

VEL TECH HIGH TECH

Dr. RANGARAJAN Dr. SAKUNTHALA ENGINEERING COLLEGE
An Autonomous Institution



DIABETIC RETINOPATHY PREDICTION USING CNN

A MINI PROJECT REPORT

Submitted by

B.SAM JEFFY(113021243043)

A.HARINISH(113021243018)

I.BALAJI(113021243007)

B.Tech – ARTIFICIAL INTELLIGENCE & DATA SCIENCE

NOVEMBER 2022

VEL TECH HIGH TECH

Dr. RANGARAJAN Dr. SAKUNTHALA ENGINEERING COLLEGE
An Autonomous Institution



BONAFIDE CERTIFICATE

Certified that this mini project entitled “**DIABETIC RETINOPATHY PREDICTION USING CNN**” is the bonafide work of “**B.SAM JEFFY, A.HARINISH, I.BALAJI**” who carried out the work under my supervision.

SIGNATURE

MR. R. KARTHIKEYAN M.E.,
HEAD OF THE DEPARTMENT
ASSISTANT PROFESSOR OF THE
DEPARTMENT OF ARTIFICIAL
INTELLIGENCE AND DATA SCIENCE
VEL TECH HIGH TECH
DR. RANGARAJAN AND
DR. SAKUNTHALA ENGINEERING
COLLEGE

SIGNATURE

MRS. VEERASUNDARI R.
SUPERVISOR
ASSISTANT PROFESSOR OF THE
DEPARTMENT OF ARTIFICIAL
INTELLIGENCE AND DATA SCIENCE
VEL TECH HIGH TECH
DR. RANGARAJAN AND
DR. SAKUNTHALA ENGINEERING
COLLEGE

ABSTRACT

Diabetic Retinopathy (DR) is a fastly spreading disease that may lead to loss of vision if not quickly detected and treated. Early-stage detection is beneficial to restrict the progress of disease and reduces the recovery expenditure. The current detection process of DR heavily depends on domain experts. Machine-dependent approaches are gain attention with large-scale fundus image repositories to overcome this difficulty. Recent techniques with deep learning are successful in getting noticeable results with pre-trained networks. However, the increase of memory occupancy and runtime with existing models is the bottleneck. We propose Binary Convolutional Neural Networks (BCNN), which drastically reduces memory consumption and faster the execution process to combat this problem. Our model is hardware friendly and efficient in DR classification with large scale fundus images. Experiments conducted using the Kaggle dataset reduce memory consumption by 37% and increase runtime by 49% compared to the base model

ACKNOWLEDGEMENT

We take this occasion to thank God, almighty for blessing us with his grace and taking our endeavor to a successful culmination. We would like to express our obeisance to the following persons for their invaluable help rendered. We wish to express our sincere thanks and gratitude to our chairman Col. Prof. **Dr. RANGARAJAN B.E. (Elec.), B.E. (Mech.), M.S (Auto.), DSC** and vice-chairman **Dr. SAKUNTHALA RANGARAJAN M.B.B.S.**, for providing us with a comfort zone for doing this project work. We express our thanks to our principal **Dr. E.KAMALANABAN B.E., M.E., Ph.D.**, for offering us all the facilities to do the project.

We also express our sincere thanks to **Mr. M. R KARTHIKEYAN, M.E., Head of the Department, Assistant Professor**, Department of Artificial Intelligence & Data Science for providing support to do this project work.

We extend our sincere and heartfelt thanks to our esteemed Project supervisor, **MRS. VEERASUNDARI R. Assistant professor**, Department of Artificial Intelligence & Data Science, for providing us with the right guidance and advice at the crucial junctures and for showing me the right way. We also take this opportunity to express a deep sense of gratitude to our **supporting staffs of the AI&DS** Department for their cordial support, valuable suggestions and guidance. Last but not the least, we would like to thank our friends and family for the support and encouragement they have given us during the course of our work.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	iii
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	viii
1	INTRODUCTION	1
	1.1 PROJECT AIMS AND OBJECTIVES	1
	1.2 BACKGROUND OF PROJECT	2
2	SYSTEM ANALYSIS	3
	2.1 SYSTEM OBJECTIVES	3
	2.2 SYSTEM REQUIREMENTS	3
	2.2.1 SOFTWARE REQUIREMENTS	3
	2.2.2 HARDWARE REQUIREMENTS	4
	2.3 EXISTING VS PROPOSED	4
	2.3.1 EXISTING SYSTEM	4
	2.3.2 PROPOSED SYSTEM	4
	2.4 INTRODUCTION TO DEEP LEARNING	5
	2.5 ARTIFICIAL NEURAL NETWORK PROCESS	6

	2.6 CONVOLUTIONAL NEURAL NETWORK	8
3	LITERATURE SURVEY	11
4	SYSTEM DESIGN	15
	4.1 SYSTEM ARCHITECTURE	15
	4.2 SEQUENCE DIAGRAM	17
	4.3 UML DIAGRAM	17
	4.4 USE CASE DIAGRAM	18
	4.5 ACTIVITY DIAGRAM	19
	4.6 COLLABORATION DIAGRAM	20
5	SYSTEM IMPLEMENTATION	21
	5.1 MODULE DESCRIPTION	21
	5.1.1 DATACOLLECTION	21
	5.1.2 CONVOLUTIONAL NEURAL NETWORK	21
	5.1.3 PREDICTION	22
6	CODINGS	23
	6.1 SOURCE CODE	23
	6.1.1 INDEX.HTML	23
	6.1.2 REGISTRATION.HTML	30
	6.1.3 LOGIN.HTML	33
	6.1.4 SUCCESS PAGE.HTML	34

	6.1.5 FILE UPLOADED FROM.HTML	35
	6.1.6 DIABETIC RETINOPATHYWEB.PY	37
7	RESULT AND ANALYSIS	42
	7.1 HOME PAGE	42
	7.2 REGISTRATION PAGE	43
	7.3 LOGIN PAGE	43
	7.4 UPLOAD PAGE	44
	7.5 RESULT PAGE	44
8	CONCLUSION AND DISCUSSION	45
	8.1 FUTURE ENHANCEMENT	45
9	REFERENCE	46

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
4.1	SYSTEM ARCHITECTURE	16
4.2	SEQUENCE DIAGRAM	17
4.3	USE CASE DIAGRAM	18
4.4	ACTIVITY DIAGRAM	19
4.5	COLLABORATION DIAGRAM	20
7.1	HOME PAGE	42
7.2	REGISTRATION PAGE	43
7.3	LOGIN PAGE	43
7.4	UPLOAD PAGE	44
7.5	RESULT PAGE	44

LIST OF ABBREVIATIONS

ABBREVIATION	DESCRIPTION
DR	DIABETIC RETINOPATHY
CNN	CONVOLUTIONAL NEURAL NETWORK
NN	NEAREST NEIGHBOUR
SVM	SUPPORT VECTORE MACHINE

CHAPTER -1

INTRODUCTION

This chapter gives an overview about the aim, objectives, background and operation environment of the system.

1.1 PROJECT AIMS AND OBJECTIVES

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

- To detect the diabetic retinopathy disease in the earlier stage using Deep learning method.
- Diabetic Retinopathy is a disease that can lead to partial or complete blindness.
- Research shows that it contributes around 5 percent of the total cases of blindness.
- Usually it takes about two weeks for the diagnosis of disease; time and money both are wasted. The proposed system aims to eradicate the above problem.
- Convolutional Neural Network (CNNs) is widely used in pattern and image recognition problems as they have a number of advantages compared to other techniques.

1.2 BACKGROUND OF PROJECT

Diabetic retinopathy or diabetic eye disease is caused by diabetes mellitus which manifests itself in the eye retina. Diabetic eye disease is one of the most frequent causes of complete blindness in many developed countries. The detection of retinal pathologies became much easier using automated retinal image analysis whereas other methods like dilation of eye pupil are time consuming and patient has to suffer for some time.

Diabetic retinopathy occurs when high blood glucose damages the small vessel that provides nutrients and oxygen to the retina. There are two types of major diabetes. First (called juvenile-onset or insulin-dependent) diabetes, In this the body completely stops producing insulin, a hormone that enables the body to use glucose found in foods for energy. Second (called adult-onset or non-insulin-dependent).

The proposed system detects stages or levels of diabetes and gives respective prescription for the same by using different classification techniques. Retinal images are used for the detection of diabetes stage which are compared with the other samples of diabetes and if found severe then immediate diagnoses is done and prescription is been provided for the same.

CHAPTER – 2

SYSTEM ANALYSIS

In this chapter, we will discuss and analyze about the developing process and comparison between existing and proposed system .Besides that, existing vs proposed provides a view of how the proposed system will be more efficient than the existing one.

