eye_report

April 13, 2017

1 Summary Report for Eye Projects

Darvin Yi (darvinyi[at]Stanford[dot]edu) Carson Lam (carsonl[at]Stanford[dot]edu)

We can give the summary reports for the following projects: - Data Preprocessing - Exudate/Microaneurysm Segmentation - Diabetic Retinopathy Classification - Sliding Window Abnormality Detection

2 Data Preprocessing

We do a few things, but we'll focus on showing off the following: - Tight Cropping - Adaptive Histogram Equalization

2.1 Tight Cropping

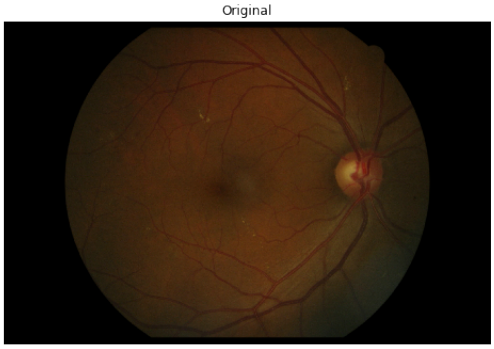
Most of the image will be black background. Thus, we want to crop very tightly around the eye as much as possible to give the network the most information as possible.

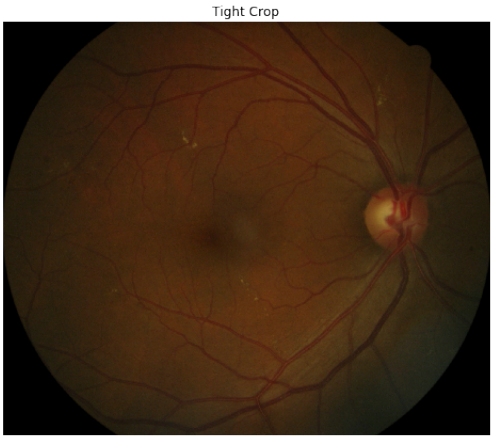
```
In [1]: import numpy as np
        import scipy.misc
        from skimage.filters import threshold_otsu
        from skimage import measure,exposure

        import matplotlib.pyplot as plt
        %matplotlib inline

        # Function to do cropping
        def tight_crop(img,size=None):
            img_gray = np.mean(img, 2)
            img_bw = img_gray > threshold_otsu(img_gray)
            img_label = measure.label(img_bw, background=0)
            largest_label = np.argmax(np.bincount(img_label.flatten())[1:]) + 1
            img_circ = (img_label == largest_label)
            img_xs = np.sum(img_circ, 0)
            img_ys = np.sum(img_circ, 1)
            xs = np.where(img_xs > 0)
            ys = np.where(img_ys > 0)
```

```
    y_lo = np.min(ys)
    y_hi = np.max(ys)
    x_lo = np.min(xs)
    x_hi = np.max(xs)
    img_crop = img[y_lo:y_hi, x_lo:x_hi, :]
    return img_crop
```

In [2]:
```
# Filepath
path_img = "figures/examples/C0021833.jpg"

# Reading in Image
img = scipy.misc.imread(path_img)
img = img.astype(np.float32)
img /= 255

# Cropping Image
img_crop = tight_crop(img)

# Plotting
plt.rcParams['figure.figsize'] = [18,12]
fig,ax = plt.subplots(1,2)
ax[0].imshow(img)
ax[0].axis('off')
ax[0].set_title('Original')
ax[1].imshow(img_crop)
ax[1].axis('off')
ax[1].set_title('Tight Crop')
```

