A Project Report
on

DIABETIC RETINOPATHY DETECTION

submitted in partial fulfillment of the requirements for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

17WH1A0594	**Ms. V. NEHA CHOWDARY**
17WH1A05A1	**Ms. J. NIKITHA**
18WH5A0518	**Ms. S. SHIRISHA**

under the esteemed guidance of

Mr. M. DYVA SUGNANA RAO
Assistant Professor

Department of Computer Science and Engineering
BVRIT HYDERABAD
College of Engineering for Women
(NBA Accredited – EEE, ECE, CSE and IT)
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Bachupally, Hyderabad – 500090

June, 2021

DECLARATION

We hereby declare that the work presented in this project entitled **"DIABETIC RETINOPATHY DETECTION"** submitted towards completion of Project Work in IV year of B.Tech., CSE at 'BVRIT HYDERABAD College of Engineering For Women', Hyderabad is an authentic record of our original work carried out under the guidance of Mr. M. Dyva Sugnana Rao, Assistant Professor, Department of CSE.

Sign. with date:
Ms. V. NEHA CHOWDARY
(17WH1A0594)

Sign. with date:
Ms. J. NIKITHA
(17WH1A05A1)

Sign. with date:
Ms. S. SHIRISHA
(18WH5A0518)

BVRIT HYDERABAD
College of Engineering for Women
(NBA Accredited – EEE, ECE, CSE and IT)
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Bachupally, Hyderabad – 500090

Department of Computer Science and Engineering

Certificate

This is to certify that the Project Work report on "**DIABETIC RETINOPATHY DETECTION**" is a bonafide work carried out by Ms. V. NEHA CHOWDARY (17WH1A0594) ; Ms. J. NIKITHA (17WH1A05A1) ; Ms. SURANNAGARI SHIRISHA (18WH5A0518) in the partial fulfillment for the award of B.Tech. degree in **Computer Science and Engineering, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Head of the Department
Dr. K. Srinivasa Reddy
Professor and HoD,
Department of CSE

Guide
Mr. M. Dyva Sugnana Rao
Assistant Professor

External Examiner

Acknowledgements

We would like to express our sincere thanks to **Dr. K V N Sunitha, Principal**, **BVRIT HYDERABAD College of Engineering for Women**, for providing the working facilities in the college.

Our sincere thanks and gratitude to our **Dr. Ch. Srinivasulu, Professor**, Department of CSE, **BVRIT HYDERABAD College of Engineering for Women** for all the timely support and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Mr. M. Dyva Sugnana Rao, Assistant Professor**, Department of CSE**, BVRIT HYDERABAD College of Engineering for Women** for his constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank our Project Coordinator, all the faculty and staff of **CSE** Department who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

Sign. with date:

Ms. V. NEHA CHOWDARY

(17WH1A0594)

Sign. with date:

Ms. J. NIKITHA

(17WH1A05A1)

Sign. with date:

Ms. S. SHIRISHA

(18WH5A0518)

ABSTRACT

Diabetic retinopathy (DR) is an eye disease caused by the complication of diabetes and we should detect it early for effective treatment. As diabetes progresses, the vision of a patient may start deteriorate and lead to diabetic retinopathy. As a result, two groups were identified, Diabetic retinopathy effected retina and Non – Diabetic retinopthay retina. In this paper, to diagnose diabetic retinopathy, three models like K-Nearest Neighbours (KNN), Support vector machine (SVM), Random Forest (RF) are used and their performances are compared. Experimental results show that SVM and RF has an highest accuracy of 96.2% and KNN has an accuracy of 92.5 %. This infers that the SVM and RF model outperforms the other model.

LIST OF FIGURES

S. No	Fig No.	Topic	Page No.
1	1.1	Normal retina (left) & Diabetic retinopathy (right)	1
2	1.2	Normal and DR effected vision images	2
3	1.3	Image of Dataset path in Google drive	3
4	1.4	Image before AHE	5
5	1.5	Image of Original Histogram	5
6	1.6	Image after AHE	6
7	1.7	Histogram image after AHE	6
8	1.8	Architecture diagram & proposed diagram	14
9	1.9	RGB colour retinal image	14
10	2.0	Retinal image after grayscale conversion	15
11	2.1	Mathematical formula for AHE	15
12	2.2	Mathematical formula of min,max values for AHE	16
13	2.3	Image after Adaptive Histogram Equalization	16
14	2.4	Mathematical formula for DWT	17
15	2.5	Image plotting of co-efficient matrices	17
16	2.6	Mathematical formula for Gabor Kernel	18
17	2.7	Image after applying Gabor Kernel	19
18	2.8	Hyper-plane representation of multi-classification	20
19	2.9	Mathematical formula of Euclidean Distance	20
20	3.0	Table of Accuracies for Classifiers	29

Contents

S.No.	Topic	Page No.
	Abstract	i
	List of Figures	ii
1.	Introduction	1
	1.1 Objectives	1
	1.2 Methodology	1
	1.2.1 Dataset	2
	1.2.2 The Proposed Approach	3
	1.2.2.1 Grayscale Conversion	4
	1.2.2.2 Adaptive Histogram Equalization	4
	1.2.2.3 Discrete Wavelets Trasform	6
	1.2.2.4 Gabor Kernel	7
	1.3 Organization of Project	7
2.	Theoretical Analysis of the proposed project	9
	2.1 Requirements Gathering	
	2.1.1 Software Requirements	9
	2.1.2 Hardware Requirements	9
	2.2 Technologies Description	9
3.	Design	13
	3.1 Introduction	13
	3.2 Architectural Diagram & Proposed System	13
	3.3 Pre-processing	
	3.3.1 Grayscale Conversion	14
	3.3.2 Adaptive Histogram Equalization	15
	3.3.3 Discrete Wavelet Traansform	16
	3.4 Feature Extraction	
	3.4.1 Gabor Kernel	17
	3.5 Classification	
	3.5.1 Support Vector Machine	19
	3.5.2 K-Nearest Neighbours	20

	3.5.3 Random Forest	21
4.	Implementation	
	4.1 Code	22
	4.2 Output	
	4.2.1 Prediction Accuracies of Classification	29
5	Conclusion & Future Scope	30
6.	References	31

1. INTRODUCTION

Diabetes is a group of metabolic diseases in which a person has high blood sugar, either because the body does not produce enough insulin, or because cells do not respond to the insulin that is produced. Diabetic retinopathy (DR) is one of the common complications of diabetes. DR is one of the most prevalent chronic debilitating diseases worldwide.