2.1 SYSTEM OBJECTIVES

All of existing methods are good in some measures for detection and segmentation of exudates but still raise some problems with low intensity, low accuracy, less color contrast and sensitivity, non-uniform illumination images. Therefore, our proposed algorithm and techniques have ability to solve these problems by preprocessing techniques

2.2 SYSTEM REQUIREMENTS

This section describes the software and hardware requirements of the system

2.2.1 SOFTWARE REQUIREMENTS

- ✓ Operating System : Windows 7 , 8, 10 (64 bit)
- ✓ Software : Python 3.7
- ✓ Tools : Anaconda (Jupyter Note Book IDE)

2.2.2 HARDWARE REQUIREMENTS

- Hard Disk : 500GB and Above
 - RAM : 4GB and Above
- Processor : I3 and Above

2.3 EXISTING vs PROPOSED SYSTEM

2.3.1 EXISTING SYSTEM

- In Existing system, they provide a provision for only manual consultation or BCNN. Patient has to travel longer distance to consult the doctor.
- If not diagnosed early but in most of the cases it is diagnosed at the later stage. Research shows that it contributes around 5 percent of the total cases of blindness.
- Patient has to wait for hours for the dilation of eyes so as to widen the pupil. After dilation the doctor has to check for abnormal blood vessels, swelling, retinal detachment, test your vision & cataracts.
- This process is too slow and consumption of more time. We want to increase the speed and accuracy by using the deep learning.

2.3.2 PROPOSED SYSTEM

- A model is proposed which uses CNN for the automated detection of Diabetic Retinopathy.
- The client on their first login has to register themselves on the Web Application. The web Application created by Flask.
- Once the user logs into the system he gets all the access for predicting the

diabetic retinopathy by using the input image.

- After submitting the inputs, it's move on to the trained model for comparison.

Already trained model were trained by deep learning algorithms. So, we get accuracy results in this project using CNN.

2.4 INTRODUCTION DEEP LEARNING

Deep Learning is a specialized form of Machine Learning that uses supervised, unsupervised, or semi-supervised learning to learn from data representations.

It is similar to the structure and function of the human nervous system, where a complex network of interconnected computation units works in a coordinated fashion to process complex information.

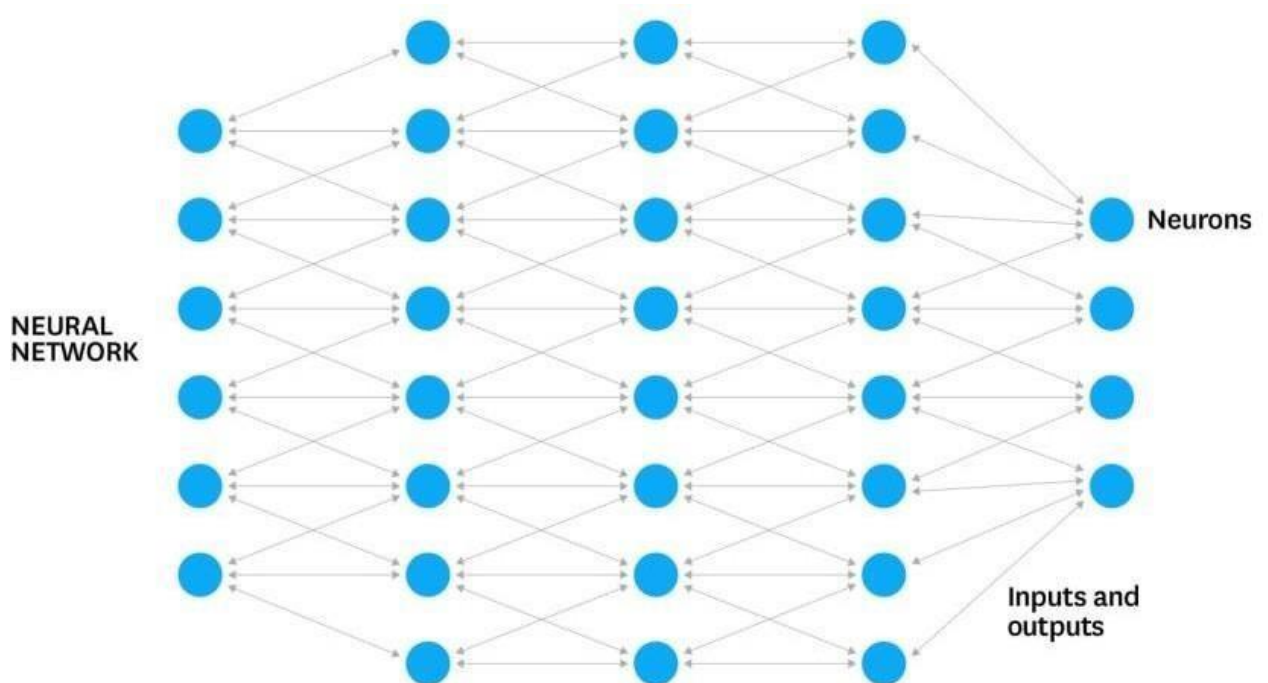
Machine Learning is an approach or subset of Artificial Intelligence that is based on the idea that machines can be given access to data along with the ability to learn from it. Deep Learning takes Machine Learning to the next level.

There are many aspects of Deep Learning as listed below

1. Multiple levels of hierarchical representations
2. Multi-layered neural networks
3. Training of large neural networks
4. Multiple non-linear transformations
5. Pattern recognition
6. Feature extraction
7. High-level data abstractions model

Human Brain vs. Artificial Neural Networks

The computational models in Deep Learning are loosely inspired by the human brain. The multiple layers of training are called Artificial Neural Networks (ANN).

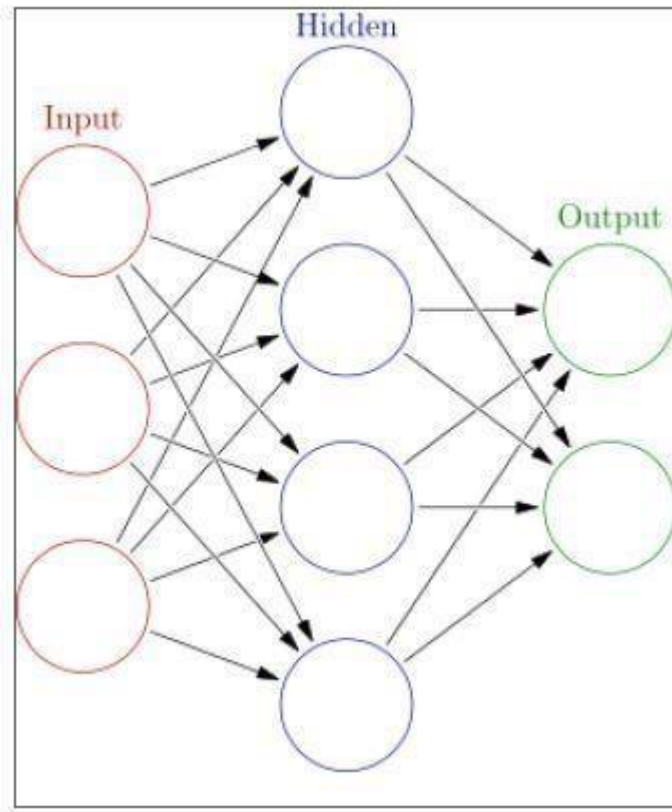


ANNs are processing devices (algorithms or actual hardware) that are modeled on the neuronal structure of the mammalian cerebral cortex but on a much smaller scale. It is a computing system made up of a number of simple, highly interconnected processing elements which process information through their dynamic state response to external inputs.

2.5 ARTIFICIAL NEURAL NETWORKS PROCESS

Artificial Neural Networks consist of the following four main parts:

- Neurons
- Nodes
- Input
- Output



Neuron

Artificial Neural Networks contain layers of neurons. A neuron is a computational unit that calculates a piece of information based on weighted input parameters. Inputs accepted by the neuron are separately weighted.

Inputs are summed and passed through a non-linear function to produce output. Each layer of neurons detects some additional information, such as edges of things in a picture or tumors in a human body. Multiple layers of neurons can be used to detect additional information about input parameters.

Nodes

Artificial Neural Network is an interconnected group of nodes akin to the vast network of layers of neurons in a brain. Each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

Inputs

Inputs are passed into the first layer. Individual neurons receive the inputs, with each of them receiving a specific value. After this, an output is produced based on these values.

Outputs

The outputs from the first layer are then passed into the second layer to be processed. This continues until the final output is produced. The assumption is that the correct output is predefined.

