Diabetic Retinopathy Prediction

using

Convolutional Neural Networks

A Project Report

submitted by

RAJIB DAS BHAGAT

in partial fulfilment of the requirements

for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

April 2019

THESIS CERTIFICATE

This is to certify that the thesis titled **Diabetic Retinopathy Prediction using Convolutional Neural Networks**, submitted by **Rajib Das Bhagat (CS17M034)**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

B. Ravindran

Prof. Balaraman Ravindran
Project Guide,
Professor,
Dept. of Computer Science and Engineering,
IIT-Madras, 600 036.

Place: Chennai

Date: 04-05-2019

ACKNOWLEDGEMENTS

I am very much thankful to my project guide Prof. Balaraman Ravindran for his prolific encouragement and guidance. He consistently allowed this thesis to be my own work, and steered me in the right direction whenever he thought I needed it.

I would like to extend my profound gratitude to the faculty at the Department of Computer Science and Engineering, and our honourable Director Prof. Bhaskar Ramamurthi, for their broad and liberal views towards promoting students to explore their curricula beyond defined horizons. Also, I would like to extend my sincere esteems to all staff in the laboratory for their untimely support.

I, earnestly thank members of the Reconfigurable and Intelligent Systems Engineering Group (RISE lab), IIT Madras for providing access to GPU cluster and providing necessary information regarding the same to run my experiments for the project.

I would also like to thank my friends and Mr. Karthik Thiagarajan (MS Scholar, Dept. of Computer Science & Engineering, IIT Madras) and Mr. Saurabh Desai (Project Associate, RBC-DSAI laboratory, IIT Madras) for making the 2 years of my stay here most cherishable and helping me out whenever I was stuck. Their presence unfolded numerous eventful experiences that I will always remember.

Finally, I express my very profound gratitude to my parents and to my extended family for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Rajib Das Bhagat

ABSTRACT

KEYWORDS: Diabetic Retinopathy, Convolutional Neural Network, Diabetes, Image Pre-processing, Feature Extraction, Image Classification, Eye.

Diabetic Retinopathy can be termed as any damage caused to the retina of an eye, which causes vision impairment to the people suffering from diabetes. Diabetic Retinopathy refers to retinal vascular disease, or damage to the retina caused by abnormal blood flow. Identifying features related to abnormalities in a retinal image is a tedious process that requires expert knowledge. This work develops a solution that automates the diagnosis of diabetic retinopathy using deep learning based techniques trained on retinal images. Firstly, the diagnosis is formulated as an end-end binary classification problem that tries to detect the presence or absence of the disease given a retinal image using a Convolutional Neural Network (CNN). Secondly, this solution is extended to automate the more nuanced diagnosis of predicting the severity of the disease, and is formulated as a multi-class classification problem. The performance of any trained models are evaluated using metrics such as precision, recall and F1 score.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Deep learning is the study of a class of machine learning models called deep neural networks that try to learn a hierarchy of features from data. Deep learning is an application of artificial intelligence (AI) that frameworks the capacity to consequently take in and improve for a fact without being unequivocally modified. AI centers around the advancement of computer programs that can get information and use it to learn for themselves.

The way toward learning begins with perceptions or information, for example, models, direct involvement, or guidance, so as to search for examples/patterns in information and settle on better choices. The essential point is to allow the computer to learn and adapt consequently without human assistance or help and adjust accordingly. Consider the scenario of a person suffering from diabetic retinopathy. As per medical reports, patients suffering from diabetic retinopathy are highly susceptible to vision loss (11). This complication occurs in individuals with high glucose levels, where an excessive accumulation of glucose damages tiny blood vessels in the retina . This entanglement, can lead to prompt swelling and blood vessels can spill out of the veins and cause permanent damage to the retina.

1.2 Problem Statement

Due to the severity of the disease, a diabetic person may completely lose vision and might lead to other complications. Thus, the study is to explore how we can automate the process of correctly classifying retinal images into different stages of DR and take precautionary measures before hand.

In this thesis, deep learning models (CNNs) are trained on retinal images in two settings: binary and multi-class classifications. In the first phase of the project, an understanding of deep learning models was obtained and a pipeline for pre-processing retinal images was setup. While in the second phase, experiments for the binary classification were performed. For the final phase of the project, experiments on multi-class classification on the fundus images were performed.

1.3 Diabetic Retinopathy and Stages

At first, diabetic retinopathy may cause no symptoms or just mild vision issues. The longer a person suffering from diabetes and the less controlled the blood sugar is, the more likely one is to build up this eye complication. In brief, it can cause visual impairment.

In non-proliferative DR [Table 1.1], the walls of the blood vessels of the retina weakens. Micro-aneurysms develops from the vessel walls of the smaller vessels, sometimes leaking fluid and blood into the retina. Larger retinal vessels can begin to dilate and become irregular in shape, as well. Non-proliferative DR can progress from mild to severe, as more blood vessels become blocked. Nerve fibers in the retina may begin to swell, and the central part of the retina (macula) begins to swell (macular edema) too.