About 63 million people, 1 in 11 are affected by DM, and 1 in 2 people are undiagnosed . The most threatening microvascular complication of DM is Diabetic Retinopathy (DR). Early detection of DR is essential for treatment success. However, at the early stage, this disease has no specific symptoms, therefore making it really challenging to diagnose. The diabetic patients must be screened annually for DR. However, it is practically not viable to achieve this goal due to the large volume of diabetics, lack of resources, economic burden, and cost of screening procedures.

As it is a severe and widely spread eye disease. It damages the small blood vessels in the retina resulting in loss of vision. The risk of the disease increases with age and therefore, middle aged and older diabetics are prone to Diabetic Retinopathy. Non proliferative diabetic retinopathy is an early stage of diabetic retinopathy. In this stage, tiny blood vessels within the retina leak blood or fluid. The leaking fluid causes the retina to swell or to form deposits called exudates. Proliferative diabetic retinopathy, PDR, is an attempt by the eye to grow or re-supply the retina with new blood vessels (neovascularization), due to widespread closure of the retinal blood supply.

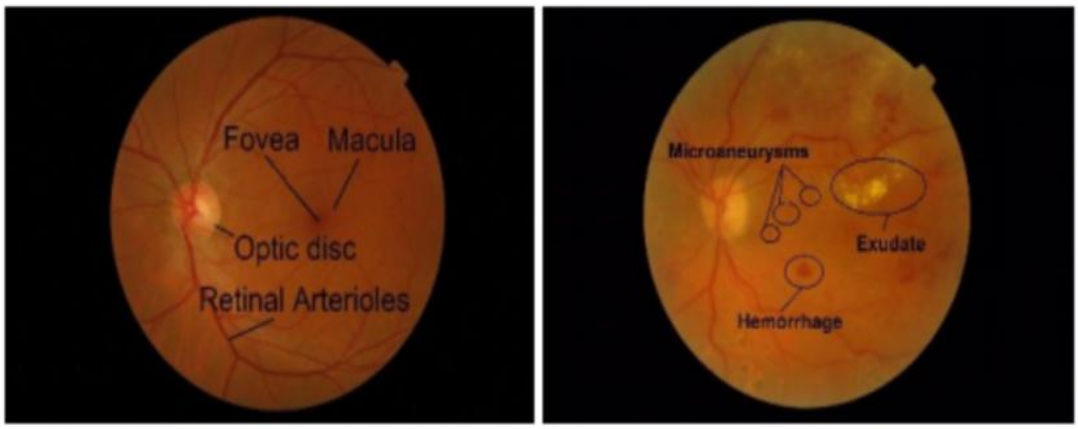

Figure 1.1 : Normal retina (left) & Diabetic retinopathy (right)

Here is an example image of how a DR affected person would have the vision,

(a) Normal Eye[4] (b) DR affected eye[4]

Figure 1.2 : Normal and DR effected vision images

1.1 Objectives

This project mainly focuses on the prediction of diabetic retinopathy and analyses the performance of different algorithms. Machine learning algorithms such as KNN, RF, SVM can be trained by providing training datasets to them and then these algorithms can predict the data by comparing the provided data with the training datasets. Our objective is to train our algorithm by providing training datasets to it and our goal is to detect diabetic retinopathy using different types of classification algorithms.

1.2 Methodology

To predict Diabetic retinopathy, retinal fundus images are required. These images are downloaded from DIARETDB1 database. In this section, the methodology followed is discussed in detail.

1.2.1 Dataset

Proper Dataset is required for all classification research during the training and the testing phase. The Dataset for the project is downloaded from **DIARETDB1** database which contains different colour retinal fundus images and their labels. The database consists of 89 colour fundus images of which 84 contain at least mild non-proliferative signs (Microaneurysms) of the diabetic retinopathy, and 5 are considered as normal which do not contain any signs of the diabetic retinopathy.

Download the images

The first step is to download the dataset. As the programming environment is Google Colab the dataset is uploaded to the google drive.

Place all the images in the folder structure as listed below.

/content/drive/Mydrive/Dataset/

Under the Dataset, the folder structure should as shown below

image001.png

image002.png

image003.png

.

.

.

image089.png

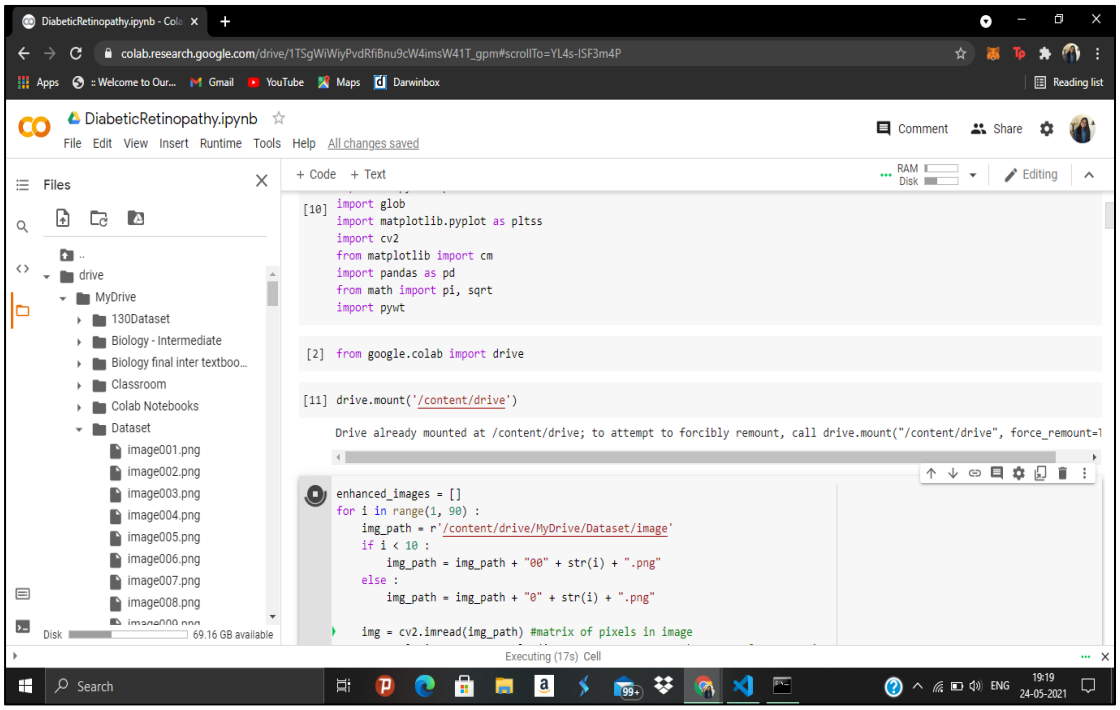

Figure 1.3 : Image of Dataset path in Google drive

1.2.2 The Proposed Approach :

The proposed approach involves the following pre-processing and Feature extraction techniques.