Each time data is passed through the network, the end result is compared with the correct one, and tweaks are made to their values until the network creates the correct final output each time.

2.6 CONVOLUTIONAL NEURAL NETWORKS (CNN)

Introduction

Convolutional neural networks (CNN) sounds like a weird combination of biology and math with a little CS sprinkled in, but these networks have been some of the most influential innovations in the field of computer vision. 2012 was the first year that neural nets grew to prominence as Alex Krizhevsky used them to win that year's ImageNet competition (basically, the annual Olympics of computer vision),

dropping the classification error record from 26% to 15%, an astounding improvement at the time. Ever since then, a host of companies have been using deep learning at the core of their services. Facebook uses neural nets for their automatic tagging algorithms, Google for their photo search, Amazon for their product recommendations, Pinterest for their home feed personalization, and Instagram for their search infrastructure.

However, the classic, and arguably most popular, use case of these networks is for image processing. Within image processing, let's take a look at how to use these CNNs for image classification.

Pre-Processing:

Image pre-processing is the initial step to identify the affected area. Multiple steps are performed in the preprocessing phase to make the image suitable for the feature extraction process. The abnormalities in the input image are detected and preprocessed for the following purpose:

- To avoid uneven illumination
- To enhance the contrast among image background pixels and exudates
- To eliminate the noise in the input image

In this research work, the techniques used for the preprocessing phase are:

- Image resizing.
- Color transformation(RGB to Gray)

Image Resizing:

An image size can be changed in several ways. One of the simpler ways of increasing image size is nearest-neighbor interpolation, replacing every pixel with the nearest pixel in the output; for up scaling this means multiple pixels of the same colour will be present. Image resizing is necessary when you need to increase or decrease the total number of pixels, whereas remapping can occur

when we are correcting for lens distortion or rotating an image. Zooming refers to increase the quantity of pixels, so that when you zoom an image, we will see more detail.

Color Transformation:

The retinal images are taken from the fundus camera in the form of RGB (Red, Green, and Blue). Grayscale is a range of shades of gray without apparent color. The darkest possible shade is black, which is the total absence of transmitted or reflected light. The lightest possible shade is white, the total transmission or reflection of light at all visible wavelengths. Intermediate shades of gray are represented by equal brightness levels of the three primary colors (red, green and blue) for transmitted light for reflected light. In the case of transmitted light (for example, the image on a computer display), the brightness levels of the red (R), green (G) and blue (B) components are each represented as a number from decimal 0 to 255, or binary 00000000 to 11111111. For every pixel in a red-green-blue (RGB) grayscale image, $R = G = B$. The lightness of the gray is directly proportional to the number representing the brightness levels of the primary colors. Black is represented by $R = G = B = 0$ or $R = G = B = 00000000$, and white is represented by $R = G = B = 255$ or $R = G = B = 11111111$. Because there are 8 bits in the binary representation of the gray level, this imaging method is called 8-bit grayscale.

In some cases, rather than using the RGB or CMY color models to define grayscale, three other parameters are defined. These are hue, saturation and brightness. In a grayscale image, the hue (apparent color shade) and saturation (apparent color intensity) of each pixel is equal to 0. The lightness (apparent brightness) is the only parameter of a pixel that can vary. Lightness can range from a minimum of 0 (black) to 100 (white).

CHAPTER-3

LITERATURE SURVEY

**TITLE: DIABETIC RETINOPATHY USING MACHINE LEARNING
ALGORITHM**

**AUTHOR: S.V Chichmalatpure , Snehal Mate , Rahul Kumar , Kushal Sharma
, Yash Jaiswal**

YEAR: 2018

In this paper, the proposed technique firstly applied Machine Learning and Nearest Neighbor classifier filter to remove the effect of high density noise in retinal images. Identifies the presence of disease by applying ensemble of machine learning algorithms .This experiments does not have clear solution in sensitivity, accuracy and error rate.

**TITLE: DIAGNOSIS OF DIABETIC RETINOPATHYUSING MACHINE
LEARNING CLASSIFICATION ALGORITHM**

AUTHOR: Karan Bhati , Shikhar Arora, Ravi Tomar

YEAR: 2016

This review paper focuses on decision about the presence of disease by applying ensemble of machine learning algorithms on features extracted from output of different retinal image processing algorithms, like diameter of optic disk, lesion specific (microaneurysms, exudates), image level (prescreening,AM/FM, quality assessment). By using alternating decision tree, adaBoost, Naïve Bayes, Random Forest and SVM , Decision making for predicting the presence of diabetic retinopathy was performed.

TITLE: AN INTEGRATED APPROACH FOR DIABETIC RETINOPATHY EXUDATE SEGMENTATION BY USING GENETIC ALGORITHM AND SWITCHING MEDIAN FILTER

AUTHOR: Amanjot Kaur, Prabhpreet Kaur

YEAR: 2016

In this paper, the proposed technique firstly applied switching median filter to remove the effect of high density noise in retinal images and then genetic algorithm will come in action to locate exudates in these images. This experimental result have clearly shown that the proposed technique outperforms over the available techniques in terms of sensitivity, accuracy and error rate.

TITLE: RETINOPATHY DETECTION USING DEEP CONVOLUTIONAL NEURAL NETWORKS

AUTHOR: Darshit Doshi, Aniket Shenoy, Deep Sidhpura, Prachi Gharpure

YEAR: 2016

This paper presents the design and implementation of GPU accelerated deep convolutional neural networks. This automatically diagnose and thereby classify high-resolution retinal images into 5 stages of the disease based on severity.

TITLE: AUTOMATED DETECTION OF DIABETIC RETINOPATHY USING FLUORESCIN ANGIOGRAPHY PHOTOGRAPHS

AUTHOR: Marco Alban, Tanner Gilligan

YEAR: 2016

State-of-the-art convolutional neural networks (CNNs) and denoising techniques were used to diagnose the presence and severity of Diabetic Retinopathy from Fluorescein Angiography photographs. Data was provided by Eye- Pacs consisting of fudus

photographs with varying ranges of DR severity labeled by clinicians. A convolutional neural network classifier engineered from GoogLeNet for 5-class severity classification performed best with an AUC of 0.79% and an accuracy of 0.45%. This paper improves on past work in the scale and heterogeneity of the dataset used, to the best of our knowledge, no other published work on DR screening deals with a dataset of our magnitude.

TITLE: TRANSFER LEARNING BASED DETECTION OF DIABETIC RETINOPATHY FROM SMALL DATASET,

AUTHOR: Shri Kant, Misgina Tsighe Hagos

YEAR: 2013

Annotated training data insufficiency remains to be one of the challenges of applying deep learning in medical data classification problems. Transfer learning from an already trained deep convolutional network can be used to reduce the cost of training from scratch and to train with small training data for deep learning. This raises the question of whether we can use transfer learning to overcome the training data insufficiency problem in deep learning based medical data classifications. Deep convolutional networks have been achieving high performance results on the ImageNet Large Scale Visual Recognition Competition (ILSVRC) image classification challenge.

TITLE: DETECTION OF DIABETIC RETINOPATHY USING DEEP LEARNING METHODOLOGY

AUTHOR: Gazala Mushtaqand Farheen Siddiqui

YEAR: 2015

Diabetic retinopathy is a complication of diabetes that targets the eyes by damaging the retinal blood vessels. Initially it is asymptomatic or causes fluctuating vision problems. As it becomes severe, it affects both the eyes and eventually causes

partial or complete vision loss. Primarily occurs when the blood sugar level is unmanageable. Therefore, the person with diabetes mellitus is always at a high risk of acquiring this disease. The early detection can deter the contingency of complete and permanent blindness. Thus, requires an efficient screening system.

**TITLE: DIAGNOSIS OF DIABETIC RETINOPATHY USING
MORPHOLOGICAL PROCESS AND SVM CLASSIFIER**

AUTHOR: Mahendran Gandhi, Dhanasekaran

YEAR: 2013

This paper focuses on automatic detection of diabetic retinopathy through detecting exudates in colour fundus retinal images and also classifies the rigorousness of the lesions. The paper firstly applied the Supply vector Machine is used to identify the Diabetic retinopathy in its early stages but there is no clear solution as it shows the incorrect accuracy level and error rates are also increased.

CHAPTER – 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

All of existing methods are good in some measures for detection and Segmentation of exudates but still raise some problems with low intensity, low accuracy, less color contrast and sensitivity, non-uniform illumination images. Therefore, our proposed algorithm and techniques have ability to solve these problems by preprocessing techniques. All process is demonstrated and step-wise details are explained below.

Step 1: First of all, take a diabetic RGB human retinal image

Step 2: Apply Gaussian Filter to remove noise from image.

