

# **Detection and Classification of Diabetic Retinopathy**

## **Using Deep Learning**

Babatunde Azeez Olatunbosun

Applied project Submitted in partial fulfillment of the requirement for

the degree of

[MSc Data Analytics]

at Dublin Business School

Supervisor: Shubham Sharma

August 2020

## **DECLARATION**

‘I declare that this applied project that I have submitted to Dublin Business School for the award of [MSc Data Analytics] is the result of my own investigations, except where otherwise stated, where it is clearly acknowledged by references. Furthermore, this work has not been submitted for any other degree.’

Signed: Babatunde Azeez Olatunbosun

Student Number: 10531672

Date: 25-08-2020

## **ACKNOWLEDGEMENTS**

I would like to express my deepest gratitude to God Almighty, My unequivocal gratitude also goes to my supervisor Dr. Shubham Sharma for her consistent whole-hearted support throughout the course of this thesis and for her spontaneous feedback at different stages of this project as she motivate me to carry out this research.

I would also like to thank Terri Hoare, my instructor for “Data Mining” and Abhishek Kaushik, instructor for “Machine learning” who both thought me very valuable concept in this field.

My warm appreciations is due to my darling wife for her immeasurable love and patience during the course of this study.

Finally, I’d like to thank my parents, family and friends those who were my true support system throughout the period of this project.

## **ABSTRACT**

Diabetic retinopathy (DR) is a widely known cause of blindness in recent times especially in people who are quite older. The DR which has several types and equally has diverse causes that can be easily put to bed with early detection. When manual medical methodologies are employed, early detection of DR is difficult even though it takes so long to complete, results are often imprecise. Hence the need arises for better method of detection and prediction of DR in patients. This paper therefore seeks to make use of deep learning algorithm to classify and detect diabetic retinopathy in patients. The deep learning algorithm that is used in this research work is the Convolutional neural network (CNN) which used a custom 4 layer VGG net and an additional neural network making it a 5 layer network. The data used for training the system is taken from the Indian Diabetic Retinopathy Image Dataset (IDRID). The system achieved an accuracy, precision, recall and F1-score of 92% which did very well when compared to other deep learning systems like KNN, SVM and logistic regression which had an accuracy of 86%, 66% and 52% respectively.

**Keywords:** Diabetic Retinopathy, Deep Learning, Classification, Detection, CNN, Diabetes

## TABLE OF CONTENTS

DECLARATION .....	ii
ACKNOWLEDGEMENTS.....	iii
ABSTRACT.....	iv
TABLE OF CONTENTS.....	v
TABLE OF FIGURES .....	viii
LIST OF TABLES.....	xi
CHAPTER ONE: INTRODUCTION .....	12
1.1 Introduction and Background .....	12
1.1.1 Emergence of deep learning in Classification and detection of Diabetic Retinopathy Fundus Images in Medical Domain.....	12
1.1.2 Types of Diabetes.....	12
1.1.3 Diabetic Retinopathy .....	13
1.1.4 Classes of Diabetic Retinopathy.....	14
1.1.5 The need for Deep learning .....	16
1.2 Research Purpose.....	18
1.3 Research Objectives and Research Questions .....	19
1.4 Thesis Structure .....	19
CHAPTER TWO: LITERATURE REVIEW .....	21
2.0 Literature Review - Introduction.....	21
2.1 Automated Detection and Classification of Fundus Diabetic Retinopathy Images using Synergic Deep Learning Model .....	21
2.2 Diabetic Retinopathy Detection Using Deep Neural Network.....	22
2.3 Detection and Classification of Diabetic Retinopathy using Retinal Images .....	22
2.4 Classification of Diabetic Retinopathy Images by Using Deep Learning Models .....	23
2.5 Automated Detection of Diabetic Retinopathy using Deep Learning.....	24
2.6 An Enhanced Diabetic Retinopathy Detection and Classification Approach Using Deep Convolutional Neural Network.....	24
2.7 Deep Transfer Learning Models for Medical Diabetic Retinopathy Detection.....	25
2.8 Diabetic Retinopathy Detection using Machine Learning .....	26
2.9 Detection and Classification of Diabetic Retinopathy in Fundus Images using Neural Network. ....	27
2.10 Diabetic Retinopathy Stage Classification using CNN .....	27
2.11 Automated Detection of Diabetic Retinopathy using Deep Learning.....	28

2.12 Classifying Diabetic Retinopathy using Deep Learning Architecture .....	28
2.13 Computer-Assisted Diagnosis for Diabetic Retinopathy Based on Fundus Images Using Deep Convolutional Neural Network .....	29
CHAPTER THREE - METHODOLOGY .....	31
3.0 Introduction .....	31
3.1 Deep Learning .....	31
3.2 Convolutional Neural Network (CNN) .....	32
3.3 Deep Neural Network .....	34
3.4 Deep learning methodology .....	34
3.4.1 Support Vector Machine (SVM) .....	35
3.4.2 KNN (K-Nearest Neighbor) .....	35
3.4.3 Logistic Regression (LR) .....	36
3.5 Classification .....	36
3.5.1 Dataset Collection .....	37
3.5.2 Data Preprocessing .....	38
3.5.3 Data Splitting and Label Binarization .....	42
3.5.4 Data Augmentation .....	42
3.5.5 Data Modeling and Training .....	47
3.5.5.1 Model Architecture Construction .....	47
3.5.5.2 Optimization .....	49
3.5.5.3 Compilation .....	49
3.5.5.4 Training and Classification .....	49
3.5.6 Code Snippet for Model training and classification .....	50
3.6 Detection .....	51
3.6.1 Input image .....	52
3.6.2 Image Pyramid .....	53
3.6.3 Sliding window .....	55
3.6.4 Taking region of interest .....	55
3.6.5 Applying Non maxima Suppression .....	55
3.6.6 Returning Results .....	56
3.6.7 Code Snippet for detection .....	57
3.7 Segmentation .....	61
3.7.1 Segmentation method .....	63

3.7.2 Code Snippet for Segmentation.....	67
CHAPTER FOUR: RESULTS AND EVALUATION .....	70
4.0 Results and Evaluation.....	70
4.1 Experimental Setup.....	70
4.1.1 Evaluation Metrics .....	71
4.2 Comparative analysis with other deep learning algorithms .....	72
4.2.1 KNN .....	72
4.2.2 Logistic Regression (LR).....	73
4.2.3 Support Vector Machine (SVM) .....	74
4.3 Comparison of Results .....	74
4.3.1 VISUALIZATION.....	76
4.4 Comparative Analysis of the proposed system with existing works.....	77
CHAPTER FIVE: CONCLUSION AND RECOMMENDATION.....	81
5.1 Conclusion.....	81
5.2 Research Limitations .....	82
5.3 Recommendation and Future works .....	82
5.4 Contribution to Knowledge.....	83
References .....	84
Appendix.....	88

## TABLE OF FIGURES

Figure	Title	Page
Figure 1:	Classifications of Diabetic Retinopathy	16
Figure 2:	Types of Diabetic Retinopathy	19
Figure 3:	Structure of Thesis	22
Figure 4:	A basic architecture of a CNN	35
Figure 5:	Processes in classification of the system	39
Figure 6:	Images used in training the data used for the system	40
Figure 7:	Graph of Gaussian distribution	42
Figure 8:	Images before and after preprocessing	43
Figure 9:	Two cars; A and B to explain augmentation	45
Figure 10:	Code Snippet for Augmentation	47
Figure 11:	Sample of augmentation images	48
Figure 12:	Snippet for loading augmented images	49
Figure 13:	Architecture for the 5-layer deep neural network.	50
Figure 14:	Model Training	53
Figure 15:	Stages for effective DR detection	54
Figure 16:	A sample image showing how image pyramid works	56



Figure 17:	A sample image showing the application of image sliding on an image	57
Figure 18:	Comparison of an image before and after non maxima suspension	58
Figure 19:	Sample of DR detection	
Figure 20:	Code Snippet for detection	62
Figure 21:	Original image used for the segmentation process	64
Figure 22:	Images showing the masks	65
Figure 23:	Results from Hemorrhages' segmentation;	66
Figure 24:	Results from Optic disks' segmentation;	67
Figure 25:	Results from Micro aneurysms' segmentation;	67
Figure 26:	Results from Soft exudates' segmentation;	68
Figure 27:	Results from Hard exudates' segmentation;	69
Figure 28:	Code Snippet for Segmentation	71
Figure 29:	Plot of the Training loss and accuracy graph	72
Figure 30:	Result of Deep CNN Model	73
Figure 31:	Result of KNN model	74
Figure 32:	Result of Logistics regression	75

Figure 33:	Result of SVM	75
Figure 34:	Comparism of Performance Matrices across different classes of DR	77
Figure 35:	Comparism of Accuracy Matrices across different models	78
Figure 36:	Comparative Analysis of the current research with existing works	79

**LIST OF TABLES**

Tables	Title	Page
Table 1:	Result of different Models	77
Table 2:	Comparative analysis with existing Models	80

## **CHAPTER ONE: INTRODUCTION**

### **1.1 Introduction and Background**

This chapter covers the contextual background for the research work and also discusses what this study is all about and how the project will be effectively achieved, the research objectives, necessities for the research and the overall structure of this thesis is likewise discussed in this chapter.

#### **1.1.1 Emergence of deep learning in Classification and detection of Diabetic Retinopathy Fundus Images in Medical Domain**

Diabetes is a condition in which the levels of glucose or sugar in the blood are too high (The Foundation of the American Society of Retina Specialists, 2016). It is a disease that usually happens when the pancreas does not secrete insulin enough or it can't be processed properly by the body system (Vislisel & Oetting, 2010). In addition, The American Diabetes Association referred to Diabetes as a metabolic disease which is characterized by hyperglycemia which may sometimes happen as a consequences or lack both insulin secretion and action or inactions. The chronic hyperglycemia of diabetes mostly results in severe breakdown of the body and with these damages usually being long-term, it also results in dysfunction and failure of different body organs, ranging from the eyes, kidneys, nerves, heart, and blood vessels. There are so many pathogenic processes in the development of diabetes. These procedures go from autoimmune system obliteration of the pancreatic  $\beta$ -cells with resulting insulin deficiency to the anomalies that happen in protection from insulin (American Diabetes Association, 2014). World Health Organization (WHO) put diabetes as the seventh most lethal disease (Wan et al., 2018). tel

#### **1.1.2 Types of Diabetes**

According to (National Institute of Diabetes and Digestive and Kidney Diseases, 2016), there are three main types of diabetes which are Type 1, Type 2 and gestational diabetes. The type 1 diabetes is such that the body of the patients do not produce any form of insulin because their immune system fights against and destroy the cells that produces insulin in the pancreas (National Institute of Diabetes and Digestive and Kidney Diseases, 2016). The type 1 diabetes is mostly common in young adults and children although it can affect people of any age but it is predominant in those categories of people (Lal, 2016). Taking insulin daily helps people with the type 1 stay healthy, alive and makes them feel better. Type 2 diabetes on the other hand occurs mostly in people who are middle-aged and older, it is also peculiar among fat and inactive individuals; The type 2 diabetes may occur as a result of the body not producing enough insulin to function properly, this is usually called insulin resistance – fat, muscle, and liver cells do not carry glucose into the body's cells using insulin for energy (National Institute of Diabetes and Digestive and Kidney Diseases, 2013). The gestational diabetes however, develops in pregnant women and it leaves mostly after the birth of the baby although most people who have had gestational diabetes are more likely to develop type 2 diabetes in the nearest future. (National Institute of Diabetes and Digestive and Kidney Diseases, 2016). Gestation diabetes is usually as a result of the body not being able to produce enough insulin to meet the extra demands of their pregnancy (Ministry of Public Health and Sanitation, Kenya, 2013).

### **1.1.3 Diabetic Retinopathy**

Diabetes develop gradually such that it affects the circulatory system and other organs as well as the retina and this occur as a result of damaged blood vessels which then accumulate blood for a long time thus causing decline in the vision of the patient leading to what is called diabetic

retinopathy (Verma et al., 2011). Diabetic retinopathy (DR) is a vascular malady of the retina that is prevalent in patients with diabetes mellitus (Visliser & Oetting, 2010). According to (Chandrakumar and Kathirvel, 2016) Diabetic retinopathy is when harm happens to the retina due to causes that arise from diabetes. It's a foundational infection, which influences up to 80 percent of all its patients for a long time – about 20 years or more. (Chandrakumar & Kathirvel, 2016). After some time, almost all patients with type 1 diabetes and between 60 to 80 percent of patients with type 2 diabetes usually have retinopathy (Fong et al., 2003)

The diabetic retinopathy (DR) is one of the major causes of irreversible blindness among the American working-class (The Foundation of the American Society of Retina Specialists, 2016) while about 5.4 percent of the people in America above the age of 40 have diabetic retinopathy (Felman, 2017). Also, in India, there is a 3-fold increment in the rate of diabetes among people living in the rural areas for about a decade. The numbers grew from about 2.2 percent in 1989 to 6.3 percent in 2003. (Chandrakumar & Kathirvel, 2016). Likewise, Diabetic retinopathy is typically the reason for new visual impairment cases among grown-ups between the ages of 20–74 years in many nations. Eye disorders like cataracts, glaucoma and other disorders is more evident and happens more frequently in people living with diabetes (Solomon et al., 2017)

#### **1.1.4 Classes of Diabetic Retinopathy**

As shown in Figure 1, according to (Visliser & Oetting, 2010), Diabetic retinopathy can be divided into two main classes and they are Proliferative diabetic and Non-proliferative diabetic retinopathy (NPDR). The PDR occurs when the fluids sent by the retina for nourishment causes the

advancement of new blood vessels to bypass blood (Vislisl & Oetting, 2010). In doing this, they grow along the retina and on the surface of the vitreous gel and then fill the eyes. Thus, the eye vitreous shrinks and it causes the fragile vessels to shrink with age and tear. Once they discharge blood, real vision difficulty and lack of visual then comes to fruition. (Kesar et al., 2018). In PDR stage, this abnormal blood vessels are formed rapidly and results in severe vision loss. The NPDR is an initial stage of DR where tiny red spots occur in the retina. These spots are the hemorrhage while the irregular pouching of blood vessels is usually the micro-aneurysms. The layer of these blood vessels can become so damaged so much that it can permit the leakage of fluid and fatty materials called exudates to the retinal surface (Revathy et al., 2020).

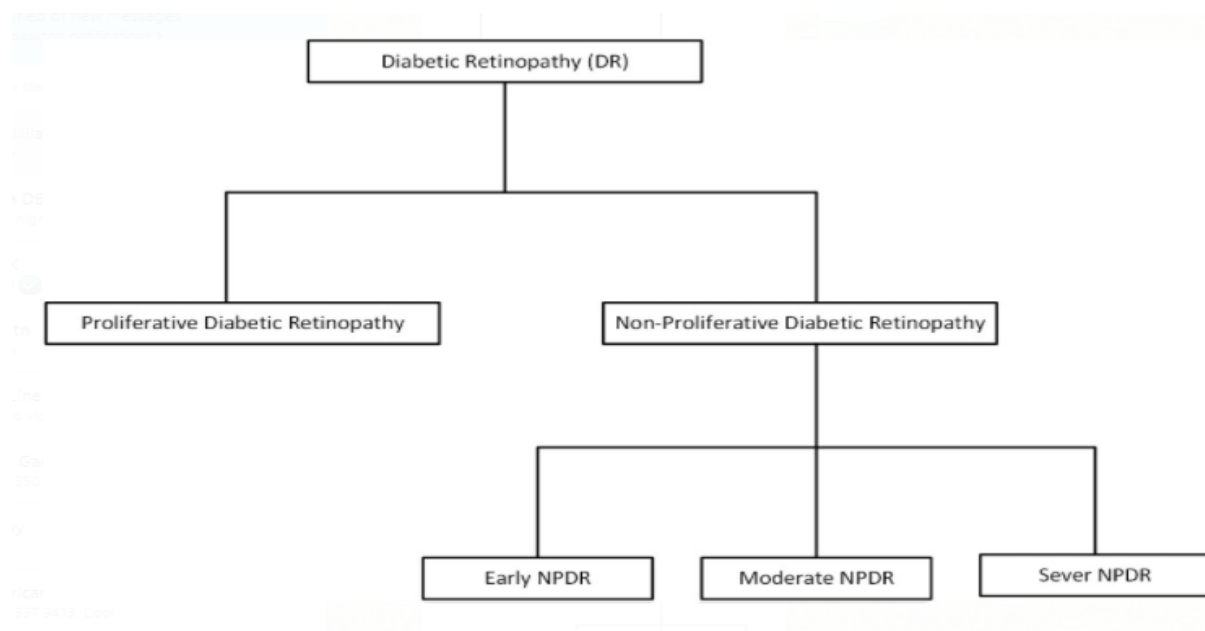


Figure 1: Classifications of Diabetic Retinopathy Source: (Azeez Babatunde)

The NPDR, based on its level of severity is divided into different sub categories (Shankar et al., 2020), while (Vislisl & Oetting, 2010) divided NPDR into three categories which are Early, Moderate and Severe NPDR. The early NPDR or mild NPDR is when just one or few micro-aneurysm is in the retina while the moderate NDPR shows more micro-aneurysm, dots and spots

as well as venous beading during a retina scan. A patient with severe NPDR means that condition is worst and will most likely result to a PDR soon. Severe NPDR is characterized by more microaneurysm spots than the mild NPDR (Vislislis & Oetting, 2010).

No matter the type of DR a patient has, it affects the vision of the patient within a short time. Therefore, it is expedient for a diabetic patient to regularly undergo retinal diagnosis (Shankar *et al.*, 2020). Despite all the aforementioned statistics, (Chandrakumar & Kathirvel, 2016) opined that about 90 percent of new diabetic retinopathy cases could be reduced if appropriate and attentive handling coupled with detailed monitoring was given to the eyes. A patient who has been suffering from diabetes for a lengthy period has a very high chance of having diabetic retinopathy. (Chandrakumar & Kathirvel, 2016). Around the globe, in 2017 about 285 million has diabetes and one-third of this population shows symptoms of diabetic retinopathy (Felman, 2017); however, it has been discovered that most times patients do not show symptoms of diabetic retinopathy until it is late and most times become too late such that the treatment of the disease will most likely become ineffective (Vislislis & Oetting, 2010) with about 49.7% of the people with diabetes undiagnosed because many are asymptomatic for years (Li, 2018)

### **1.1.5 The need for Deep learning**

Early detection of general diabetes and diabetic retinopathy can easily ensure that it is put to bed before it gets worse. While it is still at an early stage, there could still be a remedy to correcting the anomalies in the retina. As established, diabetic retinopathy is on the rise throughout the world and as it stands, manual method may be unable to keep up with the demand for early detection thereby leading to the rise in demand for automated screening techniques as early as possible. The World health organization (WHO) and the International Diabetes Federation (IDF) through its advisory group met in Geneva to review some of the WHO's guidelines in relation to ending DR



as raging disease, with the number of its patients potentially rising to 592 million in 2035. As discussed earlier, this rampaging disease has the potential to cause more complications – one of which is the DR, as such these health bodies had to give it their ultimate attention. Early detection of this disease is crucial to ensuring it doesn't graduate to a state where it is too late to be treated or managed. Diverse researches are currently in progress to improve early detection of diabetic retinopathy, although the manual method of grading and detection works quite well but the process is slow compared to the number of available cases, it also requires highly skilled medical personnel to detect it at an early stage. These further highlight the need by WHO and IDF to be open to computerized scanning systems which scans based on the images of the color fundus. Figure 2 shows an example of the retina images with NPDR and PDR. The computerized system however, still had few underlying issues that calls for development of alternative systems that are more intelligent and can make quick predictions. This problem is an open one hence enough data has been made available for as many people or outfits that can proffer solutions to the DR pandemic. Artificial intelligence, machine learning and most recently, deep learning has been brought to the fore because these mathematically inclined systems can adapt easily to different problems in diverse fields. These systems will be trained through most times a pre-existing model to be able to detect DR which will help reduce the stress on the part of the medical personnel while also helping to solve the problem faster as the number of people with diabetes keeps growing. In diagnosing diabetes, there are traits medical experts look for or signs that point towards a patient being diabetic or having have DR. These identified signs or traits the doctors look for to determine a diabetic patient or patients having DR and what class of DR they have is exactly what the machine will learn to also determine whether a person is diabetic or not. (Hemanth et al., 2019)

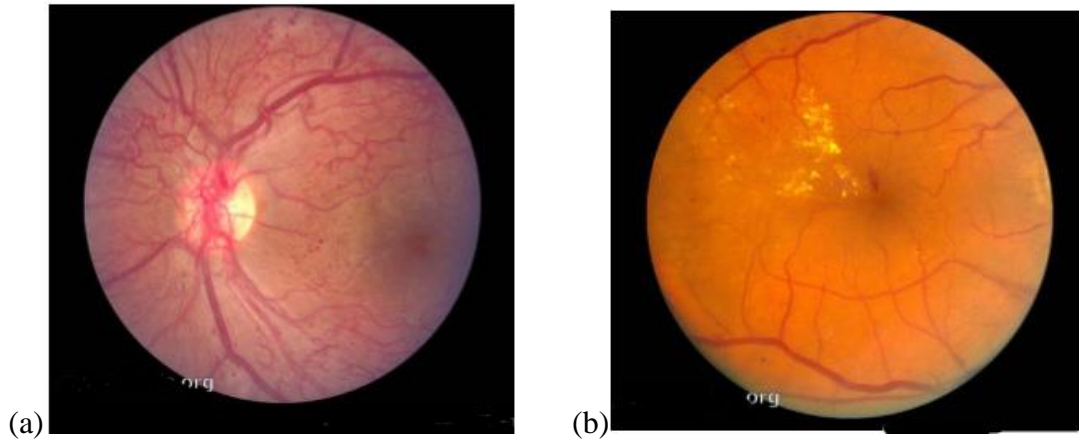


Figure 2: Types of Diabetic Retinopathy; (a) shows Proliferative diabetic retinopathy (PDR) while (b) shows Non-proliferative diabetic retinopathy (NPDR). Source: (Vislislis & Oetting, 2010)

The use of deep learning in this system ensures that it works more human-like than machine learning. Deep learning is not limited to the health sector but can equally be used in diverse sectors. It is also an important method to make use of image processing algorithm in getting near accurate diagnosis of medical image data.