Out[2]:
```
<matplotlib.text.Text at 0xa9933c8>
```





2.2 Adaptive Histogram Equalization

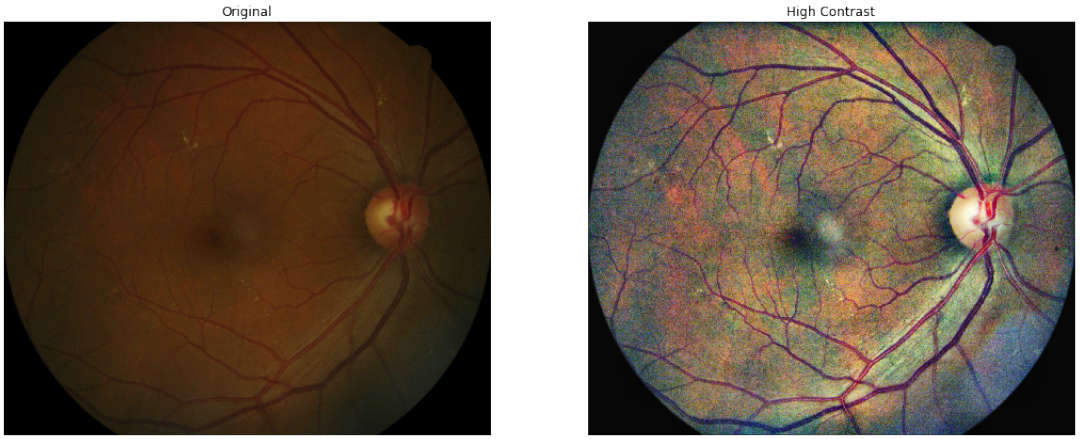
Basically, working with 8-bit images where the signal is quite subtle, we chose to put each channel through an adaptive histogram equalization to make sure interesting features were not lost.

```
In [3]: # Function for AHE
        def channelwise_ahe(img):
            img_ahe = img.copy()
            for i in range(img.shape[2]):
                img_ahe[:,:,i] = exposure.equalize_adapthist(img[:,:,i], clip_limit
            return img_ahe
```

```
In [4]: # Do Contrast Enhancement
        img_ahe = channelwise_ahe(img_crop)

        # Plotting
        plt.rcParams['figure.figsize'] = [18,12]
        fig,ax = plt.subplots(1,2)
        ax[0].imshow(img_crop)
        ax[0].axis('off')
        ax[0].set_title('Original')
        ax[1].imshow(img_ahe)
        ax[1].axis('off')
        ax[1].set_title('High Contrast')
```

```
C:\Users\yidar\AppData\Local\Continuum\Anaconda2\lib\site-packages\skimage\util\dty
  "%s to %s" % (dtypeobj_in, dtypeobj))
```

```
Out[4]: <matplotlib.text.Text at 0xa681710>
```



To see why this contrast enhancement is helpful, we can look at microaneurysms.

```
In [5]: img_MA = img_crop[825:925,250:450,:]
        img_MA_ahe = img_ahe[825:925,250:450,:]

        # Plotting
        plt.rcParams['figure.figsize'] = [18,12]
        fig,ax = plt.subplots(1,2)
        ax[0].imshow(img_MA)
        ax[0].axis('off')
        ax[0].set_title('Original')
        ax[1].imshow(img_MA_ahe)
        ax[1].axis('off')
        ax[1].set_title('High Contrast')

Out[5]: <matplotlib.text.Text at 0xa882320>
```

Even zoomed in, it's hard to tell the microaneurysms in the original image. However, in the high contrast image, it's a little easier. Easier for the human will most likely be easier for the machine.

2.3 Exudate/Microaneurysm Segmentation

2.3.1 Data: A Color Fundus Image Database

The data comes from a French group called ADCIS. They apparently work on biomedical image analysis.

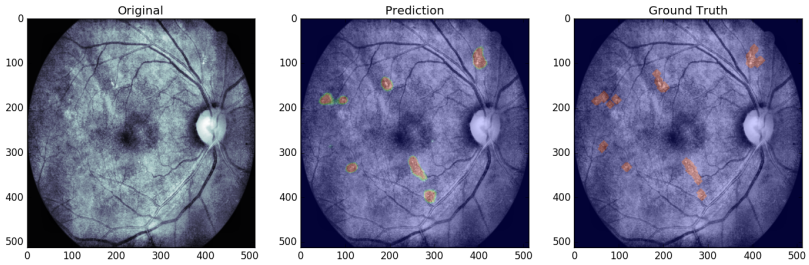
The main data that we have to work with has the following images: - 34 Patients with Exudates labeled as binary mask - 67 Patients with Microaneurysms labeled as binary mask - 100+ Normal Patients