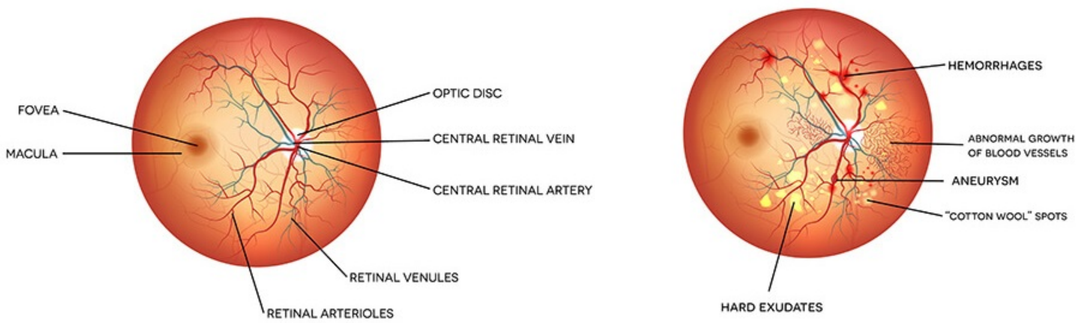
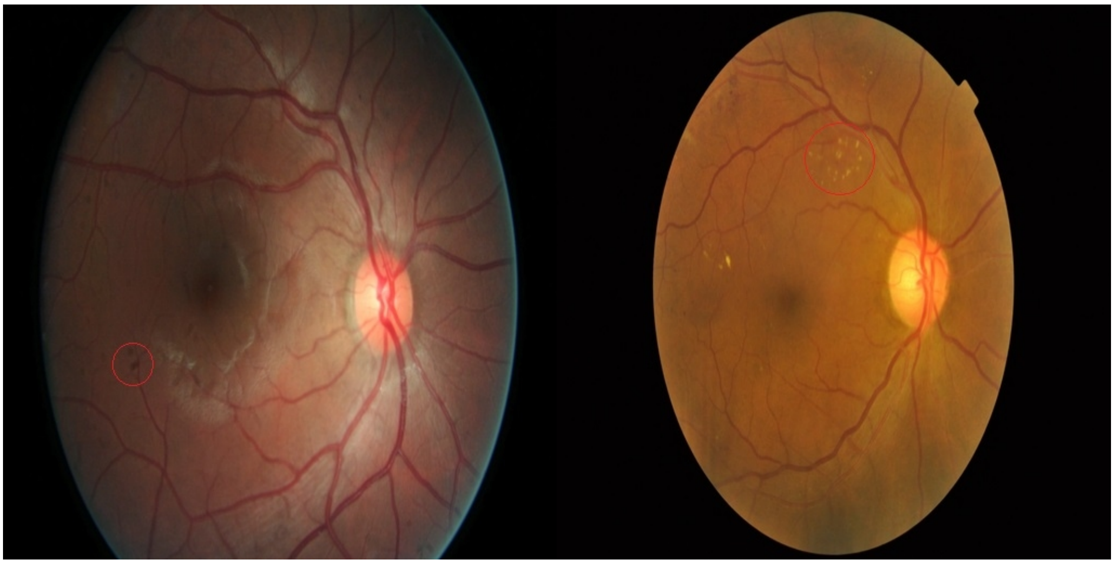


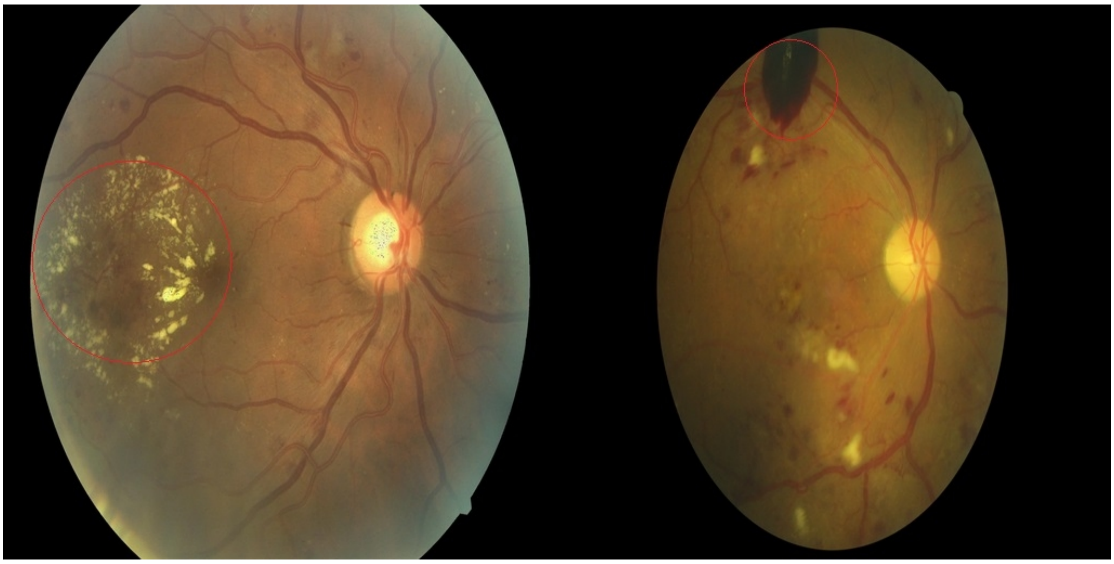
Figure 1.1: A normal eye [left] and related abnormalities [right] (12).

In this proliferative type [Table 1.1], damaged blood vessel clogs off, causing the growth of new abnormal blood vessels in the retina, and can leak lipids, a jelly-like substance that fills the center of an eye (vitreous).

Eventually, the growth of new blood vessels may cause the retina to detach from the back. Pressure may build up in the eye, causing damages to the nerve that carries images from your eye to your brain (optic nerve), resulting in glaucoma.



(a) Mild (b) Moderate



(c) Severe (d) Proliferative

Figure 1.2: Fundus image of DR stages [related abnormalities are circled] (13)

1.3.1 DR Stages

The different stages related to diabetic retinopathy (7) is described below.

i. Normal

An eye with no nearness of irregular sores or any other kind of deformities is characterized as a normal eye [Fig.1.1 (left)].

ii. Mild non-proliferative

The earliest visible detectable lesions are micro-aneurysms. Micro-aneurysms are usually round, red, tiny dots sore like, which may leak fluid into the retina. Micro-aneurysms may occur in a group or in isolation, this is classified as mild non-proliferative [Fig.1.2a].

iii. Moderate non-proliferative

As a mild diabetic retinopathy progresses, the blood vessels that nourish the retina may swell or distort a bit. The blood vessel loses their ability to transport blood. The lesion present in this is known as soft exudates. The lesion is like cotton wool spots or micro-infarctions. Soft exudates are small, white lesions not clearly visible and are termed as a moderate non-proliferative [Fig.1.2b].

iv. Severe non-proliferative

Lesion present as hard exudates spills lipid from the debilitated blood vessels. Lesions are ordinarily yellow colored and clear edges are waxy alike. Hard exudates tend to accumulate in groups. This stage is severe non-proliferative [Fig.1.2c].

v. Proliferative

When micro-aneurysm ruptures, spillage of blood occurs from the blood vessels. This is known as hemorrhages. Hemorrhages are the red dot, flame-like presence in cluster or rings. New blood vessels also tend to develop, which are typically weak and gets torn off easily. This draining can cause perpetual vision loss. This happens due to the absence of oxygen and known as neovascularization. The presence of hemorrhage and neovascularization is the final stage of DR and is termed as proliferative [Fig.1.2d].