Generally, classification is done by separating the features from the images, putting them in different categories and then identifying the grouped classes using the trained data with labeled classes. These features are then classified based on the severity of the disease in the patient. (Chandrakumar & Kathirvel 2016),

## 1.2 Research Purpose

This study focuses on using deep learning algorithms and custom models to not just detect but also classify diabetic retinopathy as accurately as possible. In this project the researcher plan to use an algorithm called RNet, which is based on a deep learning techniques that analyzes preprocessed image data, then builds on it to provide a platform to detect DR from a previously trained model. The model will be experimented on four algorithms which are; RNet, VGG16, VGG19 and

ResNet50.RNet will be used for the classification in the DR prediction model. The weight of the classification model will be directly used in the detection of the DR.

### **1.3 Research Objectives and Research Questions**

The research objectives are defined as follows:

1. Application of deep learning algorithms to identify and classify different classes of Diabetic retinopathy of different retina images.
2. To establish the near accurate method of detecting diabetic retinopathy and also to showcase the opportunities that are present in the use of deep CNN to help effectively detect diabetic retinopathy for quick medical attention.
3. To research on several machine learning and deep learning techniques for efficient comparative analysis.

The research Questions are therefore defined as:

1. What are the benefits and shortfall of application of deep learning algorithm and machine learning techniques in detection and classification of Diabetic retinopathy?
2. How can computer aided diagnostic (CAD) in the healthcare industry help to automate the decision making process, improve visualization of data, and refine the extraction of complex features from medical images?
3. How can deep learning algorithm and machine learning techniques be easily, effectively accessible for all to detect DR without medical experience?

### **1.4 Thesis Structure**

This section defines the structure of the thesis, the different chapters with a concise explanation of the content of chapters are discussed in the figure shown below:

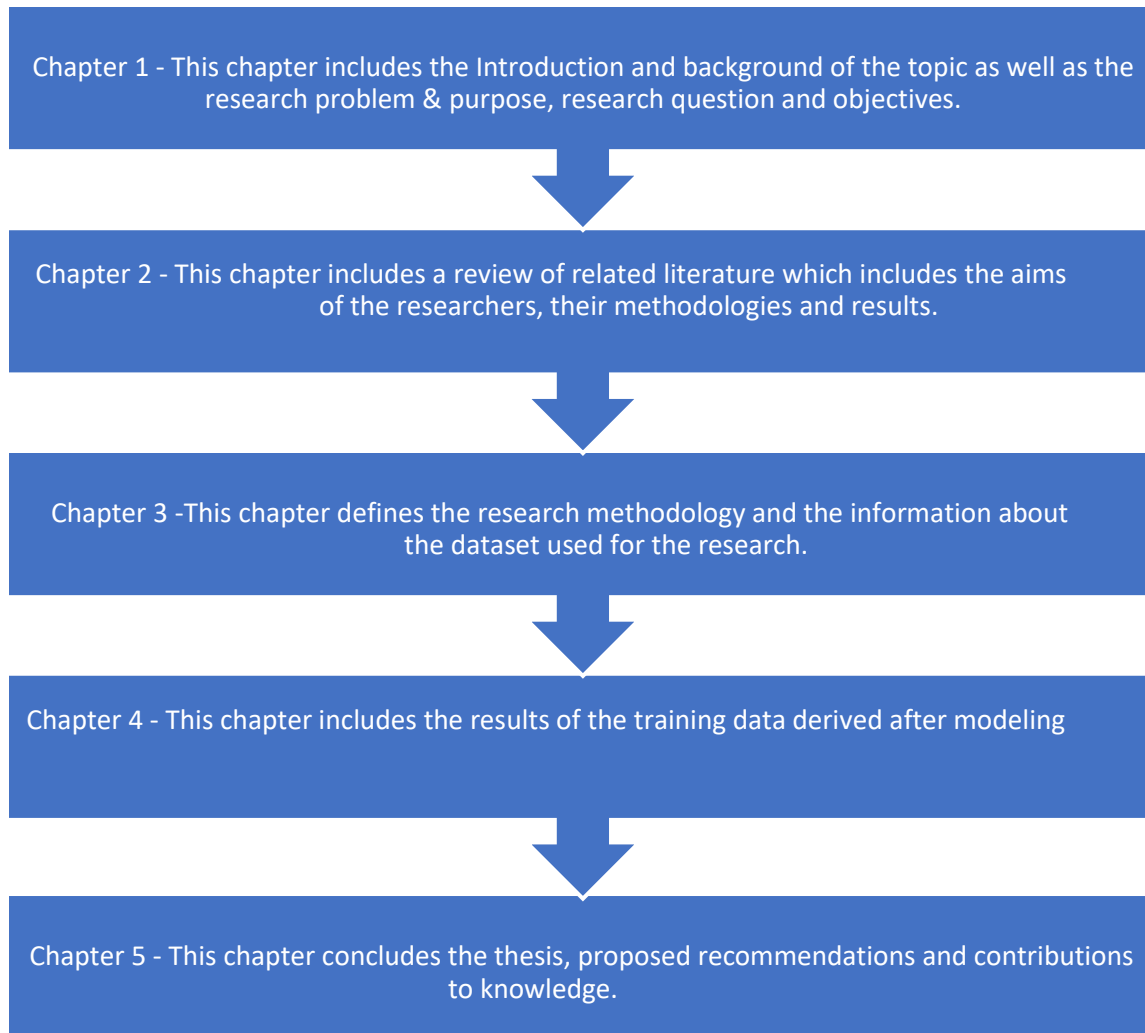


Figure 3: Structure of thesis (Azeez Babatunde)

## **CHAPTER TWO: LITERATURE REVIEW**

### **2.0 Literature Review - Introduction**

As Isaac Newton once said “If I have seen further than others, it is by standing upon the shoulders of giants” hence to have a great system, a thorough review of related works was carried out. Few of the many literatures which were more appealing are discussed in this section.

### **2.1 Automated Detection and Classification of Fundus Diabetic Retinopathy Images using Synergic Deep Learning Model**

The researcher developed an Automated Detection and Classification of Fundus Diabetic Retinopathy Images using Synergic Deep Learning Model (Shankar et al., 2020). Their project focused on the idea of classifying DR fundus images on the basis of severity level with the use of a deep learning model thus, they proposed a detection and classification model for fundus diabetic retinopathy. There were few steps in the work namely; preprocessing, segmentation and classification. The methods began with the preprocessing stage whereby unnecessary noise in the edges were removed, a histogram based segmentation was also used to extract the useful regions and a Synergic deep learning (SDL) model was used to classify the DR fundus images into various severity models while they justified it with an extensive simulation on Messidor DR dataset. The results of the evaluation of their research indicated that the presented SDL model had better classification over the existing models. The work however still had room for improvement though it had an accuracy of 99.28, sensitivity of 98.54 and specificity of 99.38 but the model could be improved on through the use of filtering techniques to improve the quality of the images before processing. The proposed method could also be extended by exploring more AlexNet and

Inception methods to provide improved performance by tuning the hyper parameters which include learning rate, epoch, dropout, number of hidden layers, momentum and activation function.

## **2.2 Diabetic Retinopathy Detection Using Deep Neural Network**

The author developed a diabetic retinopathy detection application using deep neural network, A desktop application was developed to help in identifying diabetic retinopathy (Akhila et al., 2019). It was a real-time system which needed little or no use of the internet. The framework was created with a tensor flow deep neural network design model. The model was trained and tested with several thousands of images. Five layers, 2 pool layer and 2 convolution layer and one fc layer were formed in the creation of deep neural network. The Fc layer was utilized for identifying definite global configurations of the features that the lower layers in the net detected. The model gave room for two choices for screening which were; one for image screening and one for instantaneous screening. The work however was limited in that it could only work in binary form, meaning it could only give an alert if an eye is infected with DR or not, it could not classify the DR, it also could not tell which part of the eye is infected. It was also working only on desktop at a time when the world is tending towards mobile devices.

## **2.3 Detection and Classification of Diabetic Retinopathy using Retinal Images**

Developed a system for the detection and classification of diabetic retinopathy using retina images (Verma et al., 2011). The aims of the study were to identify the blood vessel, identify hemorrhages and classify diabetic retinopathy into three different stages which are normal, moderate and non-proliferative diabetic retinopathy (NPDR). In this research, the classification of the various phases of diabetic retinopathy depended on the discovering and quantification of blood vessels and hemorrhages that are in the retinal image. The Retinal vascular was then divided using the contrast

between the variance in the blood vessels and encompassing background. Those with hemorrhage issues were detected by utilizing density analysis and bounding box methods. Grouping the stages of eye issues was done via the Random Forests procedure which has its basis on the area and perimeter of the blood vessels and hemorrhages. It was found in the wake of testing the accuracy assessment of the characterized yield that ordinary cases were grouped with 90% accuracy while moderate and severe NPDR cases had an accuracy of 87.5% (Verma et al., 2011). The room for improvement in the work is that it can include more instances of retinal images to construct a robust classifier which is needed for detecting the stages of diabetic retinopathy to get a higher accuracy. The efficiency of the classification can likewise be enhanced on increasing the extraction of features done on the images.

## **2.4 Classification of Diabetic Retinopathy Images by Using Deep Learning Models**

Researched on Classification of Diabetic Retinopathy Images by Using Deep Learning Models (Dutta et al., 2018). The aim of their research was to propose an automated knowledge model to identify the key antecedents of DR. They trained their model with three different algorithms which are, back propagation NN, Deep Neural Network (DNN) and Convolutional Neural Network (CNN). After testing the models, the CNN and DNN proved the most effective due to the CPU training time. They were also able to remove the surrounding noise around the images efficiently then yielded high accuracy results. They also used weighted Fuzzy C-means algorithm to identify the target class threshold that was needed. A way to build the system better would be to integrate the model with existing NPDR screening algorithms for improved prioritization and resourcefulness of the eye-care delivery.

## **2.5 Automated Detection of Diabetic Retinopathy using Deep Learning**

The researcher developed an Automated Detection of Diabetic Retinopathy using Deep Learning (Naithani et al., 2019) This research utilized a programmed DR evaluating framework that could characterize the images depending on the disease pathologies from four severity stages. They could distinguish that convolutional neural network (CNN) would convolve well with an input image that has a characteristic weight matrix to extract specific image highlights without losing spatial arrangement information. They utilized CNN to leverage on lots of images they got from raw pixels and a physician interpreted screening. The high chances and low inclination of the model meant that it could let CNNs diagnose a wider range of nondiabetic diseases as well. AlexNet and GoogLeNet, were used to test and train the system, these were then split in three model, which are 2-ary, 3-ary and 4- ary classification models. They were tuned to ideally work on a training dataset with several procedures which include batch normalization, L2 regularization, dropout, learning rate policies and gradient descent update rules. They conducted studies using two primary data sources, a Kaggle dataset of 35,000 retinal images with 5-class labels (normal, mild, moderate, severe, end stage) and a physician-approved Messidor-1 dataset of 1,200 color fundus images with 5-class labels. The performance of this work however degraded with the increasing number of classes.

## **2.6 An Enhanced Diabetic Retinopathy Detection and Classification Approach Using Deep Convolutional Neural Network**

Proposed an enhanced diabetic retinopathy detection and classification approach using deep convolutional neural network (Hemanth et al., 2019).The research proposed a method that incorporated the employment of image processing with histogram equalization, alongside contrast limited adaptive histogram equalization procedures. This technique was then performed by



classifying convolutional neural network. The process was approved by comparing it with 400 retinal fundus images within the MESSIDOR database, and average values for various performance evaluation parameters obtained were accuracy 97%, sensitivity (recall) 94%, specificity 98%, precision 94%, FScore 94%, and GMean 95% (Hemanth et al., 2019).. In contrast to other related studies, their technique was efficient and fruitful enough at detecting diabetic retinopathy from retinal fundus images. This work could do better by including the use of alternative solution mechanisms. The system could also use more fundus images in a large number by using different databases like Kaggle which contains thousands more images.

## **2.7 Deep Transfer Learning Models for Medical Diabetic Retinopathy Detection**

The researcher analyzed deep transfer learning models for Medical Diabetic Retinopathy Detection (Khalifa et al., 2019). This paper was aimed at using advances in computer science techniques, such as artificial intelligence (AI) and deep learning (DL) to help increase the detection of DR at the early stages for increased chances of recovery and the possibility of reduction in vision loss in patients. In this paper, they examined deep transfer learning models for medical DR detection. The Deep Learning models were trained and tested using the Asia Pacific Tele-Ophthalmology Society (APTOS) 2019 dataset. The research was a pioneer of the APTOS 2019 dataset. The models used in this research were AlexNet, Res-Net18, SqueezeNet, GoogleNet, VGG16, and VGG19. These models were selected because they consisted of a small number of layers when compared to larger models, such as DenseNet and InceptionResNet. Data augmentation techniques were used to make the models more robust and overcome the overfitting problem. The AlexNet model had the maximum accuracy at 97.9%. However, the model that was used in this research used a small number of layers and it in turn diminished the training time and the computational complexity.

## **2.8 Diabetic Retinopathy Detection using Machine Learning**

The researcher developed Diabetic Retinopathy Detection using Machine Learning (Revathy et al., 2020). They embarked on this research due to the problem with the manual method of detection. They opined that the Manual detection of diabetic retinopathy by ophthalmologist took lots of time and patients were made to suffer as a result of this. Hence they proposed an automated system which could help detect diabetic retinopathy as quick as possible and then follow-up with required treatments to avoid further degradation to the eyes. The study showed a machine learning method for extracting three features which are exudates, hemorrhages, and micro aneurysms and they equally achieved classification using hybrid classifier which is a combination of support vector machine, k nearest neighbor, random forest, logistic regression, multilayer perceptron network. To detect DR, the system counted the number of micro aneurysms that occurred, counted the number of hemorrhages and the number of exudates that occurred in the image. These features were then calculated then fed to the SVM, KNN and Random Forest classifier. The models used the extracted features to calculate the disease grade and group it as normal or abnormal. The evaluation of their research work showed, the highest accuracy values of 82%. Hybrid approach achieved a precision score of 0.8119, Recall score of 0.8116 and f-measure score of 0.8028. A deep learning approach might have been more suitable for this work to enable the system not just make informed decisions but also make unsupervised intelligence decisions. Just as stated in the research on Automated Diabetic Retinopathy Detection using Pre-trained Deep Neural Network (Boral & Thorat, 2020). Their study attempted to solve the problem of DR by detecting it early and using the application of machine learning but because the efficiency of machine learning algorithm depended on the quality of feature extraction which requires domain knowledge. The

work thereby used deep learning algorithm which automatically identifies the pattern and classifies the retina images into one of the five class based.

## **2.9 Detection and Classification of Diabetic Retinopathy in Fundus Images using Neural Network.**

The researcher developed a system for the Detection and Classification of Diabetic Retinopathy in Fundus Images using Neural Network (Sankar et al., 2018). The aim of the research was to automatically detect and classify the severity of diabetic retinopathy. At first stage, the lesions on the retina especially blood vessels, exudates and micro aneurysms were segmented. Features such as area, perimeter and count from the lesions were used to classify the stages of the disease by applying artificial neural network (ANN). The method used 214 fundus images from DIARECTDB1. The system gave the classification accuracy of 96%, precision of 95% and accuracy 96% respectively. The execution time took about 28.064 seconds and was quite helpful to ophthalmologists. A better work can still be done on the system by optimizing the classifier performance with more images, extracting details features and using a different classifier.

## **2.10 Diabetic Retinopathy Stage Classification using CNN**

Researched and Designed of deep learning model for the Classification of Diabetic Retinopathy Stage using CNN (Nikhil & Angel, 2019). In their research, classified color fundus retinal images DR into five stages using a CNN. Images with diabetic retinopathy were classified into five groups from the opinion of an expert of ophthalmologists. Three Convolutional Neural Networks are deployed for stage classification of DR. By the concatenation of these three networks, VGG16, AlexNet, and InceptionNet V3. After evaluating the system, an accuracy of 80.1% (Nikhil & Angel, 2019) was obtained. The model can be worked on for better accuracy.

### **2.11 Automated Detection of Diabetic Retinopathy using Deep Learning**

Researched and Develop an Automated Detection of Diabetic Retinopathy using Deep Learning (Lam et al., 2018). The research work demonstrated the use of convolutional neural networks (CNNs) on color fundus images for the recognition task of diabetic retinopathy staging. Their network models were able to achieve a test metric performance akin to that of the traditional literature results, with validation sensitivity of 95%. They also used multinomial classification models, and showed that errors usually occur due to the misclassification of mild disease as normal and this occurs as a result of the CNNs not being able to detect subtle disease features. They deduced that preprocessing with contrast limited adaptive histogram equalization and making sure that dataset fidelity by expert verification of class labels helps in enhancing the recognition of subtle features. The learning model on pre-trained GoogLeNet and AlexNet models from ImageNet improved peak test achieved accuracy figures of 74.5%, 68.8%, and 57.2% on 2-ary, 3-ary, and 4-ary classification models (Lam et al., 2018). This research work however would be better if they worked on improving detection of mild disease and transitioning to more challenging and beneficial multi-grade disease detection while also improving on its accuracies.

### **2.12 Classifying Diabetic Retinopathy using Deep Learning Architecture**

A research for proposed strategy about Deep Learning Fundus Image Analysis for Diabetic Retinopathy and Macular Edema Grading (Chandrakumar & Kathirvel, 2016). They proposed a deep learning approach such as Deep Convolutional Neural Network (DCNN) which gives high accuracy in classification of these diseases through spatial analysis. The DCNN is more complex architecture inferred more from human visual perspectives. The research proposed this solution to find a better and optimized way of classifying the fundus image with little pre-processing techniques. The proposed architecture used with dropout layer techniques had about 94-96%

accuracy. Also, it was tested with popular databases such as STARE, DRIVE, Kaggle fundus images datasets. The limitations of the work are however that an additional stage augmentation was needed for the images taken from different camera with different field of view. The network architecture was quite complex and also computational-intensive which required high-level graphics processing unit to process the high-resolution images when the level of layers stacked more.

### **2.13 Computer-Assisted Diagnosis for Diabetic Retinopathy Based on Fundus Images Using Deep Convolutional Neural Network**

The researcher also developed a Computer-Assisted Diagnosis for Diabetic Retinopathy based on Fundus Images Using Deep Convolutional Neural Network (Li et al., 2019). In their work they presented a novel algorithm based on deep convolutional neural network (DCNN). Unlike the traditional DCNN approach, they replaced the commonly used max-pooling layers with fractional max-pooling. Two of these DCNNs with a different number of layers were trained to derive more discriminative features for classification. After combining features from metadata of the image and DCNNs, the researcher trained a support vector machine (SVM) classifier to learn the underlying boundary of distributions of each class. For the experiments, the study used the publicly available DR detection database provided by Kaggle. 34,124 training images and 1,000 validation images were used to build the model and tested with 53,572 testing images. The proposed DR Classifier classified the stages of DR into five categories, labeled with an integer ranging between zero and four. The experimental results showed that the proposed method achieved a recognition rate up to 86.17% (Li et al., 2019). In addition to designing a machine learning algorithm, the researcher also developed an app called “Deep Retina.” This app was Equipped with a handheld ophthalmoscope, this was to help the average person take fundus images by themselves and obtain an immediate

result, calculated by the algorithm. The research was able to help in terms of home care, remote medical care, and self-examination. The accuracy of the work at 86.17% was however low compared to that of other related models.

This current research work aims to improve on the limitations of some of these research works, especially the work of (Hemanth et al., 2019) & (Sankar et al., 2018) by training the system with a larger dataset, improvement in prediction accuracy in the system of (Nikhil & Angel, 2019) and (Lam et al., 2018), Also to improve on the work of (Akhila et al., 2019) through the use of recently developed deep learning algorithms as opposed to machine learning used by (Revathy et al., 2020) as well.

