WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Detecting blindness in diabetic individuals using machine learning.

K. Sai Charan[1] V. Pardhiv Aryan[2]

[1.]UG Student Department of Computer Science Engineering, SRM UNIVERSITY Vijayawada,AP,India

[2.] UG Student Department of Computer Science Engineering, MIC college of technology, Vijayawada,AP,India

ABSTRACT

Many people suffer from diabetic retinopathy, which is most common among working-age adults. The resources available to detect diabetic retinopathy are limited, and in rural regions, it is quite challenging. Several scientific and medical procedures are available to screen and detect this disease, although the majority of detections are made utilizing retinal fungal imaging. The goal is to have the ability to automatically screen photos for diabetic retinopathy. This will be accomplished by developing a mobile application that uses a machine learning model and fundus pictures to assess the patient's level of blindness based on an examination of the eye image. This technique will help to reduce time while screening for diabetic retinopathy.

Keywords : Diabetic retinopathy, Machine learning, Fundus photographs, Android application.

Introduction

Diabetes is a chronic organ ailment characterized by high blood sugar levels. diabetes retinopathy is a primary cause of vision loss in diabetes individuals and results from long-term damage to the blood vessels of the human eye. Diabetes is often classified into two types: non-proliferative and proliferative diabetic retinopathy. The first phase of DR is NPDR, in which blood vessels begin to deteriorate and leak more fluid into the eye. This phase comprises microaneurysms (red dots), exudates (fatty tissue drops), and hemorrhages (small blood spots that leak into the retina). Proliferate Diabetic Retinopathy is a condition in which the retina's blood veins shut, reducing blood flow. PDR causes deterioration to both peripheral and central vision, resulting in significant vision loss. AI is a potentially scalable solution for DR screening. It serves to lessen the manual strain on ophthalmologists and overcome the barriers associated with teleophthalmology. Recent breakthroughs in machine learning and convolutional neural networks have enabled large-scale data analysis, pattern recognition, and report generation. AI algorithms created for DR screening (such as Google AI, EyeArt, and IDxDR) run on cloud-based platforms. The collected photographs are submitted online, and the algorithm returns an output within a reasonable amount of time. These methods are not widely used in low- and middle-income countries because to restricted internet access or capacity.In addition, most cameras integrated with AI software are the traditional expensive, large fundus cameras which require the operator to capture a dilated retinal image.

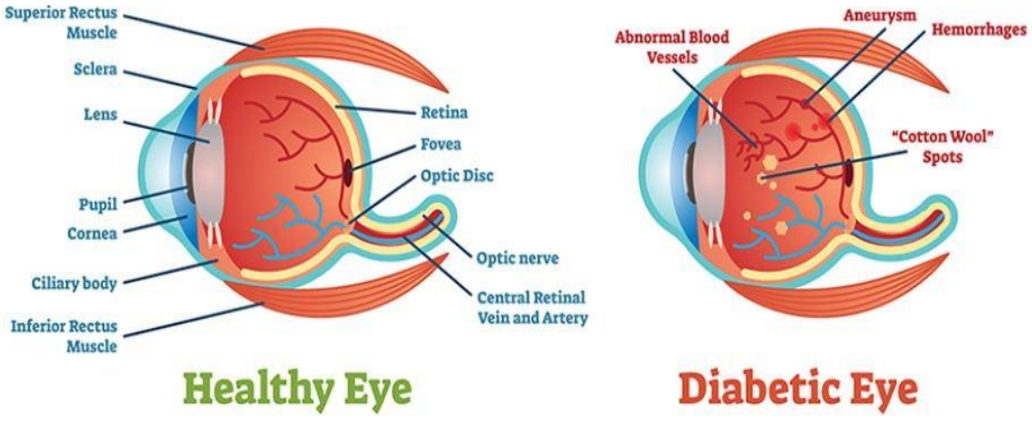


Fig. 1. DR vs Non-DR

Existing Methodology vs Proposed Methodology

Existing work	Proposed work
In previous research, implementation was done using Inception version 3.	Here, the implementation is done using ResNet50.
Overall output accuracy is 88.4%	Here, overall output accuracy is gained to 98.8%
In existing, the dataset used is of 400 images which is limited.	Here, the image dataset is increased to around 3500 images which helps in predicting more accurately.
This is implemented using C language.	This is implemented using Python language.
For android application, tensorflow lite is used and for IOS, core ML model is used.	Here, tensorflow lite is used for android application as well as IOS.

