DS 5220: Supervised Machine Learning and Learning Theory

Fall 2019

Project Title: Diabetic Retinopathy Detection

Team Number: 16

TA: Christopher

Team Members: Aishwarya Vantipuli, Aaron Moskowitz, and Qiang Fang

Problem Description:

Diabetic retinopathy is the leading cause of blindness in the working-age population of the developed world and is estimated to affect over 93 million people. Currently, detecting DR is a time-consuming and manual process that requires a trained clinician to examine and evaluate photographs of the retina. This poses a challenge in areas that traditionally lack access to skilled clinical facilities. Moreover, the manual nature of DR screening methods promotes widespread inconsistency among readers.

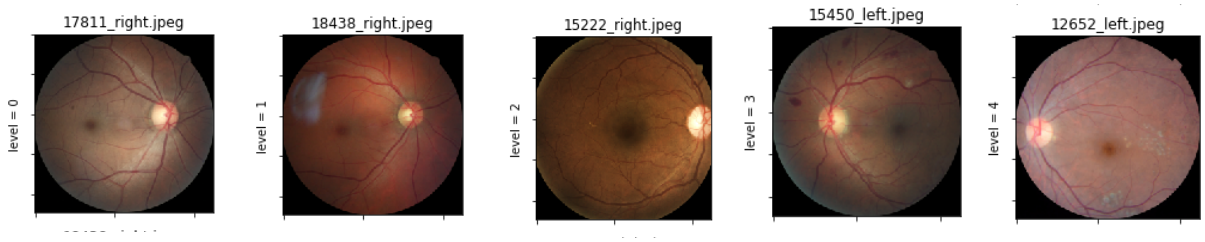
With color fundus photography as input, the goal of the project is to introduce an automatic DR grading system capable of classifying images based on disease pathologies from four severity levels. Throughout this study, we aim to elucidate a more effective means of classifying early-stage diabetic retinopathy for potential clinical benefits.

Dataset:

Diabetic retinopathy images were acquired from a Kaggle dataset of 35,000 images of size 84GB (California Healthcare Foundation 2015)[6]. The dataset comprises high-resolution retina images taken under a variety of imaging conditions that vary in height and width between the low hundreds to low thousands of pixels.

The left and right fields are provided for every subject. Images are labeled with a subject id as well as either left or right (e.g. 1_left.jpeg is the left eye of patient id 1). A clinician has rated the presence of diabetic retinopathy in each image on a scale of 0 to 4, according to the following scale:

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



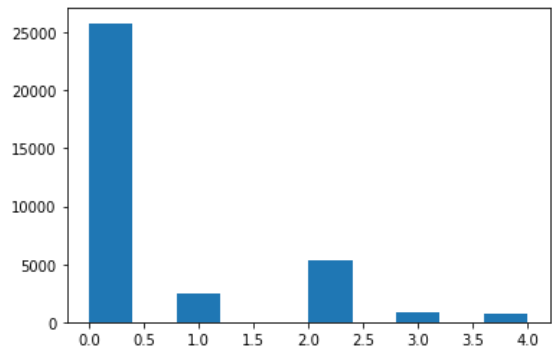
A visual examination of 5 images of each class was reviewed. The class 0 and class 1 images looked almost identically. It is possible that class 1 has slightly darker blood vessels, but they look effectively the same to an untrained eye. A small number of class 2 images either have a slightly redder color to them, but are otherwise similar to class 0 and class 1 eyes. However, most class 2 eyes has very visible droplets, or large dark clusters. Class 3 eyes often have more pronounced droplets or large dark clusters compared to class 2. Class 4 eyes tend to be distinct compared to even class 3 eyes due to common widespread blackness or a bright red color. However, around 1/3rd of the class 4 eyes look very similar to class 3 eyes or class 2 eyes.

Therefore, a conclusion of the EDA is that class 0 and class 1 eyes look very similar, and it will be difficult to differentiate them. There are certain characteristics that appear in class 2 of droplets or dark clusters that become more pronounced as the intensity of the disease increases. However, there is not a clear difference of class 2, 3, and many class 4 eyes. Class 4 eyes have

some unique characteristics of widespread black or red throughout the eye. The similarities between each of the classes make it seems that an error one class away would likely be very common.