1.2.2.1 Gray scale conversion :

Grayscale is a range of monochromatic shades from black to white. Therefore, a grayscale image contains only shades of gray and no color. While digital images can be saved as grayscale (or black and white) images, even color images contain grayscale information. This is because each pixel has a luminance value, regardless of its color. Luminance can also be described as brightness or intensity, which can be measured on a scale from black (zero intensity) to white (full intensity).

RGB image contains lots of data which may not be required in the image processing. When a RGB image is converted into Gray scale lots of information is discarded which is not required for processing.

Incase of a RGB scale image, for each of the component, i.e. R,G,B, image holds different intensity labels. RGB image is represented by 3 channels. each of the channel generally consists of 8-bits. Therefore, a color image will have intensities for each scale. Hence, there will be lots of data to store and or manipulate. So, gray scale conversion is used as a preprocessing technique.

1.2.2.2 Adaptive Histogram Equalization :

Adaptive histogram equalization (AHE) is a computer image processing technique used to improve contrast in images. It differs from ordinary histogram equalization in the respect that the adaptive method computes several histograms, each corresponding to a distinct section of the image, and uses them to redistribute the lightness values of the image. It is therefore suitable for improving the local contrast and enhancing the definitions of edges in each region of an image.

Ordinary histogram equalization uses the same transformation derived from the image histogram to transform all pixels. This works well when the distribution of pixel values is similar throughout the image. However, when the image contains regions that are significantly lighter or darker than most of the image, the contrast in those regions will not be sufficiently enhanced.

This is done to improve contrast in images. AHE is applied to all the gray scale images. As a result of AHE, the dark area in the input eye image that was badly illuminated will become brighter in the output eye image while the side that was

highly illuminated remains or reduces so that the whole illumination of the eye image is same.

1. To appreciate better the results of equalization, let's introduce an image with not much contrast, such as:

Figure 1.4 : Image before AHE

With original histogram as of below image

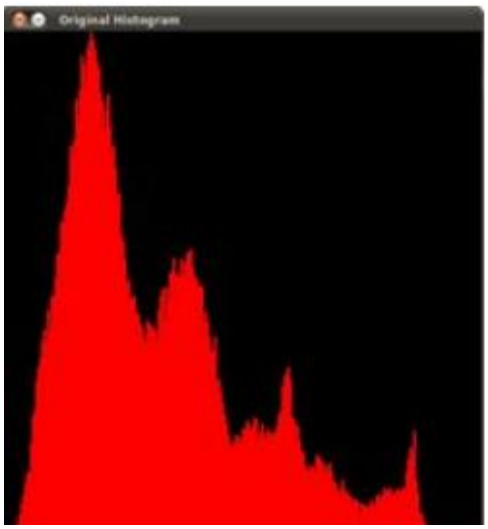

Figure 1.5 : Image of original histogram

Notice that the pixels are clustered around the center of the histogram.

2. After applying the equalization with our program, we get this result:

Figure 1.6 : Image after AHE

This image has certainly more contrast. Check out its new histogram like this:

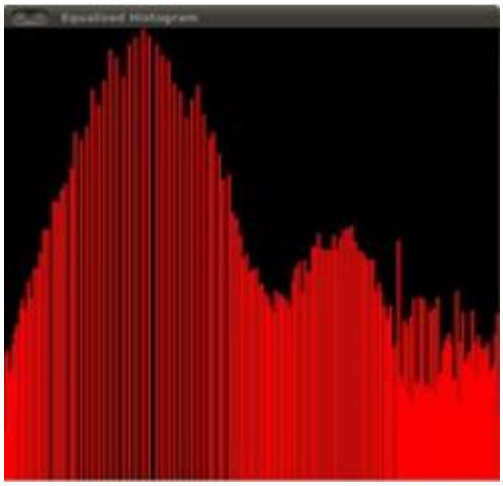

Figure 1.7 : Histogram Image after AHE

Notice how the number of pixels is more distributed through the intensity range.

1.2.2.3 Discrete Wavelet Transform :

Image processing is emerging research area which seeks attention in biomedical field. There are lots of image processing techniques which are not only useful in extracting useful information for analysis purpose but also saves computation time and memory space. Transformation is one such type of image processing technique. Examples of transform techniques are Hilbert transform, Fourier transform, Radon Transform,

wavelet transform etc. Transform technique may be chosen based on its advantages, disadvantages and applications. The wavelet transform is a technique which assimilates the time and frequency domains and precisely popular as time-frequency representation of a non stationary signal.

The transform of a signal is just another form of representing the signal. It does not change the information content present in the signal. The Discrete Wavelet Transform (DWT), which is based on sub band coding, is found to yield a fast computation of Wavelet Transform. It is easy to implement and reduces the computation time and resources required. Wavelet transform decomposes a signal into a set of basis functions. These basis functions are called wavelets. Wavelets are obtained from a single prototype wavelet $\psi(t)$ called mother wavelet by dilations and shifting.

1.2.2.4 Gabor Kernel :

In image processing, a Gabor filter, named after Dennis Gabor, is a linear filter used for texture analysis, which essentially means that it analyzes whether there is any specific frequency content in the image in specific directions in a localized region around the point or region of analysis.

Frequency and orientation representations of Gabor filters are claimed by many contemporary vision scientists to be similar to those of the human visual system. They have been found to be particularly appropriate for texture representation and discrimination. In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. Thus, image analysis with Gabor filters is thought by some to be similar to perception in the human visual system.

A set of Gabor filters with different frequencies and orientations may be helpful for extracting useful features from an image. 2-D Gabor filters have rich applications in image processing, especially in feature extraction for texture analysis and segmentation.

1.3 Organization of the project

The technique which is developed imports the dataset and performs pre-processing techniques on all the images. Then Feature extraction is performed on these images. After that, the resulting data is split into different ratios of training and testing data.

The Training data is then fed to the ML algorithms. Finally, the accuracies of the algorithms are compared.

We have three modules in this project.

