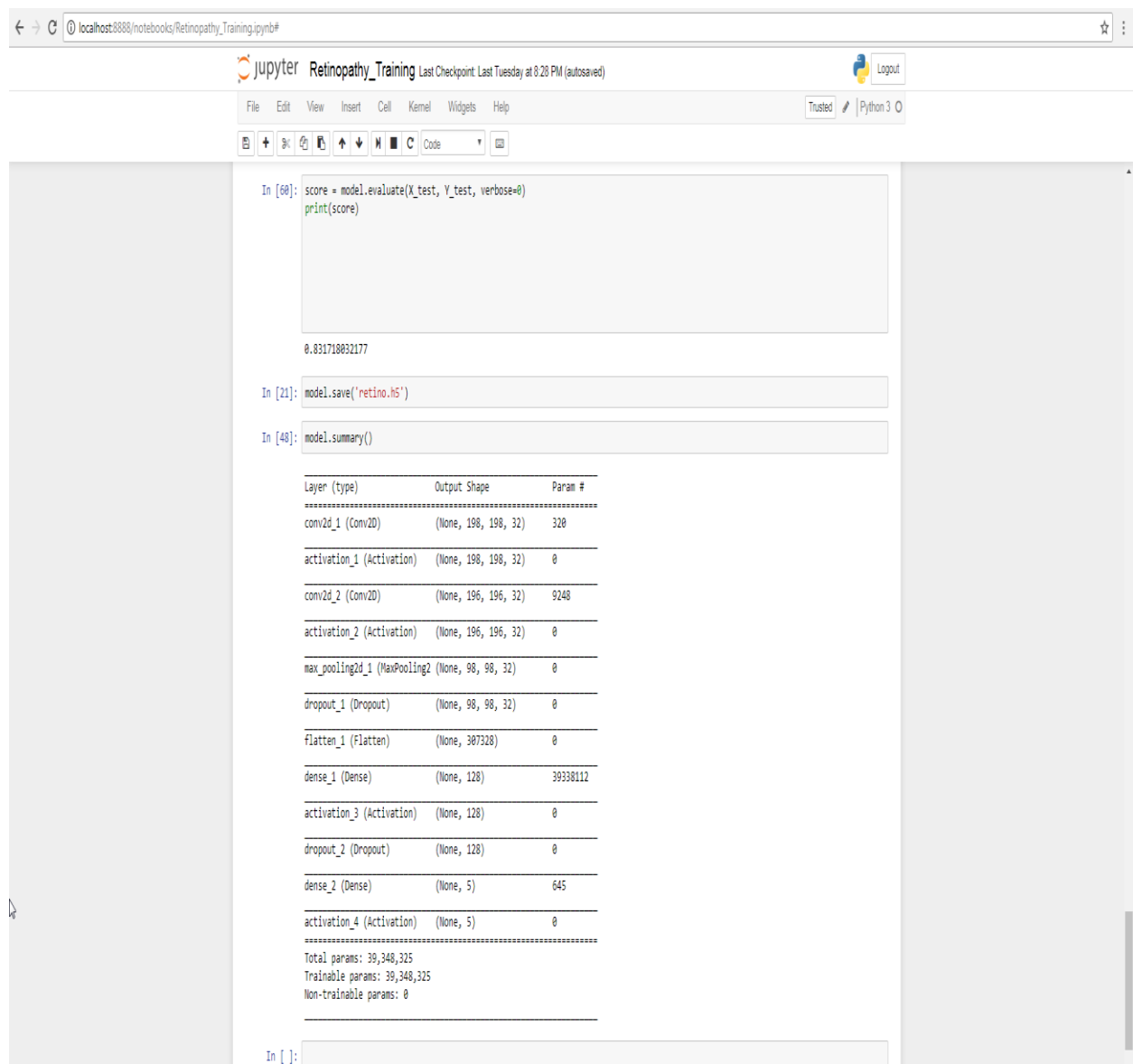
Method: “Random lebel tagging Approach”

