Machine Learning Engineer Nanodegree
Capstone Proposal
Winfred Selwyn Ooh
March 2018

Diabetic Retinopathy Detection

Identify signs of diabetic retinopathy in eye images

1.Domain Background

(source: **https://www.kaggle.com/c/diabetic-retinopathy-detection**)

Diabetic retinopathy is the leading cause of blindness in the working-age population of the developed world. It is estimated to affect over 93 million people.

The US Center for Disease Control and Prevention estimates that 29.1 million people in the US have diabetes and the World Health Organization estimates that 347 million people have the disease worldwide. Diabetic Retinopathy (DR) is an eye disease associated with long-standing diabetes. Around 40% to 45% of Americans with diabetes have some stage of the disease. Progression to vision impairment can be slowed or averted if DR is detected in time, however this can be difficult as the disease often shows few symptoms until it is too late to provide effective treatment.

Neural networks can will be used to automatic feature learning and classifier building. Images will be processed via a pipeline (i.e. labeled, resized and edited for model augmentation).

Similar work with Deep Convolutional Neural Networks have been done to detect anomalies in mammogram images for computer vision aided diagnosis.

- source : Pengcheng Xi, Chang Shu and Rafik Goubran, **“Abnormality Detection in Mammography using Deep Convolutional Neural Networks”** arxiv, 1803.01906, 2 Mar 2018
- link: https://arxiv.org/pdf/1803.01906.pdf

2. Problem Statement

Currently, detecting DR is a time-consuming and manual process that requires a trained clinician to examine and evaluate digital color fundus photographs of the retina. By the time human readers submit their reviews, often a day or two later, the delayed results lead to lost follow up, miscommunication, and delayed treatment.

Clinicians can identify DR by the presence of lesions associated with the vascular abnormalities caused by the disease. While this approach is effective, its resource demands are high. The expertise and equipment required are often lacking in areas where the rate of diabetes in local populations is high and DR detection is most needed. As the number of individuals with diabetes continues to grow, the infrastructure needed to prevent blindness due to DR will become even more insufficient.

The need for a comprehensive and automated method of DR screening has long been recognized, and previous efforts have made good progress using image classification, pattern recognition, and machine learning. With color fundus photography as input, the goal is to push an automated detection system to the limit of what is possible – ideally resulting in models with realistic clinical potential.

This problem is a **classification problem** within the domain of computer aided analysis and medical diagnostic imaging.

The model should be able to process an image and give it a score based on the following scale:

0 - No DR
1 - Mild
2 - Moderate
3 - Severe
4 - Proliferative DR

3. Datasets and Inputs

(source: https://www.kaggle.com/c/diabetic-retinopathy-detection/data)

Datasets for training and testing are provided by EyePACS, a free platform for retinopathy screening.

Dataset sizes:

- training: 32GB
- test: 49GB

The images in the dataset come from different models and types of cameras, which can affect the visual appearance of left vs. right. Some images are shown as one would see the retina anatomically (macula on the left, optic nerve on the right for the right eye). Others are shown as one would see through a microscope condensing lens (i.e. inverted, as one sees in a typical live eye exam). There are generally two ways to tell if an image is inverted:

- It is inverted if the macula (the small dark central area) is slightly higher than the midline through the optic nerve. If the macula is lower than the midline of the optic nerve, it's not inverted.
- If there is a notch on the side of the image (square, triangle, or circle) then it's not inverted.If there is no notch, it's inverted.

Like any real-world data set, noise will be encountered in both the images and labels. Images may contain artifacts, be out of focus, underexposed, or overexposed. The aim is to develop robust algorithms that can function in the presence of noise and variation.

3a. Characteristics of the dataset

The images of retinas are in .jpg format with varying dimensions (3000 x 2000 on average) and color space. The training set has a total of **35126** images, while the testing set has a total of **53576** images requiring a score.

4. Solution Statement

The task is to create an automated analysis system capable of assigning a score based on this scale:

0 - No DR
1 - Mild
2 - Moderate
3 - Severe
4 - Proliferative DR

5. Benchmark Model

The engineer who produced the best model posted links to his:

- report/results: https://kaggle2.blob.core.windows.net/forum-message-attachments/88655/2795/competitionreport.pdf
- ConvNet library used for the competition: https://github.com/btgraham/SparseConvNet
- reference on max-pooling: http://arxiv.org/abs/1412.6071

His final Kappa score is **0.84958.**

A link to reading material on Cohen's Kappa: https://en.wikipedia.org/wiki/Cohen%27s_kappa

Primary benchmark:

- **Step 1:** A CNN trained with VGGNet will be used for training and accuracy testing on a smaller subset of the dataset.
- **Step 2:** The accuracy of the model will be recorded for use as a benchmark against the final implementation.

Secondary benchmark: The model should achieve a Kappa score of at least **0.7**.