Step 3: Convert the RGB image into grey scale level

Step 4: Apply CNN Algorithm to this image.

Step 5: Compare the Resulted image with Test data.

Step 6: Detect the stages.

Step 7: Finally higher values of accuracy, sensitivity and lower value of error rate are obtained.

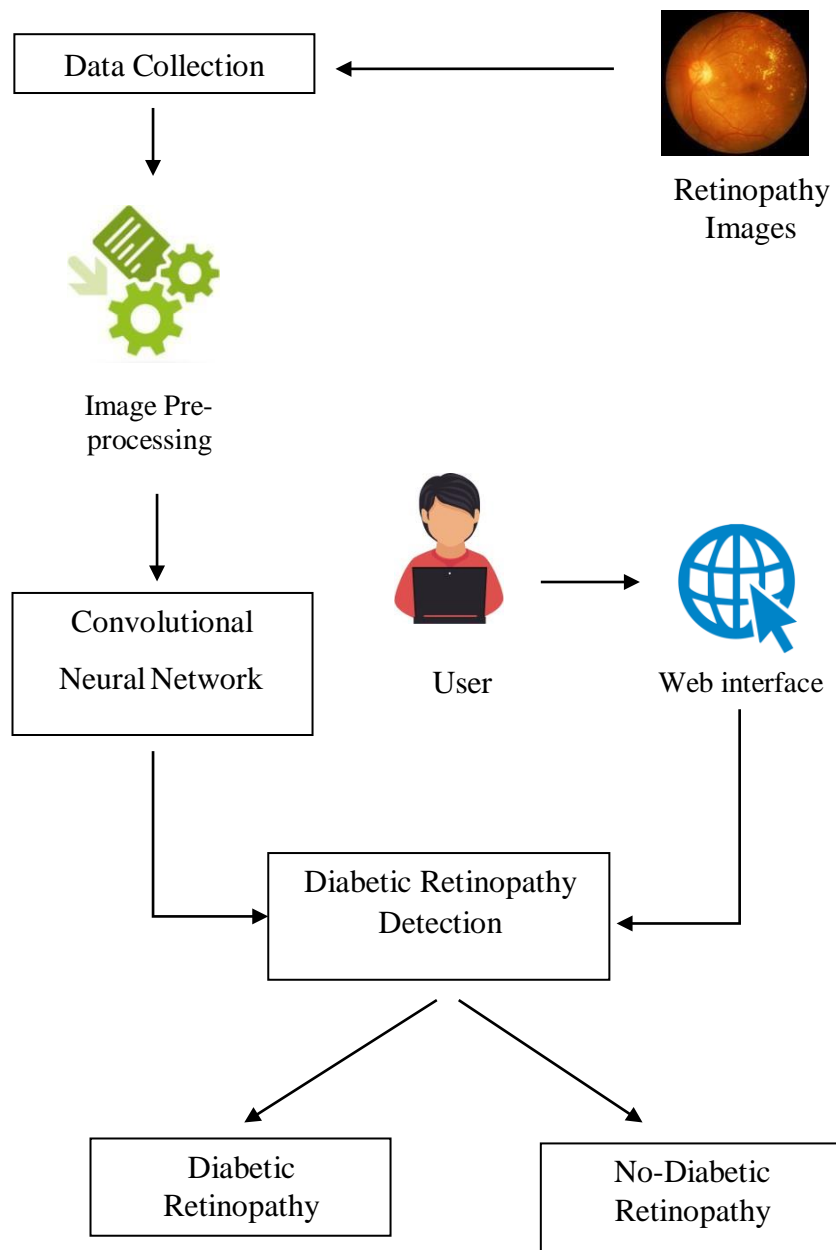


Fig 4.1 System Architecture

4.2 SEQUENCE DIAGRAM:

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

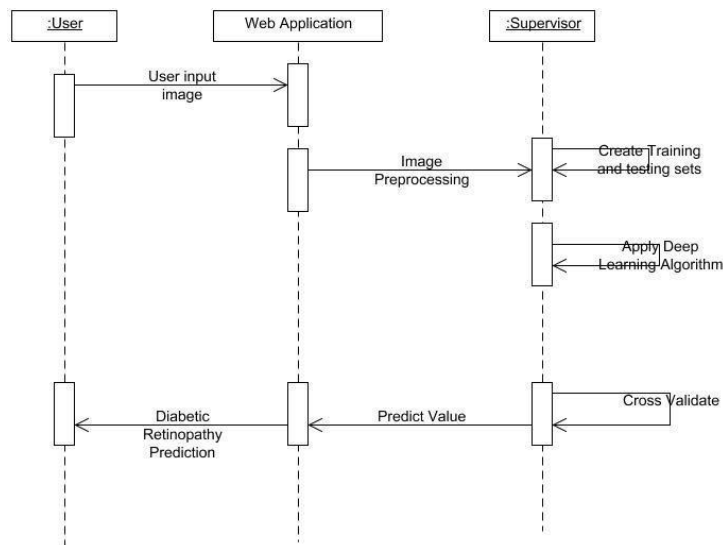


Fig 4.2 Sequence diagram

4.3 UML DIAGRAM:

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development.

4.4 USECASE DIAGRAM

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

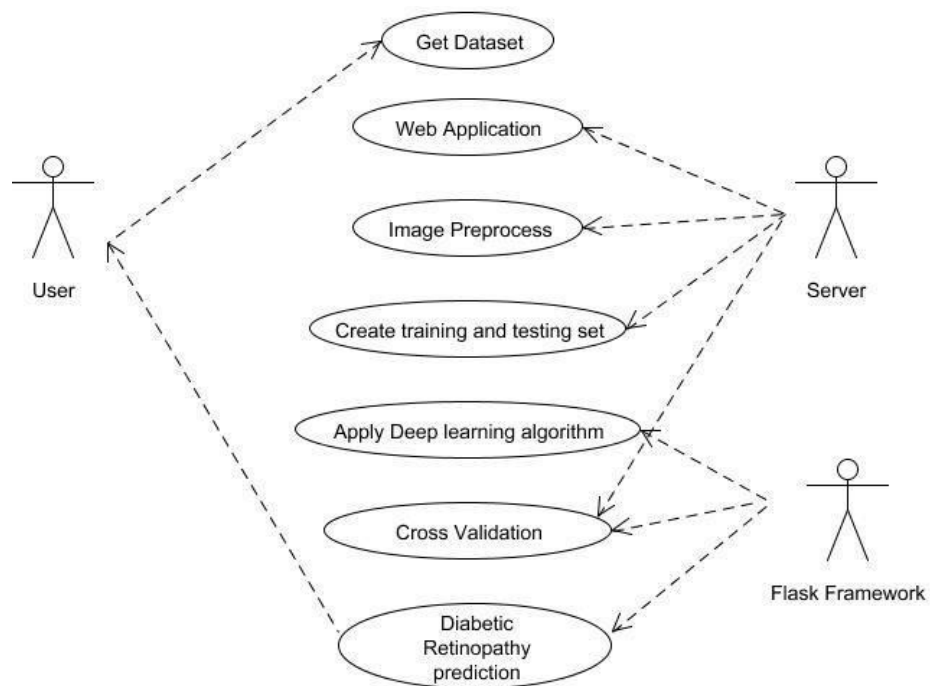


Fig 4.3 Use Case Diagram

4.4 ACTIVITY DIAGRAM:

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.
- An encircled circle represents the end of the workflow.