## **CHAPTER THREE - METHODOLOGY**

### **3.0 Introduction**

In automating the classification and detection of DR, few stages must be analyzed. The retinal lesions must be carefully extracted. These retinal lesions include the exudates, micro-aneurysms and hemorrhages. These features must be well classified to ensure the accurate detection of DR in a patient. If the classification is not done as accurately as possible, it could lead to challenges like alternate lightening and contrast across the images and also the retinal lesions may be too similar to identify just as the blood vessels, optic disc and fovea (Paranjpe & Kakatkar, 2014). There are thin lines in some of these features meaning that proper classification must be done, the larger hemorrhages for example, have the same color as the blood vessels but have different geometrical features while the smaller micro-aneurysms and hemorrhages have similar color, contrasts and geometrical features with the thin and smaller blood vessels (Paranjpe & Kakatkar, 2014). The optic disk also has the same color with the exudates, the same way the fovea shares the same color as the hemorrhages and micro-aneurysms. These little features must be carefully examined and classified for accurate detection

### **3.1 Deep Learning**

According to (Aladawi et al., 2019), deep learning (DL) is a growing or deeper technology of machine learning which has transformed artificial intelligence since its inception in the 2000s. One of its most crucial property and upgrade on machine learning is that it can easily learn feature representation. It also uses several learning methods or models to process or train data without needing to manually engineer the system for automatic recognition of intricate structures in high dimension data and this has proved very successful over the years (Aladawi et al., 2019). Deep

learning algorithm learns by using the features from a large chunk of data that is based on learning features from data by processing huge amount of data and extracting the meaningful patterns (Raman et al., 2018). Deep learning has widely helped in computer vision and natural language processing (Khalifa et al., 2019), it has been used in medicine and has had its effect in the early detection of diabetic retinopathy (Aladawi et al., 2019). The deep learning algorithm that will be used in this work is the Convolutional Neural Network (CNN).

### **3.2 Convolutional Neural Network (CNN)**

The CNN is a supervised learning method which takes an image as input and marks the image with different labels to give it distinct features for better recognition. This is to ensure that the images can be differentiated from one another (Aladawi et al., 2019) CNN uses kernels or filters to detect the needed features it needs to learn in the image. CNN is expected to just as the name suggests convolve the kernel (a matrix or array of trained values or weights) on an input image to check if the feature to be detected is present in the image. A convolution operation is taken to determine how much of that feature is present by computing the dot product of the kernel and the input area that the kernel overlaps (O'Mahony et al., 2019). A Convolutional Neural Network (CNN) has an input and output layer, as well as several hidden layers. These layers are divided into three types: CONV, POOL, and Fully Connected FC (Sakib et al., 2018). The CNN majors on data processing with different convolution layers. These layers are convolution layers and for the sake of this study, 5-layer convolution network shall be used. These layers are the core building blocks of the CNN, they filter the images which were served as input and extract the useful data from it for that specific task (Aladawi et al., 2019). Pooling layers is also important in CNN but in its case, it is used in maintaining the means of providing efficiency during the training of the models by exchanging the network's output at specific times and places where extraction of core features is needed so as to



achieve minimal amount of rotation, decrease in memory consumption and positional invariance (Coşkun et al., 2017). CNN works by focusing on the idea that the data fed into it contains images making the architecture to be built in a way that most accurately fits the necessity for coping with the particular form of information. Although one very important variation is that the layers in CNN has its neurons arranged into three dimensions also known as spatial dimensionality of the input which is the height, width and depth. The architecture of a CNN is shaped in such a way that its advantage is from the 2D structure of the input image which is done through neighboring connections and tied weights, some pooling – which results in translation invariant characteristics. The CNN has lots of convolutional and subsampling layers that are occasionally followed by fully connected layers as shown in figure 3. The input and output of the stages are sets of arrays denoted as feature maps. When the image is colored, the feature maps will consist of a 2D array with a color channel of the image, a 3D array for video and 1D for the audio. The output would then denote the characters extracted from the locations on the data (Sakib et al., 2018).

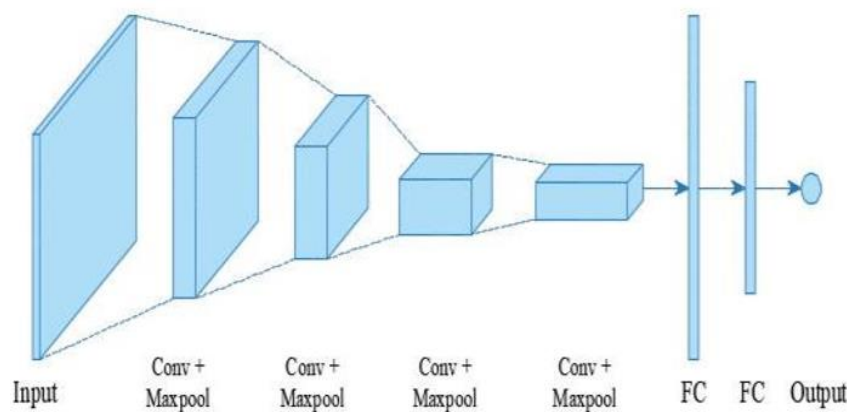


Figure 4: A basic architecture of a CNN. Source: (Sakib et al., 2018)

The advances in CNN – which evolved to deep CNN has ensure there is an increase in the accuracy of image classification and detection. Its pre-trained models have succeeded in making improvements over the years and are more accurate and efficient. Thus, ensuring that deep CNN has been one of the major algorithms use for tackling detection and classification related subjects.

### **3.3 Deep Neural Network**

The deep neural network or deep convolutional neural network is an artificial network which can be used for predictive modellings which can be trained using datasets. The network has its foundations on convolution layers which ranges in figures, the convolution layers based on these figures can be further divided into depth and point wise convolutions. The depth wise convolution is used when a single filter is to be applied on the input layer channels while the point wise comes to play when a linear combination of the output layer from the depth wise layer is to be formed (Akhila et al., 2019).

### **3.4 Deep learning methodology**

The classification and learning methodology that will be used in this project is the Convolutional Neural Network (CNN) as mentioned earlier. The use of this deep CNN for this system will undergo three major phases for effectiveness. These phases include the classification, detection, and segmentation.

Although there are several other proven methodologies that can also be used to detect DR as seen in chapter two where several scholars used some of these proven methods. The other three methods which stood out during the course of this research will be reviewed. These three methods are the Support Vector Machine, the KNN or K-Nearest Neighbor and the Logistics Regression method.

### 3.4.1 Support Vector Machine (SVM)

The support vector machine, like the CNN is a supervised learning algorithm used for classification problems (Revathy et al., 2020). This algorithm has also been proven in solving the classification problems of DR (Carrera et al., 2017)&( Ramya, 2018).SVM is characterized by an isolated hyperplane. The preparation phase of the SVM is connected to check the prepared information for locating the best approach to categorize the images in their specific classes (i.e. Normal, Proliferate and Non-Proliferate DR). According to (Ramya, 2018), in the case of nonlinear bends, SVM uses a part capacity to outline information into an alternate space where a hyper plane would be useable to perform the partition. SVM algorithm assume there are  $n$  features, these data will then be plotted such that each data has  $n$  features as a point in  $n$  dimensional space meaning each of the features denotes the value of specific coordinates in the  $n$  dimension space allowing for classification of the plotted data points into  $n$  classes via the use of an hyperplane (Revathy et al., 2020).

### 3.4.2 KNN (K-Nearest Neighbor)

The KNN algorithm is also known as the K-nearest neighbors. Like the SVM, is likewise a supervised learning algorithm which is also used mainly to solve classification problems (Revathy , et al., 2020). According to (Bethanney et al., 2015) KNN is a proven method for classifying DR. it works by using the idea of similarity and uses the distance between the closeness of objects (points) on a graph. The objects are assigned to most common class among its  $k$  distance nearest neighbors. If 1 is assigned to  $K$  (i.e.  $K=1$ ), the algorithm simply becomes nearest neighbor algorithm and the object is classified to the class of its nearest neighbor (Bethanney et al., 2015). Neighbors are determined by the distance and the number given to  $K$ , This distance is what the researcher would use to classify the data, thus the more the distance between the objects, the lesser the similarity and lesser distance means higher similarity. The data used in this method usually

comprise of a set of vectors and class label which is associated with the vector positions of the objects. Few advantages of KNN according to (Bethanne et al., 2015) is that it is analytically tractable, easy to implement although, on the other hand, they require large storage capacities and computing the recall is somewhat intensive.

### **3.4.3 Logistic Regression (LR)**

According to (Rahimloo & Jafarian, 2016), Regression analysis is a statistical method used to examine the link between the dependent variable and independent variable with the aim of predicting which variable is independent and which is dependent. The Logistic Regression is a type of Regression Analysis and the model is best used in making predictions to determine the chances of an individual to develop DR (Senthilvel et al., 2012). Logistics Regression shows the probability of a zero or one; it can help determine the presence or absence of a thing and it is based on chi-square. The Logistic regression model is best for a two-way dependent variables such as illness or health, death or life (Rahimloo & Jafarian, 2016) meaning it can only be used in determining if a person has DR or not. For effective classification of DR, the Logistics Regression technique has to be used in a hybrid system where other methodologies are featured alongside the LR algorithm (Revathy et al., 2020).

## **3.5 Classification**

The classification of the system is in five major steps, which includes collection of the dataset, the preprocessing of the data, data splitting and label binarization, augmentation and data modeling and training by passing it through the deep convolution network.

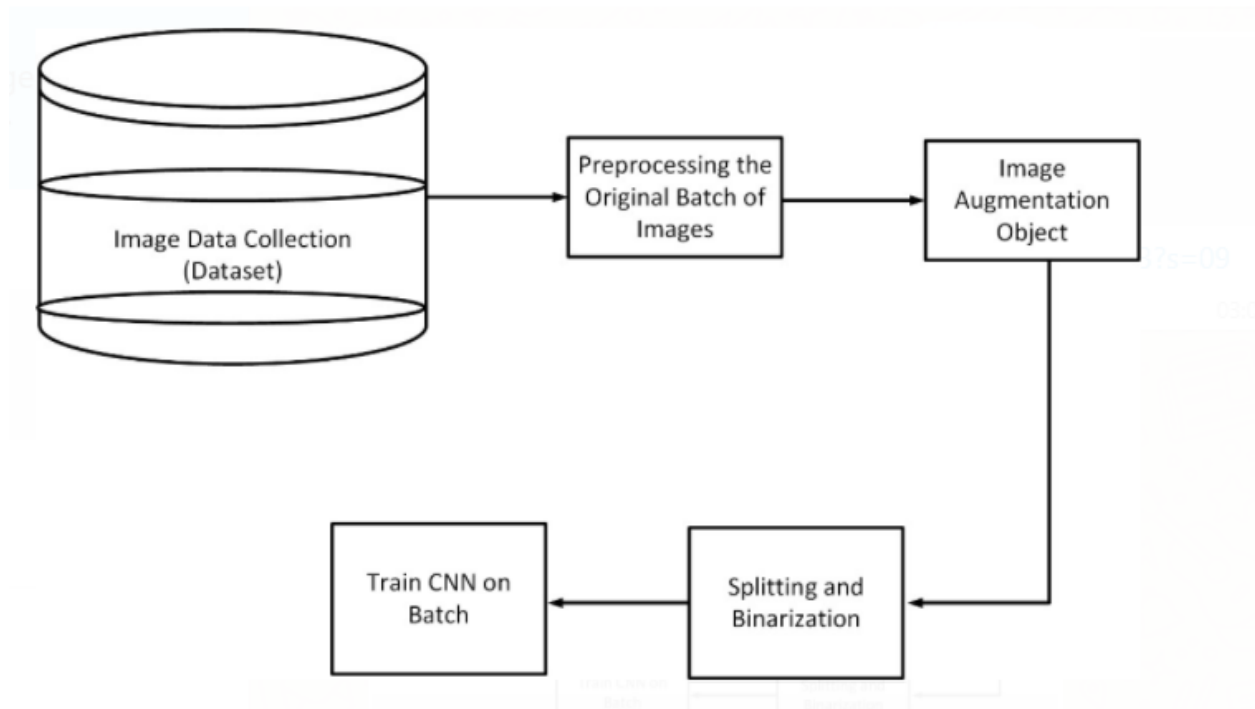


Figure 5: Processes in classification of the system Source: (Azeez Babatunde)

### 3.5.1 Dataset Collection

The collection or gathering of the necessary dataset is one of the foremost things to do in any machine or deep learning system as it is this collected data that will be processed and trained to enable the system gather as much information as it will need for effective classification and detection. The needed images for the training will be collected (Akhila et al., 2019). For this research work, 413 images of DR Dataset containing annotations and images were obtained from the Indian Diabetic Retinopathy Image Dataset (IDRID) which was gathered from <https://ieee-dataport.org/open-access/indian-diabetic-retinopathy-image-dataset-idrid>, it is an open source data collection community. The documented number of images that was used in this research work was 413 original color fundus images, The datasets consisted of color images that differs in height, width, pixels, and matrices. The collected images from IDRID were splits into

five characteristics labels which are normal, mild, moderate, severe and proliferate. Figure 6 are few of the images that were used for training the model in this project.

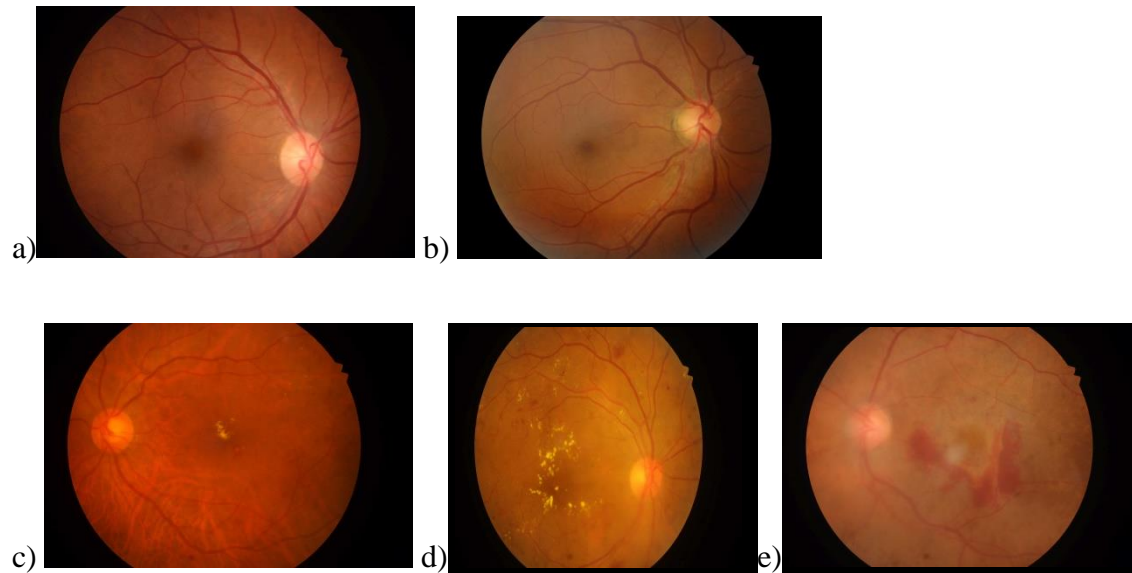


Figure 6: Sample of images used in training the model. **a** is an image with no Diabetic Retinopathy, **b** is an image with mild DR, **c** is an image with moderate DR, **d** is an image with severe DR while **e** is an image with proliferate DR Source: (Prasanna Porwal, 2018)

### 3.5.2 Data Preprocessing

After identifying and gathering the datasets, then the next step taken by the researcher was to build an efficient system to preprocess the data. The data is first converted to a hierarchical data format so that it can be used for the purpose it was intended. This format ensure accurate preprocessing, augmentation and training of the model (Lam et al., 2018). The main aim of preprocessing the data is to remove noise and filter images from the collected data which will in turn help the organization of the images while training with the neural network (Akhila et al., 2019).

Usually, when the images are collected from the databank, it may vary in sizes, color, some may not even be clear enough so it is expedient that the images are preprocessed for uniformity sake, it

must have the same size, improved visibility, just as said earlier, noise must be removed for clarity. In preprocessing, the data must undergo few steps (Lam et al., 2018) & (Li et al., 2019) divided these steps into three steps which are, rescaling the images to the same size since they are mostly circular and then the images were resized to the same diameter, color divergence caused by different ophthalmoscopes is removed by ensuring the local average color value is set at 50% grayscale and lastly the periphery is moved by clipping 10% from the image borders; (Akhila et al., 2019) divided the steps into two which is filtering and Conversion- they defined filtering as using a convolution filter to ensure the images are smoother and more reliable by removing the noise, shadows and refining the color variations while they termed conversion as the resizing of the images in the dataset to 256 by 256 pixels. (Lam et al., 2018) On the other hand, subdivided the steps into two which are, cropping the image to isolate the circular colored image of the retina and adjusting the contrast or normalizing the image using a Contrast Limited Adaptive Histogram Equalization (CLAHE) filtering algorithm. (Dutta et al., 2018) Warned however that it is dangerous not to preprocess images as it will lead to lower accuracy and inefficient classification and detection.

For this research, the Images that were contained in the dataset were of various sizes, hence the need to resize the images in the bid to make them uniform and usable by our model. The resizing was done with a custom code. The custom code is written in such a way that it can chain together different kind of processes techniques. First, in filtering the images the Gaussian blur was used to remove the noise and create images that can improve the learning ability of the model. The Gaussian blur used is a 5 by 5 kernel. The Gaussian blur or Gaussian smoothing as the name implies is used to “blur” images, it is a 2D convolution operator used to remove noise and other seemingly unnecessary details. It uses a kernel which is bell-shaped to represent the shape of a

Gaussian hump. Figure 6 shows a comparison between a 1D image (before using Gaussian blur) and the result after using Gaussian blur

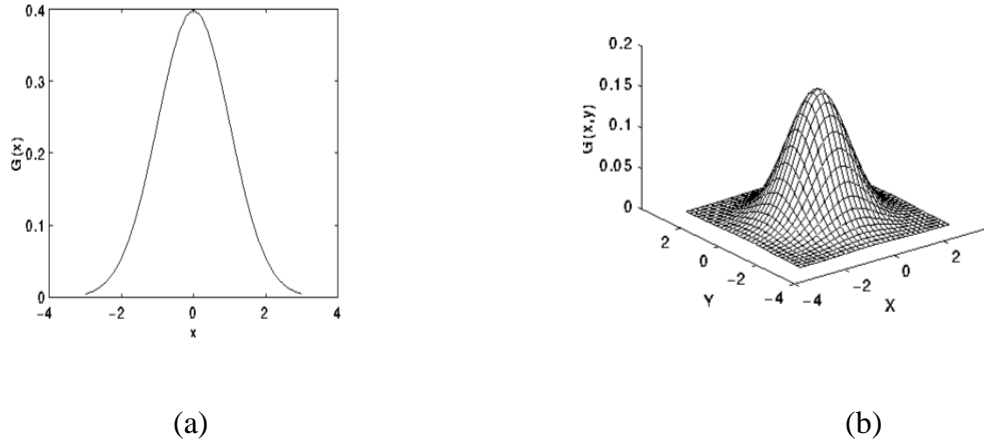


Figure 7: Graph of Gaussian Distribution; (a) is a 1-D Gaussian distribution with mean 0 and  $\sigma=1$  while (b) is a 2-D Gaussian distribution with mean (0,0) and  $\sigma=1$ . Source: (Fisher et al., 2003)

The Gaussian distribution in figure 6a has the form of

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

Where  $\sigma$  is the standard deviation of the distribution. Assuming that the distribution has a mean of zero (i.e. it is centered on the line  $x=0$ ).

The Gaussian distribution in 6b has the form of

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

Gaussian smoothing is expected to use the 2D distribution as a “point spread” function which is performed by convolution. (Fisher , et al., 2003) These images are assembled as discrete pixels



needed to produce a discrete approximation to the Gaussian function before convolution can be performed on the images. Figure 8 are examples of images before and after preprocessing.