6. Evaluation Metrics

(source: https://www.kaggle.com/c/diabetic-retinopathy-detection#evaluation)

Results are scored based on the quadratic weighted kappa, which measures the agreement between two ratings. This metric typically varies from 0 (random agreement between raters) to 1 (complete agreement between raters). In the event that there is less agreement between the raters than expected by chance, this metric may go below 0. The quadratic weighted kappa is calculated between the scores assigned by the human rater and the predicted scores.

Images have five possible ratings, 0,1,2,3,4. Each image is characterized by a tuple *(ea,eb)*, which corresponds to its scores by *Rater A* (human) and *Rater B* (predicted). The quadratic weighted kappa is calculated as follows. First, an 'N x N' histogram matrix *O* is constructed, such that *Oi,j* corresponds to the number of images that received a rating *i* by *A* and a rating *j* by *B*. An *N-by-N* matrix of weights, *w*, is calculated based on the difference between raters' scores:

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2}$$

An *N-by-N* histogram matrix of expected ratings, *E*, is calculated, assuming that there is no correlation between rating scores. This is calculated as the outer product between each rater's histogram vector of ratings, normalized such that *E* and *O* have the same sum.

From these three matrices, the quadratic weighted kappa is calculated as:

$$\kappa = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}}.$$

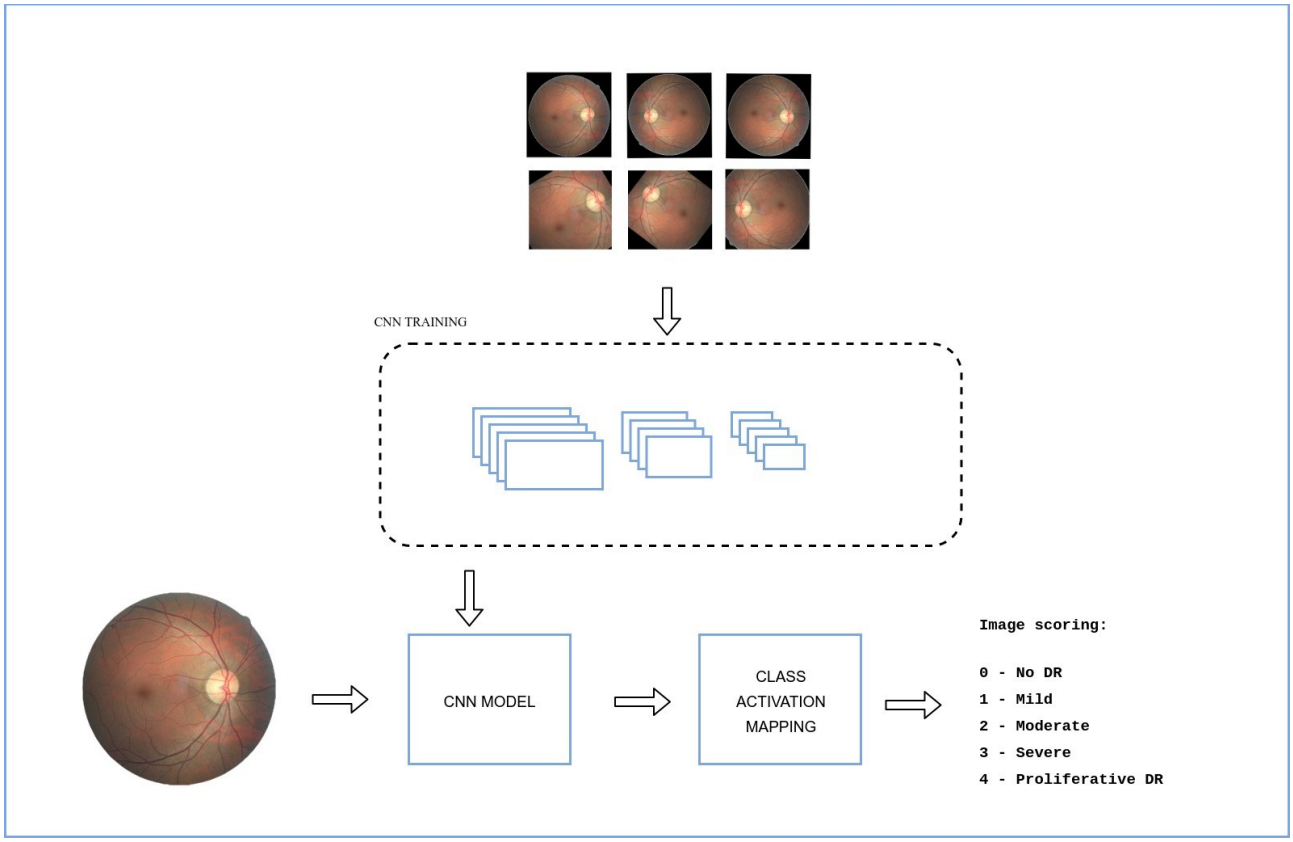
The final .csv file must have an image name and predicted DR level for each image. It should look like the following:

Image	Level
1_left	0
1_right	0
2_left	0
2_right	0

(*continued overleaf*)

7. Project Design

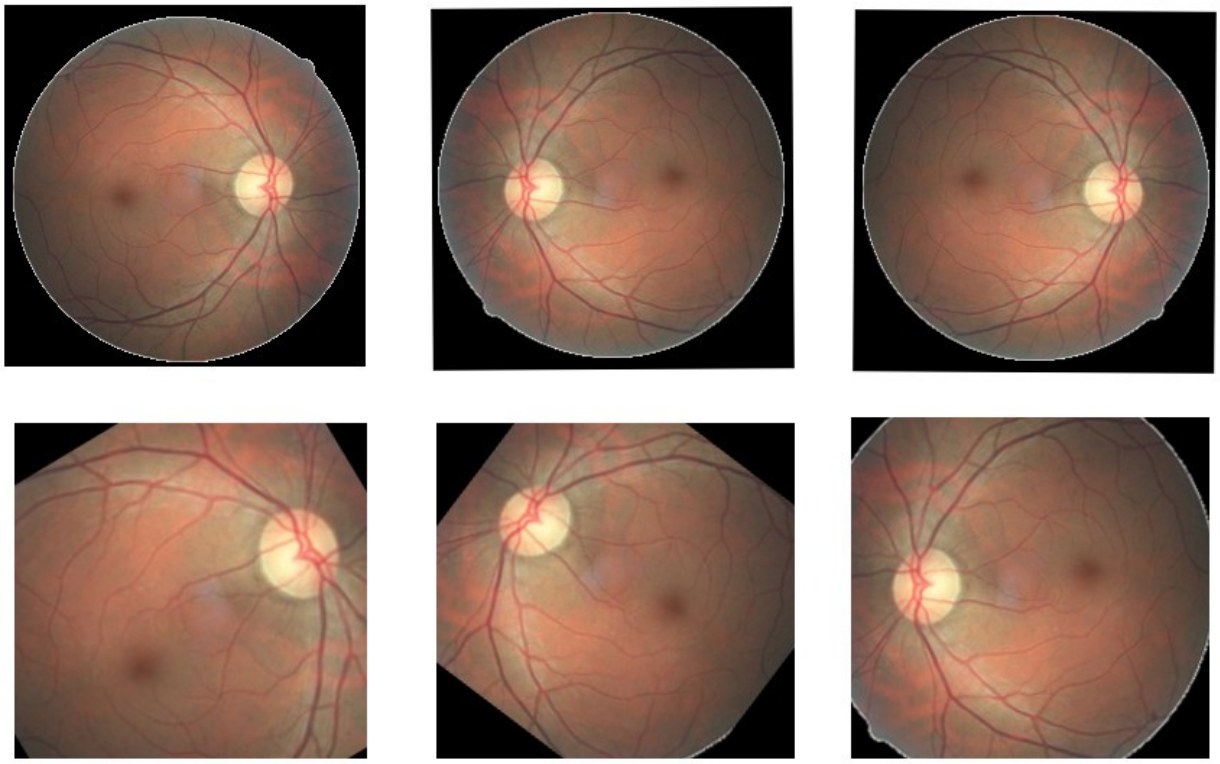
The plan is to use Tensorflow and Keras to predict the DR level for each image.



7a. Preparing dataset

This dataset is *huge* with the size of the training set at 32GB and the testing set approaching 50GB. The images will have to be loaded into memory in smaller batches.
The process will be managed with TFRecord and images preprocessed:

- remove unnecessary pixels (borders, etc),
- attach labels,
- resize to 299x299,
- rotate and mirror images,
- noise removal,
- color processing,
- convert images to arrays of NumPy arrays



Images on the second row have been cropped, resized and rotated to remove noise and augment the model.

7b. Pre-training

Pick and use a pre-trained model to avoid training the model from scratch. Tentatively, one of the following pre-trained networks will be used:

- AlexNet
- VGGNet
- GoogleNet
- Resnet

The deep CNN architectures are designed for 1000-class classification tasks. For adaptation, up to three layers from each network will be removed from each network.

Please refer to the diagram in **APPENDIX A** for my proposed model with VGG16.

7c. Training and Testing.

The model will be a Neural Network with a Tensorflow backend, similar to the dog classification CNN.

The network's architecture will probably consist of a few convolution layers (3x3) with a pooling layer.

Final layer will be a dense layer with an output layer (probably softmax).

7d. Output

Output will be a **Kappa Score** and a csv file with each image's name and the predicted DR level for each image

Additional sources:

- http://blog.kaggle.com/2015/08/14/diabetic-retinopathy-winners-interview-4th-place-julian-daniel/
- http://blog.kaggle.com/2015/09/09/diabetic-retinopathy-winners-interview-1st-place-ben-graham/
- http://blog.kaggle.com/2015/08/10/detecting-diabetic-retinopathy-in-eye-images/

APPENDIX A

BENCHMARK MODEL CNN ARCHITECTURE