16725- Methods in Medical Image Analysis
Krishna Bairavi Soundararajan
ksoundar@andrew.cmu.edu
April 2019

Retinal Vessel Segmentation for Diabetic Retinopathy Study using U-Net and SegNet

Abstract:

Diabetic Retinopathy, a disease caused in the retina due to prolonged diabetes can lead to blindness. Early diagnosis is a must that can prevent the patient from getting blind. Since it's a disease specific to retina, studying the blood vessels present in the retina turns out to be valuable for diagnosis. Using the state of the art Deep Learning and Computer Vision techniques, we can build Deep Neural Networks that can perform semantic segmentation of the retinal images. In my work, I have implemented two Segmentation Algorithms namely U-Net [1] and Seg-Net [2] and trained it for same number of epochs and compared the model metrics to see which algorithm performs better. I have used a loss function called Focal Loss [3] since it performs better than cross entropy loss.

Dataset Used:

DRIVE Image Dataset [4] was used for the study. It contained a total of 40 images (20- training and 20- testing). Each image had a corresponding mask and a gold standard image to train with and compare results in testing

Methodology:

(1) **Pre- Processing**:

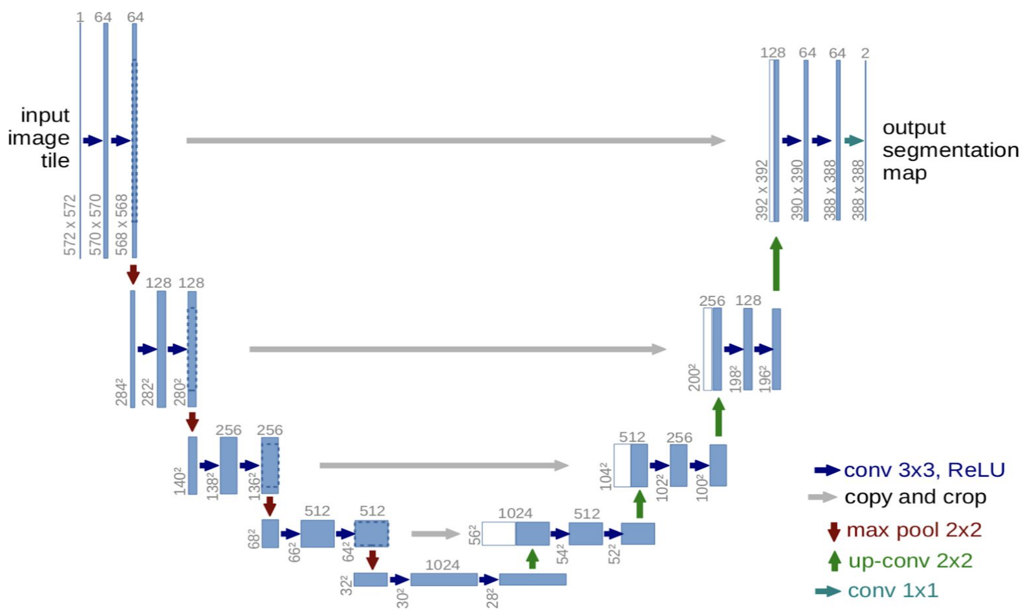
To the normalised image, I added Gaussian Noise, rescaled the Brightness Intensity and applied Adaptive Histogram Equalizer to produce a preprocessed image. I implemented the preprocessing steps using Simple ITK. However, my resultant image wasn't very clear and since the number of training images were less, I had to split the train images into small patches for training and I was afraid that splitting into patches could lead to further clarity loss to the preprocessed image, so I proceeded with just the normalized image. I took only the green channel of the image into consideration after doing a literature survey [5]

(2) **Data Augmentation**:

Since the amount of training images were less, the images had to be split into multiple patches. For training, the patches were randomly generated by finding random center points and For testing, ordered patches were generated. The generated patches were all of dimension 48x48. Totally 190000 patches (9500 patches per image) were generated for training. For testing, 3200 patches (150 patches per image) were generated. The predicted patches were later reconstructed and compared to the reference image. Throughout the process of patch generation and reconstruction process, I found the work done by Orobix group [6] to be very useful and I used few of the functions in my code.