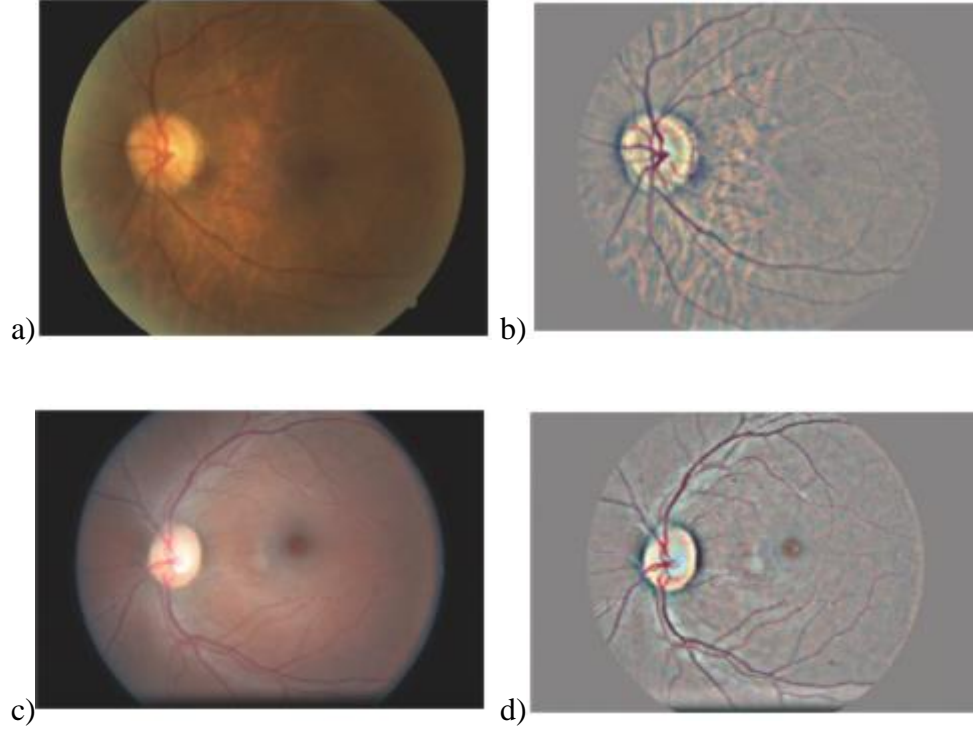


Figure 8: **a and c** shows images before preprocessing while **b and d** shows the images after preprocessing. Source: (Li et al., 2019)

After using the Gaussian blur to remove the noise, the images were then converted to array. This is because machine learning and deep learning algorithms are algorithms made with strong mathematical paradigm; they consume numeric values not images, texts, or videos which was the form of data that was downloaded. Images are simply an array of numbers. So, each image was converted to its array equivalent that can be feed to a neural net. The images were thereafter scaled to within the range of [0,1], since images consist of pixels within the range of [0, 255]. The normalization formula for converting the range to [0, 1] is

$$x_{i_{new}} = \frac{x_i - x_{min}}{x_{max} - x_{min}} \quad (1)$$

The formula in equation 1 is used to transform the image from its range of  $[0, 255]$  to  $[0,1]$

Where  $x_{i_{new}}$  is the new range the image was transformed to,  $x_{max}(255)$  is the maximum range it was transformed from while  $x_{min}(0)$  is the minimum range it was transformed from.

### **3.5.3 Data Splitting and Label Binarization**

After preprocessing the data and making sure that it is a good fit for modeling, there is the need to split that data into training and validation (Test). The data used in this research work was split into 80% training and 20% validation. Also, the labels which are categorical (i.e. are split into different categories) are then converted to numbers so it can have 5 unique values which are further split into binarized integer values that reflects the category which it is classified into.

### **3.5.4 Data Augmentation**

Data augmentation helps to make the model more robust and versatile by increasing the no of the images through aforementioned processes, the model tends to learn through various sizes of the images of the dataset as against the original images captured as provided. To explain better in lay man's language, when data is imported from the data source and has been preprocessed, the network will need to be fed as simply as possible so it doesn't learn the wrong information.

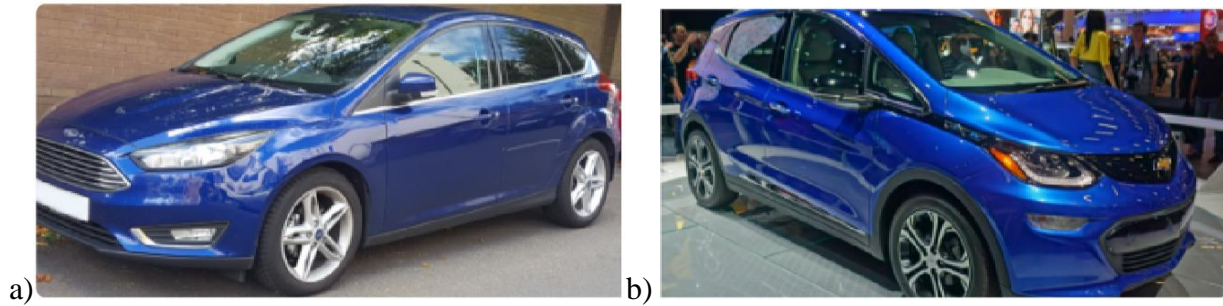


Figure 9: Two cars; (a) represents Ford facing the left, (b) represents Chevrolet facing the right.

Source: (Gandhi, 2018)

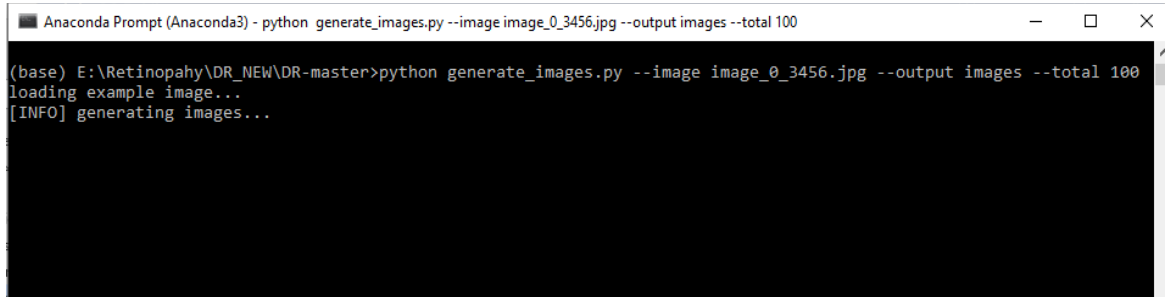
If these images in figure 8 are fed into the network, the network may automatically train itself such that once it sees a car facing the left, its first intuition is that it is a Ford car and when a car faces the right, it predicts it as a Chevrolet car. Augmentation is a means of avoiding positional variance just as seen in figure 8 where the system predicts due to the position the image is facing or the position the image takes (i.e. if the image faces left, the system calls it Ford and if the image faces right, the system calls it Chevrolet because of the varying position of the images). Variance is not just in the changing of position but also in its shape or color. Overfitting occurs because of model prediction due to the high variances in the image. Thus, what augmentation does is that, depending on the type of augmentation technique used, it tweaks the image in several other ways. Data augmentation encompasses a wide range of techniques used to generate new training samples from the original ones by applying random perturbations while making sure that the class labels are not changed. There are techniques that could flip the image to some specific degree; zoom the image, crop the image, lighten or deepen its color and so on just so the system can learn the image in different ways and not just see it in “black and white”. With the addition of the new augmented images, the dataset becomes more robust hence we have more dataset than we originally imported to train the model. These helps the system become more accurate, as stated about the limitation of

the work of (Verma et al., 2011) where the work was not robust enough hence reducing its accuracy. With data augmentation, the data is expected to be more robust as according to (Hemanth et al., 2019), the more robust the data, the better the accuracy.

Since our dataset consists of only 413 images which in turn is not deemed fit for this research, and for the system to make better detection and classifications, the images were augmented to 10,119 so as to increase the number of images which will apparently suffice during the training process. Data augmentation also ensures that the model becomes more invariant for any form of noise or transformation. Our augmentation technique makes use of four different methods, the first of the methods is the rotation method, where the images are rotated to 30° degree; the second method is the shearing method, where the image is randomly sheared by 15% but if it is just the width or height that is to be sheared, the width of images are either sheared to the right, left by 20% and the height is sheared up or down by 20%; another method used is the zoom where the image is either zoomed out or in by 15% and the last of the methods is the flipping method which can either be flipped horizontally or vertically. These four methods ensure that new images generated thus become more taking the number of images to be trained to 10,119 precisely.

This approach also ensured that the system does not need to memorize data as much, while also making the data more accountable and robust for training and testing.

The code snippet for **augmenting** the images is as follows;



```

Anaconda Prompt (Anaconda3) - python generate_images.py --image image_0_3456.jpg --output images --total 100

(base) E:\Retinopathy\DR_NEW\DR-master>python generate_images.py --image image_0_3456.jpg --output images --total 100
loading example image...
[INFO] generating images...

```

---

```

4  from tensorflow.keras.preprocessing.image import ImageDataGenerator, img_to_array, load_img
5  import numpy as np
6  import argparse
7
8
9  # construct the argument parser and parse the arguments
10 ap = argparse.ArgumentParser()
11 ap.add_argument("-i", "--image", required=True,
12               help="path to the input image")
13 ap.add_argument("-o", "--output", required=True,
14               help="path to output directory to store augmentation examples")
15 ap.add_argument("-t", "--total", type=int, default=100,
16               help="# of training samples to generate")
17 args = vars(ap.parse_args())
18
19
20 # load the input image, convert it to a NumPy array, and then
21 # reshape it to have an extra dimension
22 print("loading example image...")
23 image = load_img(args["image"])
24 image = img_to_array(image)
25 image = np.expand_dims(image, axis=0)
26
27
28 # construct the image generator for data augmentation then
29 # initialize the total number of images generated thus far
30 aug = ImageDataGenerator(
31     rotation_range=30,
32     zoom_range=0.15,
33     width_shift_range=0.2,
34     height_shift_range=0.2,
35     shear_range=0.15,
36     horizontal_flip=True,
37     fill_mode="nearest")
38 total = 0
39
40
41 # construct the actual Python generator
42 print("[INFO] generating images...")
43 imageGen = aug.flow(image, batch_size=1, save_to_dir=args["output"],
44                    save_prefix="image", save_format="jpg")
45
46
47 # loop over examples from our image data augmentation generator
48 for image in imageGen:
49     # increment our counter
50     total += 1
51     # if we have reached the specified number of examples, break
52     # from the loop
53     if total == args["total"]:
54         break

```

Figure 10: Code Snippet for Augmentation. Source: (Azeez Babatunde)

The images used in training the data model for this research work are augmented to some of the images shown in figure 11.

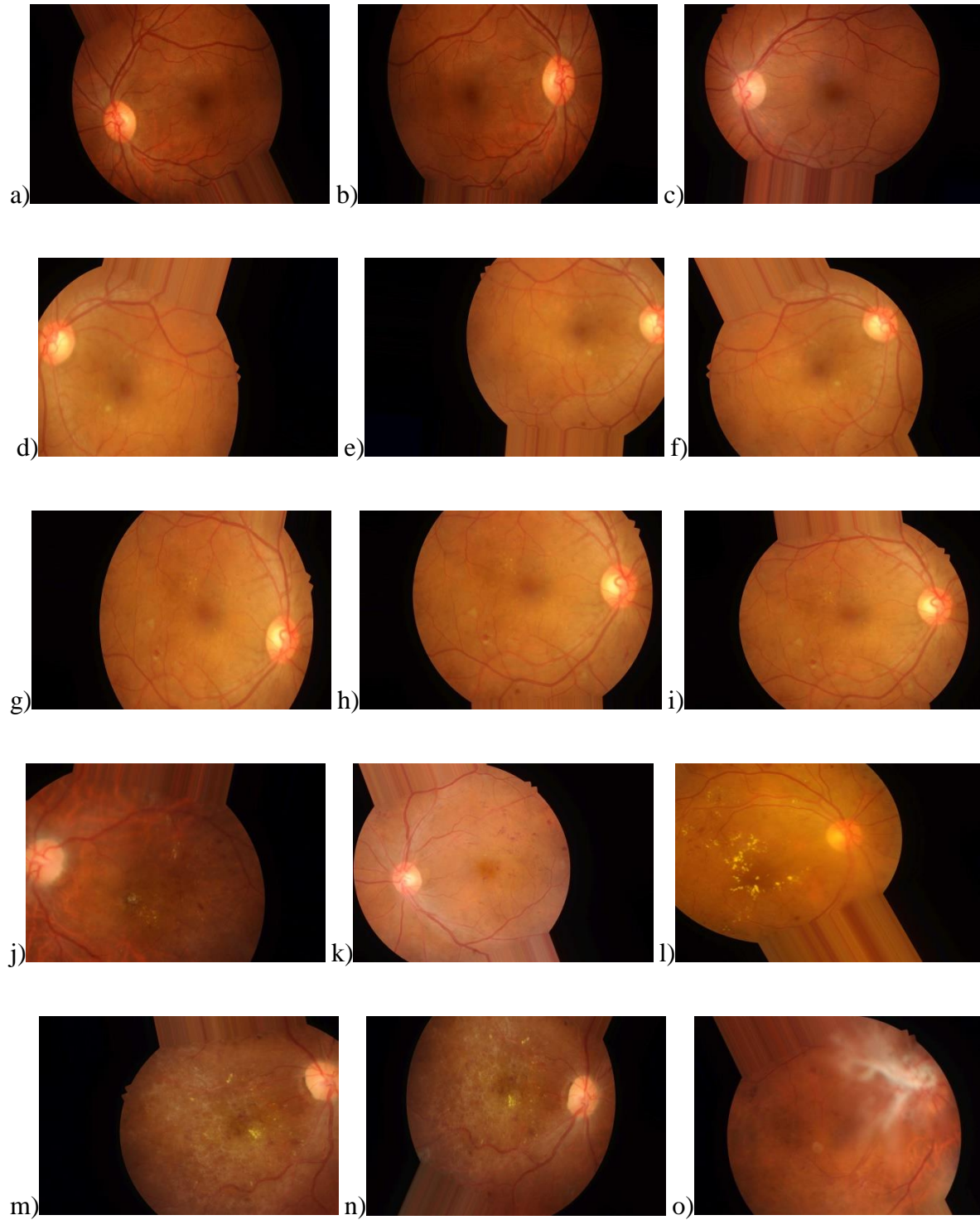


Figure 11: **a, b and c** are images with no DR; **d, e and f** are images with mild DR; **g, h and i** are

images with moderate DR; **j, k and l** are images with severe DR while **m, n and o** are images with proliferate DR. Source: (Azeez Babatunde)

### 3.5.5 Data Modeling and Training

In training and modeling the data, there are some features in the data that the system will have to look out for to be able to successfully group them into its successful classifications. For effectiveness, the architecture of the system for learning has been modeled into a 5-layer convolutional network which can also be called RNet. RNet is a custom name given to the 5-layer network for fluidity and easy comparison with other networks. This RNet comprises of a mini VGG network which has 4 layers of convolutional network and a custom one-layer network making the architecture 5 layers. The modeling phase is subdivided into model architecture construction, the optimization phase, the compilation phase, and the training phase.

#### 3.5.5.1 Model Architecture Construction

The architecture contains 5 layers of convolution, structured in a way to help improve generalization. The model architecture is as shown in figure 10 below.

```
(base) E:\Retinopathy\DR_NEW\DR-master>python RNET_ON_RETINO1.py --dataset Augmented_images --model demo100.hdf5
Loading in image data
[INFO] processed 500/10119
[INFO] processed 1000/10119
[INFO] processed 1500/10119
[INFO] processed 2000/10119
[INFO] processed 2500/10119
[INFO] processed 3000/10119
[INFO] processed 3500/10119
[INFO] processed 4000/10119
[INFO] processed 4500/10119
[INFO] processed 5000/10119
[INFO] processed 5500/10119
[INFO] processed 6000/10119
[INFO] processed 6500/10119
[INFO] processed 7000/10119
[INFO] processed 7500/10119
[INFO] processed 8000/10119
[INFO] processed 8500/10119
[INFO] processed 9000/10119
[INFO] processed 9500/10119
[INFO] processed 10000/10119
2020-08-15 23:10:53.121340: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
2020-08-15 23:10:53.132321: I tensorflow/core/common_runtime/process_util.cc:147] Creating new thread pool with default inter op setting: 2. Tune using
lism_threads for best performance.
compiling..... Please Wait...
Model: "sequential"
```

Fig 12 Loading the augmented images into the model with customized script. Source : ( Azeez Babatunde)



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	896
activation (Activation)	(None, 32, 32, 32)	0
batch_normalization (Batch Normalization)	(None, 32, 32, 32)	128
conv2d_1 (Conv2D)	(None, 32, 32, 32)	9248
activation_1 (Activation)	(None, 32, 32, 32)	0
batch_normalization_1 (Batch Normalization)	(None, 32, 32, 32)	128
max_pooling2d (MaxPooling2D)	(None, 16, 16, 32)	0
dropout (Dropout)	(None, 16, 16, 32)	0
conv2d_2 (Conv2D)	(None, 16, 16, 64)	18496
activation_2 (Activation)	(None, 16, 16, 64)	0
batch_normalization_2 (Batch Normalization)	(None, 16, 16, 64)	256
conv2d_3 (Conv2D)	(None, 16, 16, 64)	36928
activation_3 (Activation)	(None, 16, 16, 64)	0
batch_normalization_3 (Batch Normalization)	(None, 16, 16, 64)	256
conv2d_4 (Conv2D)	(None, 16, 16, 64)	36928
activation_4 (Activation)	(None, 16, 16, 64)	0
batch_normalization_4 (Batch Normalization)	(None, 16, 16, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 8, 8, 64)	0
dropout_1 (Dropout)	(None, 8, 8, 64)	0
flatten (Flatten)	(None, 4096)	0

Figure 13: Architecture for the 5-layer deep neural network. Source: (Azeez Babatunde)



### 3.5.5.2 Optimization

For the architecture to learn anything about the data, there is need to make use of an optimizer. An optimizer's function is to reduce the distance between the true value of a label and the predicted label.

This system made use of the SGD (Stochastic Gradient Descent) optimizer. This optimizer used the following arguments which were:

- Learning rate(lr) = 0.005
- Nesterov acceleration = True
- Momentum = True
- Decay\_rate = 1e-6

### 3.5.5.3 Compilation

The model is compiled using the categorical cross-entropy loss (this is used because this task has more than two classes to be predicted) and the optimizer function described earlier. The performance of the model is then judged using the “accuracy” metric.

### 3.5.5.4 Training and Classification

The researcher compiled the model and then initiate the training process. The model was configured to learn from the data in 100 times (number of epoch). The deep learning model is trained and it in turn creates a knowledge base. This knowledge based is what helps in classifying the data into the five different groups; normal, mild, moderate, severe, and proliferative DR. This

will in turn help during the detection when the system is making its own predictions on a real life or downloaded data.

### 3.5.6 Code Snippet for Model training and classification

The code snippet for classification of the system is as follows;

```
Anaconda Prompt (Anaconda3) - python RNET_ON_RETINO1.py --dataset Augmented_images --model demo100.hdf5
[CAUTION] Training about to occur, do not interrupt
Train on 8095 samples, validate on 2024 samples
Epoch 1/100
8095/8095 [=====] - 36s 5ms/sample - loss: 1.7140 - accuracy: 0.4476 - val_loss: 2.4166 - val_accuracy: 0.2890
Epoch 2/100
8095/8095 [=====] - 37s 5ms/sample - loss: 0.8091 - accuracy: 0.6566 - val_loss: 5.2959 - val_accuracy: 0.2890
Epoch 3/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.5469 - accuracy: 0.7643 - val_loss: 4.2367 - val_accuracy: 0.3162
Epoch 4/100
8095/8095 [=====] - 37s 5ms/sample - loss: 0.4488 - accuracy: 0.8114 - val_loss: 1.7624 - val_accuracy: 0.4921
Epoch 5/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.3927 - accuracy: 0.8332 - val_loss: 0.5605 - val_accuracy: 0.7984
Epoch 6/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.3521 - accuracy: 0.8537 - val_loss: 0.2789 - val_accuracy: 0.8780
Epoch 7/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.3323 - accuracy: 0.8582 - val_loss: 0.2596 - val_accuracy: 0.8745
Epoch 8/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.3041 - accuracy: 0.8703 - val_loss: 0.2351 - val_accuracy: 0.8953
Epoch 9/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2965 - accuracy: 0.8724 - val_loss: 0.2239 - val_accuracy: 0.9056
Epoch 10/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2900 - accuracy: 0.8730 - val_loss: 0.2180 - val_accuracy: 0.9046
Epoch 11/100
8095/8095 [=====] - 37s 5ms/sample - loss: 0.2857 - accuracy: 0.8770 - val_loss: 0.2174 - val_accuracy: 0.9027
Epoch 12/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2739 - accuracy: 0.8854 - val_loss: 0.2342 - val_accuracy: 0.8982
Epoch 13/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2706 - accuracy: 0.8845 - val_loss: 0.2400 - val_accuracy: 0.8809
Epoch 14/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2684 - accuracy: 0.8828 - val_loss: 0.2126 - val_accuracy: 0.9081
Epoch 15/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2696 - accuracy: 0.8836 - val_loss: 0.2129 - val_accuracy: 0.9101
Epoch 16/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2563 - accuracy: 0.8928 - val_loss: 0.2094 - val_accuracy: 0.9081
Epoch 17/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2521 - accuracy: 0.8915 - val_loss: 0.2065 - val_accuracy: 0.9101
Epoch 18/100
8095/8095 [=====] - 36s 4ms/sample - loss: 0.2496 - accuracy: 0.8893 - val_loss: 0.2519 - val_accuracy: 0.8824
```

```
1  #matplotlib.use("Agg")
2  import numpy as np
3  import argparse
4  import matplotlib
5  import matplotlib.pyplot as plt
6  # importing packages
7  from tensorflow.keras.applications import ResNet50
8  from tensorflow.keras.layers import Input
9  from sklearn.preprocessing import LabelBinarizer
10 from sklearn.metrics import classification_report
11 from refine.preprocessing import simplepreprocessor
12 from refine.preprocessing import imagetoarraypreprocessor
13 from refine.datasets import simpdatasetloader
14 from RNET import RNET
15 from tensorflow.keras.optimizers import SGD, Adam, Adamax
16 from sklearn.model_selection import train_test_split
17 from imutils import paths
18 import tensorflow as tf
19 from tensorflow.keras.layers import AveragePooling2D
20 from tensorflow.keras.applications import VGG16
21 from tensorflow.keras.layers import Dropout
22 from tensorflow.keras.layers import Flatten
23 from tensorflow.keras.layers import Dense
24 from tensorflow.keras.layers import Input
25 from tensorflow.keras.models import Model, Sequential
26 from tensorflow.keras.utils import plot_model
27 from tensorflow.keras.preprocessing.image import ImageDataGenerator
28
```

```

29 lr_init = 0.0001
30 ap = argparse.ArgumentParser()
31 #ap.add_argument("-o", "--output", required = True, help = "path to the output plot")
32 ap.add_argument("-d", "--dataset", required=True, help="path to input dataset")
33 ap.add_argument("-m", "--model", required=True,
34 help="path to output model")
35 args = vars(ap.parse_args())
36 classLabels = ["Mild", "Moderate", "No_DR", "Proliferate_DR", "Severe"]
37
38 print("Loading in image data")
39 imagePaths = list(paths.list_images(args["dataset"]))
40 #idxs = np.random.randint(0, len(imagePaths), size=(10,))
41 #imagePaths = imagePaths[idxs]
42
43 sp = simplepreprocessor.SimplePreprocessor(32, 32)
44 iap = imagetoarraypreprocessor.ImageToArrayPreprocessor()
45
46 sdl = simpledatasetloader.SimpleDatasetLoader([sp, iap])
47
48 (data, labels) = sdl.load(imagePaths, verbose = 500)
49 data = data.astype("float") / 255.0
50 trainX, testX, trainY, testY = train_test_split(data, labels, test_size = 0.20, random_state = 0)
51 trainY = LabelBinarizer().fit_transform(trainY)
52 testY = LabelBinarizer().fit_transform(testY)
53 #print(testX.shape)
54 opt = Adam(lr = lr_init, decay = lr_init/100)
55 model = RNET.build(width = 32, height = 32, depth = 3, classes = 5)
56 print("compiling..... Please Wait...")
57 model.compile(loss = "categorical_crossentropy", optimizer = opt, metrics = ["accuracy"])
58 #plot_model(model, to_file = "RNET.png")
59 print(model.summary())
60
61 print("[CAUTION] Training about to occur, do not interrupt")
62 H = model.fit(trainX, trainY, validation_data=(testX, testY), batch_size=64, epochs=100, verbose=1)
63 #model.fit_generator(trainGen, batch_size=32, epochs=100, verbose=1, shuffle = True)
64
65 print("Serializing Network")
66 model.save(args["model"])
67 # evaluating the network
68 predictions = model.predict(testX, batch_size=32)
69 print(classification_report(testY.argmax(axis=1),
70 predictions.argmax(axis=1), target_names=["Mild", "Moderate", "No_DR", "Proliferate_DR", "Severe"]))
71
72 # plot the training loss and accuracy
73 plt.style.use("ggplot")
74 plt.figure()
75 plt.plot(np.arange(0, 100), H.history["loss"], label="train_loss")
76 plt.plot(np.arange(0, 100), H.history["val_loss"], label="val_loss")
77 plt.plot(np.arange(0, 100), H.history["accuracy"], label="train_acc")
78 plt.plot(np.arange(0, 100), H.history["val_accuracy"], label="val_acc")
79 plt.title("Training Loss and Accuracy")
80 plt.xlabel("Epoch #")
81 plt.ylabel("Loss/Accuracy")
82 plt.legend()
83 plt.show()