- Pre-processing (Grayscale conversion, AHE, DWT)
- Feature Extraction using Gabor Kernel
- Classification using SVM, KNN, RF

2. THEORETICAL ANALYSIS OF THE PROPOSED PROJECT

2.1 Requirements Gathering

2.1.1 Software Requirements

Programming Language: Python 3.6

Dataset : DIARETDB1

Packages : Opencv (cv2), Numpy, Pywt, Matplotlib, Scikit-learn

Tool : Google Colaboratory (Colab)

2.1.2 Hardware Requirements

Operating System : Windows 10

Processor : Intel Core i3-2348M

CPU Speed : 2.30 GHz

Memory : 4 GB (RAM)

2.2 Technologies Description

Python 3.6

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric,

but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

DIARETDB1 Dataset

The database consists of 89 colour fundus images of which 84 contain at least mild non-proliferative signs (Microaneurysms) of the diabetic retinopathy, and 5 are considered as normal which do not contain any signs of the diabetic retinopathy according to all experts who participated in the evaluation.
Images were captured using the same 50 degree field-of-view digital fundus camera with varying imaging settings. The data correspond to a good (not necessarily typical) practical situation, where the images are comparable, and can be used to evaluate the general performance of diagnostic methods. This data set is referred to as "calibration level 1 fundus images".

Opencv(CV)

Opencv(CV) is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

PyWavelets

PyWavelets is a free Open Source wavelet transform software for Python programming language. It is written in Python, Pyrex/Cython and C for a mix of easy and powerful high-level interface and the best performance.

PyWavelets is very easy to start with and use. Just install the package, open the Python interactive shell and type:

```
>>> import pywt

>>> cA, cD = pywt.dwt([1, 2, 3, 4], 'db1')
```

PyWavelets is a Python programming language package and requires Python 2.4, 2.5 or 2.6 installed. The only external requirement is a recent version of NumPy numeric array module.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc.,

with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis
- Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

Google Colab

Colaboratory, or "**Colab**" for short, is a product from **Google** Research. **Colab** allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

3. DESIGN

3.1 Introduction

The use of Machine Learning (ML) in medical science is an emerging frontier, the tasks can be accomplished by the minimal involvement of humans and somehow with more accuracy. In ophthalmology, the use of Artificial Intelligence (AI)/ML can focus on diagnosing DR, Glaucoma, Age-related Macular Degeneration (ARMD), Retinal Vein Occlusion (RVO), Cataract and Retinopathy of Prematurity (ROP). There are many clinical benefits of using ML algorithms in detecting irreversible visual impairment diseases like glaucoma, DR and ARMD. Integrating ophthalmology and ML has the capability to transform diagnosing the disease pattern that can help in generating a compelling clinical effect. A study conducted in Spain used three different classification ML algorithms for detecting the DR; Random Forest (RF) performed the best with an accuracy of 80%. The Food and Drug Administration (FDA), USA also developed an AI-algorithm for the detection of DR, with Support Vector Machine (SVM), results showed an accuracy of 82%. There are few available ML algorithms for the detection of DR, but their performance has been criticized in the literature. This leads to develop a strong interest in proposing a ML model that can help in detecting DR in Saudi diabetic patients by inspecting the socio-demographic and clinical data. This is the first study in KSA that have used the ML algorithms in predicting the DR; various supervised ML algorithms have be applied like SVM, K-Nearest Neighbor (KNN), RF and Ranger Random Forest (RRF, on the DR data. The accuracy of each of the methods was evaluated and is presented in the subsequent sections.

3.2 Architecture Diagram & Proposed System

The evaluation of the proposed automated diagnosis system of diabetic retinopathy has been performed by using a set of 89 images which is a combination of normal (healthy) and diseased eyes. The image which is of size 1152×1500 is converted to gray scale image. After that, adaptive histogram equalization is applied to improve the contrast of the image. Then, DWT is applied. Then, Gabor Kernel is applied to extract features. Finally, modeling techniques like SVM, KNN and RF are used and their

performances are compared. Finally, the images are classified into healthy and affected eyes.

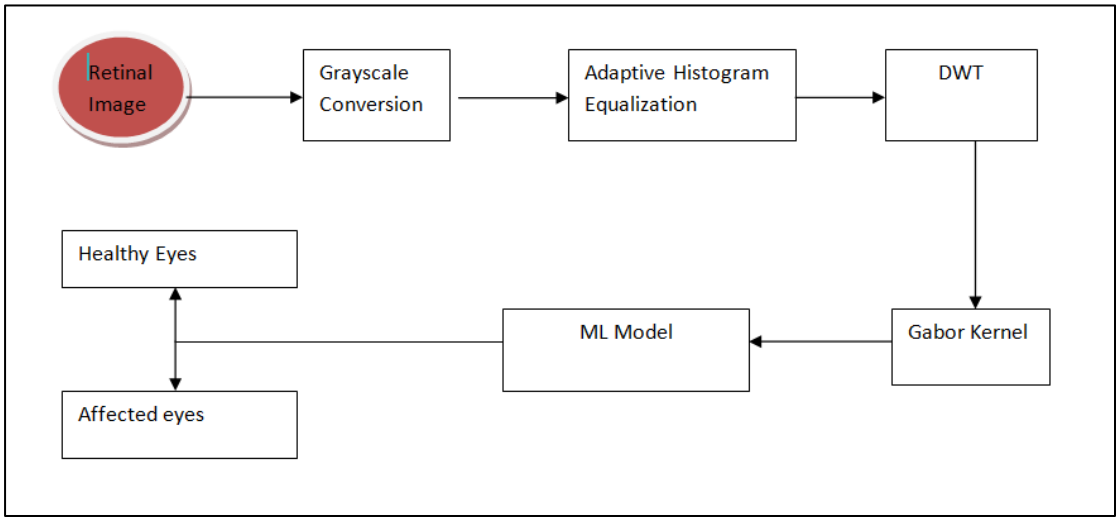

Figure 1.8 : Architecture diagram of proposed system of Diabetic Retinopathy

3.3 Pre-processing

3.3.1 Grayscale conversion

The RGB colour retinal fundus images in the dataset are converted into Grayscale using opencv.