Training Data: 16000

Validation [Data](#):2000

Activation Function: All Sigmoid, Final Layer Relu

Output :



```
In [60]: score = model.evaluate(X_test, Y_test, verbose=0)
print(score)

0.8317108632177

In [21]: model.save('retino.h5')

In [48]: model.summary()
```

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 198, 198, 32)	320
activation_1 (Activation)	(None, 198, 198, 32)	0
conv2d_2 (Conv2D)	(None, 196, 196, 32)	9248
activation_2 (Activation)	(None, 196, 196, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 98, 98, 32)	0
dropout_1 (Dropout)	(None, 98, 98, 32)	0
flatten_1 (Flatten)	(None, 387328)	0
dense_1 (Dense)	(None, 128)	39338112
activation_3 (Activation)	(None, 128)	0
dropout_2 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 5)	645
activation_4 (Activation)	(None, 5)	0

Total params: 39,348,325
Trainable params: 39,348,325
Non-trainable params: 0

Observation of Step 2:

So, by this `method our training model got an accuracy of 84% approximately, That is, this approach works better, our next step used 36K dataset with combinations of activation functions.

Step 3:

Method: “Random lebel tagging Approach”

Training Data: 36000

Validation [Data:8000](#)

substep:3.1

Activation Function: All Relu

32/286 [==>.....] - ETA: 14s
64/286 [=====>.....] - ETA: 13s
96/286 [=====>.....] - ETA: 11s
128/286 [======>.....] - ETA: 9s
160/286 [======>.....] - ETA: 7s
192/286 [======>.....] - ETA: 5s
224/286 [======>.....] - ETA: 3s
256/286 [======>....] - ETA: 1s
286/286 [=====] - 17s 60ms/step
[0.95570885468196198, 0.74125874000829417]

Layer (type)	Output Shape	Param #
=====		
conv2d_1 (Conv2D)	(None, 198, 198, 32)	320
=====		
activation_1 (Activation)	(None, 198, 198, 32)	0
=====		
max_pooling2d_1 (MaxPooling2D)	(None, 99, 99, 32)	0
=====		
conv2d_2 (Conv2D)	(None, 98, 98, 32)	4128
=====		

activation_2 (Activation)	(None, 98, 98, 32)	0
<hr/>		
max_pooling2d_2 (MaxPooling2)	(None, 49, 49, 32)	0
<hr/>		
dropout_1 (Dropout)	(None, 49, 49, 32)	0
<hr/>		
conv2d_3 (Conv2D)	(None, 47, 47, 32)	9248
<hr/>		
activation_3 (Activation)	(None, 47, 47, 32)	0
<hr/>		
max_pooling2d_3 (MaxPooling2)	(None, 23, 23, 32)	0
<hr/>		
dropout_2 (Dropout)	(None, 23, 23, 32)	0
<hr/>		
flatten_1 (Flatten)	(None, 16928)	0
<hr/>		
dense_1 (Dense)	(None, 128)	2166912
<hr/>		
dense_2 (Dense)	(None, 5)	645
<hr/>		
=====		
Total params: 2,181,253		
Trainable params: 2,181,253		
Non-trainable params: 0		

Observation of Step 3.1:

So, by this `method our training model got an accuracy of 74.12% approximately, .

Step 3:

Method: “Random label tagging Approach”

Training Data: 36000

Validation [Data:8000](#)

substep:3.2 Activation Function: All Relu, Final Sigmoid

[0.30201189897277136, 0.896503469327113]

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 198, 198, 32)	320
activation_1 (Activation)	(None, 198, 198, 32)	0
max_pooling2d_1 (MaxPooling2)	(None, 99, 99, 32)	0
conv2d_2 (Conv2D)	(None, 98, 98, 32)	4128
activation_2 (Activation)	(None, 98, 98, 32)	0

Observation of Step 3.2:

max_pooling2d_2 (MaxPooling2) (None, 49, 49, 32) 0
So, by this `method our training model got an accuracy of 89.65% approximately, .