```

Figure 14: Model Training and Classification. Source :( Azeez Babatunde)

### 3.6 Detection

The method of solving the classification problem as described in the sections 3.5 will help in solving the detection problem. A deep learning architecture was built and then the dataset was passed through it. The architecture has learnt the patterns available in the dataset, most especially patterns that makes up a Diabetic Retinopathy image and its variance. Hence the model was

serialized and used in the detection of DR when an image is passed through the model. In effectively detecting diabetic retinopathy, the detection process would pass through five basic stages before returning the detected result. The five major stages are image input, applying image pyramid, applying sliding window, taking the region of interest (ROI) and applying non-maxima suspension and after that the results are generated.

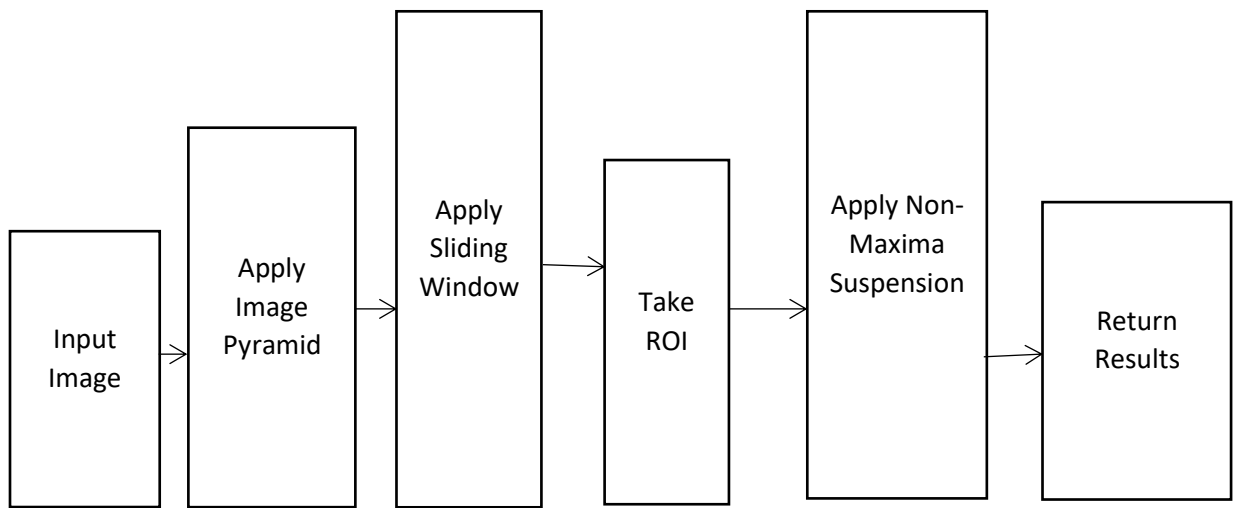


Figure 15: Stages for effective DR detection. Source: (Azeez Babatunde)

### 3.6.1 Input image

In this stage, the images that are to be observed or that the detection action is supposed to be performed on is loaded into the system and as much image as possible can be loaded into the system and the model would detect the different class of DR simultaneously. The data were pre-processed before they are passed into the network for classification. The model only considered pre-processed images for classification. Noise was then removed with Gaussian blur and then resized the images accordingly.

### 3.6.2 Image Pyramid

The image pyramid is a multiscale representation of any image developed by the image processing technique. The signal of an image is subjected to repeated smoothing and subsampling. The aim of using image pyramid is to help in improving the computing speed. This is because the smaller the filter, the faster the computing speed and the larger the filter, the slower the computing speed (Wang, 2016). The pyramid downsizes the images first as shown in figure 12 and uses a smaller filter on the downsized image. Downsizing the image makes the image smaller than the original image without taking away its features. The usage of image pyramid also avoids issues pertaining to memory overflow (Ullah & Petrosino , 2016). The step to using image pyramid begins from when the image is fed into the system as written earlier and then starts reducing the unimportant features thus giving it a pyramid structure. This will not just retain and improve accuracy but it will also reduce the number of parameters or features hence it uses less memory space which works perfectly well with lesser computational power.

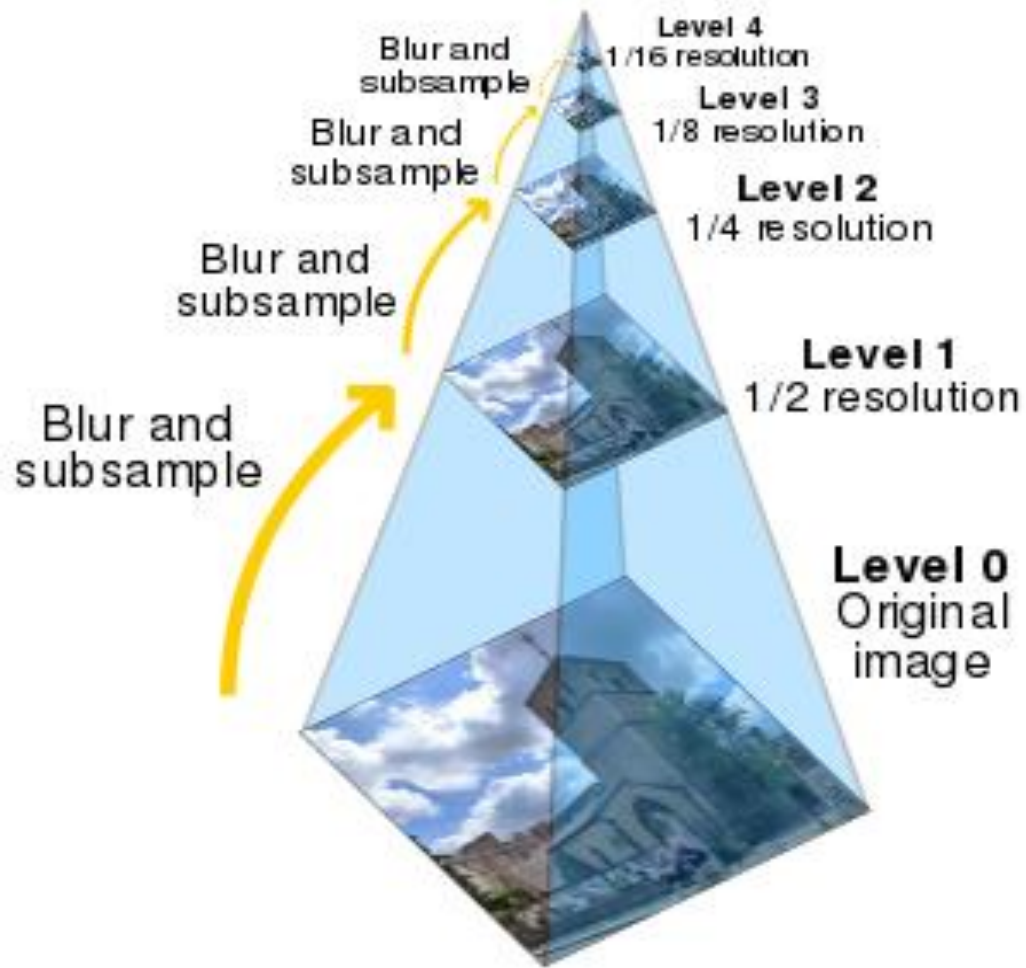


Figure 16: A sample image showing how image pyramid works. Source: (Wikipedia, the free encyclopedia, 2010)

Utilizing this technique allows us to find objects in the image at different scales or sizes. At the bottom of the pyramid, we have the original image at its original size (in terms of width and height). And at each subsequent layer, the image is resized (subsamped) and optionally smoothed via Gaussian blurring. The image is progressively subsampled until some stopping criterion is met, which is normally when a minimum size has been reached and no further subsampling needs to take place.

### 3.6.3 Sliding window

After this, an additional technique called sliding window is applied. This sliding window is a fixed rectangle of fixed size that slides from left to right and up to bottom within an image. In the sliding window method, a window of any size  $m \times n$  is used to perform a search or an operation on the target image (Hekwan & Ozsahin, 2017).

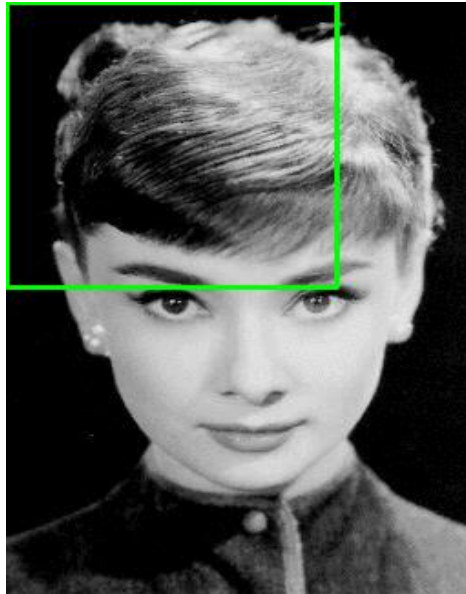


Figure 17: A sample image showing the application of image sliding on an image.

Source:(Rosebrock, 2014)

### 3.6.4 Taking region of interest

At each stop of this window (fixed rectangle), the Region of interest (ROI) is extracted and passed through the trained classifier and the output predictions are hereafter obtained.

### 3.6.5 Applying Non maxima Suppression

The final key step is non-maxima suppression, due to the application of the pre-trained model on several parts of the image (each window), the classifier can then predict and draw boxes on several part of the image as shown in figure 18.

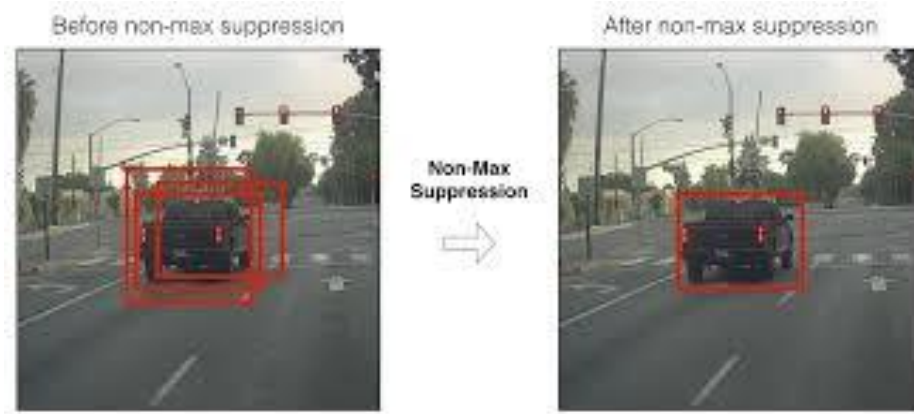


Figure 18: Comparison of an image before and after non maxima suppression is applied.

Source: (Sambasivarao, 2019))

The number of boxes is reduced to where the prediction confidence is the highest (right image), after applying non-max suppression.

### 3.6.6 Returning Results

After applying all these detection techniques, the model returned results and the images used for tests in the study after proper classifications and detection. The results of detection is shown in the figure below.



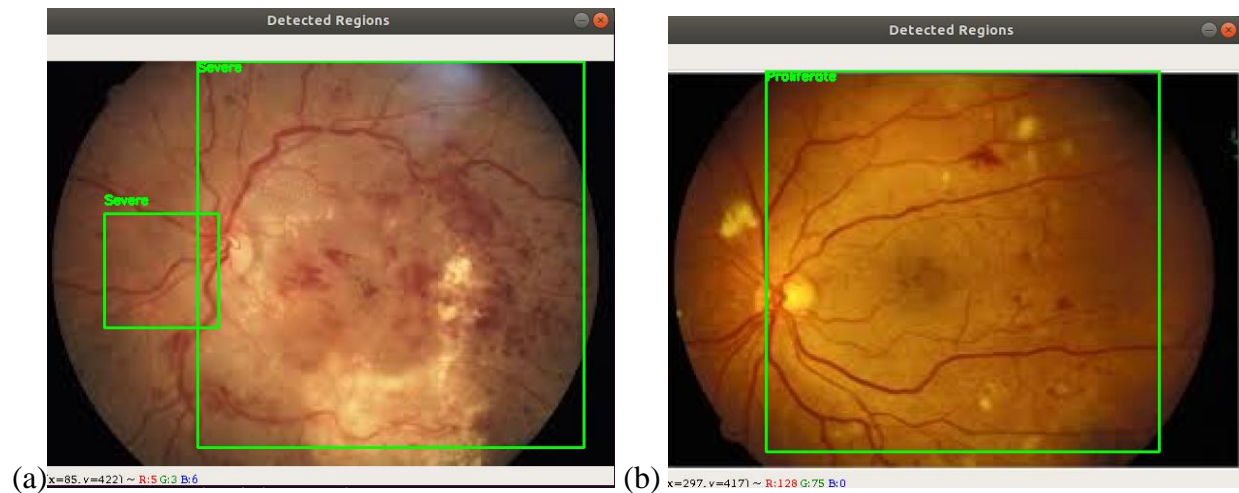


Figure 19: Images of detected DR, (a) Shows the detections of severe DR while (b) shows the detection of proliferate DR. Source: (Azeez Babatunde)

### 3.6.7 Code Snippet for detection

The code snippet for detecting the retinopathy is as follows;

```
Anacoda Prompt (Anaconda3) - python detector.py --image images/image_0_65.jpg --model RNET_final.hdf5 --size "(32,32)" --min-conf 0.90

(base) E:\Retinopathy\DR_NEW\DR-master>python detector.py --image images/image_0_65.jpg --model RNET_final.hdf5 --size "(32,32)" --min-conf 0.90
[INFO] loading network...
2020-08-22 16:02:36.665864: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX
AVX2
2020-08-22 16:02:36.671413: I tensorflow/core/common_runtime/process_util.cc:147] Creating new thread pool with default inter op setting: 2. Tune using inter_op_parallelism
threads for best performance.
Model: "sequential"

Layer (type)                Output Shape                Param #
-----
conv2d (Conv2D)              (None, 32, 32, 32)          896
activation (Activation)       (None, 32, 32, 32)          0
batch_normalization (BatchN (None, 32, 32, 32)          128
conv2d_1 (Conv2D)            (None, 32, 32, 32)          9248
activation_1 (Activation)     (None, 32, 32, 32)          0
batch_normalization_1 (Batch (None, 32, 32, 32)          128
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)          0
dropout (Dropout)            (None, 16, 16, 32)          0
conv2d_2 (Conv2D)            (None, 16, 16, 64)          18496
activation_2 (Activation)     (None, 16, 16, 64)          0
batch_normalization_2 (Batch (None, 16, 16, 64)          256
conv2d_3 (Conv2D)            (None, 16, 16, 64)          36928
activation_3 (Activation)     (None, 16, 16, 64)          0
batch_normalization_3 (Batch (None, 16, 16, 64)          256
```

```

conv2d_4 (Conv2D)          (None, 16, 16, 64)      36928
activation_4 (Activation)   (None, 16, 16, 64)      0
batch_normalization_4 (Batch (None, 16, 16, 64)      256
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 64)        0
dropout_1 (Dropout)         (None, 8, 8, 64)        0
flatten (Flatten)           (None, 4096)            0
dense (Dense)               (None, 512)            2097664
activation_5 (Activation)   (None, 512)            0
dropout_2 (Dropout)         (None, 512)            0
dense_1 (Dense)             (None, 5)              2565
activation_6 (Activation)   (None, 5)              0
=====
Total params: 2,203,749
Trainable params: 2,203,237
Non-trainable params: 512
None
[INFO] looping over pyramid/windows took 0.10396 seconds
[INFO] classifying ROIs...
[INFO] classifying ROIs took 2.38859 seconds
(1394, 5)
[INFO] showing results for 'Proliferate'
[INFO] showing results for 'Moderate'
[INFO] showing results for 'Mild'

```

```

1  # import the necessary packages
2  #from tensorflow.keras.applications import ResNet50
3  from tensorflow.keras.applications.resnet import preprocess_input
4  from tensorflow.keras.preprocessing.image import img_to_array
5  from detectors.detection_helpers import decode_predictions
6  from imutils.object_detection import non_max_suppression
7  from detectors.detection_helpers import sliding_window
8  from detectors.detection_helpers import image_pyramid
9  from tensorflow.keras.models import load_model
10 import tensorflow as tf
11 import numpy as np
12 import argparse
13 import imutils
14 import time
15 import cv2
16
17 # construct the argument parse and parse the arguments
18 ap = argparse.ArgumentParser()
19 ap.add_argument("-i", "--image", required=True,
20                 help="path to the input image")
21 ap.add_argument("-m", "--model", required = True,
22                 help = "path to pre-trained model")
23 ap.add_argument("-s", "--size", type=str, default="(200, 150)",
24                 help="ROI size (in pixels)")
25 ap.add_argument("-c", "--min-conf", type=float, default=0.9,
26                 help="minimum probability to filter weak detections")
27 ap.add_argument("-v", "--visualize", type=int, default=-1,
28                 help="whether or not to show extra visualizations for debugging")
29 args = vars(ap.parse_args())
30

```

```

31 # initialize variables used for the object detection procedure
32 WIDTH = 600
33 PYR_SCALE = 1.5
34 WIN_STEP = 16
35 ROI_SIZE = eval(args["size"])
36 INPUT_SIZE = (32,32)
37
38 # load our network weights from disk
39 print("[INFO] loading network...")
40 model = load_model(args["model"])
41
42 print(model.summary())
43 #model = tf.keras.Model(model.inputs, model.layers[-2].output)
44 # load the input image from disk, resize it such that it has the
45 # has the supplied width, and then grab its dimensions
46 orig = cv2.imread(args["image"])
47 orig = imutils.resize(orig, width=WIDTH)
48 (H, W) = orig.shape[:2]
49
50 # initialize the image pyramid
51 pyramid = image_pyramid(orig, scale=PYR_SCALE, minSize=ROI_SIZE)
52 # initialize two lists, one to hold the ROIs generated from the image
53 # pyramid and sliding window, and another list used to store the
54 # (x, y)-coordinates of where the ROI was in the original image
55 rois = []
56 locs = []
57 # time how long it takes to loop over the image pyramid layers and
58 # sliding window locations
59 start = time.time()
60

