Detection and Grading of Diabetic Retinopathy on the basis of Severity using Convolution Neural Network

SHUBHRANSHU MALHOTRA
Dept. of CSE
SRM Institute of Science and Technology
Chennai, India

VAIBHAV NANGIA
Dept. of CSE
SRM Institute of Science and Technology
Chennai, India

Abstract

Diabetic retinopathy is a diabetic condition that is a significant cause of blindness worldwide. Diabetes damages the small blood vessels inside the retina, resulting in a variety of vision-related issues. On the retina, these small blood vessels leak blood and fluid. Diabetic Retinopathy is classified into two types: non-proliferate diabetes retinopathy (NPDR) and proliferate diabetic retinopathy (PDR). People have been working for years to find a way to detect this issue in its early stages so that treatments can proceed. The Convolution Neural Network approach was effectively applied in this research to detect and evaluate diabetic retinopathy. Images (2240 X 1488) are taken from ADCIS's MESSIDOR, which comprises 1200 raw retinal colour fundus images. The images are pre-processed and the resolution is modified before applying the Convolution Neural Network layers (256 X 256). The quality of the images is the most important element determining the results. The attained accuracy is 90.83 %.

I. INTRODUCTION

Diabetic retinopathy is a disease that affects more than 80% of those who have had diabetes for more than 20 years. Diabetic Retinopathy has been detected in 17% to 97% of the cases after 5 and 15 years of the diagnosis of diabetes respectively. 90% of new cases of diabetic Retinopathy can be dealt with if timely treatment of the eyes is done. Each year in the USA, 12% of blindness cases are due to Diabetic Retinopathy and the trends in India are quite the same. People aged 20 to 64 are most affected by this disease. Diabetic Retinopathy is a symptomless complication of Diabetes and is also one of the most significant causes of vision impairment.

Diabetic retinopathy causes retinal damage and may result in vision loss; consequently, early diagnosis of the severity of DR is critical in treatment. The treatment consists of laser photocoagulation, vitrectomy, anti-VEGF medications, corticosteroids, and other procedures, however there are significant potential issues. The most difficult aspects of this problem are the segmentation and categorization of retinal lesions, which are classified into two types: bright and dark. Bright lesions consist of exudates and cotton wool spots while dark lesions consist of microaneurysms and hemorrhages. Scientists and professionals from all around the world are working hard to create ways for detecting symptoms more quickly and precisely, so that no one loses their eyesight as a result of this terrible disease. Automated detection using market-leading technology will result in significant time and effort savings.

II. RELATED WORKS

Many analysts have tried to work on the detection of Diabetic retinopathy. A brief insight into their works is given below:

Authors Gulshan V. et al. [1] in their paper used Deep Learning techniques to self-learn from large dataset which is both clinically acquired as well as digitally available (EYEPACS for training and MESSIDOR 2 for validating) between May 2015 and October 2015. In total, the research and testing are done on 1,28,175 images and they achieve an accuracy of 81% but the results were very vague and not very informative. The optimization algorithm used is Stochastic Gradient Descent batch Normalization and Single Network Multiple Binary Predictions based methods but it couldn’t provide the desired results.

Seluck T., Alkan A, in their 2019 published paper Detection of microaneurysms in early diagnosis of Diabetic Retinopathy [2] used a very infamous technology to detect microaneurysms, which are the biggest indicators of DR, the Ant Colony Algorithm inspired by the nature of ants or any other worker animals. Whenever the face an obstacle while traveling the shortest path between two-point they then and there change their path to the next shortest path similarly in ant colony algorithm the next best option is chosen if an obstacle occurs. Image processing techniques and Clustering Algorithms are applied to images acquired from the MESSIDOR and DIARETDB1 database which had 1200 images.

The steps involved in this process were 1) Extracting Retinal vascular structure and 2) Segmentation of microaneurysms using Ant Colony Algorithm The proposed methods were compared with a few important and famous techniques such as Watershed, Random Walker, K-means clustering, Maximum Entropy, and Region Growing and results were cross-examined. It followed a three-step methodology 1) Preprocessing using the green channel, CLAHE and Median filter 2) Vessel Segmentation using Frangi Filter, removing small region and Connected Component Labeling and lastly 3) Detection using 1) Ant Colony Algorithm and 2) Counting Microaneurysms.