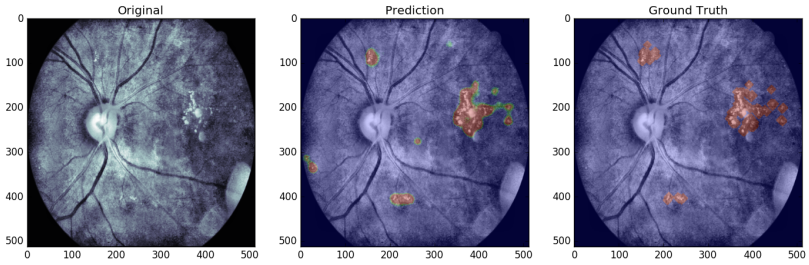
2.3.2 Preliminary work

We simply follow a simply architecture set up like the one shown by Long *et. al.*. We just put a few `conv2d_tranpose` layers on the end of GoogLe Net and we already get good results.

2.3.3 Exudates

Even with 34 patients and less than 1 hr training on a single gpu using default network parameters, we can segment out exudates pretty well. Here are some examples from the validation set



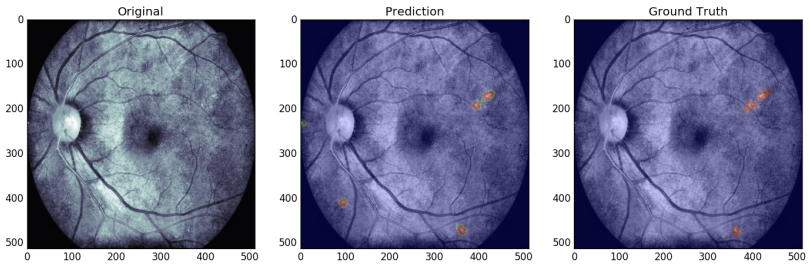


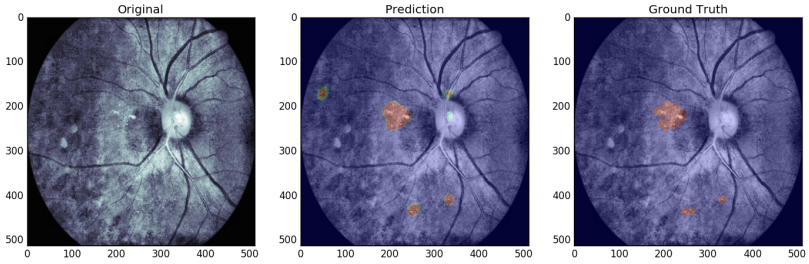
(i.e. never touched by training algorithm). Care was also taken to make sure training/validation sets were separated by patient.

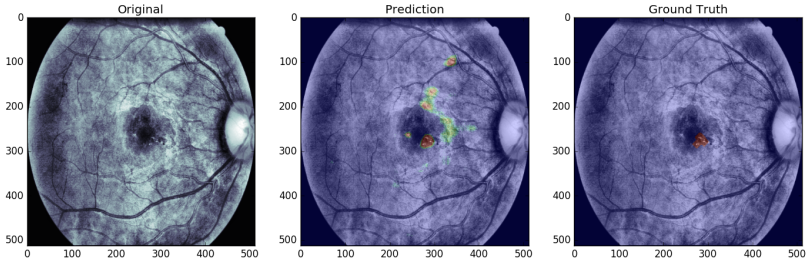
Every image below is formatted the same way. - The left image is a grayscaled version of the original image. - The right image is the original image overlayed with the grount truth segmentation (red=1, blue=0) - The middle image is the original image overlayed with our predicted softmax probability values represented by the jet-colormap going from colder blue colors for less probable and hotter red colors for more probable of being a pixel of interest.