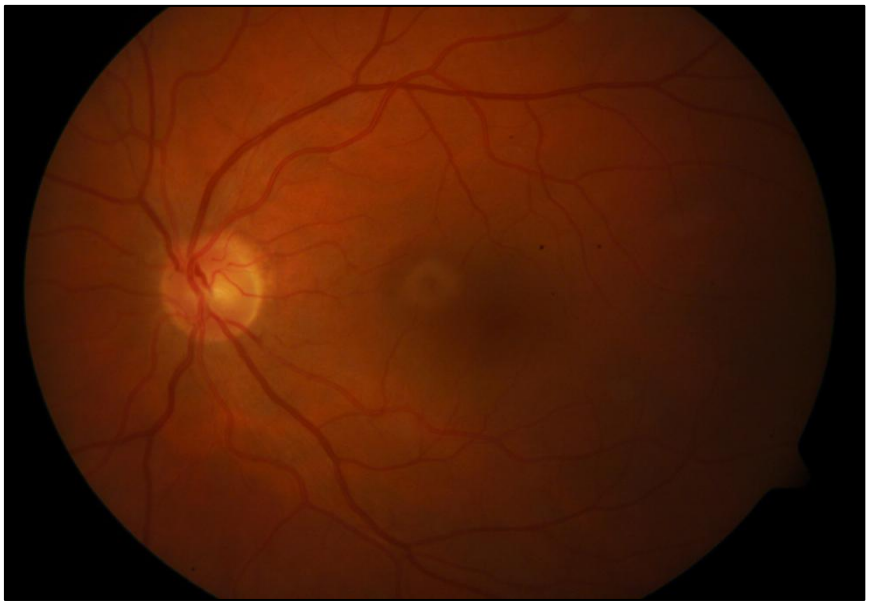

Figure 1.9 : RGB colour retinal image

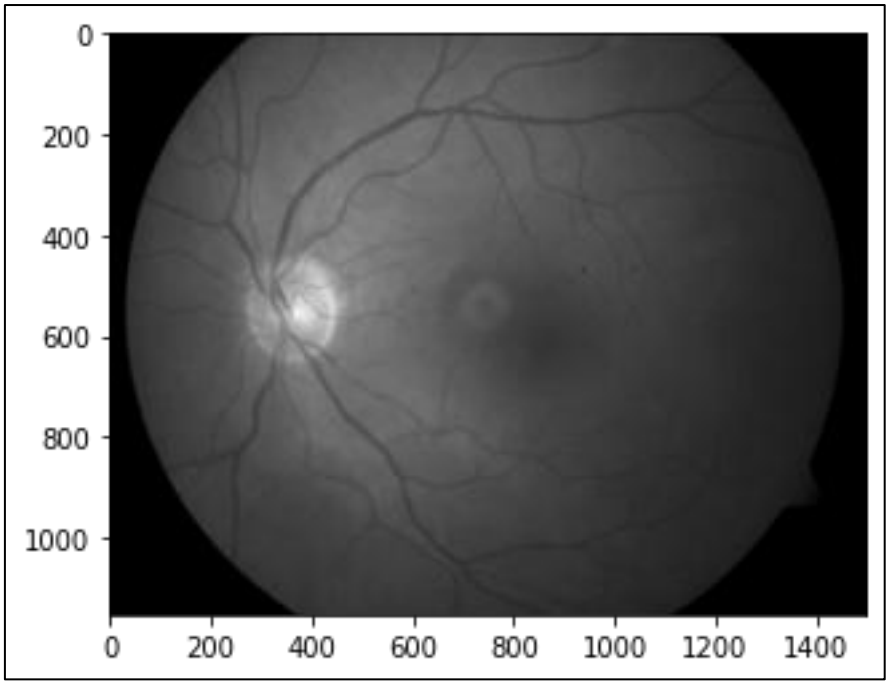

Figure 2.0 : Retinal image after Grayscale conversion

3.3.2 Adaptive histogram Equalization (AHE):

Adaptive histogram equalization which is used to improve contrast in images, is applied to the grayscale converted eye image. Consider a running sub image W of N×N pixels centered on a pixel P(i,j), the image is filtered to produce another sub image P of (N×N) pixels according to the equation below:

$$P_n = 255\left(\frac{[\phi_w(p) - \phi_w(Min)]}{[\phi_w(Max) - \phi_w(Min)]}\right)$$

Where

$$\phi_w(P) = \left[1 + \exp\left(\frac{\mu_w - p}{\sigma_w}\right)\right]^{-1}$$

Figure 2.1 : Mathematical formula for AHE

Max and Min are the maximum and minimum intensity values in the whole eye image while μw indicates the local window mean and σw indicates standard deviation which are defined as:

$$\mu_w = \frac{1}{N^2} \sum_{(i,j)\varepsilon(k,l)} P(i,j)$$

$$\sigma_w = \sqrt{\frac{1}{N^2} \sum_{(i,j)\varepsilon(k,l)} (P(i,j) - \mu_w)^2}$$

Figure 2.2 : Mathematical formula for max & min values of AHE

As a result of this adaptive histogram equalization, the dark area in the input eye image that was badly illuminated has become brighter in the output eye image while the side that was highly illuminated remains or reduces so that the whole illumination of the eye image is same.

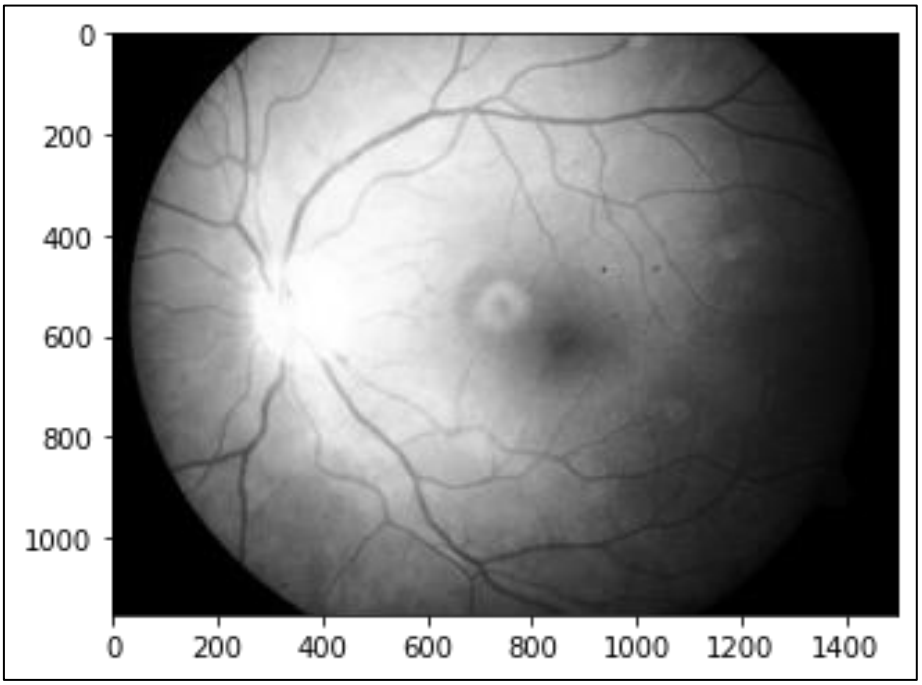

Figure 2.3 : Image after Adaptive Histogram Equalization (AHE)

3.3.3 Discrete wavelet transform (DWT):

The Discrete Wavelet Transform (DWT), which is based on subband coding, is found to yield a fast computation of Wavelet Transform. It is easy to implement and reduces the computation time and resources required. Wavelet transform decomposes a signal into a set of basis functions. These basis functions are called wavelets. Wavelets are obtained from a single prototype wavelet ψ(t) called mother wavelet by dilations and shifting:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$$

Figure 2.4 : Mathematical formula for DWT

where ‘a‘ is the scaling parameter and ‘b‘ is the shifting parameter. The mother wavelet used to generate all the basis functions is designed based on some desired characteristics associated with that function. [cA,cH,cV,cD] = dwt2(X,'wname') computes the approximation coefficients matrix cA and details coefficients matrices cH, cV, and cD (horizontal, vertical, and diagonal, respectively), obtained by wavelet decomposition of the input matrix X where X is the given input eye image

After applying adaptive histogram equalization. The 'wname' string contains the wavelet name. In this project, Haar wavelet is used. After that, inverse discrete wavelet transform is applied on the coefficient matrices to get the final transformed image.