Automated Diagnosis of Non-Proliferative Diabetic Retinopathy [3] in Fundus Images is path-breaking general published in 2015 in International Journal of Computer Applications, authored by Maher R.S., Kayte S.N., Neldhe S.T., and Dhopeshwarkar M. The images (around 130) were acquired from DIARET DB0 database and employed Support Vector Machine for automatic classification of DR. automatic classification simplifies the task for man and there is no need for ophthalmologists to confirm the presence of DR in patients' eye. RGB images were converted to Green channel images before processing them

The steps involved in this process were 1) Preprocessing the Fundus images by creating a binary mask for background and noisy areas like Histogram Equalization. 2) Morphological Operations and finally 3) Feature Extraction and SVM Classification into Normal, Non-Proliferative DR and Proliferative DR. This is done by Gabon filter. They were able to achieve 96.43% specificity, 95.9% sensitivity and 99.27% accuracy in the extraction and accurate grading of NPDR Lesions.

Another paper by Ramon Pires et al. [4] used Convolutional Neural Network on Dataset acquired at Department of Ophthalmology, Kasturba Medical College. They employed many image processing techniques, data augmentation, multiresolution training, and robust features extraction augmentation to achieve an accuracy of 93%. The dataset is acquired from the Kaggle competition dataset which is used for training purposes while for validation purposes they used MESSIDOR 2 dataset. The accuracy isn't that well but it paved the way for future studies on this topic.

In the paper "Convolution Neural Network for Diabetic Retinopathy" authored by Harry Pratt et al. [5], high-end GPUs were used, CNN was employed on Kaggle images that are available publicly and an accuracy of 75% was achieved.
Authored by Z. Mohammadpoury et al. Automatic Identification of Diabetic Retinopathy Stages by Using Fundus images and Visibility Graph Method, Measurement (2019) [6], is an extraordinary paper that introduced a new way of solving this problem by using Visibility Graph and Radon Transformation. 1200 images from the MESSIDOR database were processed to achieve an accuracy of 97.92%, specificity 98.61% and sensitivity 95.83%. Color Fundus images (RGB) were converted to grayscale by contrast limited, adaptive histogram equalization filter. Then Radon transform and Visibility Graph construction structure are applied.

Class	Severity Level
Class 0	Normal
Class 1	Mild DR
Class 2	Moderate DR
Class 3	Severe DR