(3) **Convolutional Neural Networks Architectures**:

U-Net:



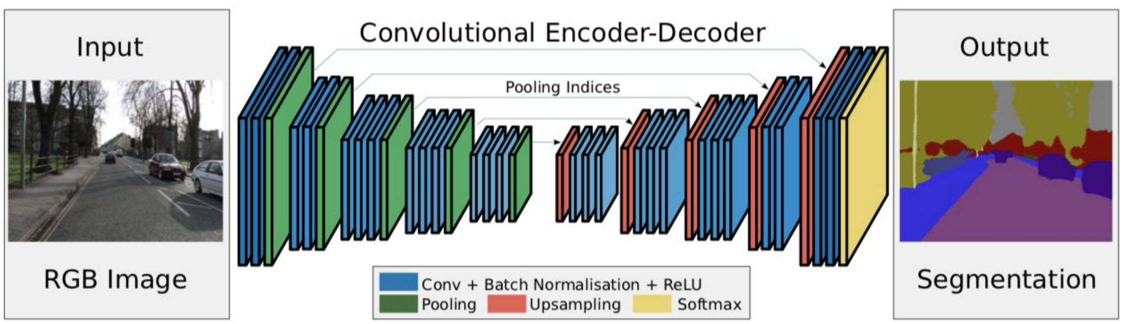
https://arxiv.org/pdf/1505.04597.pdf

U-Net was specifically developed for Biomedical Image Segmentation. The Net consists of a set of downsampling and upsampling layers. The Downsampling layer gives out the " WHAT" information about the image, helping to do a pixel wise classification. The Upsampling layers leads to reconstructing of the image to its original shape. At every step in the upsampling layer, the downsampled layer is concatenated with the previous upsampling output. The Upsampling layers give the "WHERE" information about an image and thus the reconstructed image not only knows what each pixel's class is,. But it also knows where those pixels lie in the image. Like other Convolution Networks, each Downsampling layer consists of a Convolution and a Max-pooling operation that reduces the size of the image. During the upsampling part, Inverse

Convolution operation is performed. However, training a U-Net is a bit time consuming (that's the case with every Deep learning model, yes, of course!), but , U-Net has the concatenation step in upsampling layer, that consumes a lot of memory [2]. For Keras implementation and understanding the concept of U-Net functioning, I found the blog post [7] to be very useful.

Seg-Net:

Contrary to U-Net, authors of Seg-Net provide evidence that SegNet performs better and consumes less memory [4]. This is because in case of SegNet, there is no concatenating step, and only the Max-pooling indices from the encoding layer are fed to the decoding layer. Hence it is presented as a more memory efficient algorithm compared to U-Net by the author's.



https://arxiv.org/pdf/1511.00561.pdf

Both of the architectures use a cross-entropy loss. I came across a paper on Focal Loss- from Retina Net architecture presented in the year 2017 by Facebook AI research [3] and I used Focal Loss as my Loss function and used Adam Optimiser instead of Stochastic Gradient Descent. I used a sigmoid activation since ReLU was throwing Nan errors.

(4) **Focal Loss**

Focal Loss (FL) is a variant of cross entropy loss (CE) technically and the authors of the paper put forth with promising results. In case of CE loss, easily classified negatives compromise majority of the loss and dominate the gradient [3]. In order to overcome this issue, FL reshapes the CE loss function by adding a modulating factor $\mathbf{(1 - pt)^\gamma}$. Here γ is a tunable parameter.

$$\mathbf{FL(p) = -(1 - p)^\gamma \log(p)}$$

Thus for misclassified examples when the probability p is small, modulating factor is near 1 and the loss in unaffected. When p is large and approaches 1, the modulating factor approaches 0 and the loss for correctly classified examples is downweighted. The focusing parameter γ in the modulating factor adjust the rate with which easy examples are down weighted. I have used a focusing parameter as 2, exactly as what is proposed in the paper.