Type	**Stage**	**Size**	**Shape**	**Color**
Micro-aneurysm	mild	tiny	round	darkish red
Soft Exudate	moderate	small to medium	oval	whitish
Hard Exudate	severe	varies	irregular	yellow
Hemorrhage	proliferative	varies	dot or flame like	darkish red
Neovascularization	proliferative	varies	varies	red

Table 1.1: Brief details of DR stages and lesions

1.4 Structure of the Thesis

The different abnormalities identified and related to DR are shown in Fig. 1.1 [right] and in fundus images [Fig. 1.2]. In this report, section 1 incorporate the introduction with a brief explanation of an eye structure and DR classification, section 2 relates to a brief rewind of a convolutional neural network architecture and related issues. Section 3 presents the literature survey and reviews in the related field. The dataset used is described in section 4. Section 5 highlights the methodology and strategies used. Section 6 deals with various experiments performed as an initial and final phase. While section 7 previews the results both for binary and multi-class classification and section 8 concludes with a conclusion.

CHAPTER 2

BACKGROUND

2.1 Convolutional Neural Network (CNNs)

In neural systems, convolutional neural system (ConvNets or CNNs) is one of the principle classifications to do image recognition, image classifications. CNN image classifications takes an input image, process it and classifies depending on the input and activation function.

A CNN comprises of convolution layers, pooling layers and fully connected layers. A conventional layer applies operations known as a convolution operation to the input and pass the result to the next layer. Each neuron processes data in its local receptive fields. Series of steps involved are feed-forwarding the input values and back-propagating while updating and minimizing the parameters termed as weights and bias. And, the accuracy is finally calculated.

A convolutional neural network is used in various applications such as image and speech recognition and natural language processing including medical imaging too.

The basis working steps for convolutional neural network architecture are further described below.

2.1.1 Local Receptive Fields

Input size of n x n neurons is fed into the CNN. These n x n neurons are termed as input pixels. Input pixels are connected to the first hidden layers comprising of many hidden neurons. Here, every input neurons are not connected to every hidden neuron. Generally, only a few connections are made in small. These small connections are based on localized regions for the input neurons.

The localized region for the input neurons is the local receptive field for the hidden neurons. This is like a small window on the input neurons present. Thus, the connections from input neurons to hidden neurons represents both weights and biases. The local receptive field is slided and the output is different for different hidden neurons. The sliding of the local receptive field is done in accordance with stride length. Stride can be termed as the number of pixels need to be slided for the small window present. If stride length is 1, 2, ..., n; the local receptive field can be slide by 1, 2, ..., n neurons respectively.

2.1.2 Filters

Filter is the mapping of the input layer to the next hidden layer. The weights present in the filter is termed as shared weights, while the bias is termed as shared bias. This shared weights and bias is defined as a kernel or filter. Filters are used to blur images, sharpen images, and also performs edge detection. In CNN, filters are not pre-defined. The weights in each filter is learned as the training process is carried on.

2.1.3 Pooling

Pooling layers are used after each and every convolution layer. Pooling technique combines the neuron cluster output at one layer into a single neuron in the next layer. The feature use of pooling is to simplify the output from the convolutional layer. Pooling progressively reduces the number of parameters and the huge number of computation associated with CNN. This simplification is done to generate a compressed model of the feature map. The pooling techniques available are max-pooling, average-pooling and l2-pooling.

2.2 Issues in CNN

Neural networks are fundamentally difficult to train, because of many known issues included. Few of the issues related are narrated and some of the compact solutions are described here.

While a model starts to learn, after sometime it happens that a CNN stops learning. This issue is termed as neuron saturation. Neuron saturation is a huge problem in neural networks. By using a proper choice of activation function such as ReLu or LeakyReLu, this neuron saturation issue can be solved. If the neurons saturates, the training process is stopped. This is modelled as early stopping.

The second issue in the neural system is overfitting. Overfitting is characterized as an error in training data when a model fits to the training data, but unable to generalize to new data i.e, the test data. The error tends to be huge. A general rule to prevent overfitting is to reduce the size of the network, expand the quantity of training data and while training instead of using test data, validation data to be used. While training no amount of test data to be used. Validation data is extracted from training data and also no similarity of the train data and validation data must be present. Accuracy can be computed at the end of each epoch on the validation data itself. Regularization too helps in conquering the overfitting issue. Types of different regularization are L1, L2 or mixed regularization; which is used in addition to a cost function.

Other known issues, such as a vanishing gradient problem (9) and an exploding gradient problem (9) also exists. Neurons learns much slowly in the earlier layer in contrast to neurons in the later layers. This is the vanishing gradient problem. In converse to this, is exploding gradient problem. Last, but not the least neural networks sometimes learns at an alternate pace and is known as an unstable gradient problem.

CHAPTER 3

RELATED WORK

Correctness in screening for diabetic retinopathy is the first and foremost essential perspective for further treatment. The identification precision is important for both cost and treatment effectiveness.

As expressed, a diabetic retinopathy classification for a physician is profoundly tedious. Giving an alternate substitution approach for a real-time classification is much significant for an available training data. Moreover, much work has been done on the classification of abnormal and normal diabetic retinopathy (1). The techniques used earlier were related to machine learning such as k-nearest neighbors (K-NN) classifiers and support vector machine (SVM) methods (4) (5) (6) were used.