```

```

61 # loop over the image pyramid
62 for image in pyramid:
63     # determine the scale factor between the *original* image
64     # dimensions and the *current* layer of the pyramid
65     scale = W / float(image.shape[1])
66     # for each layer of the image pyramid, loop over the sliding
67     # window locations
68     for (x, y, roiOrig) in sliding_window(image, WIN_STEP, ROI_SIZE):
69         # scale the (x, y)-coordinates of the ROI with respect to the
70         # *original* image dimensions
71         x = int(x * scale)
72         y = int(y * scale)
73         w = int(ROI_SIZE[0] * scale)
74         h = int(ROI_SIZE[1] * scale)
75         # take the ROI and preprocess it so we can later classify
76         # the region using Keras/TensorFlow
77         roi = cv2.resize(roiOrig, INPUT_SIZE)
78         roi = img_to_array(roi)
79         roi = preprocess_input(roi)
80         # update our list of ROIs and associated coordinates
81         rois.append(roi)
82         locs.append((x, y, x + w, y + h))
83
84     # check to see if we are visualizing each of the sliding
85     # windows in the image pyramid
86     if args["visualize"] > 0:
87         # clone the original image and then draw a bounding box
88         # surrounding the current region
89         clone = orig.copy()
90         cv2.rectangle(clone, (x, y), (x + w, y + h),
91                        (0, 255, 0), 2)
92

```

```

92         # show the visualization and current ROI
93         cv2.imshow("Visualization", clone)
94         cv2.imshow("ROI", roiOrig)
95         cv2.waitKey(0)
96
97     # show how long it took to loop over the image pyramid layers and
98     # sliding window locations
99     end = time.time()
100     print("[INFO] looping over pyramid/windows took {:.5f} seconds".format(
101         end - start))
102     # convert the ROIs to a NumPy array
103     rois = np.array(rois, dtype="float32")
104     # classify each of the proposal ROIs and then show how
105     # long the classifications took
106     print("[INFO] classifying ROIs...")
107     start = time.time()
108     preds = model.predict(rois)
109     end = time.time()
110     print("[INFO] classifying ROIs took {:.5f} seconds".format(
111         end - start))
112     print(preds.shape)
113     # decode the predictions and initialize a dictionary which maps class
114     # labels (keys) to any ROIs associated with that label (values)
115     preds = decode_predictions(preds, top=1, class_list_path = "index.json")
116     labels = {}
117
118     # loop over the predictions
119     for (i, p) in enumerate(preds):
120         # grab the prediction information for the current ROI
121         (imagenetID, label, prob) = p[0]
122         # filter out weak detections by ensuring the predicted probability
123         # is greater than the minimum probability
124         if prob >= args["min_conf"]:
125             # grab the bounding box associated with the prediction and
126             # convert the coordinates
127             box = locs[i]
128             # grab the list of predictions for the label and add the
129             # bounding box and probability to the list
130             L = labels.get(label, [])
131             L.append((box, prob))
132             labels[label] = L
133
134     # loop over the labels for each of detected objects in the image
135     for label in labels.keys():
136         # clone the original image so that we can draw on it
137         print("[INFO] showing results for '{}'.format(label))
138         clone = orig.copy()
139         # loop over all bounding boxes for the current label
140         for (box, prob) in labels[label]:
141             # draw the bounding box on the image
142             (startX, startY, endX, endY) = box
143             cv2.rectangle(clone, (startX, startY), (endX, endY),
144                 (0, 255, 0), 2)

```

```

145     # show the results *before* applying non-maxima suppression, then
146     # clone the image again so we can display the results *after*
147     # applying non-maxima suppression
148     # cv2.imshow("Before", clone)
149     clone = orig.copy()
150
151     # extract the bounding boxes and associated prediction
152     # probabilities, then apply non-maxima suppression
153     boxes = np.array([p[0] for p in labels[label]])
154     proba = np.array([p[1] for p in labels[label]])
155     boxes = non_max_suppression(boxes, proba)
156     # loop over all bounding boxes that were kept after applying
157     # non-maxima suppression
158     for (startX, startY, endX, endY) in boxes:
159         # draw the bounding box and label on the image
160         cv2.rectangle(clone, (startX, startY), (endX, endY),
161             (0, 255, 0), 2)
162         y = startY - 10 if startY - 10 > 10 else startY + 10
163         cv2.putText(clone, label, (startX, y),
164             cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 255, 0), 2)
165     # show the output after apply non-maxima suppression
166     cv2.imshow("Detected Regions", clone)
167     cv2.waitKey(0)

```

Figure 20: Code snippet for detecting retinopathy. Source: (Azeez Babatunde)

### 3.7 Segmentation

According to (Pandian, 2017), one of the major reasons why segmentation is done is for the identification of the expansive estimated associated frontal area district in an image i.e to recognize the damaged features of the retina. (Kesar et al., 2018) Split segmentation into the manual and computerized segmentation. These segmentation types each have their own peculiarities with the computerized method as an upgrade of the manual segmentation. One of the drawbacks of the manual segmentation which birthed the computerized segmentation is that blood vessels are small in sizes and needs to be segmented accurately. (Panayides et al., 2016) Explained how segmentation are being done and discussed that the features from the images are broken down and the fundamental bits of those images as well as its useless parts are isolated from the images thereby ensuring that the crucial parts of the images are addressed with a white area while the meaningless parts are portrayed with a dull area. After the picture is isolated the features are emptied once more. These features are considered against the readiness set features remembering

the ultimate objective to recognize the damage features. After these areas of the images has been separated, their features are emptied once more. These features are checked against the already set features with the aim of recognizing the presence of any discrepancy. (Kesar et al., 2018). (Vaishnavi et al., 2016) Further discussed segmentation in the light of CNN and described it as a phase in CNN which segments the abnormal from the normal regions with the aid of several segmentation methods. After segmentation is concluded, the images that were segmented are then classified using some classifiers like the naïve Bayes, Gaussian method, and Random Forest (Vaishnavi et al., 2016). This study made use of the Gaussian mixture model for segmentation.

For this research work, five region-based features and a Gaussian mixture model classifier was used to classify the images. With this segmentation method, the images were segmented into the bright regions which are the optic disc region and the non-bright or non-optic disk regions. The dataset used for the segmentation which contains the real images and mask equivalent was obtained from <https://ieee-dataport.org/open-access/indian-diabetic-retinopathy-image-dataset-idrid>. At first, major blood vessels are detected by thresholding and later on, the blood vessels origin was detected at the centroid of the vessels.

Figure 21 shows the original image that were used in the segmentation process while figure 22 shows the masks.

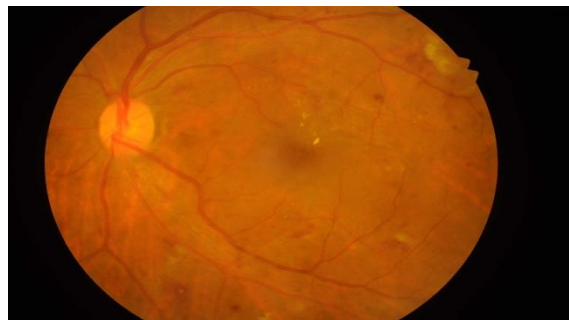


Figure 21: Original image used for the segmentation process. Source: (Prasanna Porwal, 2018)

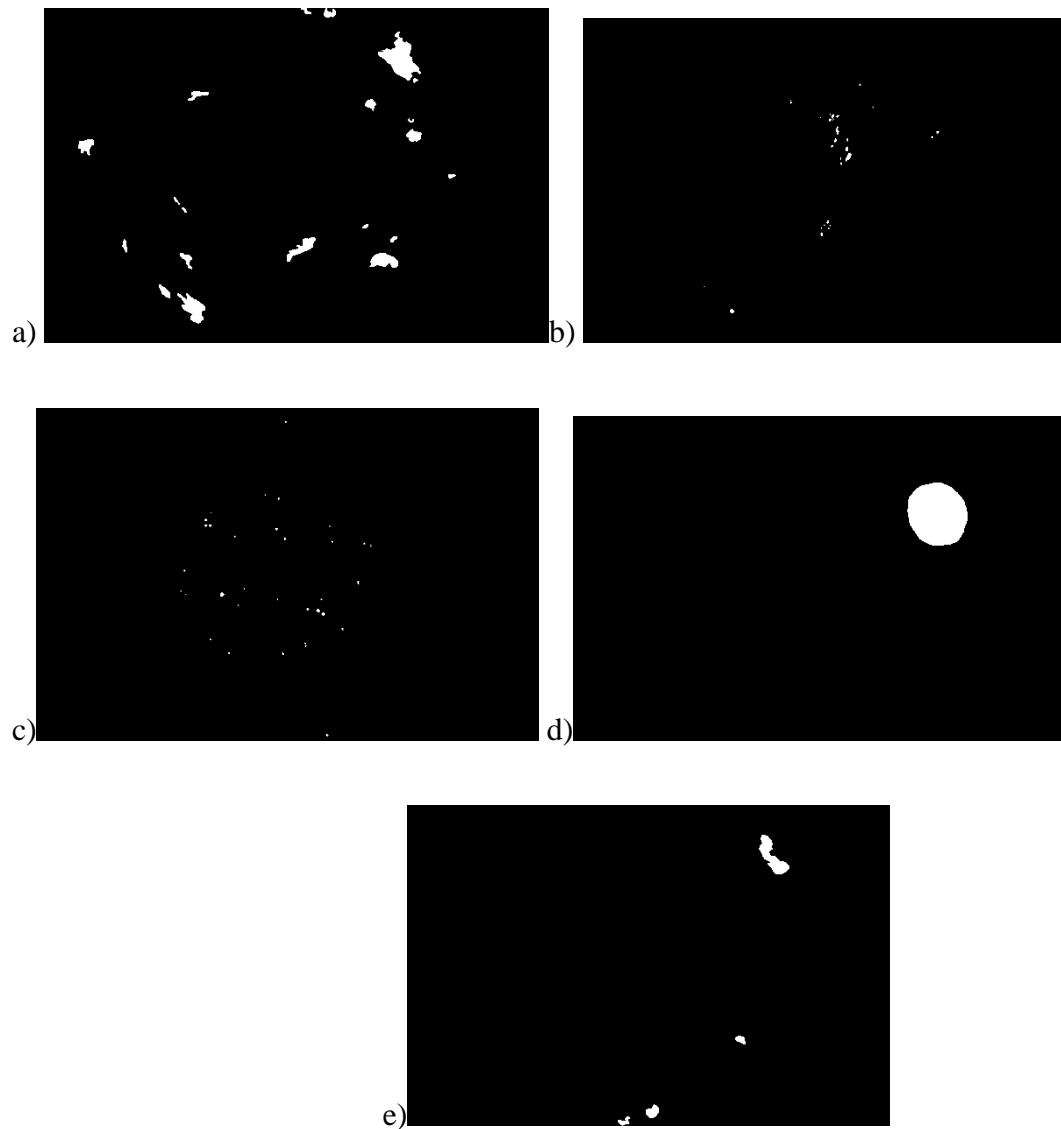


Figure 22: Images showing the masks; where **a** is showing the Hemorrhages, **b** shows the hard exudates, **c** shows the microaneurysms, **d** shows the optic disk while **e** shows the soft exudates.

Source: (Azeez Babatunde)

### 3.7.1 Segmentation method

For the segmentation done in this research work, the technique used is called GrabCut. The inbuilt opencv algorithm was used to perform this technique.

The GrabCut works by;

- Accepting an input image with a mask that approximated the segmentation
- Iteratively performing the following steps:
  1. Estimating the color distribution of the foreground and background via a Gaussian Mixture Model (GMM)
  2. Constructing a Markov random field over the pixel's labels (i.e., foreground vs. background)
  3. Applying a graph cut optimization to arrive at the final segmentation

The results from each of the masks after segmentation are shown in figure 22 – 27

### Hemorrhages

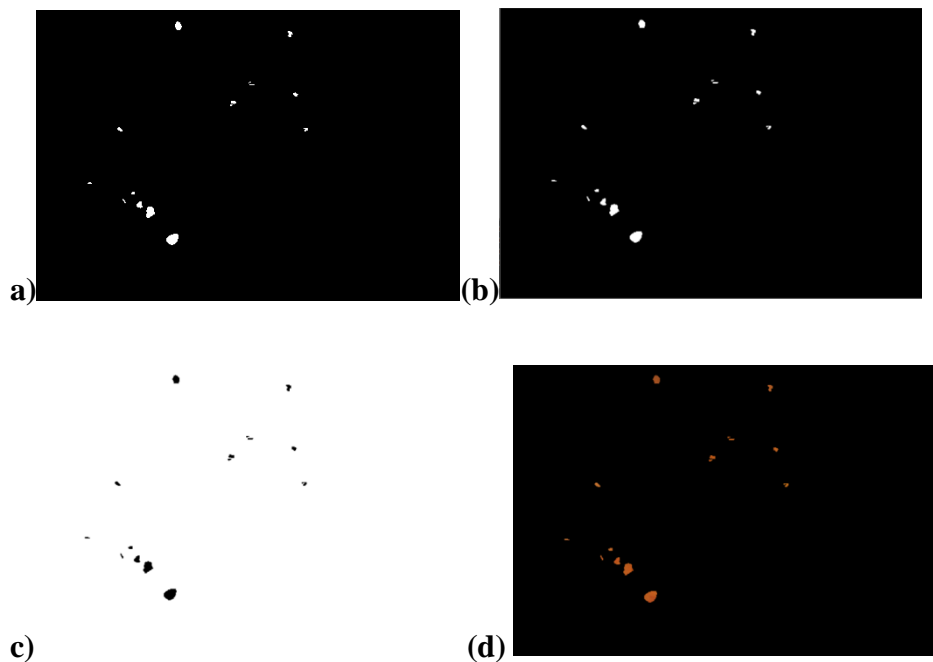


Figure 23: Results from Hemorrhages' segmentation. Source: (Azeez Babatunde)



**a** shows the mask, **b** shows the probable foreground, **c** shows the probable background while **d** shows the segmentation output for hemorrhages

### Optic Disc

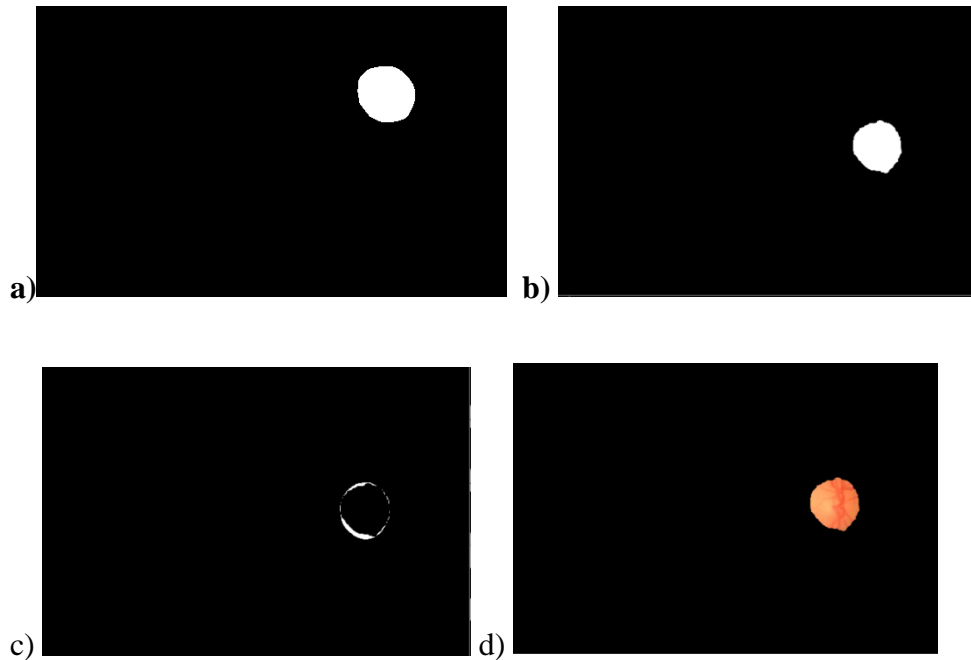
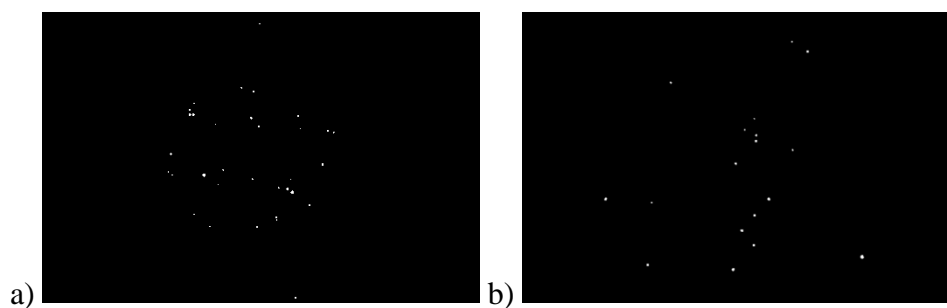


Figure: Results from Optic disks' segmentation; **a** shows the mask, **b** shows the probable foreground, **c** shows the probable background while **d** shows the segmentation output for optic disks. Source: (Azeez Babatunde)

### Microaneurysms



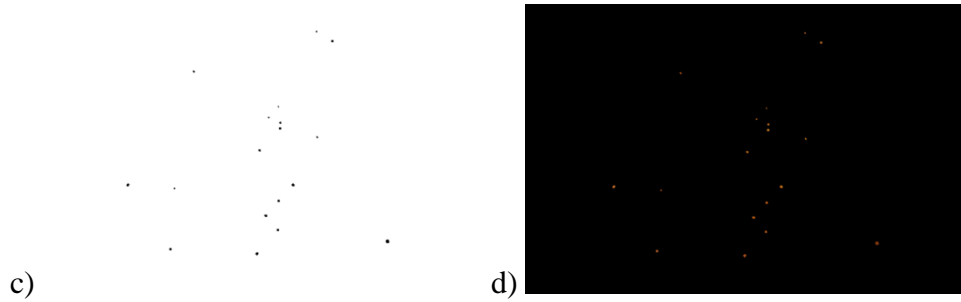


Figure 25: Results from Micro aneurysms' segmentation; **a** shows the mask, **b** shows the probable foreground, **c** shows the probable background while **d** shows the segmentation output for micro aneurysms. Source: (Azeez Babatunde)

### Soft Exudates

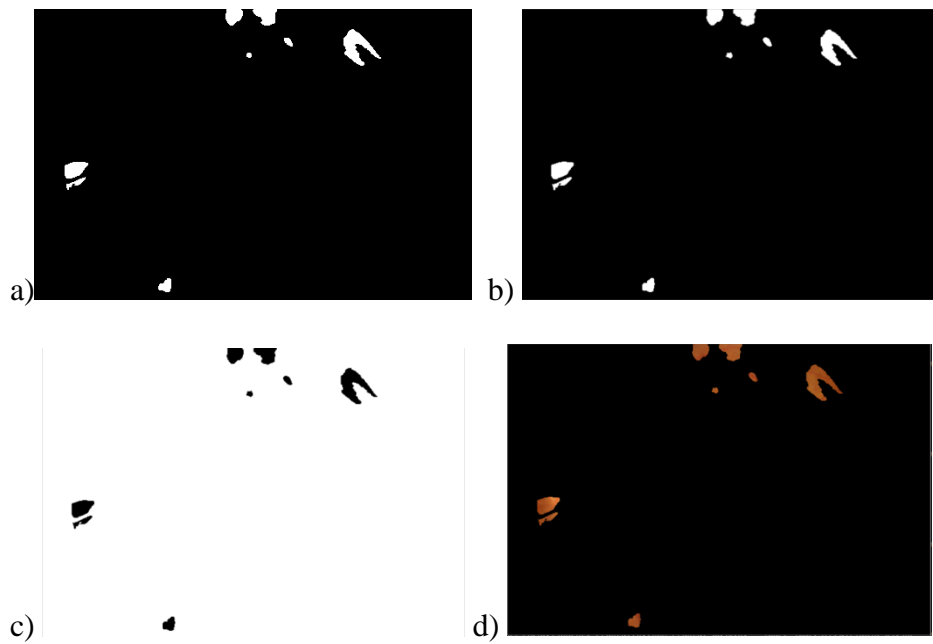


Figure 26: Results from Soft exudates' segmentation; **a** shows the mask, **b** shows the probable foreground, **c** shows the probable background while **d** shows the segmentation output for soft exudates. Source: (Azeez Babatunde)