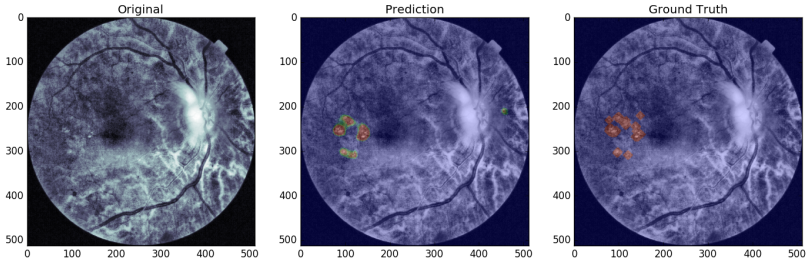
2.3.4 Microaneurisms

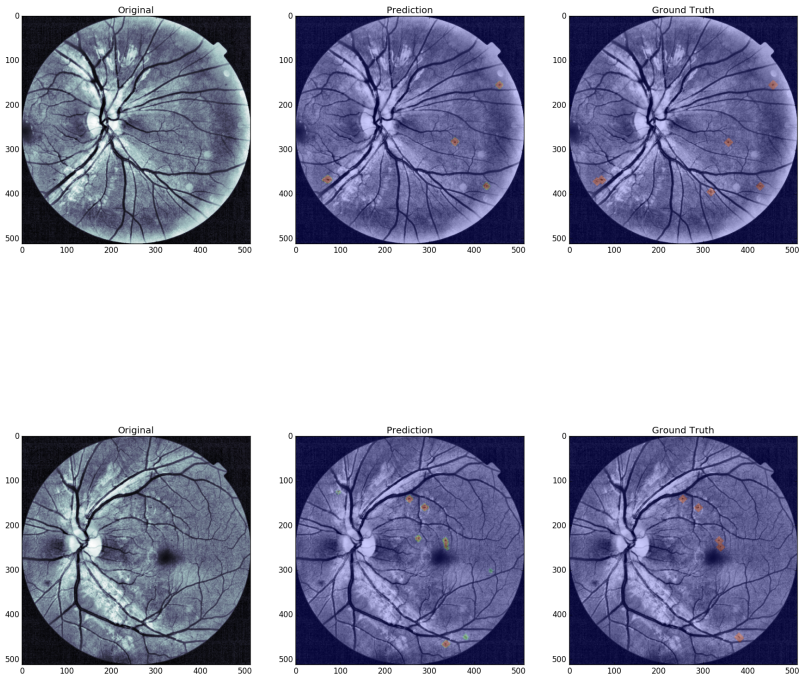
These were a bit trickier. To be honest, the exudates will train even without the histogram normalization. The microaneurisms will not train without the adaptive histogram equalization. They




Original
Prediction
Ground Truth
0
100
200
300
400
500


Original
Prediction
Ground Truth
0
100
200
300
400
500


Original
Prediction
Ground Truth
0
100
200
300
400
500



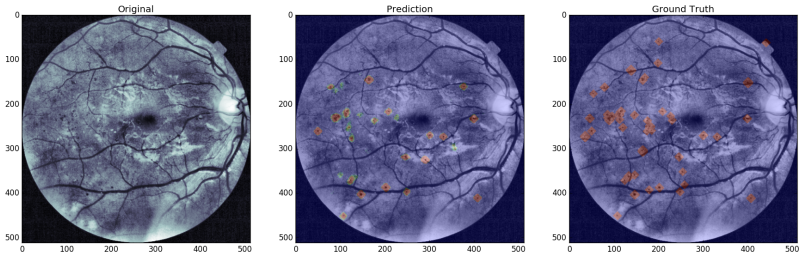
also take about twice the time to train as the exudates. This is because the features are MUCH MUCH smaller.

