Project Status Report

Diabetic Retinopathy Detection

10 March 2023

Amandip Padda (200455829)

Ramanpreet Singh (200384219)

Ihab Mohammad (200401862)

Abstract

The Blindness Deduction Project focuses on creating a computer-aided diagnosis system that leverages Convolutional Neural Network (CNN) algorithms for the early detection of Diabetic Retinopathy (DR), a leading cause of blindness among diabetic patients. The project involves the use of a large dataset of eye images obtained from Kaggle, which has been preprocessed and resized to a uniform size to enhance the performance of the CNN models. The project also involves the development of various CNN models using TensorFlow and Keras libraries, which will be trained on the preprocessed dataset. The performance of these models will be evaluated using different metrics, such as accuracy, sensitivity, and specificity. The ultimate goal of this project is to develop an accurate and efficient computer-aided diagnosis system that can assist healthcare professionals in the early detection of DR, thus reducing the risk of blindness among diabetic patients.

Project Status: On track

Summary

Diabetic Retinopathy Detection has come a long way from just initial concept idea. Our team has gone through intense research about this disease and we are continuing to learn about it. The initial idea was to think like an Ophthalmologist and consider Orbital X-Rays to be our guide in this process. We collected data from Kaggle and studied what makes a "bad" eye in comparison to "good" eye. We trained ourselves just like a machine to learn and detect Diabetic Retinopathy from Orbital Radiography. The next phase was to collect information from other models and how they perform in regards to similar datasets. We have already implemented an algorithm on a generic dataset that TensorFlow provides as a learning curve and as of right now we are building CNN for the Diabetic Retinopathy training dataset.

Main Problem

The main problem is to detect blindness in diabetic patients from test samples using Machine Learning and outputting it into 5 categories.

Other Problems

Dataset preprocessing need to be accurate. We are in the middle of figuring out on how to pre process training data; which does not affect accuracy.

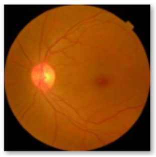
Algorithms used

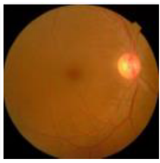
On the generic data provided by TensorFlow; we are using Supervised Learning Model because we do have both input and output labels. We are using Keras to import more functionality into our model. Our model is Sequential and we have trained generic dataset of 60,000 images; using 3 layers. The input layer is using "Flatten" and as a hyperparameter we are passing 28 by 28 shape image to collapse spatial dimensions into the channel dimension. The hidden layer is using "Dense" and as a hyperparameter we are passing 128 neural to connect the neural network deeply. For the hidden layer we are using 'relu' as an activation function. For the output layer we are passing 10 neural with activation function being 'softmax'. As for optimizer, loss and metrics; we are using 'adam', 'sparse_categorical_crossentropy', 'accuracy' respectively. We have trained the model only one time and the accuracy of the test set was 89.34%. This was only done on the generic dataset that was provided by TensorFlow.

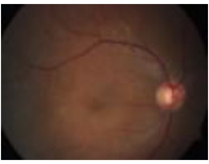
Explore

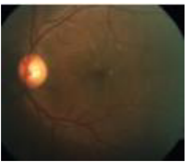
We will be exploring multiple directions in regards to this project. First, we will classify the data into 4 to 8 categories depending on the type of the image.

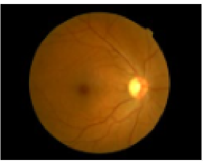
Here are some examples:

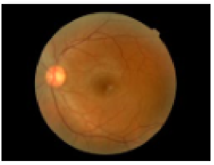

Class: Fit-Left


Class: Fit-Right


Class: Zoom-Right


Class: Zoom-Left

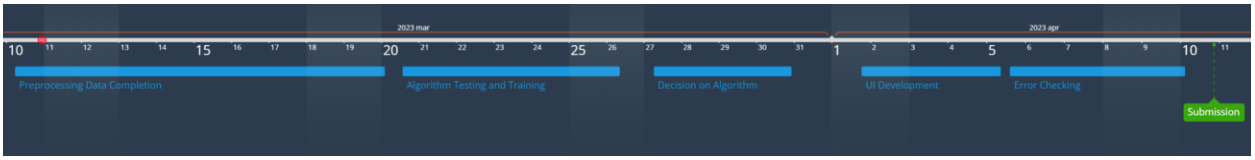

Class: Far-Right


Class: Far-Left

Once the images classify to appropriate sets, we will use Data Augmentation or Data Synthesis to increase the training set by 4 folds. This will help us increase the data set and train the machine at a much higher data sets. We will look into increasing the accuracy by toggling with display contrasts where we might increase the dataset at maximum 2 folds. Once the preprocessing has been done, we will continue to work with different hyperparameter values. The idea is to let the machine train by giving it different hypermeters and storing the result in an array to produce best possible accuracy.

We will look into Inception-v3[1], ResNet-50[4] and EfficientNet[2] and implement to see if it does increase the accuracy of the model.

Timeline of the project



Accomplishments

Group Members	Tasks
Amandip Padda	• Researched on the disease • Implemented CNN[3] using Keras • Set Neural Network using TensorFlow
Ramanpreet Singh	• Proposal of Idea • Implemented relu using Keras • Tried different loss functions and ended up with sparse_categorical_crossentropy
Ihab Mohammad	• Managed Datasets • Implemented adam using Keras • Implemented softmax as an output activation function

References

[1] Irla, T. (2019, October 8). Transfer Learning using Inception-v3 for Image Classification. Analytics Vidhya. https://medium.com/analytics-vidhya/transfer-learning-using-inception-v3-for-image-classification-86700411251b

[2] Mingxing, Tan and Quoc, V. Le. (2019, May 29). EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling. https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html

[3] Yamashita, R., Nishio, M., G, R. K., DO, & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. Insights Into Imaging, 9(4), 611–629. https://doi.org/10.1007/s13244-018-0639-9

[4] Khandelwal, R. (2021, December 11). Deep Learning using Transfer Learning -Python Code for ResNet50. Medium. https://towardsdatascience.com/deep-learning-using-transfer-learning-python-code-for-resnet50-8acdfb3a2d38