dropout_1 (Dropout)	(None, 49, 49, 32)	0
conv2d_3 (Conv2D)	(None, 47, 47, 32)	9248
activation_3 (Activation)	(None, 47, 47, 32)	0

Step 3:

Method: “Random lebel tagging Approach”

Training Data: 36000

Validation [Data:8000](#)

substep:3.2 Activation Function: All Sigmoid

[0.30030625058220817, 0.897503469327113]

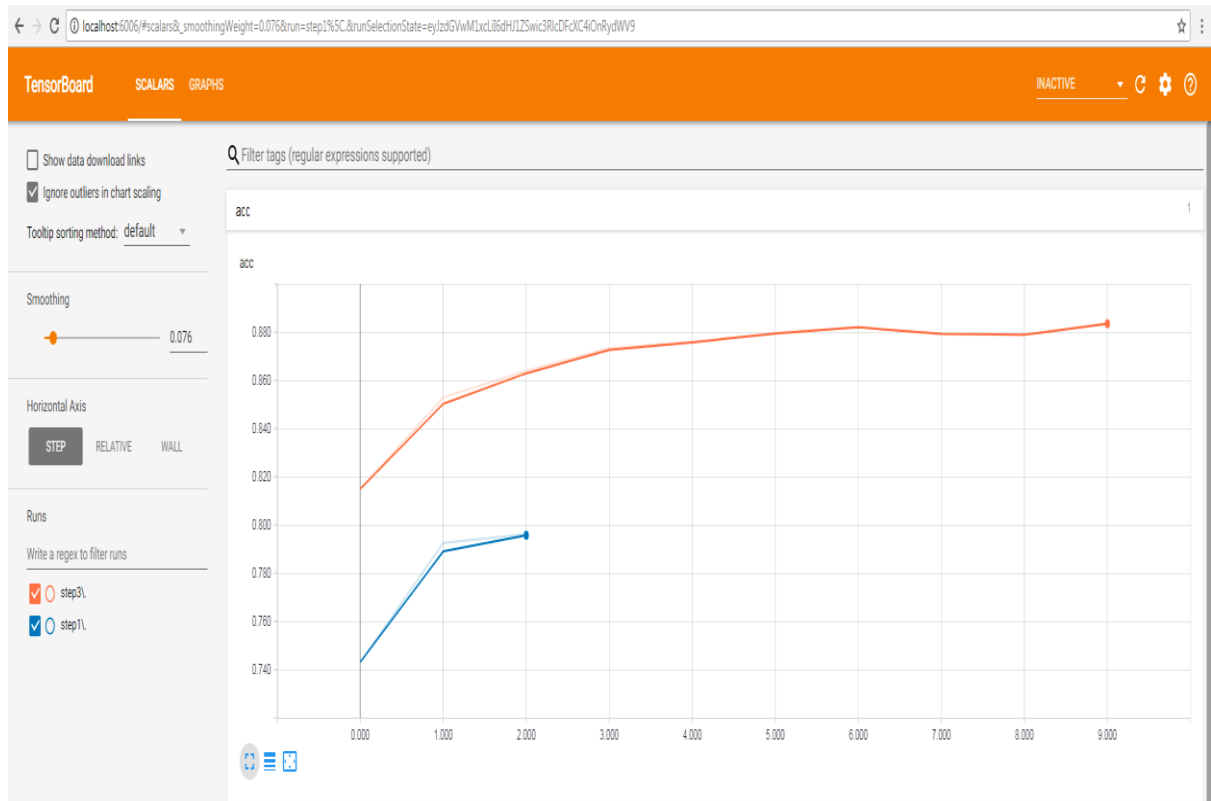
Layer (type)	Output Shape	Param #
=====		
conv2d_1 (Conv2D)	(None, 198, 198, 32)	320
activation_1 (Activation)	(None, 198, 198, 32)	0
max_pooling2d_1 (MaxPooling2)	(None, 99, 99, 32)	0
conv2d_2 (Conv2D)	(None, 98, 98, 32)	4128
activation_2 (Activation)	(None, 98, 98, 32)	0