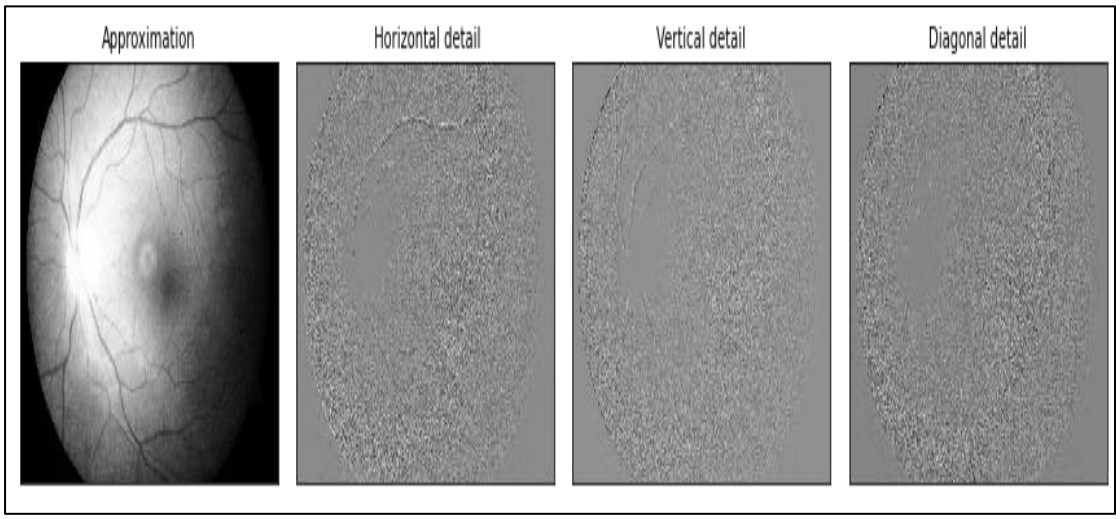

Figure 2.5 : Image plotting of co-efficient matrices

3.4 Feature Extraction

3.4.1 Gabor Kernel

Gabor filter is a linear filter with a Gaussian kernel which is modulated by a sinusoidal plane wave. Frequency and orientation representations of the Gabor filter are similar to those of the human visual system. Gabor filter banks are commonly

used in computer vision and image processing. These filters can be used for texture analysis, edge detection, feature extraction. These filters are band pass filters, meaning; they allow certain band of frequencies and reject other types.

$$g_{\lambda\theta\psi\sigma\gamma}(x,y) = exp(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2})cos(2\pi\frac{x'}{\lambda} + \psi)$$

$$x' = xcos(\theta) + ysin(\theta),$$

$$y' = ycos(\theta) - xsin(\theta).$$

Figure 2.6 : Mathematical formula for Gabor Kernel

In this equation, λ represents the wavelength of the sinusoidal factor, θ represents the orientation of the normal to the parallel stripes of a Gabor function, ψ is the phase offset, σ is the sigma/standard deviation of the Gaussian envelope and γ is the spatial aspect ratio, and specifies the ellipticity of the support of the Gabor function.

To generate gabor features, we sweep through the parameter space of cv2.getGaborKernel function and create a bunch of kernels. These kernels are called bank filters. The function createGaborFilterBank() creates and returns 16 different kernels by only changing one parameter that is theta. Theta is the orientation of the normal to the parallel stripes of a Gabor function. Simply put, angle. Then, applyFilters() function takes an image and filter bank as the input parameters and applies all the 16 kernels to the one single input image. This generates 16 convoluted images. Then, we form a feature vector by considering the maximum of the pixel values of the 16 convoluted images. Then we do this process for all the images in the dataset and form feature vector for all images.

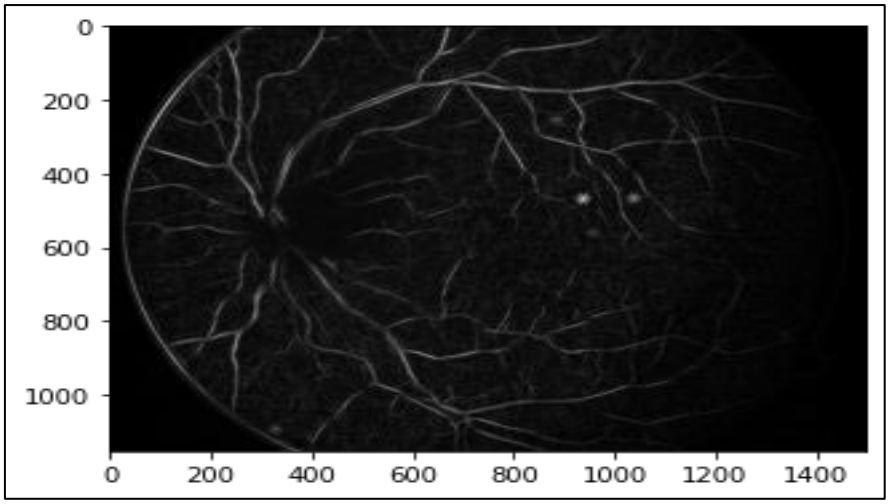

Figure 2.7: Image after applying Gabor kernel

3.5 Classification

The gabor features are divided into different ratios of training and testing data. The different ratios are :

60 : 40

70 : 30

80 : 20

Now, these different ratios of data are fed into classification algorithms and their accracies are compared

3.5.1 Support Vector Machine (SVM) :

SVM classifier: The SVM approach basically used for binary classification problems. In this we are using multi-class pattern recognition problem. Basically there are two types to solve multi-class problems by using binary SVM classifiers: The first one is "one-againstone" in this method for each possible pair of classification a binary classifier is calculated. Other approach is the "one-against-rest" in this method N different classifiers are constructed, one for each class. In this paper we built a two-class classifier over a feature vector φ(x, y′) derived from the pair consisting of the input features and the class of the data. At test time the classifier chooses the class

