**EFFICIENT CLASSIFICATION OF DIABETIC RETINOPATHY USING BINARY CNN**

**Alternate Title:**

Automated Diabetic Retinopathy Detection Using Convolutional Neural Network

**Aim:**

To detect the diabetic retinopathy disease in the earlier stage using Deep learning method

**Abstract:**

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. Aim of the project is to provide an automated, suitable and sophisticated approach using Convolutional Neural Network (CNN)

**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.

**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 logins 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.

**Modules:**

* Data Collection
* Convolution Neural network algorithm
* Prediction

**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.

**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.

**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.

**Software Requirements:**

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

**Hardware Requirements:**

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

**Architecture Diagram:**

User

Web interface

Retinopathy Images

Image Pre-processing

Convolutional Neural Network

Diabetic Retinopathy

Detection

No-Diabetic Retinopathy

Diabetic Retinopathy

Data Collection