Observation of Step 3.3:

max_pooling2d_2 (MaxPooling2 (None, 49, 49, 32) 0
So, by this `method our training model got an accuracy of 89.75% approximately,

dropout_1 (Dropout)	(None, 49, 49, 32)	0
conv2d_3 (Conv2D)	(None, 47, 47, 32)	9248
activation_3 (Activation)	(None, 47, 47, 32)	0

Accuracy gap Between Our 1st Model to 3.3rd Model:



Value Loss gap Between Our 1st Model to 3.3rd Model:

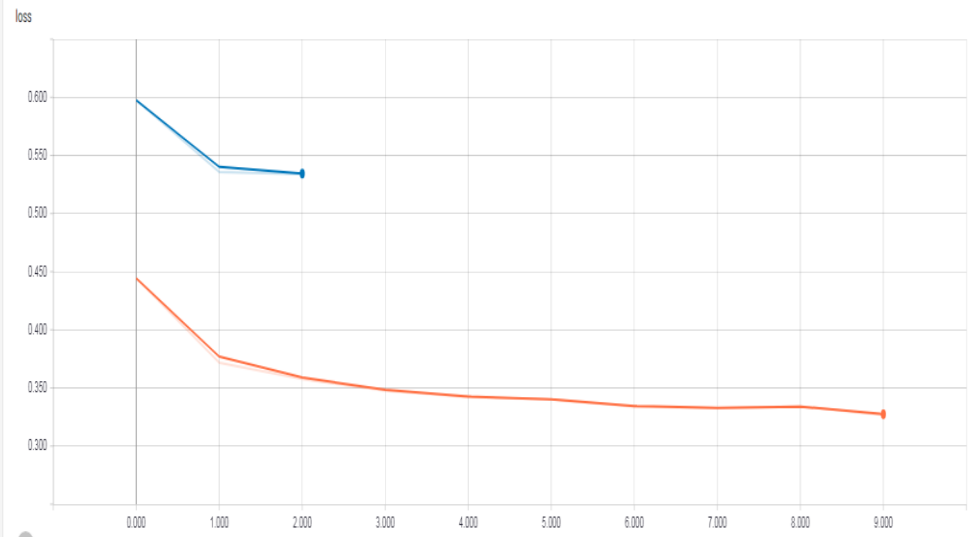
Runs

Write a regex to filter runs

- ☒ step3/
- ☒ step1/

TOGGLE ALL RUNS

step3 C:\Users\BIDS-RAY\Desktop\aa
step1 C:\Users\BIDS-RAY\Desktop\final\step2



val_acc	1
val_loss	1

Chapter 8

Results

Machine learning is the method to predict the possible future interactions among the nodes in the near future.

The algorithm is implemented in Jupyter. 36,000 Retinal images from Eyepacsc data base, containing grade 0, grade1, grade2 ,grade 3 and grade 4 images are tested with the algorithm for the machine learning using Convolutional Neural Network.

The algorithm has yielded an accuracy of 89.61%.

Bibilography

[1] kaggle Challenge Diabetic Retinopathy Detection. <https://www.kaggle.com/c/diabetic-retinopathy-detection>

[2] Wong Li Yun , U. Rajendra Acharya, Y.V. Venkatesh, Caroline Chee, Lim Choo Min, E.Y.K. Ng Identification of different stages of diabetic retinopathy using retinal optical images. July 2007

[3] Jagadis h Naya k, P Subba nna Bhat, Rajen dra Achar ya U,C. M. Lim, Manjunath Kagathi Automated Identification of Diabetic Retinopathy Stages Using Digital Fundus Images November 2007

[4] Diabetic Retinopathy http://en.wikipedia.org/wiki/Diabetic_retinopathy

[5] Morphological operations http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html#morphological-ops