### Hard Exudates

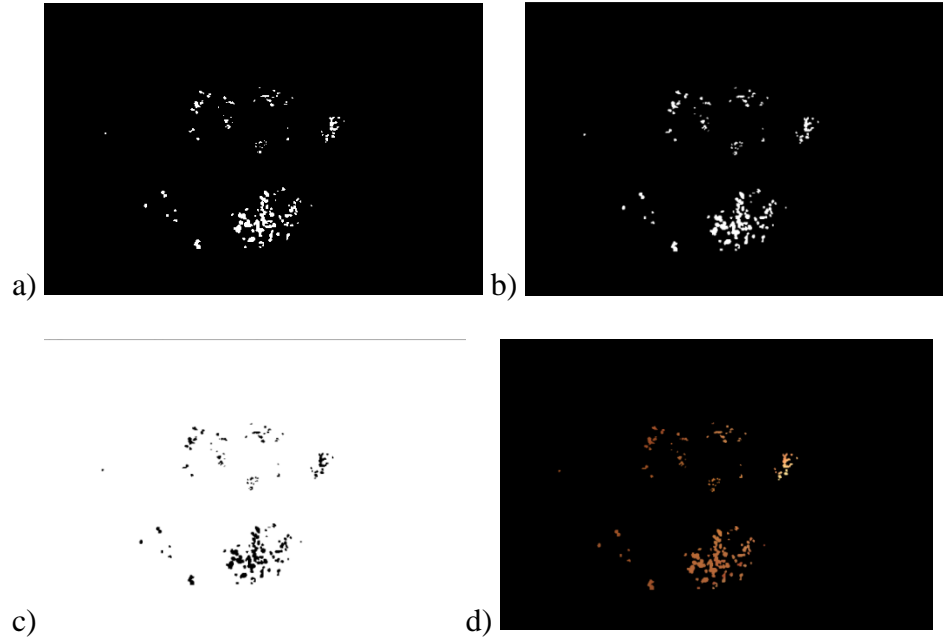


Figure 27: Results from Hard exudates' segmentation; **a** shows the mask, **b** shows the probable foreground, **c** shows the probable background while **d** shows the segmentation output for hard exudates. Source: (Azeez Babatunde)

### 3.7.2 Code Snippet for Segmentation

The code snippet for the segmentation process is as follows;

```
(base) E:\Retinopathy\DR_NEW\DR-master\segmentation>python segment.py --image images/real/IDRI01.jpg --mask images/mask/haemorrhages/IDRI01_HE.tif
[INFO] applying GrabCut took 1.33 seconds
[INFO] showing mask for 'Definite Background'
[INFO] showing mask for 'Probable Background'
[INFO] showing mask for 'Definite Foreground'
[INFO] showing mask for 'Probable Foreground'
```

```

1  # python grabcut_mask.py
2  # import the necessary packages
3  import numpy as np
4  import argparse
5  import time
6  import cv2
7  import os
8
9  # construct the argument parser and parse the arguments
10 ap = argparse.ArgumentParser()
11 ap.add_argument("-i", "--image", type=str, help="path to input image that we'll apply GrabCut to")
12 ap.add_argument("-mask", "--mask", type=str, help="path to input mask")
13 ap.add_argument("-c", "--iter", type=int, default=10,
14               help="# of GrabCut iterations (larger value => slower runtime)")
15 args = vars(ap.parse_args())
16
17 height = 600
18 width = 412
19 # load the input image and associated mask from disk
20 img = cv2.imread(args["image"])
21 image = cv2.resize(img, (height, width))
22
23 #mask_resize = cv2.resize(args["mask2"])
24 mask = cv2.imread(args["mask"], cv2.IMREAD_GRAYSCALE)
25 mask = cv2.resize(mask, (height, width))
26 # apply a bitwise mask to show what the rough, approximate mask would
27 # give us
28 roughOutput = cv2.bitwise_and(image, image, mask=mask)
29
30 # show the rough, approximated output
31 cv2.imshow("Rough Output", roughOutput)
32 cv2.waitKey(0)
33

```

```

34 # any mask values greater than zero should be set to probable
35 # foreground
36 mask[mask > 0] = cv2.GC_PR_FGD
37 mask[mask == 0] = cv2.GC_BGD
38
39 # allocate memory for two arrays that the GrabCut algorithm internally
40 # uses when segmenting the foreground from the background
41 fgModel = np.zeros((1, 65), dtype="float")
42 bgModel = np.zeros((1, 65), dtype="float")
43
44 # apply GrabCut using the the mask segmentation method
45 start = time.time()
46 (mask, bgModel, fgModel) = cv2.grabCut(image, mask, None, bgModel,
47   fgModel, iterCount=args["iter"], mode=cv2.GC_INIT_WITH_MASK)
48 end = time.time()
49 print("[INFO] applying GrabCut took {:.2f} seconds".format(end - start))
50
51 # the output mask has for possible output values, marking each pixel
52 # in the mask as (1) definite background, (2) definite foreground,
53 # (3) probable background, and (4) probable foreground
54 values = (
55     ("Definite Background", cv2.GC_BGD),
56     ("Probable Background", cv2.GC_PR_BGD),
57     ("Definite Foreground", cv2.GC_FGD),
58     ("Probable Foreground", cv2.GC_PR_FGD),
59 )
60
61 # loop over the possible GrabCut mask values
62 for (name, value) in values:
63     # construct a mask that for the current value
64     print("[INFO] showing mask for '{}'".format(name))
65     valueMask = (mask == value).astype("uint8") * 255
66

```

```

67         # display the mask so we can visualize it
68         cv2.imshow(name, valueMask)
69         cv2.waitKey(0)
70
71     # set all definite background and probable background pixels to 0
72     # while definite foreground and probable foreground pixels are set
73     # to 1, then scale the mask from the range [0, 1] to [0, 255]
74     outputMask = np.where((mask == cv2.GC_BGD) | (mask == cv2.GC_PR_BGD),
75         0, 1)
76     outputMask = (outputMask * 255).astype("uint8")
77
78     # apply a bitwise AND to the image using our mask generated by
79     # GrabCut to generate our final output image
80     output = cv2.bitwise_and(image, image, mask=outputMask)
81
82     # show the input image followed by the mask and output generated by
83     # GrabCut and bitwise masking
84     cv2.imshow("Input", image)
85     cv2.imshow("GrabCut Mask", outputMask)
86     cv2.imshow("GrabCut Output", output)
87     cv2.waitKey(0)

```

Figure 28: Code snippet for segmenting retinopathy. Source: (Azeez Babatunde)

## CHAPTER FOUR: RESULTS AND EVALUATION

### 4.0 Results and Evaluation

The experimental setup of the developed model is presented describing the dataset and the evaluation metrics used.

#### 4.1 Experimental Setup

Experiments were conducted at different stages to verify the system performance and to determine how useful and precise the recommendations were. The experiments were conducted on the IDRID dataset (<https://ieee-dataport.org/open-access/indian-diabetic-retinopathy-image-dataset-idrid>).

As with some other machine learning algorithms, this system suffered some losses while training. The loss experienced while training this network was quite minimal and it proves to show that the classifier did a great job. The job of the loss is usually to measure how good the model performed.

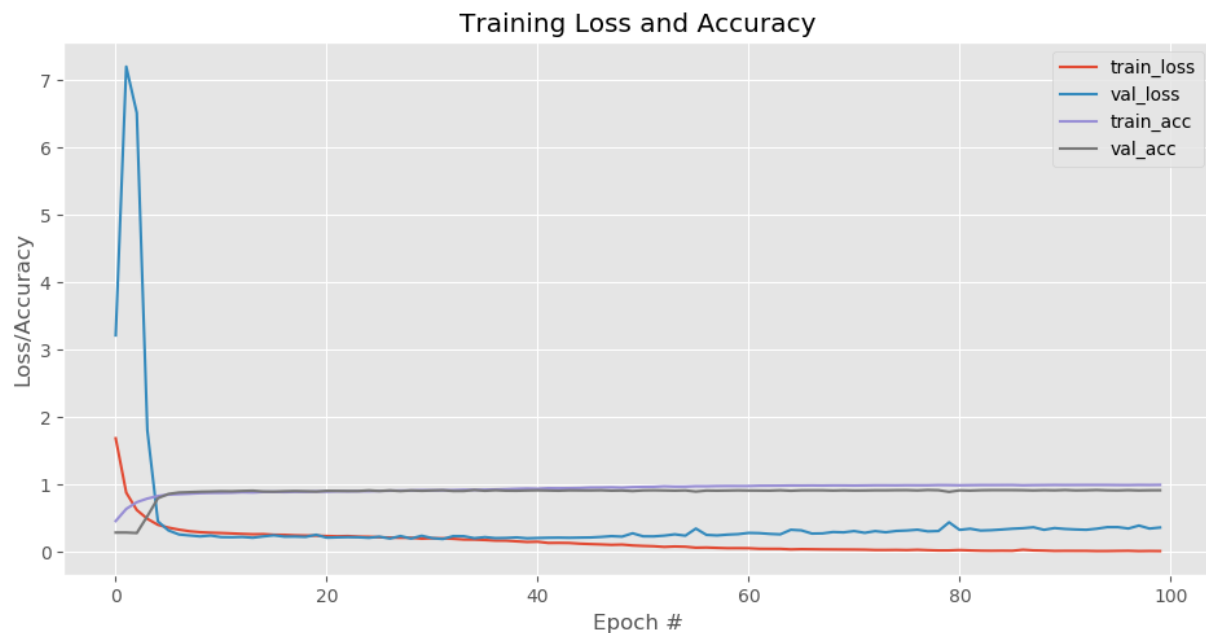


Figure : Plot of the Training loss and accuracy graph. Source: (Azeez Babatune)

#### 4.1.1 Evaluation Metrics

The evaluation of the system was carried out using the following metrics, ranked retrieval metric i.e. Accuracy, Precision, Recall and F1-Score. The report from what the system gave in comparison to the actual answer was used to evaluate the system. These evaluation measures are defined as follows:

$$Accuracy = \frac{A + D}{A + B + C + D}$$

$$Precision = \frac{A}{A + C}$$

$$Recall = \frac{A}{A + B}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Where:

*A* is true positive which is the number of DR that was correctly detected by the system

*B* is true negative which is the number of DR that was correctly detected but not by the system

*C* is false positive which is the number of DR that was not correctly detected by the system

*D* is false negative which is the number of DR that was not correctly detected and not by the system

#### 4.1.1 Evaluation of the System in terms of Accuracy, Precision, Recall and F1-Score

From the images fed into the system and with the preconceived knowledge before feeding these images into the system (i.e. the images with each DR types have been known beforehand, this is what informed our opinion on how accurate it was in making predictions), these were used to measure the accuracy, precision, recall and F1-Score of the system. Figure 23 shows the results of the average Accuracy, Precision, Recall and F1-Score of the system with how it reacted when it

was tested with images of varying degrees of diabetic retinopathy. This shows that the system performs efficiently

	precision	recall	f1-score	support
Mild	0.99	0.99	0.99	283
Moderate	0.85	0.80	0.82	315
No_DR	0.99	0.98	0.99	328
Proliferate_DR	0.94	0.90	0.92	513
Severe	0.85	0.92	0.89	585
accuracy			0.92	2024
macro avg	0.93	0.92	0.92	2024
weighted avg	0.92	0.92	0.92	2024

Figure 30: Image showing the result of the model (Deep CNN) Source :( Azeez Babatunde)

## 4.2 Comparative analysis with other deep learning algorithms

The evaluation result of the proposed system was compared with existing deep learning methodologies which were discussed in section 3.4.

### 4.2.1 KNN

KNN in itself doesn't really learn but uses nearest neighbor comparison making the nearest neighbor its class. These neighbors are what KNN uses in making predictions as discussed in section 3.4.2. The number of neighbors used in this is 5. The result of the classifier when KNN was used is shown in figure 31



```
(base) E:\Retinopathy\DR_NEW\DR-master>python classical_algo.py --dataset Augmented_images --model "knn"
loading images...
extracting image features...
      precision    recall  f1-score   support

     0       0.96       0.97       0.97        399
     1       0.66       0.75       0.70        389
     2       0.95       0.93       0.94        428
     3       0.87       0.87       0.87        625
     4       0.88       0.82       0.85        689

 accuracy         0.86         0.86         0.86       2530
 macro avg        0.86       0.87       0.87       2530
weighted avg        0.87       0.86       0.87       2530
```

Figure 31: Image showing the result of KNN model Source :( Azeez Babatunde)

#### 4.2.2 Logistic Regression (LR)

The LR was used in performing regularization. The variance (C) used in testing the classification of the logistics regression here was 0.01. The lower the value of C, the higher the correctness of the prediction. The value couldn't be reduced again when it got to 0.01 as further reductions didn't have any effect on the classification. The result of the classifier when LR was used is shown in figure 32

```
      precision    recall  f1-score   support

     0       0.49       0.53       0.51        399
     1       0.38       0.31       0.34        389
     2       0.55       0.77       0.64        428
     3       0.51       0.30       0.38        625
     4       0.57       0.68       0.62        689

 accuracy         0.52         0.52         0.52       2530
 macro avg        0.50       0.52       0.50       2530
weighted avg        0.51       0.52       0.50       2530
```

Figure 32: Image showing the result of Logistic Regression Model. Source :( Azeez Babatunde)

### 4.2.3 Support Vector Machine (SVM)

The SVM uses a linear algorithm like the Logistics Regression. The result of the classifier when is shown in figure 33

```
(base) E:\Retinopathy\DR_NEW\DR-master>python classical_algo.py --dataset Augmented_images --model "SVC"
loading images...
extracting image features...
```

	precision	recall	f1-score	support
0	0.62	0.87	0.73	399
1	0.45	0.62	0.52	389
2	0.75	0.91	0.82	428
3	0.73	0.36	0.48	625
4	0.78	0.69	0.73	689
accuracy			0.66	2530
macro avg	0.67	0.69	0.66	2530
weighted avg	0.69	0.66	0.65	2530

Figure 33: Image showing the result of the SVM Mode when different DR images are fed into it

## 4.3 Comparison of Results

After building and testing out each of the models considered in this research, the next thing was to compare the performance of each of the models to be able to decide which model performs best. Images from the selected dataset were fed into each of the models and individual results were generated.

Performance metrics such as precision, recall, f1-score were generated for each of the classes in the dataset while the accuracy of the entire model was also calculated. This section compares the result of each of the models, this is show in the table and figures below:

S/N	MODELS	CLASS	PRECISION	RECALL	F-1 SCORE	ACCURACY
1	CNN	MILD	0.99	0.99	0.99	0.92
		MODERATE	0.85	0.80	0.82	
		NO DR	0.99	0.98	0.99	
		PROLIFERATE	0.94	0.90	0.92	
		SEVERE	0.85	0.92	0.89	
2	KNN	MILD	0.96	0.97	0.97	0.86
		MODERATE	0.66	0.75	0.70	
		NO DR	0.95	0.93	0.94	
		PROLIFERATE	0.87	0.87	0.87	
		SEVERE	0.788	0.82	0.85	
3	LR	MILD	0.49	0.53	0.51	0.52
		MODERATE	0.38	0.31	0.34	
		NO DR	0.55	0.77	0.64	
		PROLIFERATE	0.51	0.30	0.38	
		SEVERE	0.57	0.68	0.62	
4	SVM	MILD	0.62	0.87	0.73	0.66
		MODERATE	0.45	0.62	0.52	
		NO DR	0.75	0.91	0.82	
		PROLIFERATE	0.73	0.36	0.48	
		SEVERE	0.78	0.69	0.73	

Table 1: Result of different Models

### 4.3.1 VISUALIZATION

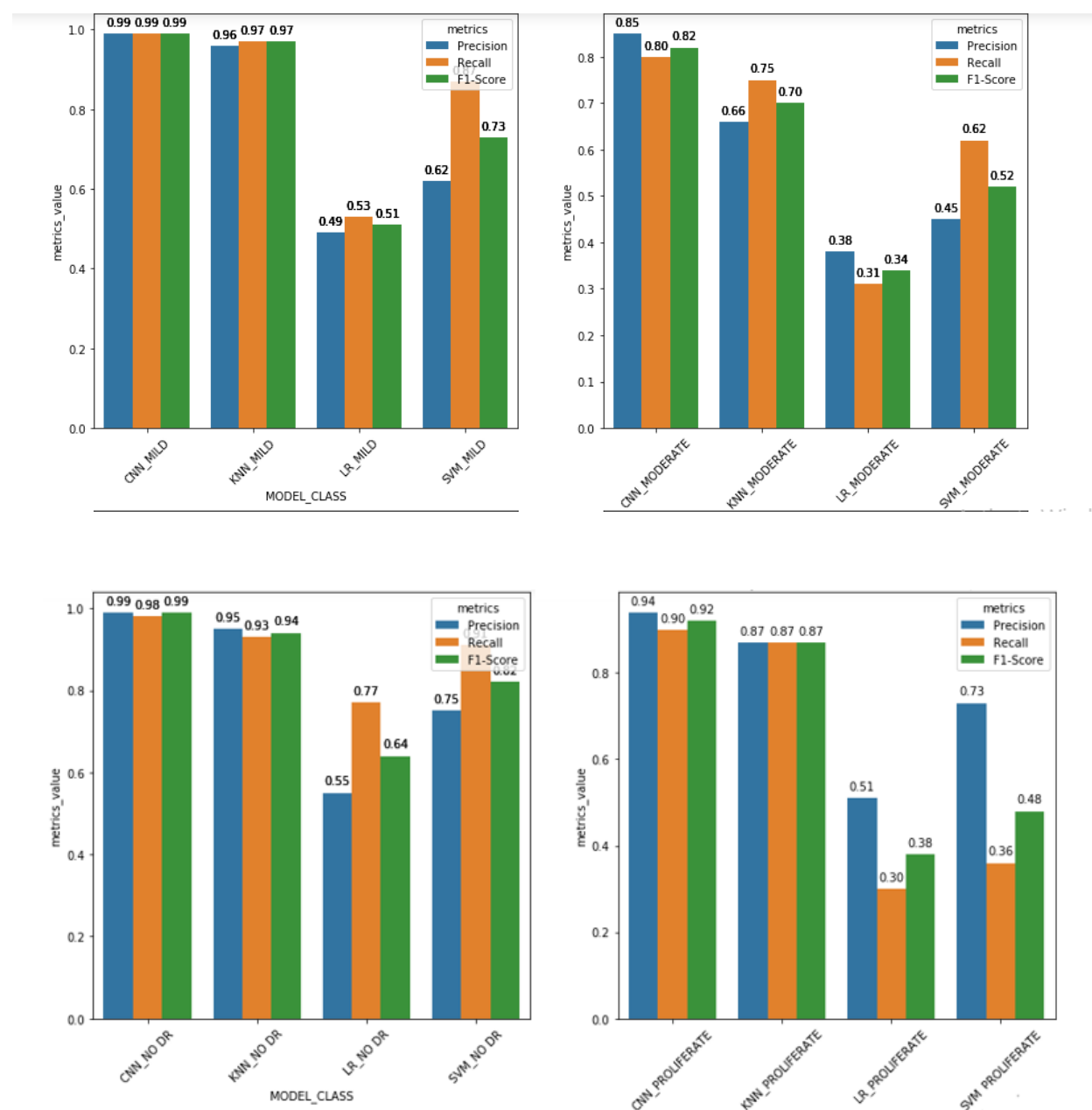


Figure 34: Comparison of the Performance Metrics across different classes of DR.  
Source:(Azeez Babatunde)

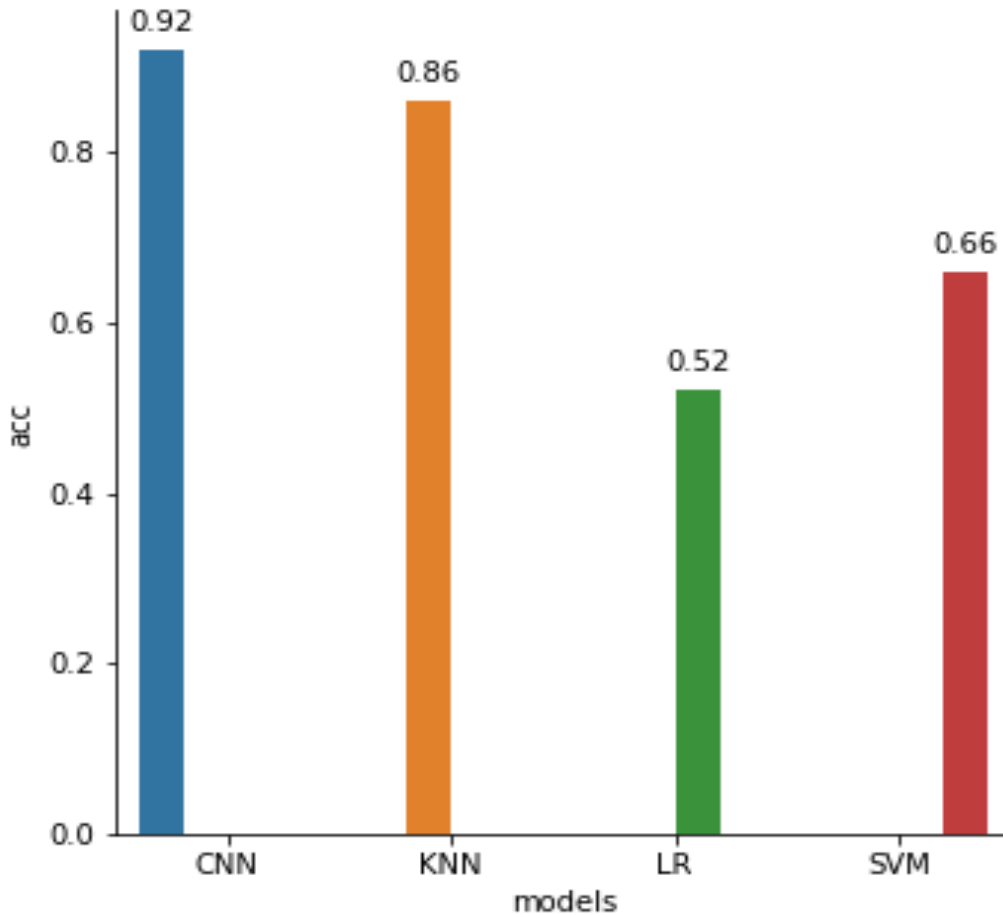


Figure: Comparison of Accuracy of the different Models, Source: (Azeez Babatunde)

#### 4.4 Comparative Analysis of the proposed system with existing works

As mentioned earlier in chapter two, this research work was proposed to ride on the scholarly researches that have performed similar studies and try to make headways and contribute to the knowledge environment, hence contributing to the growth of this research area in real life. The accuracy of this research work was therefore compared with other four scholars and summarily, it could be observed that this research work did well compared to that of those who have a similar research work to this current one. Except with the work of (Shankar et al.,

2020) that studied with only 214 images and 28.064 time taken to return the result as against the current research that modeled 10,119 images and 4hrs to return the result.