Fig. 2. Differentiation of proposed methodology compared to existing methodology

Implementation

Stages of DR in proposed model

1. No DR
2. Mild DR
3. Moderate DR
4. Severe DR
5. Proliferative DR

Labeling of DR

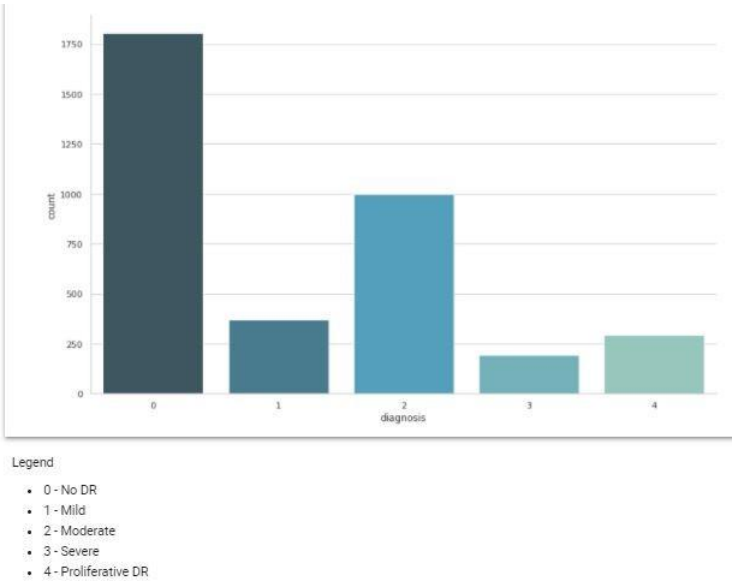


Fig. 3.Labeling of DR

Methodology

This representation illustrates the labeling of diabetic retinopathy as well as the number of photos per class. The dataset represents the input dataset provided to the system. The standard scalar approach is used to resize and reset many images at a single scale. PCA is used to extract features from photos. The next stage demonstrates image categorization using ResNet50. The output will display the detected diabetic retinopathy.