There are a total of 5 train sets each containing around 8,000 images of size 8.5 GB. Loading the data and training the models is done using the Google Colab cloud server.
We found that the classes were very imbalanced with 73% of data containing images with no severity (class 0).

Disease Class 0 - 25,810 images
Disease Class 1 - 2,443 images
Disease Class 2 - 5,292 images
Disease Class 3 - 873 images
Disease Class 4 - 708 images



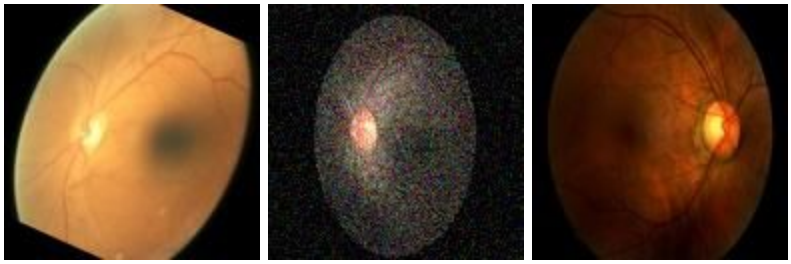
Approach and methodology:

During the first step in preprocessing, the images were reduced size to 500 x 500 pixel color images from the original pixel sizes of around 3000 x 5000 pixels. This is to ease in reducing the feature space to allow for both faster and easier to converge analysis. Color images were normalized to represent pixels in the range 0 to 1 across matrixes for red, blue and green before use in the models. To keep the images outside of memory, the images are housed in different folders for each class from inspiration of existing code (Chollet)[4]

Class 4 had the smallest number of images of 708, so a random 20% of those images (142) were designated as the validation set. Another 142 random images were selected from each disease class to form a balanced validation set. The remaining images are used for the training set.

Data Augmentation is a necessary step in this case. Class 0 images were not augmented due to the large sample size. The following are the methods used for image augmentation:

1. Random Rotation: Picking a random degree of rotation between 25% on the left and 25% on the right.
2. Random Noise: Adding random noise to the image
3. Horizontal Flip: Flipping the image array of pixels



Three augmented training sets of different sizes of images for each class were used. The smallest set of no data augmentation involved the full 566 Class 4 training images, and the first

566 training images from all other classes. Larger augmentation sets of 5,000 total images per class involved all training data of class 1-4 and data augmentation on the training images brough the size of each class up to 5,000. Finally, a set of 25,668 images per class in each training set was used to allow for utilization of all original training data in class 0.

Machine Learning Models and Model Conclusions:

KNN Model:

Besides deep learning model, we also plan to try machine learning models, we started with KNN model. We defined two methods to take an input image and converted it to a feature vector. First method simply took an input image and resizes it to a fixed size, and then flattened the RGB pixel intensities into a single list of numbers. Second method accepted an input image and constructed a color histogram to characterize the color distribution of the image.

Both methods had roughly 23% accuracy on the 566 per class sample with a K of 3. The model did not resolve when running on a local computer across the 5,000 per class size. The 28 K per class training set was not used due to the smaller 5,000 per class size did not resolve.

Random Forest:

We also tried random forest using same features, the accuracy improved a little (about 30%) compared to KNN, but with drastically lower computational time. We used 50 decision trees. We didn't put both eyes of the same person into train or test dataset. Just randomly assign to train and test. When 5,000 images per class was used, the accuracy was between 22-25%.

Convolutional Neural Network Model:

A Convolutional Neural Network (CNN) was used due to a recent success of those models from numerous work such as (Krizhevsky 2013)$_1$ showed the success in CNN models to classify images through training with Graphical Processing Units (GPUs). A Keras model architecture previously used to classify between cats and dogs were utilized as the basis for the CNN model used in this analysis (Chollet)$_4$

A CNN model had 29% validation set accuracy after training on 566 records per disease class. The accuracy was 24% on the 5,000 per class size. The 25,668 per class trained model had 26% accuracy. There was a clear trend in the accuracy of the class 0 classifier increasing as the size of the model increased to 25,668 when otherwise the accuracy was stable. This seems to indicate that more diverse data in other classes would improve the accuracy of those classes. Otherwise, different samples resulted in widely different accuracy metrics by class that swung by upwards of 30%. There was no clear pattern when each other class did not perform better when all of their non-augmented training data was used or not. The models also began to overfit as soon as the 2nd or 3rd epoch. These two facts seem to indicate that the model had large issues in converging to a lower loss for non-training data.