Table 1: Classification of Dataset

III. PROPOSED MODEL

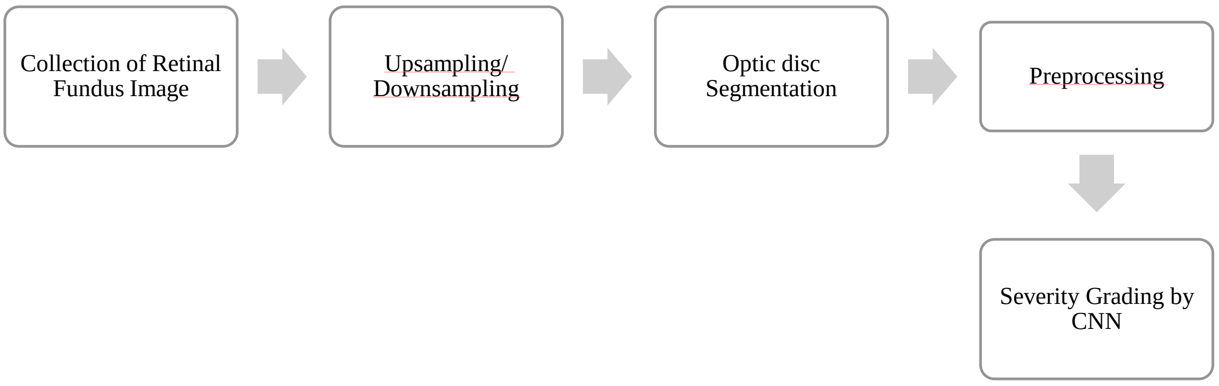


Figure 1: Flow of the Proposed Model

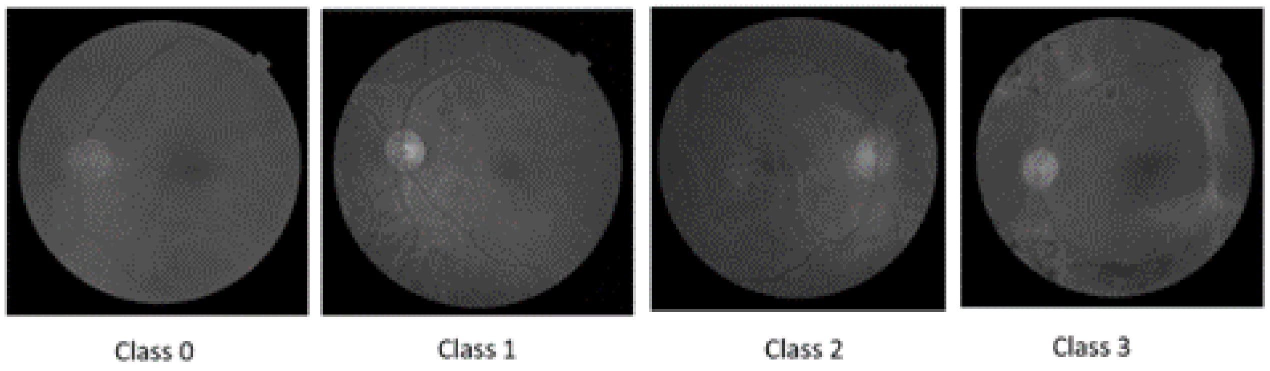


Figure 2: Sample retinal fundus images of eye belonging to each class

1. Dataset used

The main dataset used in this paper is the Messidor dataset from ADCIS. This dataset consists of a total of 1200 raw retinal color fundus images. These images have been divided into 4 classes from 0 to 3 on the basis of the severity of Diabetic retinopathy. Class 0 represents a healthy eye with no sign of Diabetic Retinopathy, Class 1 images have Mild Diabetic retinopathy, Class 2 images are the ones with moderate Diabetic Retinopathy and the Class 3 images have severe Diabetic retinopathy. The MESSIDOR dataset is imbalanced, it has more class 0 images, almost half of the images are class 0, also there is very few class 1 images, this may lead to bias in the model.

Classification of color fundus images present in Messidor dataset is shown in Table 1. Figure 2 shows sample images belonging to each class present in the dataset.

2. Methodology

The methodology used involves: (i) Upsampling/Downsampling the data, (ii) Optic Disc Segmentation, (iii) Preprocessing images to make it model ready, (iv) Feeding Preprocessed images to the CNN model for severity grading. The procedural explanation for the above 4 steps is given in this section.

2.1 Upsampling/Downsampling the data

Upsampling is the technique for balancing the dataset by duplicating images in the classes that have less presence in the data. By random duplication of data, this technique prevents the model to be biased towards a particular class.

Downsampling refers to removing samples belonging to a certain class in order to prevent it from dominating the dataset and leading to a bias towards that kind of data. By random elimination of data from the dataset, Downsampling ensures that the majority class doesn't have an ill effect on the classification.
Here, while using the Messidor dataset there was a need to upsample the class 1, 2, 3 images to the level of class 0 images to prevent this bias. Similarly, the Kaggle dataset was also very imbalanced and required Up-sampling and Down-sampling of images belonging to certain classes.
Up-sampling

The process of Up-sampling randomly duplicates observations from minority classes to bolster its signal. There are many techniques for performing Up-sampling, but the most common one is to simply add a replacement and here the same approach has been used. A new variable with up-sampled minority classes is created. Here are the steps:
1. Segregate observations of each class into separate variables.
2. Resample the minority class and add replacement images.
3. Combine the up-sampled minority class with the other class samples.

Downsampling

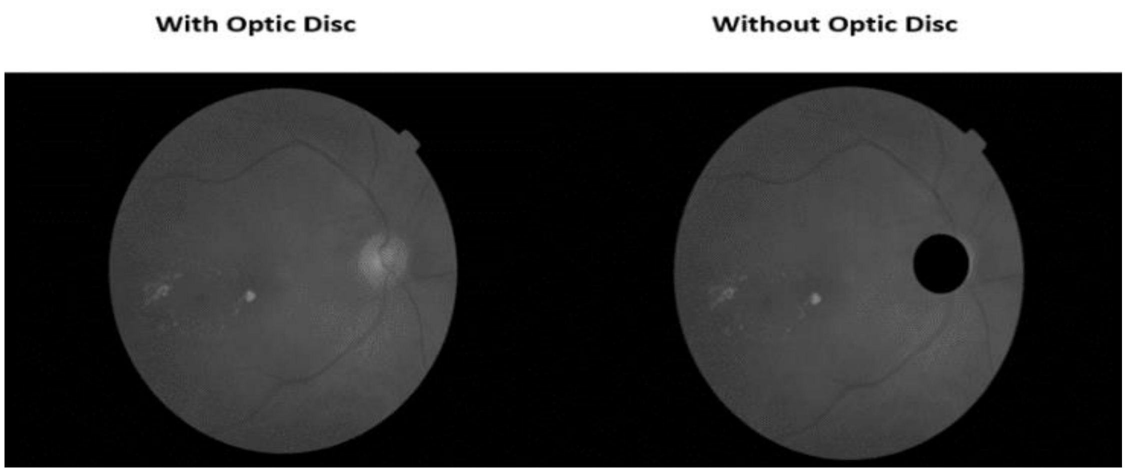
The process of down-sampling randomly removes observations from majority classes to prevent its signal from governing the learning process. The process used is similar to that of up-sampling. Here are the steps:

1. Segregate observations of each class into separate variables.
2. Resample the majority class and remove extra images.
3. Combine the down-sampled majority class with the other class samples.