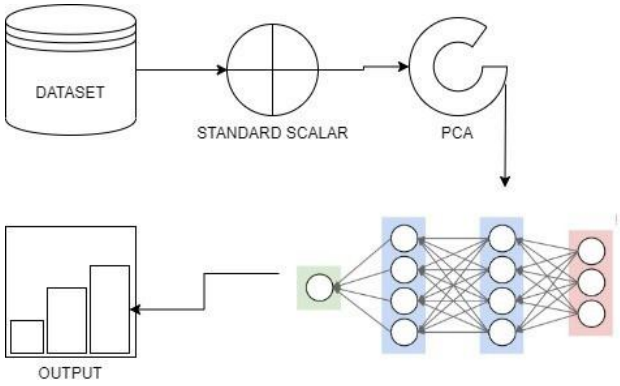


Fig. 4.Methodology

Confusion Matrix

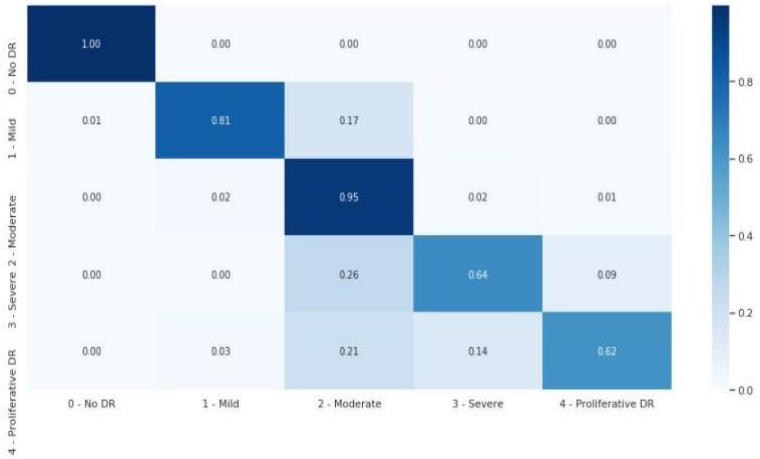


Fig. 5. Confusion Matrix

This picture is given to provide a better understanding of the confusion matrix, which contains four primary parameters.

TP (True Positive) - If the input image is No DR and it extracts features from No DR and the model predicts No DR, it is a true positive.

FP (False Positive) - If the given input is No DR and it extracts the properties of No DR, but it also returns another result, it is a false positive.

FN (False Negative) - Here, the input image is No DR, and when given as input, it extracts features of moderate DR and returns anything other than NO DR. It's termed a false negative.

TN(True Negative) - Here, input image is No DR and when given as input, if it extracts features of moderate DR yet if it gives output of No DR then is called true negative.

Model Specifications

Model

```
[ ]  def create_model(input_shape, n_out):
         input_tensor = Input(shape=input_shape)
         base_model = applications.ResNet50(weights=None,
                                            include_top=False,
                                            input_tensor=input_tensor)
         base_model.load_weights('../input/resnet50/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5')

         x = GlobalAveragePooling2D()(base_model.output)
         x = Dropout(0.5)(x)
         x = Dense(2048, activation='relu')(x)
         x = Dropout(0.5)(x)
         final_output = Dense(n_out, activation='softmax', name='final_output')(x)
         model = Model(input_tensor, final_output)

         return model
```

```
[ ]  model = create_model(input_shape=(HEIGHT, WIDTH, CANAL), n_out=N_CLASSES)

     for layer in model.layers:
         layer.trainable = False

     for i in range(-5, 0):
         model.layers[i].trainable = True

     metric_list = ["accuracy"]
     optimizer = optimizers.Adam(lr=WARMUP_LEARNING_RATE)
     model.compile(optimizer=optimizer, loss="categorical_crossentropy",  metrics=metric_list)
     model.summary()
```

Fig. 6.Model Input

Train top layers

```
[ ]  STEP_SIZE_TRAIN = train_generator.n//train_generator.batch_size
     STEP_SIZE_VALID = valid_generator.n//valid_generator.batch_size

     history_warmup = model.fit_generator(generator=train_generator,
                                  steps_per_epoch=STEP_SIZE_TRAIN,
                                  validation_data=valid_generator,
                                  validation_steps=STEP_SIZE_VALID,
                                  epochs=WARMUP_EPOCHS,
                                  verbose=1).history

     Epoch 1/2
     366/366 [==============================] - 525s 1s/step - loss: 1.2831 - acc: 0.6376 - val_loss: 3.6946 - val_acc: 0.4629
     Epoch 2/2
     366/366 [==============================] - 374s 1s/step - loss: 0.8092 - acc: 0.7001 - val_loss: 4.2149 - val_acc: 0.4586
```

Fig. 7. Training

```
[ ]  for layer in model.layers:
         layer.trainable = True

     es = EarlyStopping(monitor='val_loss', mode='min', patience=ES_PATIENCE, restore_best_weights=True, verbose=1)
     rlrop = ReduceLROnPlateau(monitor='val_loss', mode='min', patience=RLROP_PATIENCE, factor=DECAY_DROP, min_lr=1e-6, verbose=1)

     callback_list = [es, rlrop]
     optimizer = optimizers.Adam(lr=LEARNING_RATE)
     model.compile(optimizer=optimizer, loss="binary_crossentropy",  metrics=metric_list)
     model.summary()
```