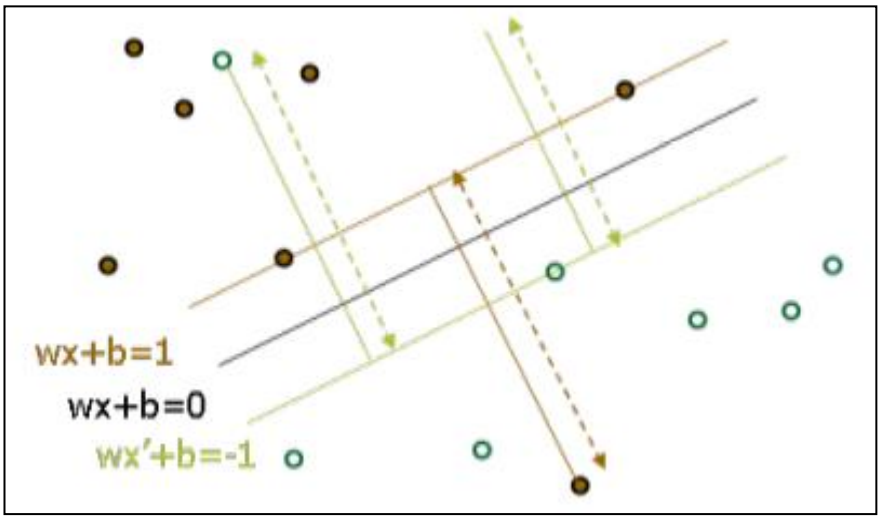

Figure 2.8 : Hyper plane representation for multi-classification.

3.5.2 K-Nearest Neighbors (KNN) :

KNN is another potential candidate for a supervised ML classifier, which utilizes the dissimilarity method for the classification of a new observation. The dissimilarity is usually computed by Euclidean distance. In KNN classification, the output is a class membership. An object is classified by its neighbors' plurality vote, with the object being assigned to the class most common among its K nearest neighbors. KNN algorithm depends on K; larger values of K reduce the effect of noise on the classification but make the boundaries between classes less distinct. It is usually tuned by cross-validation re-sampling procedure.

The Euclidean distance between sample xi and xl (l=1,2,…,n) is defined as:

$$d(x_i, x_i) \equiv \sqrt{(x_{i1} - x_{i1})^2 + (x_{i2} - x_{i2})^2 + \ldots + (x_{ip} - x_{ip})^2}$$

$$\text{A Ri} = \{x \in R_p : d(x, x_i) \leq d(x, xm), \forall \text{ i} \neq \text{m}\}$$

Figure 2.9 : Mathematical formula of Euclidean distance

3.5.3 Random Forest (RF)

RF is another prominent ML method; it is an ensemble tree-based learning algorithm. The RF classifier is a set of decision trees that are made from the randomly selected subset of training patients data. It aggregates the votes from different decision trees to decide the final class of the test object, majority votes of all predicted classes over B trees.