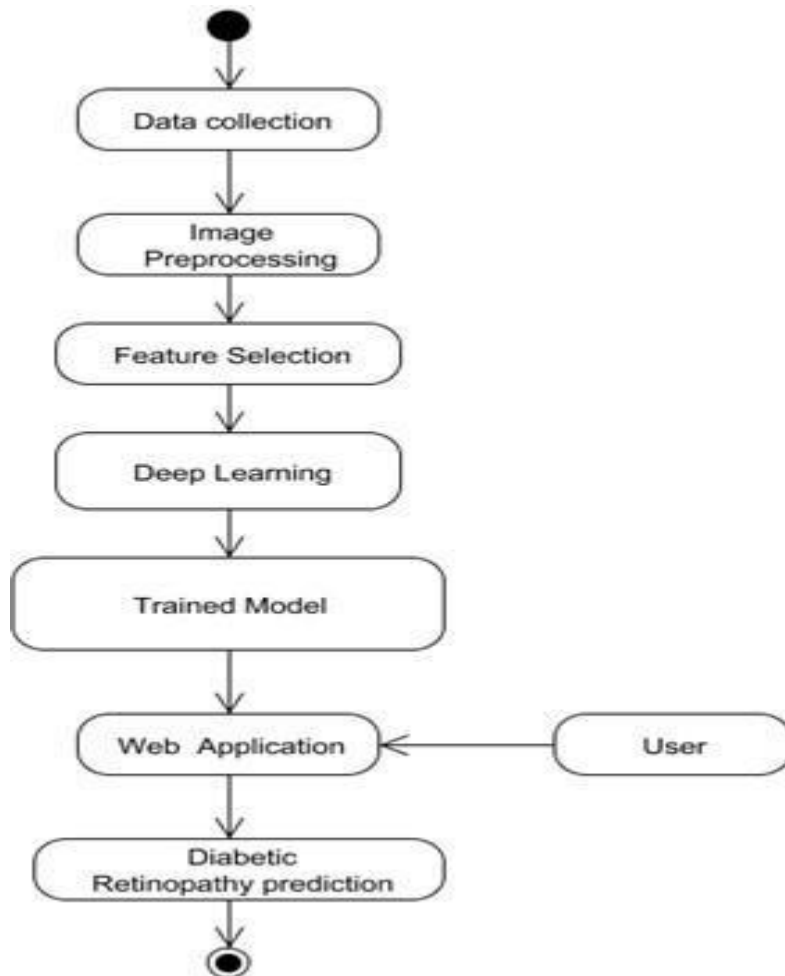


Fig 4.4 Activity diagram

4.5 COLLABORATION DIAGRAM:

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and object.

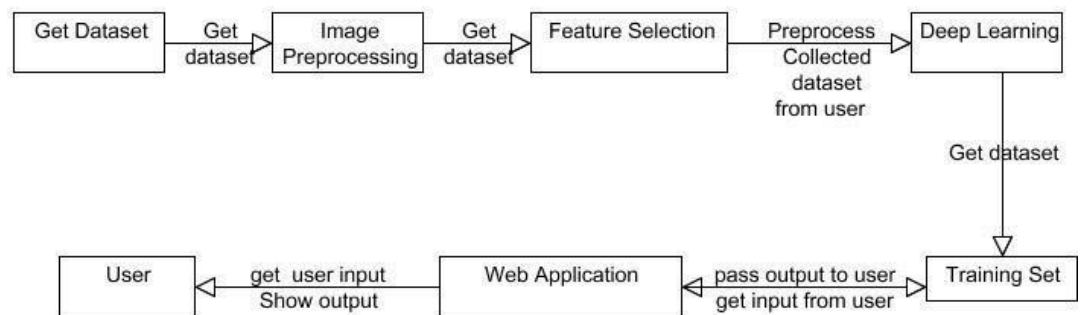


Fig 4.5 Collaboration Diagram

CHAPTER-5

SYSTEM IMPLEMENTATION

5.1 MODULE DESCRIPTION

A **module description** provides detailed information about the **module** and its supported components, which is accessible in different manners.

MODULES

- Data Collection
- Convolution Neural network algorithm
- Prediction

5.1.1 DATA COLLECTION AND PREPARATION

Data were drawn from a dataset provided via Kaggle. The dataset used is highly heterogeneous. The Kaggle DR Detection mission dataset includes color fundus photographs. We have reduced the DR classification into binary lessons. A smaller subset, of size 3662 fundus images, of the publicly available EyePacs dataset that is uploaded on Kaggle DR Detection challenge was used for model training and testing.

5.1.2 CONVOLUTIONAL NEURAL NETWORK

Convolution neural network is a subset of deep learning neural network. It is mainly used for image classification and image analysis. The goal behind CNN is to mimic how human brain analyzes the image. Convolution neural network is comprised of one or more Convolutional layers and then followed by one or more fully connected layers. The CNN consist of input, hidden and output layer. The input

layer is basically consists of arrays of pixels. The hidden layer is the most important layer as it plays the main role in image computation. Hidden layer comprises of activation functions and biases. The output layer helps us to determine the class score. The benefit of CNN's is that they are easier to train with providing high accuracies.

5.1.3 PREDICTION:

Preprocessed data are trained by DL algorithm and input given by the user from web application, goes to the trained dataset. After prediction the predict value Shown as an output on web application.

CHAPTER-6

CODINGS

This chapter describes about the coding which is used to implement this project and the outcome of the project.

6.1 SOURCE CODE

6.1.1 INDEX.HTML

```
<head>
  <title>Home page</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" type="text/css" href="aa/aa.css">
  <link rel="stylesheet" type="text/css" href="aa/an.js">
</style>
body { font-family: Arial, Helvetica, sans-serif;}
/* Full-width input fields */
input[type=text], input[type=password] {
width: 100%;
padding: 12px 20px;
margin: 8px 0;
display: inline-block;
border: 1px solid #ccc;
box-sizing: border-box;
}
/* Set a style for all buttons */
```



```

button {
    background-color: #23D9C6;
    color: white;
    padding: 14px 20px;
    margin: 8px 0;
    border: none;
    cursor: pointer;
    width: 100%;
}

button:hover {
    opacity: 0.8;
}

/* Extra styles for the cancel button */
.cancelbtn {
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}

/* Center the image and position the close button */
.imgcontainer {
    text-align: center;
    margin: 24px 0 12px 0;
    position: relative;
}

img.avatar {
    width: 40%;
    border-radius: 50%;
}

```

```

.container {
  padding: 16px;
}
span.psw {
  float: right;
  padding-top: 16px;
}
/* The Modal (background) */
.modal {
  display: none; /* Hidden by default */
  position: fixed; /* Stay in place */
  z-index: 1; /* Sit on top */
  left: 0;
  top: 0;
  width: 100%; /* Full width */
  height: 100%; /* Full height */
  overflow: auto; /* Enable scroll if needed */
  background-color: rgb(0,0,0); /* Fallback color */
  background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
  padding-top: 60px;
}
/* Modal Content/Box */
.modal-content {
  background-color: #fefefe;
  margin: 5% auto 15% auto; /* 5% from the top, 15% from the bottom and centered */
  border: 1px solid #888;
  width: 80%; /* Could be more or less, depending on screen size */
}

```

```

/* The Close Button (x) */
.close {
    position: absolute;
    right: 25px;
    top: 0;
    color: #000;
    font-size: 35px;
    font-weight: bold;
}
.close:hover,
.close:focus {
    color: red;
    cursor: pointer;
}
/* Add Zoom Animation */
.animate {
    -webkit-animation: animatezoom 0.6s;
    animation: animatezoom 0.6s
}
@-webkit-keyframes animatezoom {
    from {-webkit-transform: scale(0)}
    to {-webkit-transform: scale(1)}
}

@keyframes animatezoom {
    from {transform: scale(0)}
    to {transform: scale(1)}
}

```

```

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
    span.psw {
        display: block;
        float: none;
    }
    .cancelbtn {
        width: 100%;
    }
}
body, html {
    height: 100%;
    margin: 0;
}

.bg {
    /* The image used */
    background-image: url("{ {url_for('static', filename = 'Atlee.png')} }");

    /* Full height */
    height: 100%;

    /* Center and scale the image nicely */

    background-repeat: no-repeat;
    background-size: 1300px 500px;
}
</style>
<STYLE>A {text-decoration: none;} </STYLE>

```

```

</head>
<body link="white">
<marquee behavior="scroll" direction="left" onmouseover="this.stop();"
onmouseout="this.start();" style="background-color:#D6DDDF;font-size: xxx-large;"
>Diabetic Retinopathy Detection</marquee>

<div class="bg" >
    <center>
</br></br></br></br></br></br></br></br></br></br></br></br></br></br>
        <button
onclick="document.getElementById('id01').style.display='block'"
style="width:auto;">Login</button>
            <button onclick="document.getElementById('').style.display='block'"
style="width:auto;"><a href = "{ { url_for('register') }}" >Register</a></button>

    </center>
</div>
<div id="id01" class="modal">

    <form class="modal-content animate" action="/login" method="post">
        <div class="imgcontainer">
            <span onclick="document.getElementById('id01').style.display='none'"
class="close" title="Close Modal">&times;</span>
            
        </div>

        <div class="container">
            <label for="uname"><b>Username</b></label>