The evaluation result of the proposed system was compared with existing works in literature and this is shown in Table 4.2

**Table 4.2 Comparative Analysis of the accuracy of the proposed model vs existing works**

<b>Authors</b>	<b>Approach/Technique</b>	<b>Accuracy</b>
(Lam et al., 2018)	Deep Convolution Neural Network	66.8%
(Revathy et al., 2020)	Machine Learning	82.0%
(Shankar et al., 2020)	Synergic deep learning	99.3%
(Nikhil & Angel, 2019)	Artificial Neural Network	80.1%
Current Research (2020)	Deep Convolution Neural Network	92.5%

Table 2: Comparative analysis with existing Models

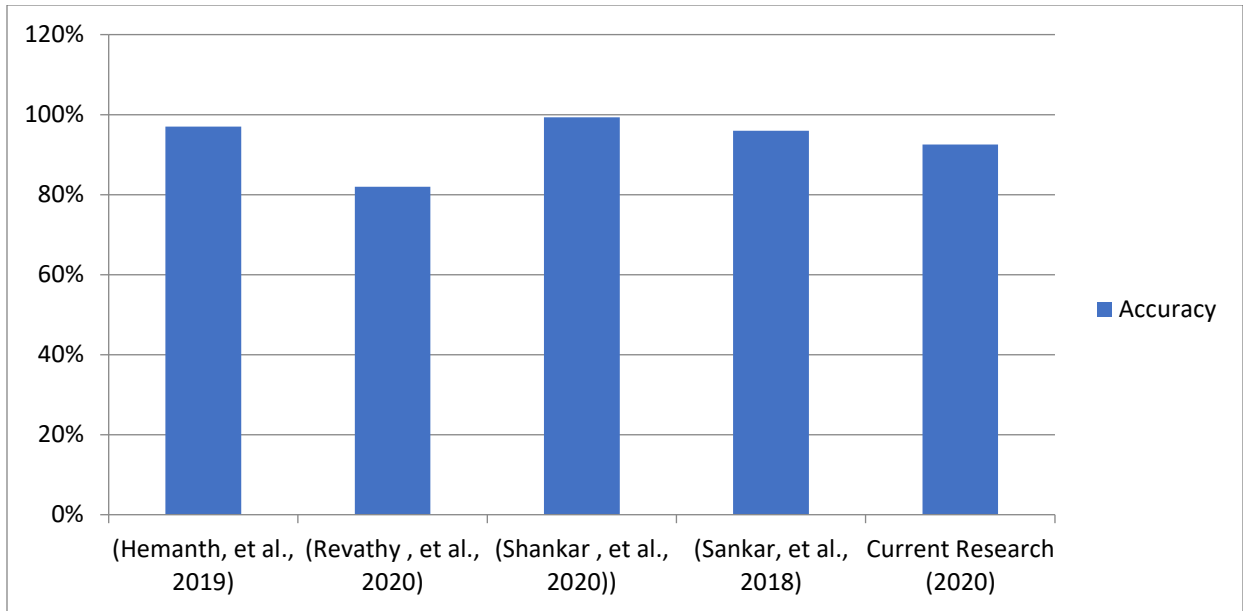


Figure: Comparative Analysis of the current research with existing works

The works of Nikhil & Angel, 2019 used three networks layers which are VGG 16, AlexNet and InceptionNet V3 while (Shankar et al., 2020) used the Messidor DR dataset as opposed to the IDRID dataset which this current system used, Shankar, et al., also used the AlexNet and InceptionNet. (Lam et al., 2018) used GoogleNet and AlexNet and got their dataset from ImageNet. (Revathy et.al., 2020) in their own research used Machine learning as opposed to deep learning used by other researchers as well as this current research. They infused various techniques like SVM, KNN and Random Forest classifier. In his research work, the DR detection was grouped into two which are normal and abnormal thus when images are fed into the system the system just classifies it as either normal (i.e no DR) or abnormal (i.e DR is present). This current research on the other hand made use of a customized RNet which comprises of 4 layers of Convolution network and a mini VGG network and got its images from the IDRID dataset. This was able to ensure this research work effectively developed a system which had an accuracy of about 92% edging the accuracies of the other research

works bar that of (Shankar et.al., 2020) hence giving this system further room for improvement while also giving pointers to the fact that this work did well compared to that of several scholars.



## **CHAPTER FIVE: CONCLUSION AND RECOMMENDATION**

### **5.1 Conclusion**

Convolutional neural network (CNN), K-nearest neighbors (KNN), Support vector machines (SVM) and Logistics Regression (LR) and other techniques are widely used in modern data and image evaluation, especially in health where it is use to classify, detect and segment diseases to mention a few and these have proven to be efficient and effective over time. Existing systems in this field has not really solved the issue completely and suffers from cold-start problems. Hence, the need for studying the past related researches so as to predict the future, convolution neural network (fuzzy logic) was adopted to help detect and classify the diabetic retinopathy (DR) in images, the model is intended also segment the DR in order to isolate the region of interest. The system uses RNet which is the name given to our 5-layer convolutional network. The RNet comprises of a mini VGG network which has 4 layers of convolutional network and a custom one-layer network making the architecture 5 layer and this system performed wonderfully well when it was tested on over 10,000 augmented images. The detection process was in 5 stages which were inputting the image, applying image pyramid, applying sliding window, taking the region of interest (ROI) and applying non-maxima suspension. At the end of these stages, the model was able to fully detect the different classes of DR, while segmentation was done as the data was split into five region-based features and a Gaussian mixture model classifier was used to classify the images and segment them.

Experimental test were conducted using a dataset of about 413 images which were augmented to 10,119 images, worth of note is the high computational power and process time required by the model in augmenting the images from the original number of images to the final experimental datasets, however the accuracy and the results showed that the system performs efficiently. The

accuracy, precision, recall and F1-score of mild DR, moderate DR, no DR, proliferate DR and severe DR gave an average result of 92%.

Therefore, this research result established a reliable deep convolution network model that is capable of solving the problem of classification and detection of diabetic retinopathy.

## **5.2 Research Limitations**

One of the major challenges encounter during the course of this research was the problem of computational power and time taken to preprocess the images, about 4hours is required to train the CNN Model. Another problem is the unavailability of primary retina fundus image datasets

## **5.3 Recommendation and Future works**

The deep learning system that was developed from this research has proven to be an efficient way of classifying and detecting diabetic retinopathy. Therefore, it is recommended for researchers especially those in the medical field to help patients with diabetic retinopathy before it gets to a stage when it can't be managed or cured because early detection of diabetic retinopathy would to a large extent help in maintaining it.

Future works could incorporate the use of mobile system technology which will make the job easier where the eyes are just scanned through the mobile device's camera. Also, more factors could be considered where the patients would be able to scan themselves from the comfort of their home without needing to visit the hospital, which would then send a prompt to the doctor for medical evaluation of patient who detects DR through the app hence improving the area of telemedicine. Evaluation of the system could be done on a larger dataset as well.

#### **5.4 Contribution to Knowledge**

This research has proposed a reliable and efficient deep learning algorithm that would classify and detect different classes of diabetic retinopathy thereby aiding the early detection of DR in patients so that adequate treatment can be administered to the patients by the medical personnel.

## References

- Akhila , T., Ambarish , A. & Unnikrishnan , K. S., 2019. Diabetic Retinopathy Detection Using Deep Neural Network. *International Journal of Computer Science and Mobile Computing*, 8(5), pp. 126-131.
- Aladawi , W., Jayakumari, C. & Vidhyalavanya , S. E. P., 2019. Recent Innovations in Automated Detection and Classification of Diabetic Retinopathy. *International Journal of Innovative Technology and Exploring Engineering*, 8(10), pp. 1997-2004.
- Altomare , F., Kherani , A. & Lovshin , J., 2018. Retinopathy. *Canadian Journal of Diabetes* , Volume 42, pp. S210-S216.
- American Diabetes Association, 2014. Diagnosis and Classification of Diabetes Mellitus. *Diabetes Care*, 37(1), pp. 581-590.
- Bethanney , J. et al., 2015. Detection and classification of exudates in retinal image using image processing techniques. *Journal of Chemical and Pharmaceutical Sciences*, 8(3), pp. 541-546.
- Boral, Y. & Thorat, S., 2020. A Review on Automated Diabetic Retinopathy Detection using Pretrained Deep Neural Network. *International Journal for Research in Applied Science & Engineering Technology*, 8(5), pp. 1568-1573.
- Carrera, E. V., Gonzalez, A. & Carrera, R., 2017. *Automated detection of diabetic retinopathy using SVM*, Ecuador: s.n.
- Chandrakumar, T. & Kathirvel, R., 2016. Classifying Diabetic Retinopathy using Deep Learning Architecture. *International Journal of Engineering Research & Technology (IJERT)*, 5(6), pp. 19-24.
- Cho, N. H., Shaw, J. E. & Karuranga, S., 2018. IDF Diabetes Atlas: global estimates of diabetes prevalence for 2017 and projections for 2045. *Diabetes Research and Clinical Practice*, Volume 138, pp. 271-281.
- Coşkun, M., Yildirim, Ö., Uçar , A. & Demir, A., 2017. An Overview Of Popular Deep Learning Methods. *European Journal of Technic*, 7(2), pp. 165-176.
- Dutta, S. et al., 2018. Classification of Diabetic Retinopathy Images by Using Deep Learning Models. *International Journal of Grid and Distributed Computing*, 11(1), pp. 89-106.
- Felman, A., 2017. *Diabetic retinopathy: Causes, symptoms, and treatments*. [Online] Available at: <https://www.medicalnewstoday.com/articles/183417> [Accessed 30 June 2020].
- Fisher , R., Perkins, S., Walker, A. & Wolfart, E., 2003. *Spatial Filters - Gaussian Smoothing*. [Online] Available at: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm> [Accessed 9 July 2020].
- Fong, D. et al., 2003. Diabetic Retinopathy. *Diabetes Care*, 26(1), pp. 99-103.
- Fu, X. et al., 2018. *Lightweight Pyramid Networks for Image Deraining*, China: s.n.
- Gandhi, A., 2018. *Data Augmentation | How to use Deep Learning when you have Limited Data — Part 2*. [Online]

Available at: <https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/>

[Accessed 9 July 2020].

Gowthaman , R., 2014. Automatic Identification And Classification Of Microaneurysms For Detection Of Diabetic Retinopathy. *International Journal of Research in Engineering and Technology*, 3(2), pp. 464-473.

Gupta, N. & Gupta, R., 2015. Diabetic Retinopathy – An Update. *JIMSA*, 28(1), pp. 54-58.

Hekwan, A. & Ozsahin, D. U., 2017. Sliding Window Based Machine Learning System for the Left Ventricle Localization in MR Cardiac Images. *Applied Computational Intelligence and Software Computing*, pp. 1-9.

Hemanth, J., Deperlioglu, O. & Kose, U., 2019. An enhanced diabetic retinopathy detection and classification approach using deep convolutional neural network. *Neural Computing and Applications*.

Ho, D., Liang, E. & Liaw, R., 2019. *1000x Faster Data Augmentation*. [Online]

Available at: [https://bair.berkeley.edu/blog/2019/06/07/data\\_aug/](https://bair.berkeley.edu/blog/2019/06/07/data_aug/)

[Accessed 9 July 2020].

Kesar, A., kaur, N. & Singh, P., 2018. Eye Diabetic Retinopathy by Using Deep Learning. *International Research Journal of Engineering and Technology*, 5(3), pp. 2504-2508.

Khalifa, N. E. M., Loey, M., Taha, M. H. N. & Mohamed, H. N. E. T., 2019. Deep Transfer Learning Models for Medical Diabetic Retinopathy Detection. *ACTA INFORM MED*, 27(5), pp. 327-332.

Kotsiliti, E. et al., 2017. A classification model for predicting diabeticretinopathy based on patient characteristics and biochemical measures. *Journal for Modeling in Ophthalmology*, 4(1), pp. 69-85.

Kunghatkar, A. A. & Panse , M. S., 2018. Detection of Lesions and Classification of Diabetic Retinopathy. *International Journal of Scientific Research in Science and Technology*, 4(8), pp. 703-707.

Labhade , J. D. & Chouthmol, L. K., 2016. Diabetic Retinopathy Detection using Random Forest. *International Journal of Modern Trends in Engineering and Research*, pp. 630-634.

Lal, S. B., 2016. Diabetes: Causes, Symptoms And Treatments. In: *Public Health Environment and Social Issues in India*. India: s.n., pp. 55-67.

Lam, C., Yi, D., Guo, M. & Lindsay, T., 2018. Automated Detection of Diabetic Retinopathy using Deep Learning. pp. 147-155.

Li, Y.-H., Yeh, N.-N., Chen , S.-J. & Chung, Y.-C., 2019. *Computer-Assisted Diagnosis for Diabetic Retinopathy Based on Fundus Images Using Deep Convolutional Neural Network*, s.l.: s.n.

Mankar, B. S. & Rout , N., 2016. Automatic Detection of Diabetic Retinopathy using Morphological Operation and Machine Learning. *International Journal of Engineering & Technology* , 3(5), pp. 12-19.

Mehta, S. & Dhillon, A. S., 2019. Classification Approach for Diabetic Retinopathy Detection. *International Journal of Computer Science and Mobile Computing* , 8(7), p. 129 – 136 .

- Ministry of Public Health and Sanitation, Kenya, 2013. *Community Diabetes Prevention and Management*. Nairobi, Kenya: Division of Non-communicable Diseases .
- Naithani, S., Bharadwaj, S. & Kumar, D., 2019. Automated Detection of Diabetic Retinopathy using Deep Learning. *International Research Journal of Engineering and Technology*, 6(4), pp. 2945-2947.
- National Institute of Diabetes and Digestive and Kidney Diseases, 2013. *Your Guide to Diabetes: Type 1 and Type 2*. Bethesda: National Diabetes Information Clearinghouse.
- National Institute of Diabetes and Digestive and Kidney Diseases, 2016. *What is Diabetes?*. [Online] Available at: <https://www.niddk.nih.gov/health-information/diabetes/overview/what-is-diabetes> [Accessed 30 June 2020].
- Nikhil, M. N. & Angel, R., 2019. Diabetic Retinopathy Stage Classification using CNN. *International Research Journal of Engineering and Technology*, 6(5), pp. 5969-5974.
- O'Mahony, N. et al., 2019. *Deep Learning vs. Traditional Computer Vision*, Tralee, Ireland : s.n.
- Panayides, A. S., Pattichis, C. S. & Pattichis, M. S., 2016. The Promise of Big Data Technologies and Challenges for Image and Video Analytics in Healthcare. p. 1278–1282.
- Pandian, S., 2017. Diabetic Retinopathy Detection using Image Processing. *International Journal for Scientific Research & Development*, 5(9), pp. 533-534.
- Paranjpe, J. M. & Kakatkar, M. N., 2014. Review Of Methods For Diabetic Retinopathy Detection And Severity Classification. *International Journal of Research in Engineering and Technology*, 3(3), pp. 619-624.
- Porwal, P., 2018. *indian diabetic retinopathy images*. INDIA: s.n.
- Prasanna Porwal, S., 2018. Indian Diabetic Retinopathy Image Dataset.
- Rahimloo, P. & Jafarian, A., 2016. Prediction of Diabetes by Using Artificial Neural Network, Logistic Regression Statistical Model and Combination of Them. *Bulletin de la Société Royale des Sciences de Liège*, 85(1), pp. 1148 - 1164.
- Raman, R. et al., 2018. Fundus photograph-based deep learning algorithms in detecting diabetic retinopathy. *Eye*, 33(1), pp. 97-109.
- Ramya, V., 2018. SVM Based Detection for Diabetic Retinopathy. *International Journal of Research and Scientific Innovation (IJRSI)*, 5(1), pp. 11-13.
- Revathy , R. et al., 2020. Diabetic Retinopathy Detection using Machine Learning. *International Journal of Engineering Research & Technology (IJERT)*, 9(6), pp. 122-126.
- Sahlsten, J. et al., 2019. *Deep Learning Fundus Image Analysis for Diabetic Retinopathy and Macular Edema Grading*, s.l.: s.n.
- Sakib, S., Ahmed, N., Kabir, A. J. & Ahmed, H., 2018. *An Overview of Convolutional Neural Network: Its Architecture and Applications*, Bangladesh : Preprints.

Sambasivarao, K., 2019. *Non Maxima Suppression*. [Online]

Available at: [towardsdatascience.com/non-maxima-suppression-nms-93ce178e177c](https://towardsdatascience.com/non-maxima-suppression-nms-93ce178e177c)

[Accessed 19 07 2020].

Sankar, U., Vijai, R. & Balajee, R. M., 2018. Detection and Classification of Diabetic Retinopathy in Fundus Images using Neural Network. *International Research Journal of Engineering and Technology*, 5(4), pp. 2630-2635.

Scientific Department, The Royal College of Ophthalmologists, 2012. *Diabetic Retinopathy Guidelines*, London: The Royal College of Ophthalmologists .

Senthilvel, V., Radhakrishnan, R. & Sathiyamoorthi, R., 2012. Prediction of diabetic retinopathy among diabetics using binary logistic regression approach. *Indian Journal Of Medical Specialities*, 3(1), pp. 18-20.

Shankar, K. et al., 2020. Automated Detection and Classification of Fundus Diabetic Retinopathy Images using Synergic Deep Learning Model. *Pattern Recognition Letters*.

Solomon, S. D. et al., 2017. Diabetic Retinopathy: A Position Statement by the American Diabetes Association. *Diabetes Care*, p. 412–418.

Song, S. J. & Wong, T. Y., 2014. Current Concepts in Diabetic Retinopathy. *Diabetes & Metabolism Journal*, Volume 38, pp. 416-425.

The Foundation of the American Society of Retina Specialists, 2016. Diabetic Retinopathy. *RETINA HEALTH SERIES | Facts from the ASRS*.

Ullah, I. & Petrosino, A., 2016. *About Pyramid Structure in Convolutional Neural Networks*, Italy: ResearchGate.

Vaishnavi, J., Subban, R., Anousouya, M. & Punitha, S., 2016. Detection of Diabetic Retinopathy based on Classification Algorithms. *International Journal of Computer Science and Information Security*, Volume 14, pp. 75-81.

Verma, K., Deep, P. & Ramakrishnan, A. G., 2011. *Detection and Classification of Diabetic Retinopathy using Retinal Images*. s.l., s.n.

Vislisel, J. & Oetting, T., 2010. *Diabetic Retinopathy: from one medical student to another*. [Online]

Available at: <http://www.EyeRounds.org/tutorials/diabetic-retinopathy-med-students/>

[Accessed 30 June 2020].

Wang, W. & Lo, A. C. Y., 2018. Diabetic Retinopathy: Pathophysiology and Treatments. *International Journal of Molecular Sciences*, Volume 19, pp. 1-14.

Wang, K. (., 2016. *Image Classification with Pyramid Representation and Rotated Data Augmentation on Torch 7*, California: Stanford University .

Wan, S., Liang, Y. & Zhang, Y., 2018. Deep convolutional neural networks for diabetic retinopathy detection by image classification. *Computers & Electrical Engineering*, Issue 72, pp. 274-282.

Whiting, D., Guariguata , L., Weil , C. & Shaw , J., 2011. IDF diabetes atlas: global estimates of the prevalence of diabetes for 2011 and 2030. *Diabetes Res ClinPract* , 12(3), pp. 311-321.

Wikipedia, the free encyclopedia, 2010. *Pyramid (Image Processing)*. [Online]

Available at: [wikiwand.com/en/Pyramid\\_\(image\\_processing\)](http://wikiwand.com/en/Pyramid_(image_processing))

[Accessed 19 August 2020].

## **Appendix**

Link to the large augmented image datasets

[https://diabetic-retinopathy1.s3-eu-west-1.amazonaws.com/Diabetic\\_Retinopathy\\_Project.rar](https://diabetic-retinopathy1.s3-eu-west-1.amazonaws.com/Diabetic_Retinopathy_Project.rar)