4. IMPLEMENTATION

4.1 Code

```
#Importing necessary libraries

from numpy import exp
from matplotlib import pyplot as plt
import numpy as np
import cv2
import pywt

from google.colab import drive
drive.mount('/content/drive')

#Loading dataset, Grayscale conversion and Adaptive histogram equalization

enhanced_images = []
for i in range(1, 90) :
   img_path = r'/content/drive/MyDrive/Dataset/image'
   if i < 10 :
      img_path = img_path + "00" + str(i) + ".png"
   else :
      img_path = img_path + "0" + str(i) + ".png"

   img = cv2.imread(img_path) #matrix of pixels in image
   grayscale_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #grayscale
conversion
   equalized_img = cv2.equalizeHist(grayscale_img) #adaptive histogram
equalization
   enhanced_images.append(np.array(equalized_img).flatten())

#Dimensions of the images

print("Dimensions of original image : ")
```

```
print(np.shape(img))
print("Dimensions of grayscale image : ")
print(np.shape(grayscale_img))
print("Dimensions of histogram equalized image : ")
print(np.shape(equalized_img))
print("Dimensions of flattened image : ")
print(np.shape(np.array(equalized_img).flatten()))
print("Dimensions of enhanced : ")
print(np.shape(enhanced_images))
print("No. of elements in enhanced : ")
print(len(enhanced_images))

#Visualizing a random image from the dataset

random_image_path = '/content/drive/MyDrive/Dataset/image079.png'
random_image = cv2.imread(random_image_path)
grayscale_img = cv2.cvtColor(random_image, cv2.COLOR_BGR2GRAY)
print("Grayscale image : ")
plt.imshow(grayscale_img, cmap='gray')
plt.show()
equalized_img = cv2.equalizeHist(grayscale_img)
print("AHE image : ")
plt.imshow(equalized_img, cmap='gray')
plt.show()

#Discrete wavelet transform

dwt_images = []
for equalized_img in enhanced_images :
   equalized_img = equalized_img.reshape((1152,1500))
   coeffs = pywt.dwt2(equalized_img, 'haar') #coeffs - (approximation coeff, details
coeff) -> (cA, (cH, cV, cD))
   wavelet_transformed = pywt.idwt2(coeffs, 'haar') #Inverse discrete wavelet
```

```
transform
    dwt_images.append(np.array(wavelet_transformed).flatten())

#Visualizing the random image after dwt

print("DWT image : ")
plt.imshow(dwt_images[78].reshape((1152,1500)),cmap='gray')
plt.show()

#Visualizing the random image's coefficient matrices returned by dwt

original = enhanced_images[78].reshape((1152, 1500))
# Wavelet transform of image, and plot approximation and details
titles = ['Approximation', ' Horizontal detail',
          'Vertical detail', 'Diagonal detail']
coeffs2 = pywt.dwt2(original, 'bior1.3')
LL, (LH, HL, HH) = coeffs2
fig = plt.figure(figsize=(12, 3))
for i, a in enumerate([LL, LH, HL, HH]):
    ax = fig.add_subplot(1, 4, i + 1)
    ax.imshow(a, interpolation="nearest", cmap=plt.cm.gray)
    ax.set_title(titles[i], fontsize=10)
    ax.set_xticks([])
    ax.set_yticks([])

fig.tight_layout()
plt.show()

#Feature extraction using Gabor filters

def createGaborFilterBank() :
    filters = []
    ksize = 31
```

```
    for theta in np.arange(0, np.pi, np.pi / 16) : #varying angles
        kernel = cv2.getGaborKernel((ksize, ksize), 6, theta, 12, 0.37, 0, ktype =
cv2.CV_32F) #creates a kernel
        kernel /= 1.5 * kernel.sum() #normalizing the kernel
        filters.append(kernel)
    return filters #returns the list of 16 kernels of size 31 x 31

def applyFilters(image, kernels) :
    images = np.array([cv2.filter2D(image, -1, k) for k in kernels])
    return np.max(images, 0) #returns the filtered image

gaborKernel_images = []
for wavelet_transformed_1d in dwt_images : #89 images after dwt
    wavelet_transformed_2d = wavelet_transformed_1d.reshape((1152,1500))
    filtered_image = applyFilters(wavelet_transformed_2d, createGaborFilterBank())
#apply gabor filters to each image
    gaborKernel_images.append(np.array(filtered_image).flatten())

#Visualizing the random image after applying gabor filters

print("Image after applying gabor filter : ")
plt.imshow(gaborKernel_images[78].reshape((1152,1500)), cmap = 'gray')
plt.show()

#Visualizing the filter bank

filterBank = createGaborFilterBank()
for kernel in filterBank :
  plt.imshow(kernel.reshape((31, 31)))
  plt.show()

#Visualizing the convoluted images
```

```
randomImage_dwt = dwt_images[78].reshape((1152,1500))
for kernel in createGaborFilterBank() :
  filtered_image = cv2.filter2D(randomImage_dwt, -1, kernel)
  plt.imshow(filtered_image, cmap = 'gray')
  plt.show()

#Defining labels for the dataset

DR_labels = np.ones(89)
DR_labels[1] = DR_labels[5] = DR_labels[7] = DR_labels[17] = DR_labels[6] = 0
#These images are marked as healthy eye images in the dataset

#Splitting the dataset into Train and test images into different ratios; 70 : 30, 60 : 40,
80 : 20

from sklearn.model_selection import train_test_split
X_train_70, X_test_30, y_train_70, y_test_30 = train_test_split(gaborKernel_images,
DR_labels, test_size = 0.3)
X_train_60, X_test_40, y_train_60, y_test_40 = train_test_split(gaborKernel_images,
DR_labels, test_size = 0.4)
X_train_80, X_test_20, y_train_80, y_test_20 = train_test_split(gaborKernel_images,
DR_labels, test_size = 0.2)

#Classification using Support Vector Machine

from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
clf = SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape=None, degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)

def svmClassification(train, test, labelsTrain, labelsTest, clf) :
  clf.fit(train, labelsTrain)
```

```
  pred_labels = clf.predict(test)
  accuracy = accuracy_score(labelsTest, pred_labels)
  return accuracy

print(svmClassification(X_train_60, X_test_40, y_train_60, y_test_40, clf)) #94.4%
print(svmClassification(X_train_70, X_test_30, y_train_70, y_test_30, clf)) #96.2%
print(svmClassification(X_train_80, X_test_20, y_train_80, y_test_20, clf)) #88.8%

#Classification using K-Nearest Neighbors Classifier

from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors = 3)

def knnClassification(train, test, labelsTrain, labelsTest, clf) :
  clf.fit(train, labelsTrain)
  p = clf.predict(test)
  accuracy = accuracy_score(labelsTest, p)
  return accuracy

print(knnClassification(X_train_60, X_test_40, y_train_60, y_test_40, neigh))
#88.8%
print(knnClassification(X_train_70, X_test_30, y_train_70, y_test_30, neigh))
#92.5%
print(knnClassification(X_train_80, X_test_20, y_train_80, y_test_20, neigh))
#94.4%

#Classification using RandomForest Classifier

from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
rfc = RandomForestClassifier(n_estimators = 25, random_state = 0)

def rfClassification(train, test, labelsTrain, labelsTest, clf) :
```

```
  clf.fit(train, labelsTrain)
  p = clf.predict(test)
  accuracy = accuracy_score(labelsTest, p)
  return accuracy

print(rfClassification(X_train_60, X_test_40, y_train_60, y_test_40, rfc)) #91.6%
print(rfClassification(X_train_70, X_test_30, y_train_70, y_test_30, rfc)) #96.2%
print(rfClassification(X_train_80, X_test_20, y_train_80, y_test_20, rfc)) #94.4%
```

4.2. OUTPUT

4.2.1 Prediction Accuracies of Classification Algorithms:

Classifier	Accuracy for 60:40	Accuracy for 70:30	Accuracy for 80: 20
SVM	94.4%	96.2%	88.8%
KNN	88.8%	92.5%	94.4%
RF	91.6%	96.2%	94.4%

Table 3.0 : Accuracy for Classifiers

After applying different amounts of data to the Classification algorithms i.e., Support Vector Machine - SVM, k-nearest neighbors - KNN, Random Forest – RF, it is observed that there is a result of highest accuracy of 96.2% accuracy in both SVM and RF when the training and testing data is split in the ratio of 70 : 30 respectively.

5. CONCLUSION AND FUTURE SCOPE

Conclusion :

By using 53 number of images for training and 36 number of images for testing, SVM and RF gave the highest accuracy of 96.2% where as KNN gave accuracy of 92.5%.

Future Scope :

In future, below mentioned extensions can be made to the project

- Increase the size of the data set
- To use better morphological analysis algorithms to get clearer features.
- Implement neural nets in a better and efficient way
- Detect different stages of DR

6. REFERENCES

[1] Diabetic Retinopathy Detection Using Machine Learning and Texture Features Mohamed Chetoui, Moulay A. Akhloufi, Mustapha Kardouchi Perception, Robotics, and Intelligent Machines Research Group (PRIME), Dept. of Computer Science, Universitéde Moncton Moncton, NB, Canada (IEEE Canadian Conference on Electrical & Computer Engineering (CCECE) - 2018)

[2] R. Priya, P. Aruna, "SVM and Neural Network based Diagnosis of Diabetic Retinopathy", IJCA, 2012.

[3] S. Mohammadian, A. Karsaz and Y. M. Roshan, "A comparative analysis of classification algorithms in diabetic retinopathy screening," 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, pp. 84-89, 2017.

[4] M. N. Ashraf, Z. Habib and M. Hussain, "Texture Feature Analysis of Digital Fundus Images for Early Detection of Diabetic Retinopathy," 2014 11th International Conference on Computer Graphics, Imaging and Visualization, Singapore, pp. 57-62, 2014.

[5] M. Tavakoli, R. Shahri, H. Pourreza, A. Mehdizadeh, T. Banaee and M. Bahreini Toosi, "A complementary method for automated detection of microaneurysms in fluorescein angiography fundus images to assess diabetic retinopathy," Pattern Recognition, vol. 46, no. 10, pp. 2740- 2753, 2013.

[6] Hemant P. Chaudhari, Amol D. Rahulkar, Chetankumar Y. Patil, "Segmentation of Retinal Vessels by the Use of Gabor Wavelet and Linear Mean Squared Error Classifier", IJEERT, 2014.