2.2 Optic Disc Segmentation

On carefully observing the dataset, it was found that the brightest pixel in the image is almost always present around the center of the optic disc, this is the result of the imaging technology used in order to obtain retinal fundus images.

This led to the technique of optic disc segmentation that has been used in the proposed system. In the current system instead of using a range of intensity values, a single brightest spot is found. Global maxima in the image is found in order to locate the brightest spot. The brightest spot is actually a single brightest pixel in the image. Once the brightest spot in the image is obtained, a circle is created with that spot as the center and then this part of the image is removed. This removes the optic disc from the color fundus images, removing unnecessary details from it. Figure 3 shows a Retinal Fundus Image before and after optic disk removal



.

Figure 3: Optic Disk Removal

3. Preprocessing

After trying a range of preprocessing techniques, it was found that canny edge detection along with resizing, interpolation and normalization after cropping and removal of the optic disk would be the most useful and effective method.

3.1 Cropping

Cropping is a basic image manipulation technique, that removes irrelevant areas and noise from an image. Cropping an image means obtaining a rectangular region containing the most useful information from the original image. This helps in focusing the attention of the model on the useful part of the image and also helps in discarding areas of the image that contain unimportant information.

The images in the dataset used have a lot of extra, unnecessary parts that can be removed without losing any useful information, these parts were cropped out in order to reduce the noise and to allow the model to work on more useful information.

3.2 Canny Edge Detection

Edges can be defined as sudden changes/discontinuities in an image and they can encode just as much information as pixels. The process behind canny edge detection includes smoothening the image to reduce the noise and then recognizing these sudden changes in intensity. It is a multi-stage process. It involves:

1. Reducing Noise
The detection of edges is highly sensitive to noise in the picture, so the primary step involves removing noise from the image using a Gaussian filter.

2. Find the intensity gradient
Sobel kernel is used to filter the image against both vertical and horizontal directions and the first derivative in both the directions is then obtained. The edge gradient and the pixel direction can then be obtained from the derived two images.

3. Non-maximum Suppressing
Once the magnitude, direction, and gradient have been obtained, the image is fully scanned in order to remove the pixels that don't compose the edges. This is done by checking each pixel along with its neighbouring pixels to verify if it is a local maximum.

4. Hysteresis Threshold
This is done to define the edges that are required. This is done by setting threshold values, minVal, and maxVal. Edges having intensity gradient above the set maxVal value are taken as edges and the ones having a value less than minVal are non-edges, so they are discarded.

Getting these two parameters right is most of the work and highly affects the precision of edges detected. These values are adjusted such that finally, only the strong edges remain in the image.