(5) **Training Details**:

I trained both the UNet and SegNet modes for 10 epochs so that I can compare my results using a limited access GPU. I have included both my trained models to the code

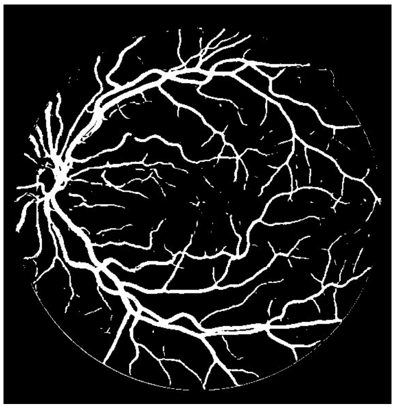
(6) **Testing Details:**

I generated 3200 ordered patches (156 patches per image) from the test dataset and used my trained models to predict the patches. Output patches were later reconstructed to produce an output image which of same dimension (584x576) as input image.

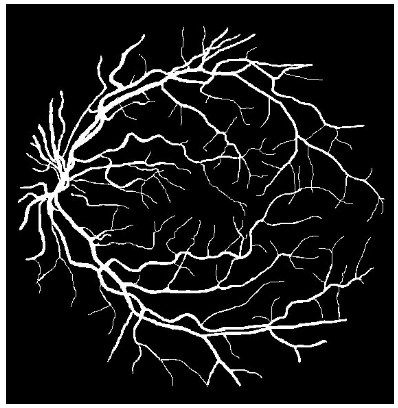
(7) Results and Discussions:

From training the model for 10 epochs and observing the predicted results per image, I felt like SegNet performed slightly better than U-Net. Please find few of the model predictions below for a visual comparison.

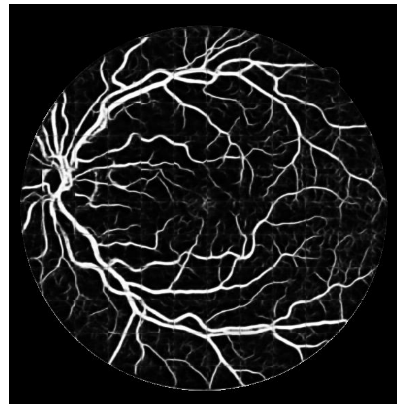
Few of my decent segmentation Images….



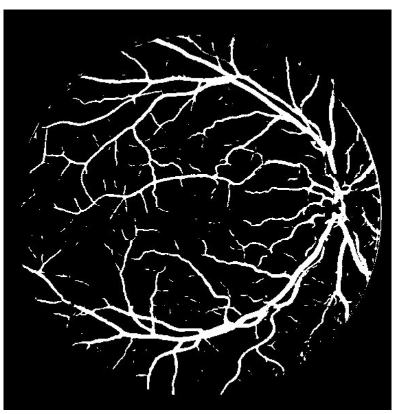
U-Net Prediction (subject 1)



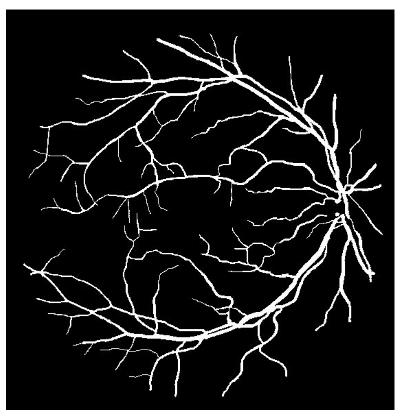
Reference Image (subject 1)



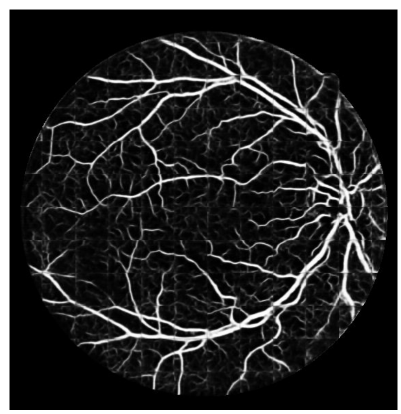
Segnet Prediction (subject 1)