```

```
<input type="text" placeholder="Enter Username" name="username" required
value="{{ request.form.username }}">
```

```
<label for="psw"><b>Password</b></label>
```

```
<input type="password" placeholder="Enter Password" name="password"
required value="{{ request.form.password }}">
```

```
<button type="submit">Login</button>
```

```
<label>
```

```
<input type="checkbox" checked="checked" name="remember"> Remember me
</label>
```

```
</div>
```

```
<div class="container" style="background-color:#f1f1f1">
```

```
<button type="button"
```

```
onclick="document.getElementById('id01').style.display='none'"
```

```
class="cancelbtn">Cancel</button>
```

```
<span class="psw">Forgot <a href="#">password?</a></span>
```

```
</div>
```

```
</form>
```

```
{% if error % }
```

```
<p class="error"><strong>Error:</strong> {{ error }}</p>
```

```
{% endif % }
```

```
</div>
```

```
<script>
```

```
// Get the modal
```

```
var modal = document.getElementById('id01');
```

```
var modal = document.getElementById('id02');
```

```
// When the user clicks anywhere outside of the modal, close it
window.onclick = function(event) {
    if (event.target == modal) {
        modal.style.display = "none";
    }
}
</script>

</body>
</html>
```

6.1.2 REGISTRATION.HTML

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
    font-family: Arial, Helvetica, sans-serif;
    background-color: black;
}

* {
    box-sizing: border-box;
}

/* Add padding to containers */
```

```

.container {
  padding: 16px;
  background-color: white;
}
/* Full-width input fields */
input[type=text], input[type=password] {
  width: 100%;
  padding: 15px;
  margin: 5px 0 22px 0;
  display: inline-block;
  border: none;
  background: #f1f1f1;
}
input[type=text]:focus, input[type=password]:focus {
  background-color: #ddd;
  outline: none;
}

/* Overwrite default styles of hr */
hr {
  border: 1px solid #f1f1f1;
  margin-bottom: 25px;
}

/* Set a style for the submit button */
.registerbtn {
  background-color: #4CAF50;
  color: white;
  padding: 16px 20px;

```



```

margin: 8px 0;
border: none;
cursor: pointer;
width: 100%;
opacity: 0.9;
}
.registerbtn:hover {
    opacity: 1;
}
/* Add a blue text color to links */
a {
    color: dodgerblue;
}
/* Set a grey background color and center the text of the "sign in" section */
.signin {
    background-color: #f1f1f1;
    text-align: center;
}
</style>
</head>
<body>
<form action="{ {url_for('register')}}" method = "POST">
    <div class="container">
        <h1>Register</h1>
        <p>Please fill in this form to create an account.</p>
        <hr>

        <label for="name"><b>Name</b></label>
        <input type="text" placeholder="Name" name="name" required>

```

```

    <label for="email"><b>Email</b></label>
    <input type="text" placeholder="Enter Email" name="username" required>

    <label for="password"><b>Password</b></label>
    <input type="password" placeholder="Enter Password" name="password"
required>
    <hr>
    <p>By creating an account you agree to our <a href="#">Terms &
Privacy</a>.</p>
    <button type="submit" class="registerbtn">Register</button>
</div>
<div class="container signin">
    <p>Already have an account? <a href="{ { url_for('login') } }">Sign in</a>.</p>
</div>
</form>
</body>
</html>

```

6.1.3 LOGIN.HTML

```

<html>
<head>
<title>login page</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="static/bootstrap.min.css" rel="stylesheet" media="screen">
</head>
<body>
<div class="container">
<h1>Please login</h1>

```

```

<br>
<form action="/login" method="post">
<input type="text" placeholder="Username" name="username" value="{{
request.form.username }}">
<input type="password" placeholder="Password" name="password" value="{{
request.form.password }}">
<input class="btn btn-default" type="submit" value="Login">
</form>
{% if error %}
<p class="error"><strong>Error:</strong> {{ error }}
{% endif %}
</div>
</body>
</html>

```

6.1.4 FILE_UPLOAD_FORM.HTML

```

<html>
<head>
  <title>upload</title>

<style>
body, html {
  height: 100%;
  margin: 0;
}
.bg {
  background-image: url('{{ url_for('static',filename = 'diabetic.jpg') }}');
  height: 100%;

```

```

background-position: center;
background-repeat: no-repeat;
background-size: 1300px 500px;
}
</style>
</head>
<div class="container">
<div class="bg">
<h1><span><center><font color="red">Diabetic Retinopathy Detection
</font><center></span></h1>
<body><br><br><br><br><br><br>
<body >

<form action = "/success" method = "post" enctype="multipart/form-data">
<center><input type="file" name="file" /> </center><br>
<center> <input type = "submit" value="Upload"> </center>
</form>
</div>
</div>
</body>
</html>

```

6.1.5 SUCCESS.HTML

```

<html>
<head>
<title>success</title>

<style>

```

```

body, html {
    height: 100%;
    margin: 0;
}
.bg {
    background-image: url('{{url_for('static',filename = 'view.jpg')}}');
    height: 100%;
    background-position: center;
    background-repeat: no-repeat;
    background-size: 1400px 500px;
}
</style>
</head>
<div class="container">
<div class="bg">
    <h1><span><center><font color="red">Diabetic Retinopathy Detection
</font><center></span></h1>
    <body><br><br><br><br><br><br>
<body>
<p style="color:red">File uploaded successfully</p>

        {% for i in list2 %}
        <h1 style="color:black">{{i}}<h1>
        {% endfor %}

</div>
</div>
</body>

```

</html>

6.1.6 DIABETIC_RETINOPATHY_WEB.PY

```
from flask import *
app = Flask(__name__)
import tensorflow as tf, sys
import tensorflow.compat.v1 as tf
tf.disable_v2_behavior()

import os
import sqlite3 as sql
import base64
from flask import Flask ,render_template,request,jsonify,session
from flask import Flask,abort,render_template,request,redirect,url_for
#from werkzeug import secure_filename
#Flask Libraries
from flask import Flask, redirect, url_for, render_template, request
import requests
import urllib
@app.route('/', methods=['GET', 'POST'])
def home():
    return render_template('index.html')

def validate(username,password):
    con = sql.connect('static/chat.db')
    completion = False
    with con:
        cur = con.cursor()
```

```

cur.execute('SELECT * FROM persons')
rows = cur.fetchall()
for row in rows:
    dbuser = row[1]
    dbpass = row[2]
    if dbuser == username:
        completion = (dbpass == password)
return completion

```

```

@app.route('/login', methods=['GET', 'POST'])
def login():
    error = None
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        completion = validate(username,password)
        if completion == False:
            error = 'invalid Credentials. please try again.'
        else:
            session['username'] = request.form['username']
            return render_template('file_upload_form.html')
    return render_template('file_upload_form.html', error=error)

```

```

@app.route('/register', methods = ['GET','POST'])
def register():
    if request.method == 'POST':
        try:

```

```

name = request.form['name']
username = request.form['username']
password = request.form['password']
with sql.connect("static/chat.db") as con:
    cur = con.cursor()
    cur.execute("INSERT INTO persons(name,username,password) VALUES
(?,?,?)",(name,username,password))
    con.commit()
    msg = "Record successfully added"
except:
    con.rollback()
    msg = "error in insert operation"
finally:
    return render_template("index.html",msg = msg)
    con.close()
return render_template('register.html')

```

```

@app.route('/list')
def list():
    con = sql.connect("static/chat.db")
    con.row_factory = sql.Row

    cur = con.cursor()
    cur.execute("select * from persons")

    rows = cur.fetchall();
    return render_template("list.html",rows = rows)

```



```

@app.route('/upload')
def upload():
    return render_template("file_upload_form.html")

@app.route('/success', methods = ['POST'])
def success():
    if request.method == 'POST':
        f = request.files['file']
        # change this as you see fit
        #image_path = sys.argv[1]
        image_path = f.filename

        # Read in the image_data
        image_data = tf.gfile.FastGFile(r"./test/"+str(image_path), 'rb').read()

        # Loads label file, strips off carriage return
        label_lines = [line.rstrip() for line
                        in tf.gfile.GFile("models/retrained_labels.txt")]

        # Unpersists graph from file
        with tf.gfile.FastGFile("models/retrained_graph.pb", 'rb') as f:
            graph_def = tf.GraphDef()
            graph_def.ParseFromString(f.read())
            _ = tf.import_graph_def(graph_def, name=")

        with tf.Session() as sess:
            # Feed the image_data as input to the graph and get first prediction
            softmax_tensor = sess.graph.get_tensor_by_name('final_result:0')

```

```

predictions = sess.run(softmax_tensor,
                        {'DecodeJpeg/contents:0': image_data})

# Sort to show labels of first prediction in order of confidence
top_k = predictions[0].argsort()[-len(predictions[0]):][::-1]
print(top_k)
m=[]
for node_id in top_k:
    human_string = label_lines[node_id]
    score = predictions[0][node_id]
    for j in human_string,score:
        print(j)
        m.append(j)