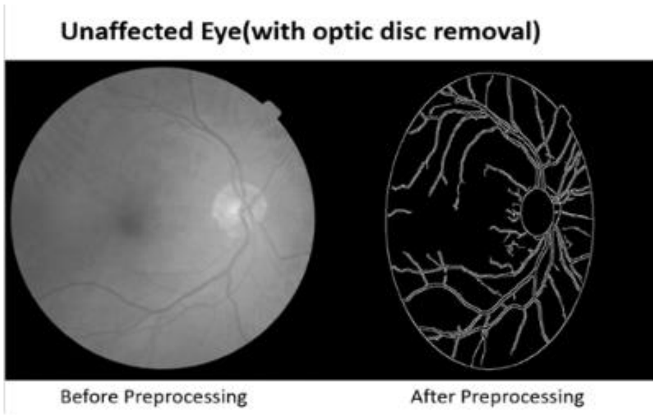


Figure 4: Unaffected eye before and after preprocessing

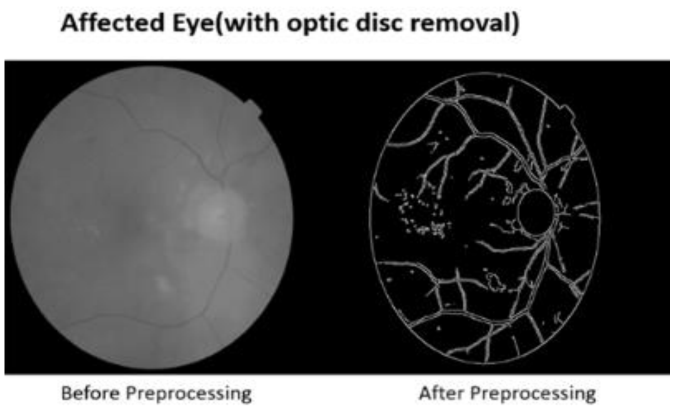


Figure 5: Diabetic retinopathy eye before and after preprocessing

3.3 Resizing

It becomes necessary to resize an image if there is a need to decrease or increase the number of pixels. The pixel information of an image changes when it is resized e.g., Reducing the size of an image will discard any unnecessary pixels or new pixel information that must be created when an image is enlarged -- based on estimation(interpolation) -- in order to attain a large sized image which generally makes an image pixelated or blurry.

The images used have a resolution of 2240 X 1488 pixels. It is computationally impossible to process such large size images as, more pixels will result in more weights associated with the image, adjusting which would require more processing power. Due to the limitation of resources, it becomes necessary to resize the images.

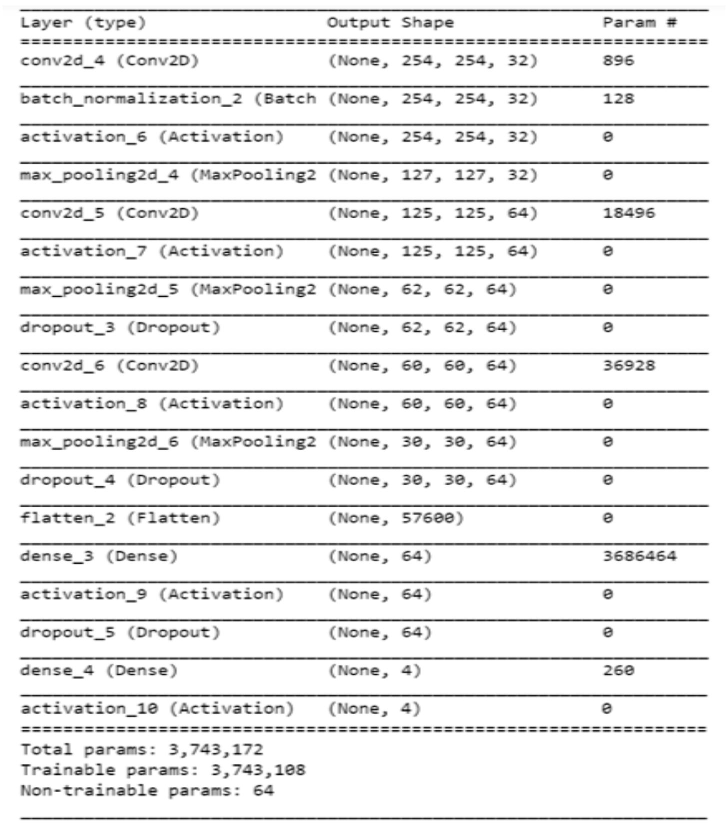
In this system resizing of images is done after the major preprocessing steps are implemented, this helps to preserve important information during preprocessing and give better-preprocessed images. After major preprocessing steps are over, the images are resized to 256 X 256 pixels before feeding them to the Neural Network for training purposes.

3.4 Interpolation

Interpolation is the method of introducing new data points within the range of already known discrete data points. It is a type of estimation function and can be used to find missing values. Interpolation becomes necessary when the image is resized or rotated. Image interpolation works in both horizontal and vertical directions and works on achieving the best approximation of pixel intensity on the basis of the pixels surrounding it. There are 2 kinds of interpolation algorithms: Adaptive/Non-adaptive. In the system, Inter-Area Interpolation is used, which is considered best for resizing.