A numerous amount of research work [Table 3.1] for DR classification such as binary classification (1), three-class classification (3) and five-class classification (4) (5) (6) has been carried out, and still in advancement on the kinds of characterization and classification of DR.

It tends to plainly observe [Table 3.1] that the significant part of the works have been carried out utilizing the SVM technique. This technique requires feature extraction methods before being feed into the SVM classifier. For five-class classification, features such as micro-aneurysm, exudates, hemorrhages were used (5). A very small number of the dataset were used for training, testing or validation purpose. For the real-time application, a convolutional neural network is much appropriate as far as characterization and classification, as well as prediction is concerned.

	Authors	**Paper**	**Year**	**Classification Type**	**Techniques Used**	**Accuracy**
1	G G Gardner, D Keating, T H Williamson, A T Elliott	Automatic detection of diabetic retinopathy using an artificial neural network:a screening tool (1)	1996	Binary	Neural Network	88.4%
2	Nayak J, Bhat PS, Acharya R, Lim CM, Kagathi M	Automated identification of diabetic retinopathy stages using digital fundus images (3)	2008	Three class	Neural Network	93.0%
3	Acharya UR, Chua CK, Ng EY, Yu W, Chee C	Application of higher order spectra for the identification of diabetes retinopathy stages (4)	2008	Five Class	SVM	82.0%
4	Acharya UR, Lim CM, Ng EY, Chee C, Tamura T	Computer-based detection of diabetes retinopathy stages using digital fundus images (5)	2009	Five Class	SVM	85.9%
5	P. Adarsh, D. Jeyakumari	Multiclass SVM-based automated diagnosis of diabetic retinopathy (6)	2013	Five Class	SVM	96.0%

Table 3.1: Literature Survey

CHAPTER 4

THE DATASET

4.1 Dataset Source

A competition previously hosted by Kaggle on the classification of DR. This dataset is the part of the competition. The image dataset was collected from Kaggle website (13). The website contains a total of 88,702 images for DR classification [Table 4.1] and the total size is of around 89 GB. The image dimensions are around 6M pixels per image. The image dimensions were really huge. The image's dimensions were re-sized to 512 x 512 pixels, in order to make-up for the memory issue. The re-sized images of testing and training data were barely 5.7 GB and 3.73 GB respectively. The training and testing files were sorted according to their respective categories as normal, mild, moderate, severe and proliferative DR from the original dataset [Table 4.2, 4.3].

	Count	**Size**	**Re-sized (512 x512)**
Test	53576	53.7 GB	5.70 GB
Train	35126	35.3 GB	3.73 GB
Total	88702	89.0 GB	9.43 GB

Table 4.1: Diabetic Retinopathy Dataset

	DR Type	**Count**
0	Normal	25810
1	Mild Non-proliferative	2443
2	Moderate Non-proliferative	5292
3	Severe Non-proliferative	873
4	Proliferative	708
Total		35126

Table 4.2: DR as per Training Dataset

	DR Type	Count
0	Normal	39533
1	Mild Non-proliferative	3762
2	Moderate Non-proliferative	7861
3	Severe Non-proliferative	1214
4	Proliferative	1206
Total		53576

Table 4.3: DR as per Testing Dataset

For binary classification, category [3,4] was grouped as abnormal and category [0] as normal was chosen for training and testing respectively. The fundamental reason for grouping the category [1,2], is that no visual symptoms is available for our naked eye to classify a particular given image. Hence, for the category [3,4] symptoms were clearly visible and it was grouped and chosen for training and testing purpose. Total of 3600 files chosen as training data and 400 was chosen for testing data for each category named abnormal and normal. Hence, the total of 8000 files was selected for training and testing purpose.

Relating to multi-class classification and in view to utilizing most of the dataset, few issues were generated. The related issues are described further and states how it was precisely handled.

4.2 Issues related to dataset:

From the Table 4.2, 4.3 provided above, it can also be seen that the classes are clearly imbalance. Imbalanced class typically refers to a problem with classification problems where the classes are not represented equally. The known issue with the imbalance classes is overfitting with respect to the highest number of instance class. Down-sampling of higher class as well as up-sampling of the lower classes were performed to address this issue. As the number of experiments increased, the process of down-sampling and up-sampling was performed repeatedly, to get some optimal results.

The dataset also contained lots of noisy data. Some images were corrupted, blur, not properly focused, thus making the pixel intensity varying a lot. Lighting effects also create unnecessary distortion. Few of the noisy images were manually removed.

CHAPTER 5

METHODOLOGY

5.1 Model Architecture

The convolutional neural network architecture used is shown in Table 5.1, this similar architecture was used by Harry Pratt (7). The architecture comprising of 10 convolutional layers and 3-fully connected layers. The input layer was fed with size 512 x 512 neurons. Each convolution layer is trailed by a leaky rectified linear unit (LeakyReLu) activation function, along with batch normalization. For pooling layers, a max-pooling technique is used with kernel size 3 x 3 and 2 x 2 strides. A kernel size is basically an odd number. Before adding to the fully-connected layer, the network from the final convolutional layer was flattened to one dimension.

Hyper-parameters was chosen dependent on experimentation strategy [Table 6.2]. For all the referred issues as stated earlier, hyper-parameters and other settings were done accordingly. The loss function, entropy cost function was used to address the saturation problem. Similarly, L2 regularization for weights and bias initialization and dropouts on the dense layer was used to abstain from overfitting. In the final hidden layer, a softmax activation was used.