Fig. 8.Fine-tuning

Output

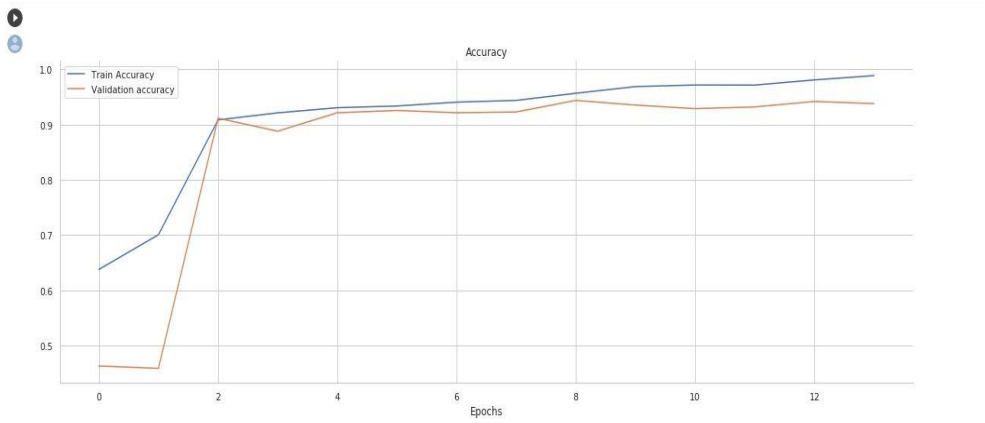


Fig. 9.This shows change in validation accuracy and training accuracy according to increase in epochs.

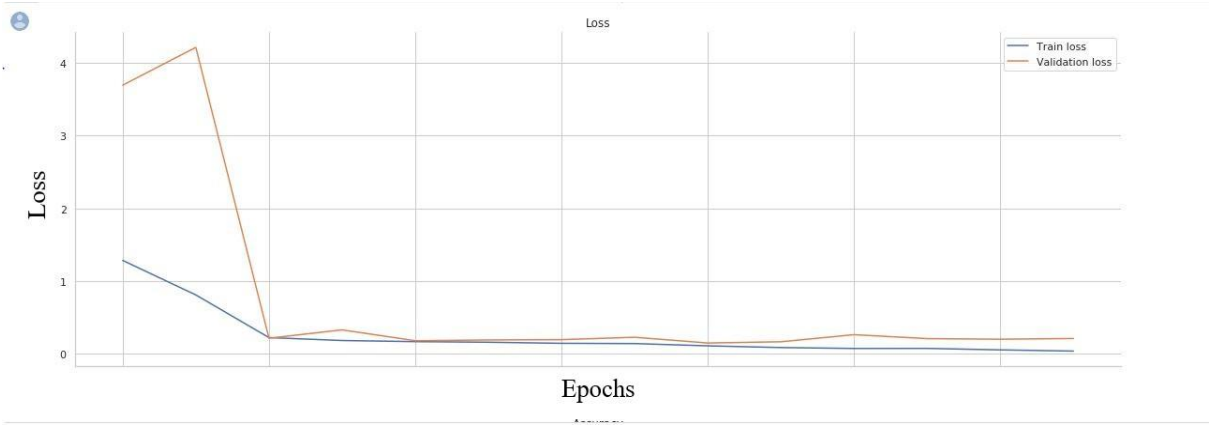
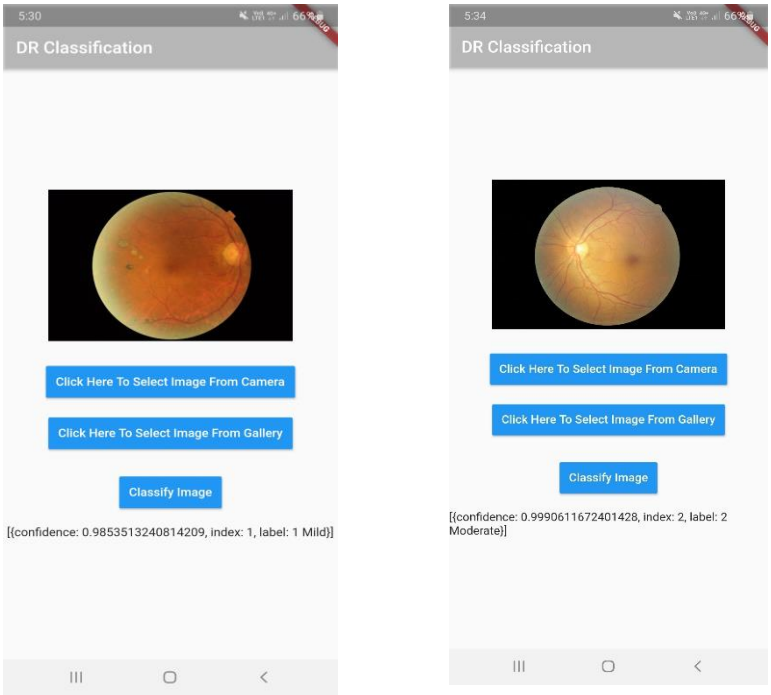