    #print('%s (score = %.5f)' % (human_string, score))

return render_template("success.html", list2 = m)

if __name__ == '__main__':
    app.run(debug = True)

```

CHAPTER-7

RESULT AND DISCUSSION

7.1 HOME PAGE

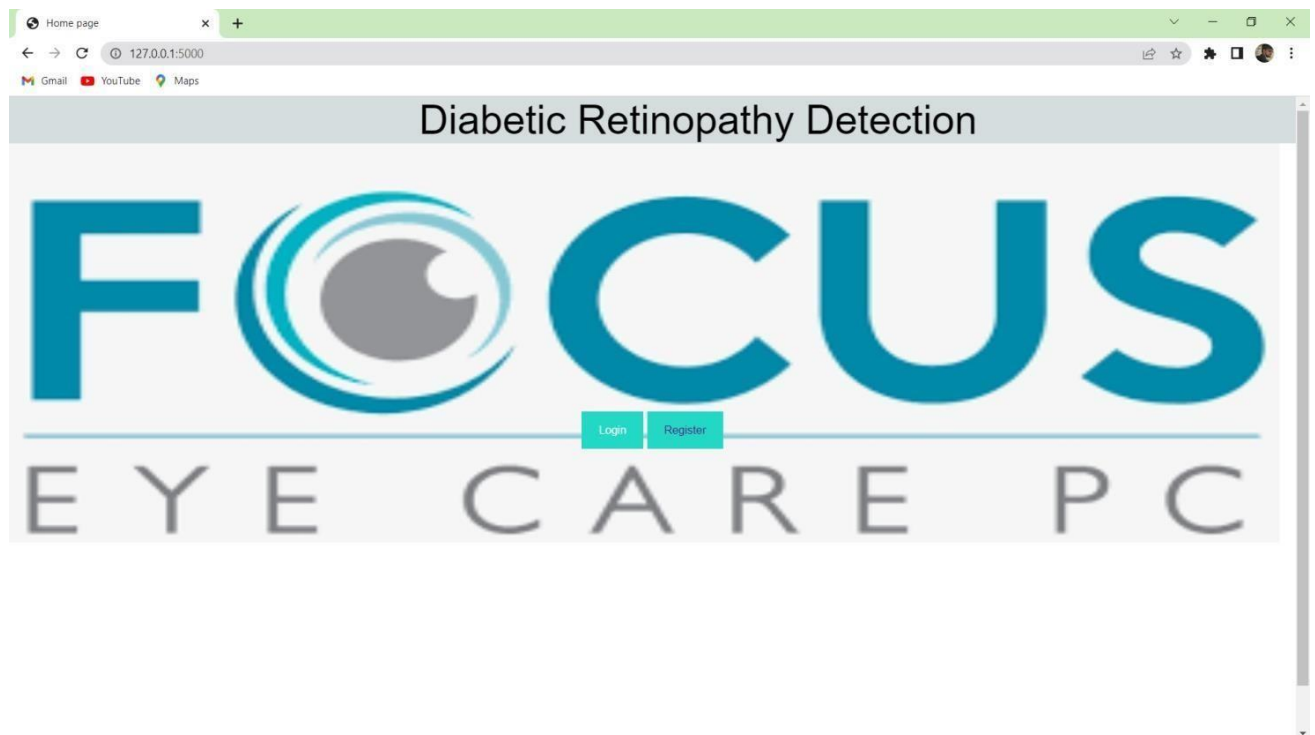
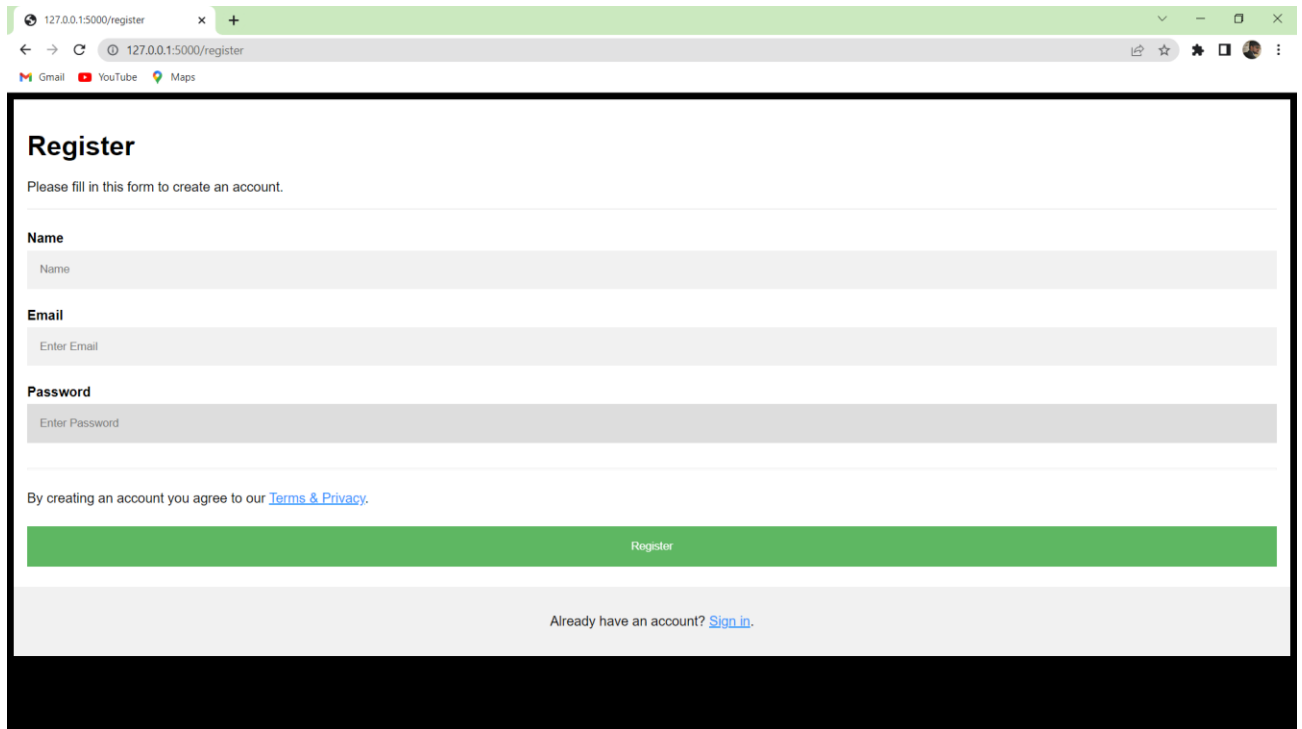


Fig 7.1 Home page

7.2 REGISTRATION PAGE



Register

Please fill in this form to create an account.

Name

Name

Email

Enter Email

Password

Enter Password

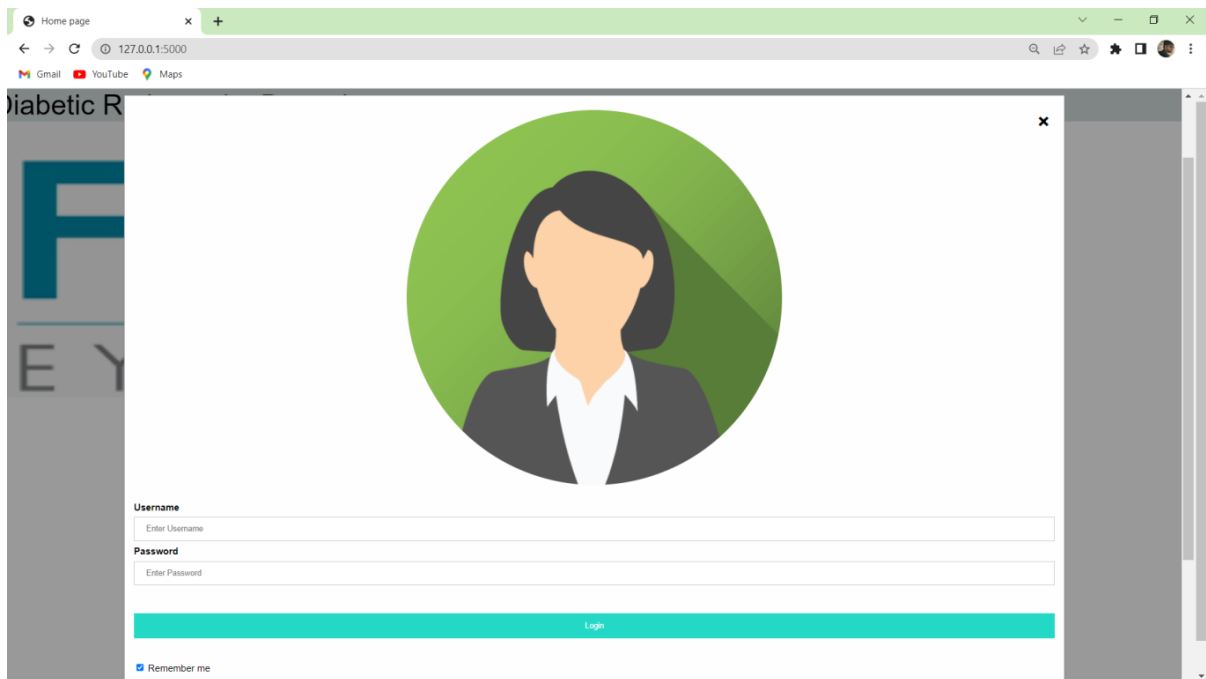
By creating an account you agree to our [Terms & Privacy](#).

Register

Already have an account? [Sign in](#).

Fig 7.2 Registration page

7.3 LOGIN PAGE



Diabetic R

F

E Y

Username

Enter Username

Password

Enter Password

Login

☒ Remember me

Fig 7.3 Login page

7.4 UPLOAD PAGE



Fig 7.4 Upload page

7.5 RESULT PAGE

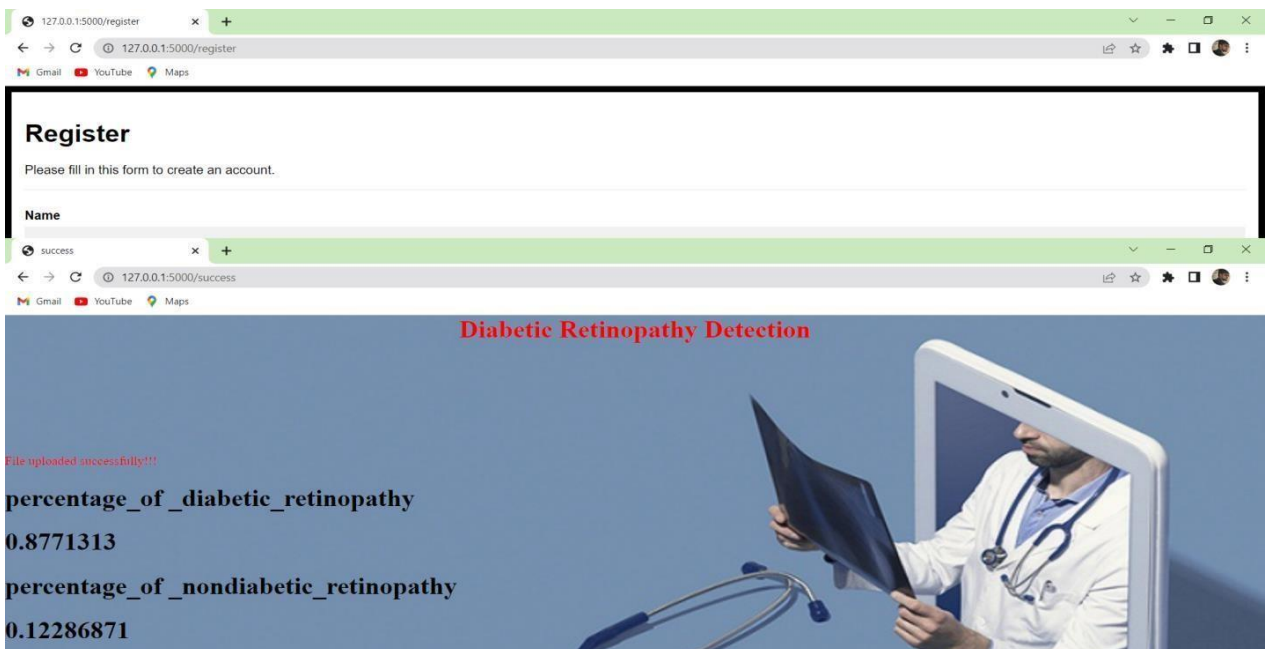


Fig 7.5 Result page

CHAPTER-8

CONCLUSION AND FUTURE SCOPE

This paper proposes the Binary Convolutional Neural Networks (BCNN) model, which performs memory- efficient classification in a shorter running time. This framework reduces the memory space by binary quantizing the weights and activations and speeds up the execution process using a hardware-friendly approach. Our model performs in restricted memory environments with massive images. Experiments conducted on Kaggle fundus images datasets proved the improvement in our method's scaling factor without much effecting the classification accuracy. This model achieves a 37.50% of reduction in memory usage and a 49.34% gain in runtime. Region importance can be measured with the help of a heatmap to improve the efficiency of the model detection capability. Experts may grade the annotations used in this data set.

8.1 FUTURE ENHANCEMENT

In the future, experiments will be conducted on real-time datasets for various real-time applications with advanced memory- efficient fast classification approaches to meet ground truth. Advanced lightweight models with low cost and high-performance embedded computing devices can be preferred for further enhancements.

CHAPTER-9

REFERENCE

- [1] Prabhu N, Bhoir D, Shanbhag N. Diabetic Retinopathy Screening using Machine Learning for Hierarchical Classification. International Journal of Innovative Technology and Exploring Engineering.. 2019;8(10):1943-8.
- [2] Tsighe Hagos M, Kant S. Transfer Learning based Detection of Diabetic Retinopathy from Small Dataset. arXiv e-prints. 2019 May;arXiv-1905.
- [3] Wang Z, Yang J. Diabetic retinopathy detection via deep convolutional networks for discriminative localization and visual explanation. arXiv preprint arXiv:1703.10757. 2017 Mar 31.