The classification is binary classification as abnormal DR and normal DR, while multi-class classification as normal, moderate non-proliferative DR, mild non-proliferative DR, severe non-proliferative DR and proliferative DR. Also, a considerable lot of other related parameters were used in this convolutional neural network [Table 5.1]. The total number of trainable and non-trainable parameters are 7,867,165 and 2,040 respectively.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 32, 512, 512)	320
leaky_re_lu_2 (LeakyReLU)	(None, 32, 512, 512)	0
batch_normalization_1 (BN)	(None, 32, 512, 512)	2048
max_pooling2d_1 (MaxPooling2)	(None, 32, 256, 256)	0
conv2d_2 (Conv2D)	(None, 32, 256, 256)	9248
leaky_re_lu_3 (LeakyReLU)	(None, 32, 256, 256)	0
batch_normalization_2 (BN)	(None, 32, 256, 256)	1024
max_pooling2d_2 (MaxPooling2)	(None, 32, 128, 128)	0
conv2d_3 (Conv2D)	(None, 64, 128, 128)	18496
leaky_re_lu_4 (LeakyReLU)	(None, 64, 128, 128)	0
batch_normalization_3 (BN)	(None, 64, 128, 128)	512
max_pooling2d_3 (MaxPooling2)	(None, 64, 64, 64)	0
conv2d_4 (Conv2D)	(None, 64, 64, 64)	36928
leaky_re_lu_5 (LeakyReLU)	(None, 64, 64, 64)	0
batch_normalization_4 (BN)	(None, 64, 64, 64)	256
max_pooling2d_4 (MaxPooling2)	(None, 64, 32, 32)	0
conv2d_5 (Conv2D)	(None, 128, 32, 32)	73856
leaky_re_lu_6 (LeakyReLU)	(None, 128, 32, 32)	0
batch_normalization_5 (BN)	(None, 128, 32, 32)	128
max_pooling2d_5 (MaxPooling2)	(None, 128, 16, 16)	0
conv2d_6 (Conv2D)	(None, 128, 16, 16)	147584
leaky_re_lu_7 (LeakyReLU)	(None, 128, 16, 16)	0
batch_normalization_6 (BN)	(None, 128, 16, 16)	64
max_pooling2d_6 (MaxPooling2)	(None, 128, 8, 8)	0
conv2d_7 (Conv2D)	(None, 256, 8, 8)	295168
leaky_re_lu_8 (LeakyReLU)	(None, 256, 8, 8)	0
conv2d_8 (Conv2D)	(None, 256, 8, 8)	590080
leaky_re_lu_9 (LeakyReLU)	(None, 256, 8, 8)	0
batch_normalization_7 (BN)	(None, 256, 8, 8)	32
max_pooling2d_7 (MaxPooling2)	(None, 256, 4, 4)	0
conv2d_9 (Conv2D)	(None, 512, 4, 4)	1180160
leaky_re_lu_10 (LeakyReLU)	(None, 512, 4, 4)	0
conv2d_10 (Conv2D)	(None, 512, 4, 4)	2359808
leaky_re_lu_11 (LeakyReLU)	(None, 512, 4, 4)	0
batch_normalization_8 (BN)	(None, 512, 2, 4)	16
max_pooling2d_8 (MaxPooling2)	(None, 512, 2, 2)	0
dropout_1 (Dropout)	(None, 512, 2, 2)	0
flatten_1 (Flatten)	(None, 2048)	0
dense_1 (Dense)	(None, 1024)	2098176
leaky_re_lu_12 (LeakyReLU)	(None, 1024)	0
dropout_2 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 1024)	1049600
leaky_re_lu_13 (LeakyReLU)	(None, 1024)	0
dense_3 (Dense)	(None, 2)	2050
Total params: 7,869,205 Trainable params: 7,867,165 Non-trainable params: 2,040		

Table 5.1: Convolutional Neural Network Architecture

5.2 Hardware and Software:

For running our CNN, GPU cluster was used. The cluster has high-end GPU. Random GPU type available were used such as k20, k80, TitanX or 1080Ti. The GPU contains deep neural network libraries for machine learning and training, with TensorFlow support at the back-end. Along with GPU, keras package which is a high-level neural network API is used in order to train and test the neural network model. Keras documentation is available at https://keras.io.

CHAPTER 6

EXPERIMENTS

6.1 The Initial Phase

In the initial phase of experiment, the fundus images were classified as abnormal and normal. Due to the presence of noisy and unbalanced training data, fewer number of data was manually chosen for training. Around, 1200 images were selected from each category and 90:10 percent split was performed. The images were pre-processed before feeding it to the convolutional neural network. And, the process of augmentation was also applied.

6.1.1 Pre-processing

As described earlier, the images contained a huge amount of noise. Thus, Gaussian Blur color normalization [Fig. 3b] was implemented using OpenCV (https://opencv.org/) package numpy. The Gaussian blur is a type of image-blurring filter channel that uses a Gaussian function for calculating the transformation to apply to each pixel in the image.

The formula of a Gaussian function in two dimension is

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{(-\frac{x^2+y^2}{2\sigma^2})}$$

where, x = the distance from the origin in the horizontal axis,
y = the distance from the origin in the vertical axis, and
$\sigma =$ the standard deviation of the Gaussian distribution.

Here, along with color normalization, the images were re-sized to 512 x 512. The pre-processing algorithm (10) was used by Ben Graham; one of the winners in Kaggle diabetic retinopathy competition and the algorithm is stated as:

Algorithm 1 Gaussian Pre-processing

```
procedure PRE-PROCESSING(source):
for file in glob.glob(source):
    inputImage = cv2.imread(file, 1)
    resizedImage = cv2.imread(inputImage, (512, 512))
    gaussBlur = cv2.GaussianBlur(resizedImage, (0, 0), 10)
    processedImage = cv2.addweigted(resizedImage, 4, gaussBlur, -4, 128)
return processedImage
```

6.1.2 Augmentation

After pre-processing the images, the images were fed into the convolutional neural network. The pre-processed images were used for training the convolutional neural network. Later, during every epoch performed the images were randomly augmented. The augmentation performed details can be found in the Table 6.1.