Fig. 10.This shows change in validation loss and training loss according to increase in epochs.

Screenshots of Mobile Application

Fig. 11.Screenshots of mobile application



Conclusion

Detected DR utilizing a mobile phone. ResNet50 is a convolutional neural network that classifies diabetic retinopathy into five stages. This model uses approximately 3500 photos for training, testing, and validation. After that, this model is transformed into a mobile application using Flutter, which can run on Android. This outcome yields 98.8 percent accuracy.

REFERENCES

1. Jianan Yuan , College of Computer Science Software EngineeringShenzhen University, Shenzhen, China, E-mail: yuanjiajiaqq@outlook.com, "Single-Label Image Classification", Review Paper.
2. Sanskruti Patel, Charotar University of Science and Technology, "A Comprehensive Analysis of Convo- lutional Neural Network Models", January 2020, International Journal of Advanced Science and Technology, 29(4):771-777.
3. Thippa Reddy Gadekallu1 · Neelu Khare1 · Sweta Bhattacharya1 · Saurabh Singh2 ·Praveen Kumar Reddy Maddikunta1 · Gautam Srivastava3,4,5, "Deep neural networks to predict diabetic retinopathy" , Received: 18 January 2020 / Accepted: 6 April 2020© Springer-Verlag GmbH Germany, part of Springer Nature 2020.
4. Abhishek Samantaa, Aheli Sahaa, Suresh Chandra Satapathya,,Steven Lawrence Fernandes b, Yo-Dong Zhangc, "Automated detection of diabetic retinopathy using convolutional neural networks on a small dataset", 0167-8655/© 2020 Elsevier B.V. All rights reserved.
5. Maximilian W.M. Wintergerst,1 Divyansh K. Mishra, 2 Laura Hartmann,1 Payal Shah, 2Vinaya K. Konana,2
6. Pradeep Sagar, 2 Moritz Berger, 3 Kaushik Murali, 2Frank G. Holz, 1 Mahesh P. Shanmugam, FRCSEd,2,* Robert P. Finger, "Diabetic Retinopathy Screening Using Smartphone-Based Fundus Imaging in India", ISSN 0161-6420/20,2020 American Academy of Ophthalmology, Published by Elsevier Inc.
7. Nikita Kashyap Dr. Dharmendra Kumar Singh Dr. Girish kumar singh, "Mobile Phone Based Diabetic Retinopathy Detection System Using ANN-DWT", 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON)GLA University, Mathura, Oct 26-28, 2017.
8. Bhavana Sosale , Sosale Ramachandra Aravind, Hemanth Murthy, Srikanth Narayana, Usha Sharma, Sahana G V Gowda, Muralidhar Naveenam, "Simple, Mobile-based Artificial Intelligence Algorithm in the detection of Diabetic Retinopathy (SMART) study", BMJ Open Diab Res Care 2020;8:e000892. doi:10.1136/bmjdrc-2019-000892.
9. Sharmin Majumder, Yaroub Elloumi, Mohamed Akil, Rostom Kachouri, Nasser Kehtarnavaz, "A deep learning-based smartphone app for real-time detection of five stages of diabetic retinopathy", HAL Id: hal-02556991, April 2020.