U-Net Prediction (subject 20)



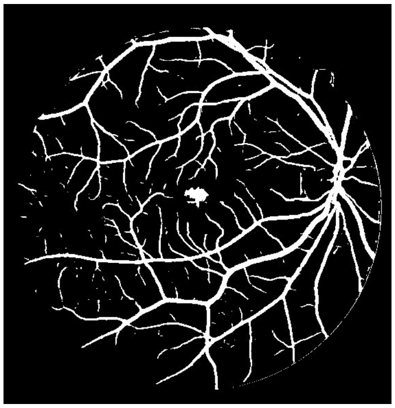
Reference Image (subject 20)



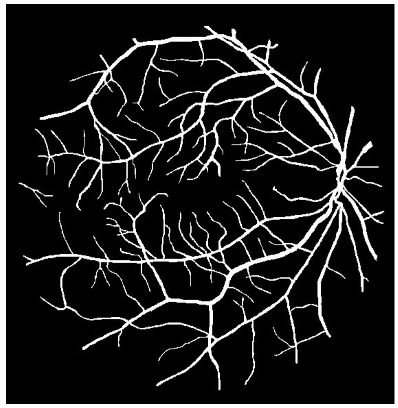
Segnet Prediction (subject 20)

Few of my average segmentation Images…

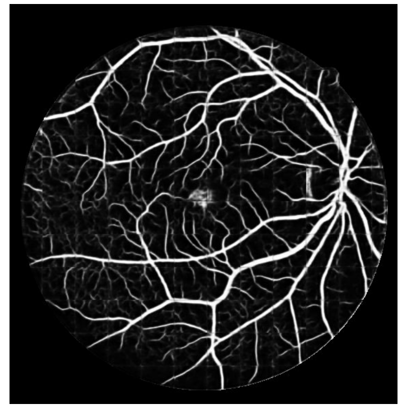
(*Please forget to see the white dots at the centre in the following predictions)*



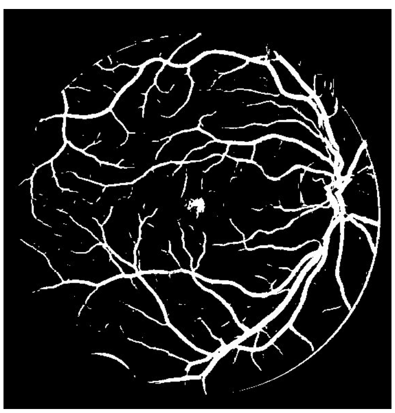
U-Net Prediction (subject 16)



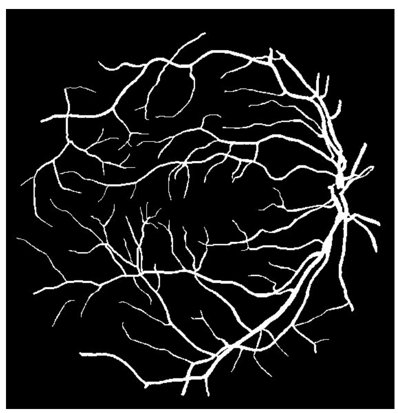
Reference Image (subject 16)



Segnet Prediction (subject 16)



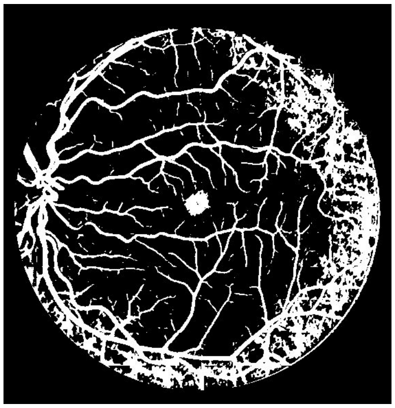
U-Net Prediction (subject 18)