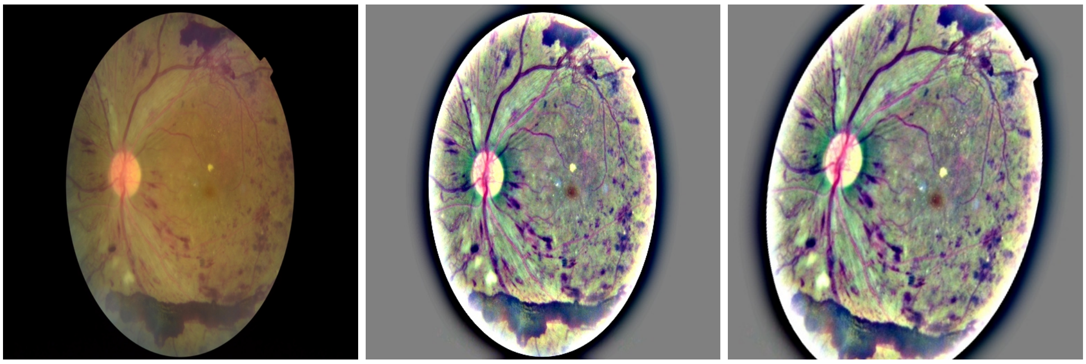
	Attributes used	Values
1	featurewise center	False
2	featurewise std nomalization	False
3	rescale	1.0/255.0
4	rotation range	120
5	horizontal/vertical flip	True
6	wide/height shift range	0.1
7	shear_range	0.2
8	zoom_range	0.2

Table 6.1: Augmentation Details

Point to remember that the augmentation is performed on training data only and not on test data. Although, re-scaling of input images is performed both on training and testing data.

6.2 The Final Phase

Multi-class classification is the final phase of the experiment, in continuation to the binary classification. As per the training dataset [Table 4.2], the classical issues as per



(a) Original (b) Gaussian-blur (c) Augmented

Figure 6.1: Showing an original, and applied pre-processing and augmentation image.

described before needed to be resolve. Few of the noisy images were manually removed such as totally blank, corrupted, unfocused, blur images. The dataset was also unfairly unbalanced. The technique of down-sampling and up-sampling was used to balance the training dataset. The category with the highest number of instances was down-sampled and other classes having lower instances were up-sampled. The balanced instances were 12,500 images per class. Initial spilt with 80:20 percent was performed and later settled with 90:10 percent for final experiment.

6.2.1 Pre-processing

The gaussian pre-processing technique [6.1.1] used for the binary classification in the initial phase was used with changes in intensity values.

6.2.2 Augmentation

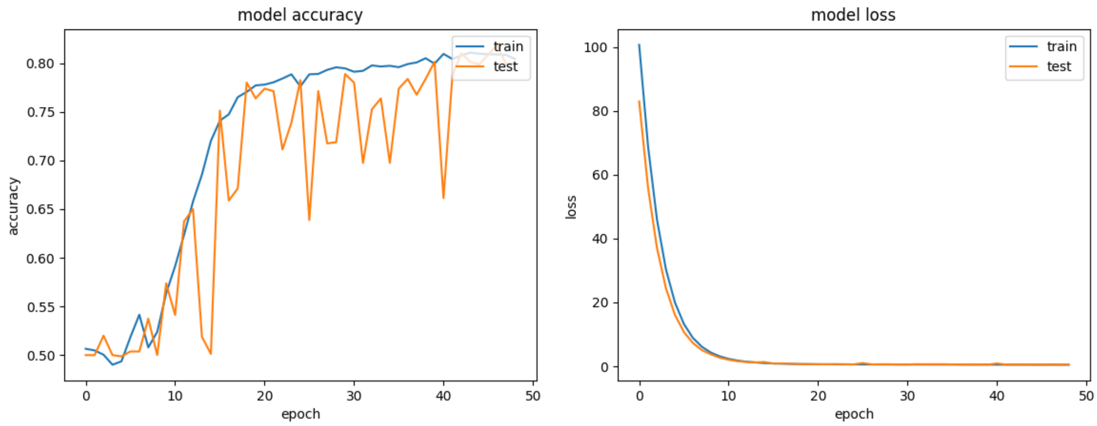
The training data was augmented with the values provided in the table 6.1, with few more added attributes to the initial phase.

6.3 Training

After all the set-up being constructed and done, the CNN was trained with split-ted training dataset into 90% as training and 10% as validation. The initial epochs with value 30 were used for training the neural network. This was basically done to know whether all the parameters used was capable of handling the neural network or not. It was found that the validation accuracy fluctuated a lot. Hyper-parameters was tuned and many experiments were performed for tuning and training the convolutional network. For tuning the learning rate if the cost loss is too high, then the learning rate is decreased by multiples of 10. And, if cost loss is too low, the learning rate is increased by multiples of 10. Stochastic gradient descent optimizer was used for training along with Nestrov momentum. With trial and error, some optimal result was found. Some of the basic hyper-parameters for different experiments performed is described in the Table 6.2.

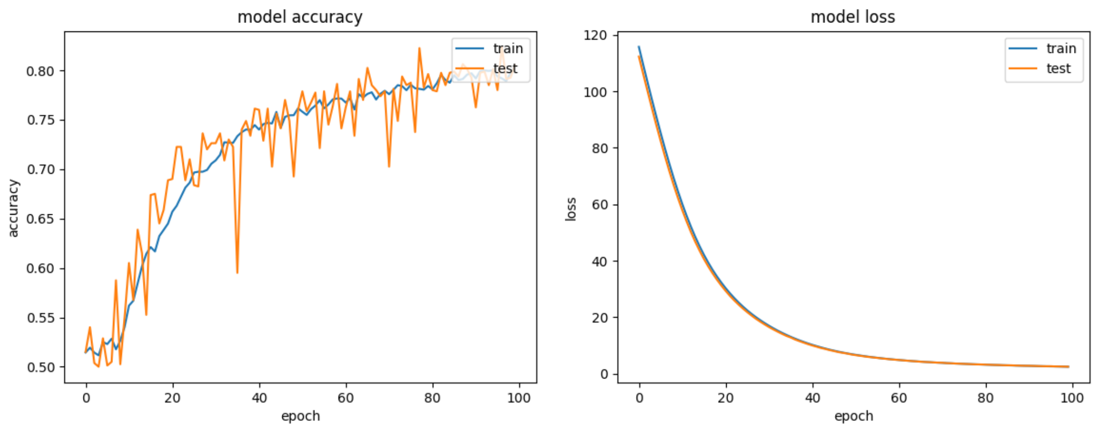
Attributes Used	**Experiment 1**	**Experiment 2**	**Experiment 3**	**Experiment 4**
classification type	binary	binary	multi	multi
batch_size	32	16	32	32
learning_rate	0.0001	0.00001	0.0001	0.003
epochs	50	100	94	57
drop_out	0.5	0.5	0.5	0.5
input_size	512 x 512	512 x 512	512 x 512	512 x 512
kernel_size	3 x 3	3 x 3	3 x 3	3 x 3
pool_size	3 x 3	3 x 3	3 x 3	3 x 3
strides	2 x 2	2 x 2	2 x 2	2 x 2
padding	same	same	same	same
color_mode	rgb	grayscale	rgb	rgb