3.5 Normalization

The process of normalization is used to change the range of values of pixel intensity. It may also be referred to as histogram stretching or contrast stretching. It is done to lessen the noise in the image and also to bring the intensity values into the 'normal' range (i.e. statistically now, it follows a normal distribution). For this, all the pixels are divided by 255 in order to bring them into range zero to one. Figure 4 shows a preprocessed Normal Eye with Canny edges and removed optic disk. Figure 5 shows a preprocessed DR affected Eye with Canny edges and removed optic disk.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 254, 254, 32)      896
_________________________________________________________________
batch_normalization_2 (Batch (None, 254, 254, 32)      128
_________________________________________________________________
activation_6 (Activation)    (None, 254, 254, 32)      0
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 127, 127, 32)      0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 125, 125, 64)      18496
_________________________________________________________________
activation_7 (Activation)    (None, 125, 125, 64)      0
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 62, 62, 64)        0
_________________________________________________________________
dropout_3 (Dropout)          (None, 62, 62, 64)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 60, 60, 64)        36928
_________________________________________________________________
activation_8 (Activation)    (None, 60, 60, 64)        0
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 30, 30, 64)        0
_________________________________________________________________
dropout_4 (Dropout)          (None, 30, 30, 64)        0
_________________________________________________________________
flatten_2 (Flatten)          (None, 57600)             0
_________________________________________________________________
dense_3 (Dense)              (None, 64)                3686464
_________________________________________________________________
activation_9 (Activation)    (None, 64)                0
_________________________________________________________________
dropout_5 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 4)                 260
_________________________________________________________________
activation_10 (Activation)   (None, 4)                 0
=================================================================
Total params: 3,743,172
Trainable params: 3,743,108
Non-trainable params: 64
_________________________________________________________________
```

Figure 6: CNN Model Summary

4. Severity grading using CNN

A Convolutional Neural Network also referred to as a ConvNet or CNN takes an image as an input, then assigns importance (weights and biases) to various aspects of the image, called features and makes it such that they can be differentiated from one another.
CNN is a Deep Learning algorithm and the amount of pre-processing required for a CNN is considerably low when compare to other classification techniques.

The functioning of Neural Networks is similar to that of a 'black box' brain.

It takes an input image and predicts output. It's better than most traditional ML algorithms in the sense that it can learn more complex non-linear mappings to deliver more accurate predictions.

CNN takes shape of the image as input and gives the output as the number of classes in the dataset. Multiple hidden layers can be added in between to make the model more complex. Adding more layers than necessary leads to overfitting while a CNN with fewer layers will cause underfitting. So, striking a balance becomes very important.

To better explain the working of CNN, you can divide its working into the following steps: You feed an image to the network, then pass it through a range of layers i.e. convolutional, nonlinear, pooling (max, sum or average pooling), and fully connected layers, and then get an output prediction. The output may be in the form of a single class or it may be given as probability of the image to be in each of the available classes.

Working of the CNN model is explained below:

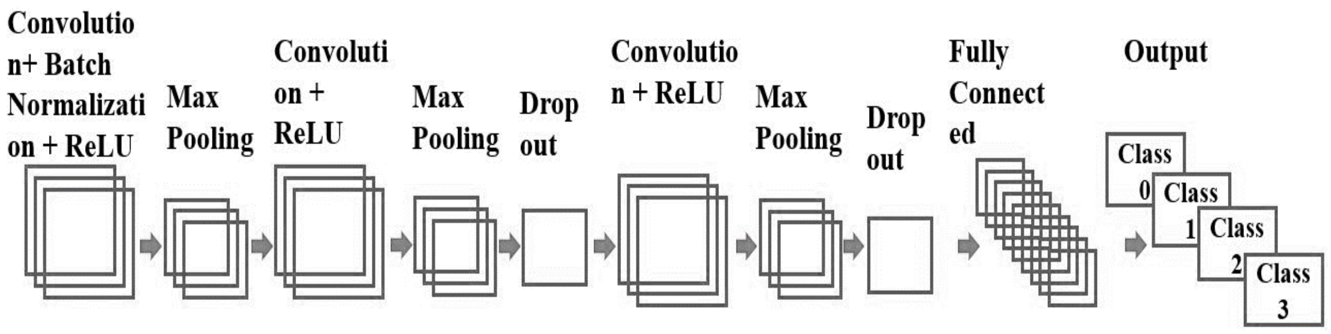


Figure 7: CNN model representation

The first layer in the CNN model is a Convolutional Layer. The convolution layer takes the image shape as input. Convolutions are executed by sliding a filter or kernel over an input image. The sliding process is a simple matrix multiplication or dot product between the kernel and the image receptive field.

An intermediate Batch Normalization layer is also added to reduce the covariance shift, which in turn speeds up learning.