- [4] Li X, Hu X, Yu L, Zhu L, Fu CW, Heng PA. CANet: Cross-Disease Attention Network for Joint Diabetic Retinopathy and Diabetic Macular Edema Grading. IEEE transactions on medical imaging. 2019 Nov 6;39(5):1483-93.
- [5] Li YH, Yeh NN, Chen SJ, Chung YC. Computer-assisted diagnosis for diabetic retinopathy based on fundus images using deep convolutional neural network. Mobile Information Systems. 2019 Jan 1;2019.
- [6] Kanungo YS, Srinivasan B, Choudhary S. Detecting diabetic retinopathy using deep learning. In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) 2017 May 19 (pp. 801-804). IEEE.
- [7] Dai Y, Zhu C, Shan X, Cheng Z, Zou B. A Survey on Intelligent Screening for

Diabetic Retinopathy. Chinese medical sciences journal= Chung-kuo i hsueh k'o
hsueh tsa chih. 2019 Jun1;34(2):120-32.

[8]Szegedy, C., Toshev, A. and Erhan, D., 2013. Deep neural networks for object
detection.

[9]Zhao ZQ, Zheng P, Xu ST, Wu X. Object detection with deep learning: A
review. IEEE transactions on neural networks and learning systems. 2019 Jan
28;30(11):3212-32.

[10]Dosovitskiy and T.Brox. Inverting visual representations with convolutional
networks. InCVPR, 2016.

[11]A. Mahendran and A. Vedaldi. Understanding deep image representations by
inverting them. InCVPR, 2015.

[12] Alban M, Gilligan T. Automated detection of diabetic retinopathy using
fluoresceinangiography photographs. Report of standford education. 2016.

[13] Razzak MI, Naz S, Zaib A. Deep learning for medical image processing:
Overview, challenges and the future. Classification in BioApps. 2018:323-50.

[14] Alyoubi WL, Shalash WM, Abulkhair MF. Diabetic retinopathy detection
through deeplearning techniques: A review. Informatics in Medicine
Unlocked. 2020 Jun 20:100377.

[15] Mateen M, Wen J, Hassan M, Nasrullah N, Sun S, Hayat S. Automatic
detection of diabeticretinopathy: a review on datasets, methods and evaluation
metrics. IEEE Access. 2020 Mar 11;8:48784-811.

- [16] Pao SI, Lin HZ, Chien KH, Tai MC, Chen JT, Lin GM. Detection of Diabetic Retinopathy Using Bichannel Convolutional Neural Network. *Journal of Ophthalmology*. 2020 Jun 19;2020.
- [17] Liu H, Yue K, Cheng S, Pan C, Sun J, Li W. Hybrid Model Structure for Diabetic Retinopathy Classification. *Journal of Healthcare Engineering*. 2020 Oct 13;2020.
- [18] Gayathri S, Gopi VP, Palanisamy P. A lightweight cnn for diabetic retinopathy classification from fundus images. *Biomedical Signal Processing and Control*. 2020 Sep 1;62:102115.
- [19] Courbariaux M, Bengio Y, David JP. Binaryconnect: Training deep neural networks with binary weights during propagations. *arXiv preprint arXiv:1511.00363*. 2015 Nov 2.
- [20] Kaggle dataset [Online]. Available, <https://kaggle.com/c/diabetic-retinopathy-detection>