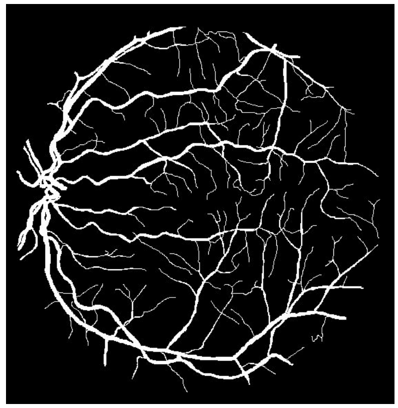
Reference Image (subject 18)

Segnet Prediction (subject 18)

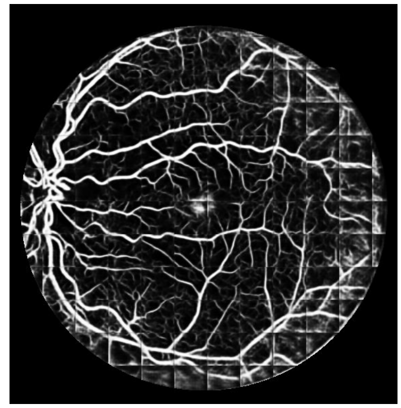
I did have a few bad predictions too…. For example:



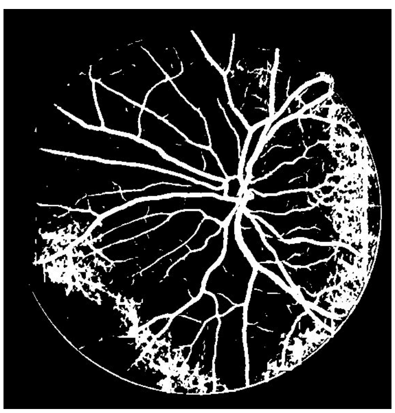
U-Net Prediction (subject 11)



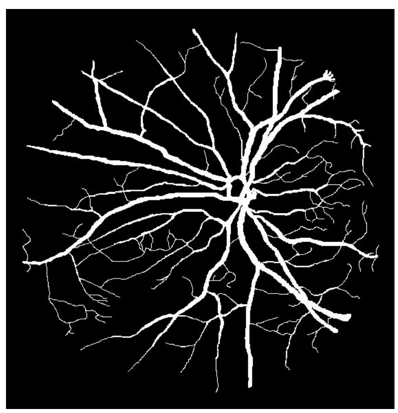
Reference Image (subject 11)



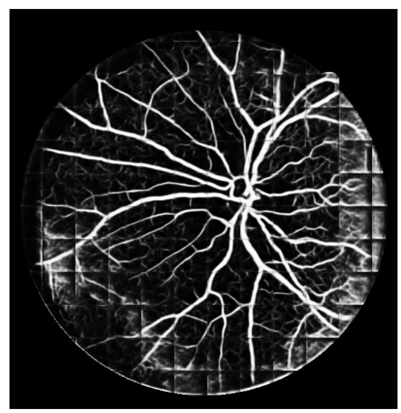
Segnet Prediction (subject 11)



U-Net Prediction (subject 4)



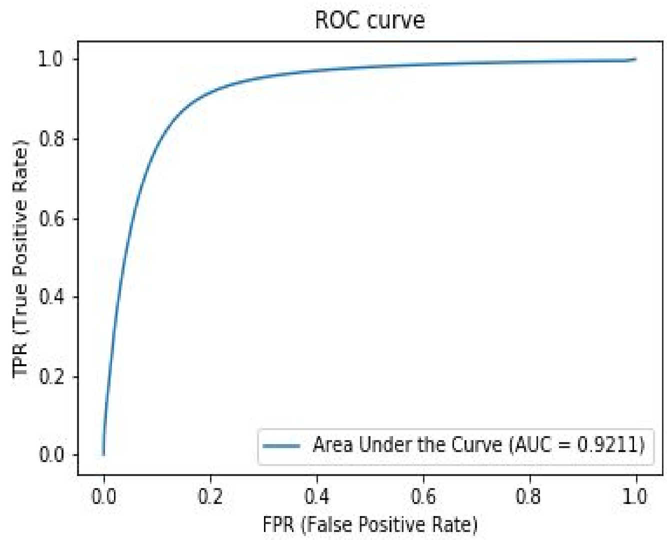
Reference Image (subject 4)