Remaining work:

Data Preprocessing Alterations: More pre-processing techniques include more robust data augmentation techniques such as by adjusting image saturation, hue, or brightness. Previous code on Kaggle involving this problem achieved accuracy of 75% with methods from (Ilovescience 2019)$_3$ of oversampling of data to result in a balanced model without the use of

augmented data. The oversampling technique can be used to compare data augmentation and sample size strategies tradeoffs before using the data to optimize models.

The best performing and best performance of non-deep learning model was random forest. However, due to the long time to train the model on such a large input, a lower dimensional representation will be created with PCA. Next, a non-deep learning model such as Random Forest or Logistic Regression will be used on the lower dimensional representation. This model will be used to optimize the preprocessing of the training set. The validation set will be further split into a test set to give a designated validation set to give insights on how to design each model without the large computational and storage cost of cross validation.

One of the greatest hurdles has been that samples larger than 500 images per class can not be easily uploaded to Google Colab. This can be resolved by connecting GitHub to Google Drive, and only uploading non-augmented images in single zip files. Lab computer may also be used to get larger hardware resources that do not require uploading to a cloud system. If there continues to be processing time concerns, the images will be further downsized to be smaller than 500 by 500 pixels.

Model Alterations: Tensorflow can allow us to attain better accuracy levels than the higher level Keras. Another model will be built (most likely multi-layer perceptron) for comparison with the KNN and CNN model.

The CNN model will be tuned with further runs with different hyper parameters to achieve a more consistent accuracy results across class 1 to 4. Insight on the CNN model will be utilized when a larger CNN model will incorporate pre-trained weights from extensively trained models. For example, the convolutional layers on VGG-18 will likely be used after previous results of accuracy of 60% (Mader 2017)$_2$. Transfer learning using ResNet50 in PyTorch resulted in a similar tier of accuracy of 75% (Ilovescience 2019)$_3$.

Goal Alterations: If the use of 5 classes is too difficult to achieve high accuracy the problem statement can be altered. A grouping of 3 classes of no disease, mild and severe may be utilized (Lam, Yi, Guo, Lindsey 2018)$_5$. In addition, a top 2 result can be used to allow the model to more easily accomplish it's task when converging. Finally, a binary model of sick or not sick can allow more easy use of all data, as well as an easier problem statement.

Team member contribution:

Aaron Moskowitz: General project planning, image downsizing and validation split, CNN model, execute data augmentation with model tests, and report writing.
Qiang Fang: project discussion, training and validating kNN, random forest model. Report writing.
Aishwarya Vantipuli: Team coordination, Dataset selection, Data Augmentation, Visual Insights, and extensive report writing.

References:

- Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever (2013) *ImageNet Classification with Deep Convolutional Neural Networks* University of Toronto$_1$
- Kevin Mader (2017) *InceptionV3 for Retinopathy (GPU-HR)* Kaggle$_2$
- Ilovescience (2019) *Diabetic Retinopathy with ResNet50, Oversampling* Kaggle$_3$
- François Chollet (2017) *Deep Learning with Python* Manning$_4$

- Carson Lam, Darvin Yi, Margaret Guo, Tony Lindsey (2018) *Automated Detection of Diabetic Retinopathy using Deep Learning* AMIA Jt Summits Transl Sci Proc[5]
- California Healthcare Foundation (2015) *Diabetic Retinopathy Detection* Kaggle[6]
- Mengye Ren, Ryan Kiros, Richard Zemel (2015) *Exploring Models and Data for Image Question Answering* University of Toronto[7]
- Intel AI Developer Program (2017) *Image Pre-Processing and augmentation* Intel[8]
- A. Zhang, Z. Lipton, and A. Smola (2019) *Dive into Deep Learning* UC Berkeley[9]