Table 6.2: Hyper-parameters and Other Details



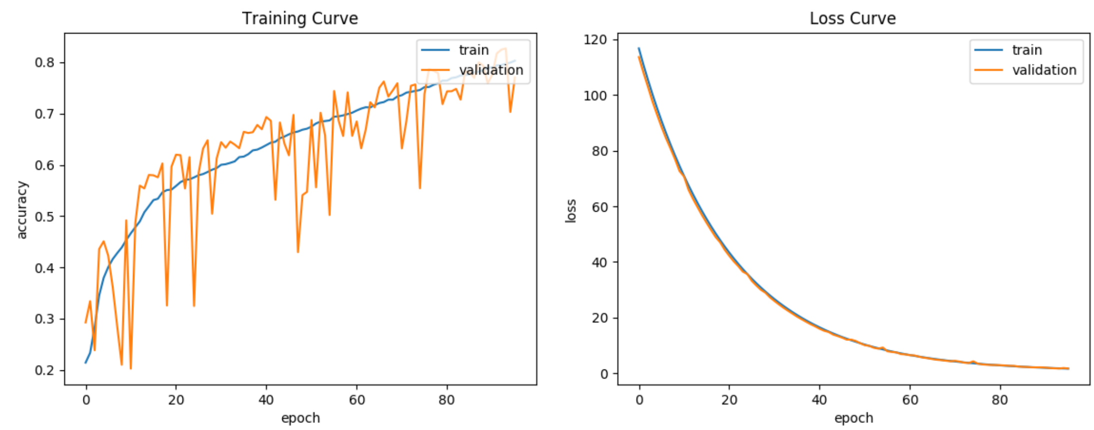
(a) Accuracy curve (b) Loss curve

Figure 6.2: Accuracy and loss curves for binary classification. (Experiment 1)



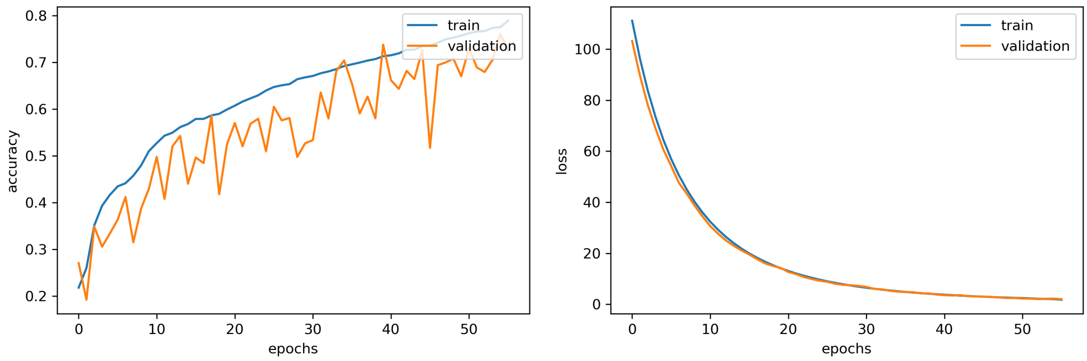
(a) Accuracy curve (b) Loss curve

Figure 6.3: Accuracy and loss curves for binary classification. (Experiment 2)



(a) Accuracy curve (b) Loss curve

Figure 6.4: Accuracy and loss curves for Multi-class classification. (Experiment 3)



(a) Accuracy curve (b) Loss curve

Figure 6.5: Accuracy and loss curves for Multi-class classification. (Experiment 4)

CHAPTER 7

RESULTS

7.1 Results

For the final prediction, a total of 800 testing images for binary classification and 53,312 testing images for multi-class belonging to different categories were used. As the classification is based on image classification. Hence, sensitivity, specificity, accuracy, precision, f-score, and support need to be defined in terms of confusion matrix and for the classification report too.

i. Sensitivity

Actual positive classes that are correctly predicted is sensitivity. Sensitivity is also termed as recall or true positive rate. Sensitivity must be as high as possible.

$$sensitivity = \frac{TruePositive}{TruePositive + FalseNegative}$$

ii. Specificity

Specificity or true negative rate can be defined as actual negative classes that are correctly predicted.

$$specificity = \frac{TrueNegative}{TrueNegative + FalsePositive}$$

iii. Accuracy

The measure of correctness is defined as accuracy.

$$accuracy = \frac{TruePositive + TrueNegative}{TruePositive + FalsePositive + TrueNegative + FalseNegative}$$

iv. Precision

Precision is the number of correct prediction wrt all the classes. Precision must be high.

$$precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

v. F-score

F-score (F1-score) is the calculation of harmonic mean of sensitivity (recall) and precision. When any model having high recall and low precision (and vice-versa), the model isn't comparable. Therefore, f-score is used to make the model comparable. The best score is identified as 1, while the worst is 0.

$$precision = \frac{2 * recall * precision}{recall + precision}$$

vi. Support

Support is the count of actual occurrences of the data present in a particular class. This indicates whether the count of data in classes are unbalanced or balanced.

$$precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

	0	1
0	**303**	97
1	92	**308**

Table 7.1: Confusion matrix for binary classification (Experiment 1).