After convolving through the image and Normalizing the output non-linearity is introduced into the model by using an activation function as the convolution process in itself is linear. Here, the ReLU activation function. A ReLU activation function converts all the negative values to 0 without affecting the positive values.

After this, a pooling layer is used. Pooling, also known as subsampling is used to reduce the size or dimensionality of the feature vector. The purpose of pooling is to reduce trainable parameters while also retaining important information. Max Pooling in used in the CNN Model, though there are 2 other pooling methods: Average and Sum pooling. MaxPooling is used in order to reduce training parameters and conserve the most useful information.

Three sets of the above-mentioned CONVOLUTION =>BATCH NORMALIZATION => RELU => MAX POOLING layers are used.

The dropout layer is also added in between these layers in order to reduce the risk of overfitting. The dropout layer reduces the possibility of overfitting by randomly eliminating useless links in the layers.

Finally, Fully Connected and Classification layer is attached. Fully connected layer also uses "RELU" as activation function. The softmax function is used for classification as it is a multi-class classifier.

Rmsprop optimizer with a learning rate of 0.001 is used along with "categorical cross-entropy" loss function.
The model is trained for 100 epochs with a batch size of 64 and shuffle ON. Callbacks are also in order to save the model with the best accuracy and least loss.

Figure 6 shows the summary of the CNN model used for predictions and Figure 7 gives a more visual representation of the layers in CNN model.

IV. RESULTS AND DISCUSSION

While researching on this topic, there were several hurdles that came up. Firstly, finding a well labeled and good quality dataset was a big challenge, even if the dataset itself had good quality images, there still was the problem of unbalanced data.

Secondly, designing the CNN required very careful configurations and a lot of trial and error to prevent overfitting and underfitting. It had to be made sure that the loss was low and the learning efficiency remained high. Getting the hyperparameters for the CNN model right was another big task. Optimal learning rate, batch size, activation function, loss function, optimizer used, number of epochs the model is to be trained, etc. had to be decided. The biggest challenge of all was the problem of limited processing resources. The predictions done require differentiating images on the basis of very minute details but due to lack of computational and memory resources, the options were very limited.

The preprocessing method used is very simple and light, i.e. it doesn’t require huge processing power. Also, lack of huge processing resources was a big challenge while doing this research but this led to discovery of techniques that required less processing. Besides this, the CNN model used despite having fewer layers has a very good learning efficiency.

CNN was used to detect the level of Diabetic retinopathy by feeding Color Fundus images of a patient’s eyes. 20% of the dataset was used for testing and 90.83% accuracy, 96.3% specificity, 88.75% sensitivity and a loss of just 0.48 was obtained on the images from MESSIDOR dataset.

Ref.	Specificity	Sensitivity	Acc.	Dataset
This Paper	96.30	88.75	90.83	Messidor
[1]	93.9	96.1	-	Messidor
[4]	83.3	95.8	95.0	Kaggle/Messidor
[5]	30	95	75.0	Kaggle
[11]	93.65	-	83.68	EYEPacs
[12]	84.7	90.6	85.7	Messidor

Table 2: comparison of specificity, sensitivity and accuracy of this work with other similar works

As compared to Gulshan V. et.al[1], where prediction accuracy of 81% was achieved using similar Deep Learning techniques, Harry Pratt et.al[5], CNN was used and accuracy of just 75% could be obtained and Garcia et al.[11], where the best accuracy was 83.68%, model used in this paper in addition to using an original self-made CNN model which is very light and simple also performs a lot better, considering the accuracy of 90.83%, which was a result of effective preprocessing techniques and the CNN architecture. Table 2 compares specificity, sensitivity and accuracy of this work with other similar works.

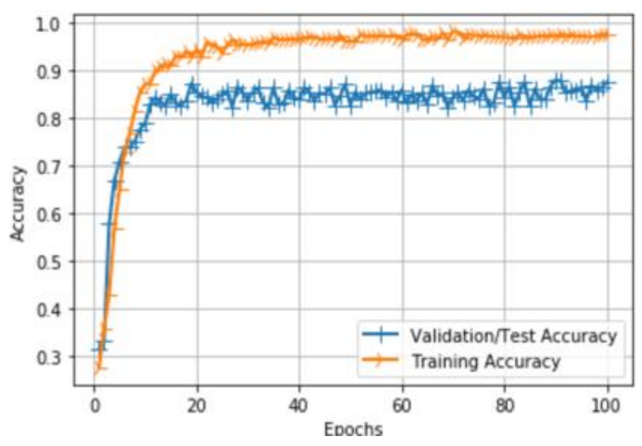


Figure 8a: Accuracy vs Epochs (Messidor Dataset)

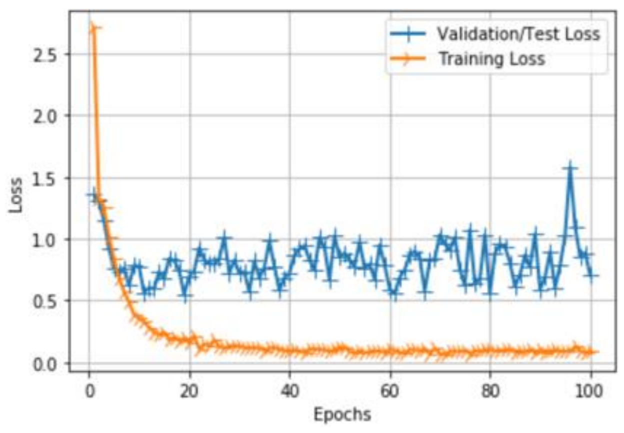
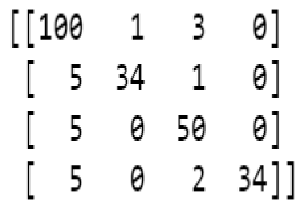


Figure 8b: Loss vs Epochs (Messidor Dataset)

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+FP+FN+TN)}$$

The learning process was visualized by drawing plots of training and validation accuracy (Figure 8a) and training and validation loss (Figure 8b) showing learning vs epochs on Messidor dataset. These helped to identify when and where the model was overfitting and helped to make changes to rectify it.

	Precision	Recall	F1-score	Support
0	0.87	0.96	0.91	104
1	0.97	0.85	0.91	40
2	0.89	0.91	0.90	55
3	1.00	0.83	0.91	41