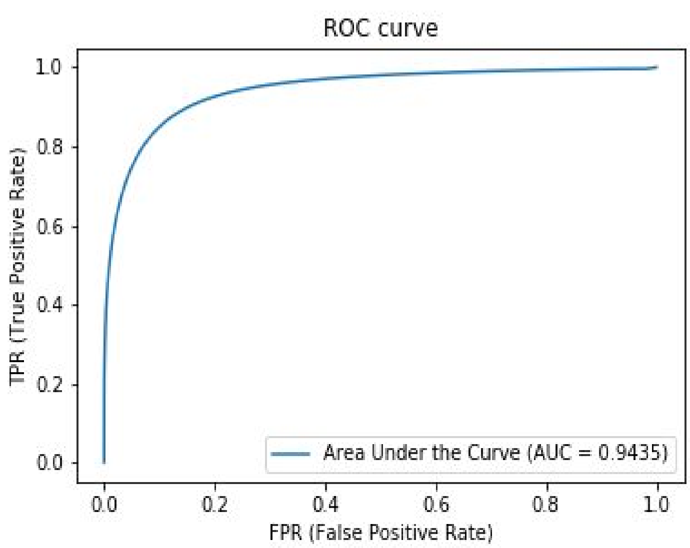
Segnet Prediction (subject 4)

ROC Plots:

Following are my result plots.



ROC Curve for U-Net predictions



ROC Curve for SegNet predictions

IOU Scores:

Test Image Number (DRIVE)	Precision Score (U-net)	Precision Score (Seg-Net)
1	0.816	0.86
2	0.742	0.831
3	0.645	0.727
4	0.688	0.786
5	0.624	0.784
6	0.873	0.912
7	0.456	0.67
8	0.565	0.719
9	0.909	0.927
10	0.709	0.807
11	0.567	0.76
12	0.802	0.851
13	0.877	0.912
14	0.62	0.68
15	0.206	0.457
16	0.841	0.885
17	0.846	0.882
18	0.772	0.821
19	0.638	0.79
20	0.792	0.832
mean	0.6994	0.79465

Precision Scores:

Test Image Number (DRIVE)	Precision Score (U-net)	Precision Score (Seg-Net)
1	0.816	0.86
2	0.742	0.831
3	0.645	0.727
4	0.688	0.786
5	0.624	0.784
6	0.873	0.912
7	0.456	0.67
8	0.565	0.719
9	0.909	0.927
10	0.709	0.807
11	0.567	0.76
12	0.802	0.851
13	0.877	0.912
14	0.62	0.68
15	0.206	0.457
16	0.841	0.885
17	0.846	0.882
18	0.772	0.821
19	0.638	0.79
20	0.792	0.832
mean	0.791	0.795

Future Work:

I would love to experiment the segmentation for larger epochs (If I get a GPU access) and train the model with more training examples and come up with a more fine conclusion about which model performs better than the other.

References:

1. Ronneberger, O., Fischer, P. and Brox, T., 2015, October. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham.
2. Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." *IEEE transactions on pattern analysis and machine intelligence* 39, no. 12 (2017): 2481-2495.
3. Lin, T.Y., Goyal, P., Girshick, R., He, K. and Dollár, P., 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980-2988).
4. J.J. Staal, M.D. Abramoff, M. Niemeijer, M.A. Viergever, B. van Ginneken, "Ridge based vessel segmentation in color images of the retina", *IEEE Transactions on Medical Imaging*, 2004, vol. 23, pp. 501-509.
5. Siva Sundhara Raja, D. and Vasuki, S., 2015. Automatic detection of blood vessels in retinal images for diabetic retinopathy diagnosis. *Computational and mathematical methods in medicine*, *2015*.
6. https://github.com/orobix/retina-unet
7. https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47