	0	1
0	**305**	95
1	55	**345**

Table 7.2: Confusion matrix for binary classification (Experiment 2).

	0	1	2	3	4
0	**38385**	303	533	1	118
1	2912	**434**	400	0	10
2	3108	376	**3855**	227	253
3	112	10	611	**307**	165
4	192	3	355	75	**567**

Table 7.3: Confusion matrix for multi-class classification (Experiment 3).

	0	1	2	3	4
0	**36041**	1178	1729	10	382
1	1318	**1867**	528	4	39
2	1141	313	**5472**	465	428
3	552	2	268	**663**	220
4	105	2	167	31	**887**

Table 7.4: Confusion matrix for multi-class classification (Experiment 4).

Experiment no.		Precision	Recall	F1-score	Support
1	0 (normal)	0.74	0.87	0.80	400
	1 (abnormal)	0.84	0.69	0.76	400
	Total/Avg	0.79	0.78	0.78	800
2	0 (normal)	0.77	0.76	0.76	400
	1 (abnormal)	0.76	0.77	0.77	400
	Total/Avg	0.76	0.76	0.76	800
3	0 (normal)	0.86	0.98	0.91	39340
	1 (mild)	0.39	0.12	0.18	3756
	2 (moderate)	0.67	0.49	0.57	7819
	3 (severe)	0.50	0.25	0.34	1205
	4 (proliferative)	0.51	0.48	0.49	1192
	Total/Avg	0.78	0.82	0.79	53312
4	0 (normal)	0.93	0.91	0.92	39340
	1 (mild)	0.55	0.49	0.52	3756
	2 (moderate)	0.67	0.70	0.69	7819
	3 (severe)	0.56	0.55	0.56	1205
	4 (proliferative)	0.45	0.74	0.55	1192
	Total/Avg	0.84	0.83	0.84	53312

Table 7.5: Classification report for binary and multi-class classification.

After experimenting for fewer number of times, the result obtained is shown in table 7.5. Confusion matrix [Table 7.1, 7.2, 7.4] shows that the CNN is able to predict most of the images as per the binary and multi-class classification. The final binary accuracy for the experiment [Fig. 6.2a, 6.2b] is around 80.00%. From the training accuracy graph [Fig. 6.2a, 6.3a, 6.4a, 6.5a], it can be found that the accuracy curve is not much smooth and more tuning of hyper-parameters needed to be done, which is an error and trail approximation.

Due to usage of drop-out, the validation accuracy curve sometimes tends to be more than the training accuracy curve obtained. Batch-normalization used after each convolutional layer, normalizes the activation values after each epochs. Hence, a steep rise of validation accuracy curve can be seen.

The loss curve seems to be good enough as the cost entropy loss is less than 1.0. The testing run time on the convolutional neural network took 14.8 seconds approximately for binary classification, while multi-class classification took around 180 seconds. The final multi-class classification accuracy achieved 84% [Fig. 6.3a].

CHAPTER 8

CONCLUSION

The report shows that the diabetic retinopathy classification screening with respect to binary and multi-class can be performed with much higher accuracy than the actual result found. However, for certain cases, where the images were out of focus, the algorithm fails to identify some of the DR highlights. Consequently, in the near future, the algorithm will be improved to handle the impact of an unfocused picture.

REFERENCES

[1] **G. G. Gardner**,**D. Keating**, **T. H. Williamson**, **A. T. Elliott.**. Automatic detection of diabetic retinopathy using an artificial neural network: a screening tool. *British Journel Ophthalmology* 1996;**80**(11):940-944.

[2] **Markku Kuivalainen**. Retinal Image Analysis Using Machine Vision. *Thesis report Lappeenranta Univ of Tech, Dept of IT* 2005.

[3] **Nayak J**, **Bhat PS**, **Acharya R**, **Lim CM**, **Kagathi M.**. Automated identification of diabetic retinopathy stages using digital fundus images. *J Med Sys* 2008;**32**(2):107-115.

[4] **Acharya UR**, **Chua CK**, **Ng EY**, **Yu W**, **Chee C.**. Application of higher order spectra for the identification of diabetes retinopathy stages. *J Med Sys* 2008;**32**(6):481-488.

[5] **Acharya UR**, **Lim CM**, **Ng EY**, **Chee C**, **Tamura T.**. Computer-based detection of diabetes retinopathy stages using digital fundus images. *P I Mech Eng H* 2009;**223**(5):545-553.

[6] **P.Adarsh**, **D.Jeyakumari.**. Multiclass SVM-based automated diagnosis of diabetic retinopathy. In: *Communications and Signal Processing (ICCSP),*2013 *International Conference on.* IEEE; 2013,p. 206-210.

[7] **Harry Pratt**, **Frans Coenen**, **Deborah M Broadbent**, **Simon P Harding**, **Yalin Zheng.**. Convolutional Neural Network for Diabetic Retinopathy. *Procedia Computer Science* 90 2016:200-205.

[8] **Linda Roach**. Artificial Intelligence, The Next Step in Diagnostics. In: *EyeNet* November 2017.

[9] **Pascanu**, **Razvan et al**. Understanding the exploding gradient problem. In: *CoRR* 2012.

[10] **Ben Graham** (2015). Kaggle Diabetic Retinopathy Detection competition report. URL http://blog.kaggle.com/2015/09/09/diabetic-retinopathy-winners-interview-1st-place-ben-graham/

[11] Contributors(2017). Diabetic Retinopathy. URL https://en.wikipedia.org/wiki/Diabetic_retinopathy

[12] Ecenbarger Eye Care. URL https://www.ecenbargereyecare.com/diabetic-retinopathy-2/

[13] Kaggle Diabetic Retinopathy Competition. URL https://www.kaggle.com/c/diabetic-retinopathy-detection/data