```
[[100   1   3   0]
 [  5  34   1   0]
 [  5   0  50   0]
 [  5   0   2  34]]
```

Table 3: classification report and confusion matrix (MESSIDOR Dataset)

Calculations done:
Precision: ratio of total number of true positive observations to the number of all positive observations.

$$\text{Precision} = \frac{TP}{(\mathrm{TP+FP})}$$

Recall: Ratio of the number of true positive observations to the total number of observations in the class.

$$\text{Recall} = \frac{TP}{(TP+FN)}$$

F1-score: The weighted average of Recall and Precision.

$$\text{F1 Score} = 2 * \frac{(Recall*Precision)}{(Recall+Precision)}$$

Support = Total data belonging to a particular class.

Table 3 shows the confusion matrix and classification report for testing on the Messidor dataset.
Misclassified images are found and then visualized to analyze what mistake could have happened. On visualizing this data, it was observed that it was difficult to classify these images even with the bare eyes. So, there is a high probability that a lot of the misclassified data was mislabeled.

Graphs of Accuracy vs Epochs and Loss vs Epochs are constructed in order to visualize the trend of improvement in the predictions over the training process.

The graphs in figure 8a and figure 8b show the reduction in loss and the increase in accuracy as the number of epochs progress.

V. CONCLUSION

A detailed analysis of existing works was done and it was observed that there were not many works in which CNN was used and good results were obtained. Even amongst the ones with good results the process used was very computationally intensive and high-end devices had to be used. In the proposed system, a very light model and other techniques that require less computational power and can be done on a Home PC have been used. Various pre-processing steps were used to remove unnecessary information. The color fundus images of eyes were classified into 4 classes based on the severity of Diabetic retinopathy: No DR, Mild DR, Moderate DR, Severe DR. Validity and correctness of the work was verified by calculating accuracy, specificity, sensitivity and comparing these with those of other works. The proposed system was able to effectively detect Non- Proliferative Diabetic Retinopathy and was also able to grade it efficiently. Variations in the techniques used, such as using a better CNN model or by further reducing noise from image during preprocessing, may result in even better results.

REFERENCES

[1] Gulshan, V., Peng, L., Coram, M., Stumpe, M. C., Wu, D., Narayanaswamy, A., & Kim, R. (2016). Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. Jama, 316(22), 2402-2410.

[2] SELÇUK, T., & ALKAN, A. (2019). Detection of microaneurysms using ant colony algorithm in the early diagnosis of diabetic retinopathy. Medical Hypotheses, 129, 109242.

[3] Maher, R. S., Kayte, S. N., Meldhe, S. T., & Dhopeshwarkar, M. (2015). Automated diagnosis non-proliferative diabetic retinopathy in fundus images using support vector machine. International Journal of Computer Applications, 125(15).

[4] Pires, R., Avila, S., Wainer, J., Valle, E., Abramoff, M. D., & Rocha, A. (2019). A data-driven approach to referable diabetic retinopathy detection. Artificial intelligence in medicine, 96, 93-106.

[5] Pratt, H., Coenen, F., Broadbent, D. M., Harding, S. P., & Zheng, Y. (2016). Convolutional neural networks for diabetic retinopathy. *Procedia Computer Science*, *90*, 200-205.

[6] Z. Mohammadpoury, M. Nasrolahzadeh, N. Mahmoodian, J. Haddadnia, Automatic Identification of Diabetic Retinopathy Stages By Using Fundus images and Visibility Graph Method, Measurement (2019), S0263-2241(19)30206-4.