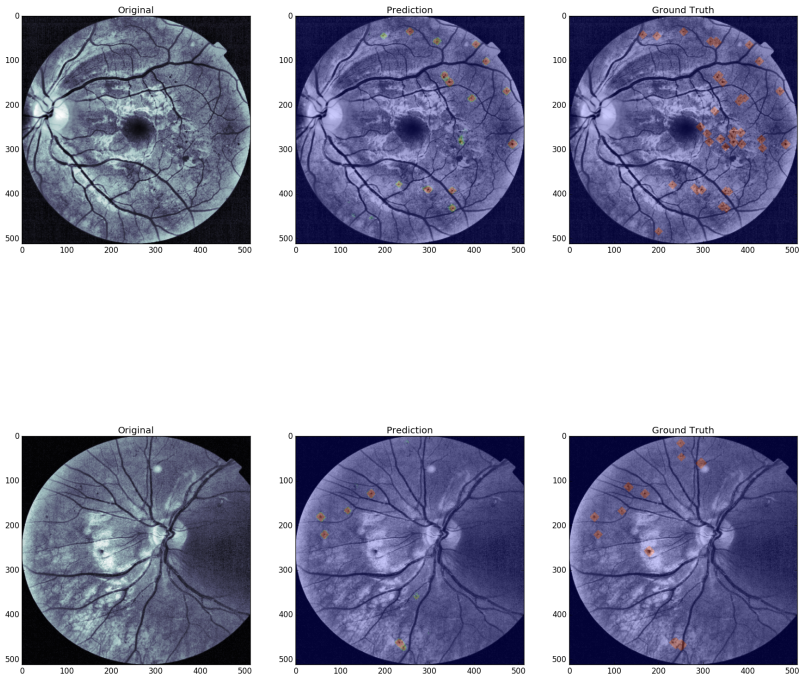
However, after training all the high contrast images for more than an hour (50+ epochs), we do see some results.

2.4 Diabetic Retinopathy Challenge

For the preliminary results, we simply separated a training and validation set. We applied the preprocessing steps as stated above (cropping and adaptive histogram equalization).

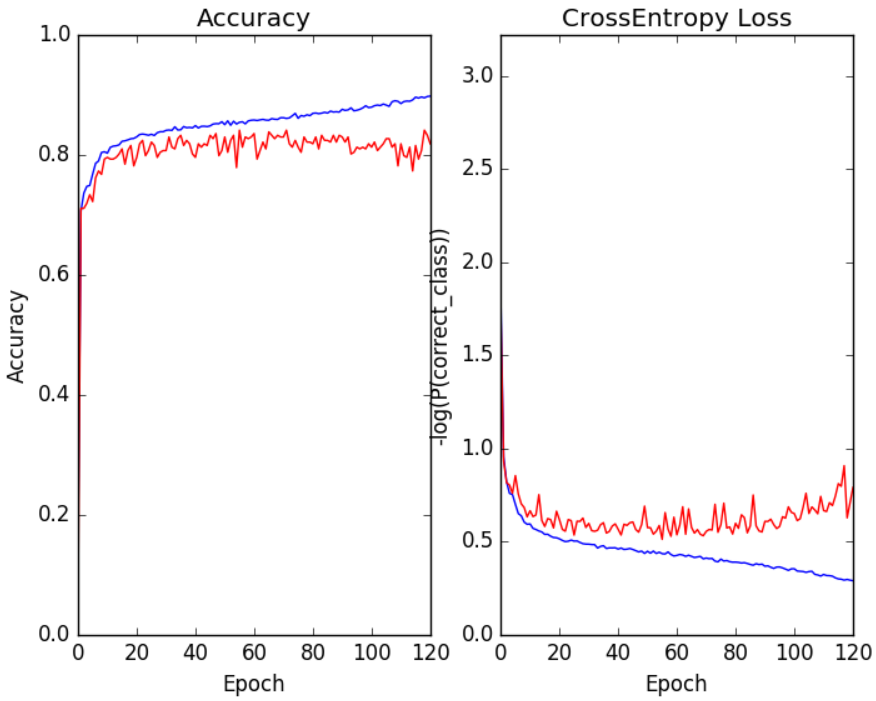
The data has the following splits: - Label 0: Normals - 73.48% - Label 1: Mild DR - 6.96% - Label 2: Moderate DR - 15.07% - Label 3: Sever DR - 2.48% - Label 4: Proliferative DR - 2.01%





Thus, we should be seeing a majority classifier of about 73.5%. We don't do much better, but we do indeed do better than the majority classifer. If we're being generous, we have a validation accuracy of about 85%, a 10% increase on the majority classifier.

In [ ]:


Accuracy
Accuracy
Epoch
CrossEntropy Loss
-log(P(correct_class))
Epoch