Abstract

The mission of the Center for Global Women's Health Technology (GWHT) is to use optical technologies produce widely accessible healthcare technologies, in particular for women's cancers. Current work involves *in vivo* models of cancer. Angiogenesis, an important hallmark of cancer, is a desirable endpoint to add. This motivated me to develop a convolutional neural network to segment vasculature. To achieve this I implemented a U-Net architecture with a trainable mask physical layer. I have achieved metrics of 0.1975 loss and 0.9244 accuracy after 30 epochs training with 134 images. I have shown promising results with the addition of a trainable mask. Training with only 20 images, I achieved training loss of 0.392 and the validation loss of 0.393, which was an improvement over the same model trained with 20 images without the trainable mask, which yielded training loss of 0.567 and the validation loss was 0.495.

Introduction

The mission of Dr. Nimmi Ramanujam's Center for Global Women's Health Technology (GWHT) is to use optical technologies produce widely accessible healthcare technologies, in particular for women's cancers such as cervical and breast cancer. As a first year PhD student in the center working on the breast cancer project, my goal is to understand and predict metabolic changes in breast cancer cells as cancer transitions from a primary tumor, to dormant disease, to a recurrent tumor. Previous work from our group has developed fluorescent probes to monitor metabolism *in vitro* in mammosphere organoid models (1), and the project has now transitioned to an *in vivo* mouse model where turning on an oncogene induces tumor growth in the mammary pad of female nude mice. Mice receive a window and are injected with fluorescent probes for optical imaging. Two probes are injected simultaneously, allowing for the investigation of two metabolic endpoints. These probes were verified to not have optical crosstalk or interfere with the metabolic processes being studied.

An important hallmark of cancer is angiogenesis. Previous work in the Ramanujam lab studied tortuosity of vessels in a hamster cheek pouch model, finding significant differences in cancerous vasculature (2). This motivates a need to include vasculature as an endpoint in *in vivo* studies. However, introduction of another probe is not preferable as it would be difficult to coordinate excitation and emission wavelengths without crosstalk amongst three probes. I posit that a virtual staining technique where a Convolutional Neural Network (CNN) segments vessels and then overlays the segmentation maps with the original image would provide a valuable endpoint for these studies.

For this project, based on available data, I trained my CNN using a publicly available dataset of fundus images of the retina. These images were collected at St. Paul's Eye Unit in Liverpool, UK (3). This dataset comes with segmentation maps by an expert from St. Paul's. It is known as the Automatic Retinal Image Analysis (ARIA) dataset.

Related Work

Previously students in the Ramanujam lab have studied angiogenesis as a hallmark of cancer and developed image processing techniques to study angiogenesis. In the article *Dark field optical imaging reveals vascular changes in an inducible hamster cheek pouch model during carcinogenesis* (2) Hu et al use an inducible hamster cheek pouch carcinogen model where carcinogen DMBA is brushed on to a hamster cheek pouch daily for 17 days, taking cross-polarized dark field images at day 12, 14 and 17. Images were processed by dividing the green channel by the red channel to obtain G/R images with pixels greater than 1 set to 1, then taking the reverse G/R image by subtracting it from 1. Next they convolved those reverse G/R images with a multi-scale Gabor algorithm to obtain Gabor responses. Finally, Dijkstra forest vessel segmentation created binary vessel masks. Using these image processing techniques, authors compared control hamster cheek pouch vessels with carcinogen-treated cheek pouches and found that vessel features such as tortuosity and length showed different trends. Vessel length was significantly lower in the carcinogen-treated cheek pouches than the controls, and there was a significant increase in tortuosity in the carcinogen-treated cheek pouches when compared to the control (2).

The U-Net architecture is continuously being updated, so here I will briefly introduce the manuscript *SA-UNet: Spatial Attention U-Net for Retinal Vessel Segmentation* (4) by Guo et al which was submitted to the 2020 25th International Conference on Pattern Recognition on March 16, 2020. This aim of this paper was to improve U-Net to train for smaller datasets such as retinal fundus image datasets, while keeping the network lightweight and improving metrics compared to current standards. The authors achieved this by replacing standard convolutional blocks with a structured dropout block, and adding a spatial attention network between the encoder and decoder sides of the "U" which suppresses unnecessary features. Compared to sate of the art segmentation algorithms such as Backbone, SD-UNet and U-Net, it achieves the best performance with sensitivity of 0.8573, accuracy of 0.9755, area under the curve of 0.9905 without adding many parameters and remaining lightweight.

Methods

Importing Dataset

I downloaded images and corresponding segmentation masks from the Automatic Retina Image Analysis (ARIA) dataset (3). There were 134 images and 134 corresponding segmentation maps. Two different people each segmented the full dataset - BDP and BSS. At random I chose to work with segmentation maps from BDP. I uploaded all images to one folder in my Google Drive named Project Data>Images, and all segmentation maps from BDP in my Google Drive folder Project Data>Labels. I mounted my Google Drive to my Google CoLab Notebook.

I imported glob and read in images using the glob.glob command. I chose this command so I could then use the .sort() command to ensure that images and labels were read in in the order they appeared in my Google Drive and would therefore be correctly corresponded with each other. I then used tf.stack to create two stacks- one of images and one of labels. As a note, I relegated 20 of the 134 images for test images with the remaining 114 as training images.

Image Pre-processing

I normalized images and labels to be between 0 and one by dividing by the max value (255). I then cropped them to be 512x512. I added a channel to the labels. Finally I had two stacks of images and labels, all 512x512, having 4 dimensions (number, height, width, channels). I repeated the processes above for 20 test images.

Implementation of U-Net

I spent a significant amount of time on my project working on coding my own U-Net algorithm. I used the architecture described in "U-Net: Convolutional Networks for Biomedical Image Segmentation" by Ronneberger et al (4). My input images were 512x512. To build my layers I used tf.keras.layers.

For the encoder side of the U-Net I started with two 2D convolutional layers each with 16 3x3 filters with a stride of one and ReLu activation function. I then added a 2D max pooling layer. Following this pattern of two convolutional layers followed by a max pooling layer, I increased the number of filters to 32, then 64, then 128, then 256.

On the decoder side of the network, I used 2D transpose convolutional layers for upsampling. Skip connections were used to preserve spatial information from the encoder side of the network by concatenating transpose convolutional layers with the corresponding layer from the encoder side. Skip connection layers were followed by dropout with a rate of 0.1, which was followed by two 2D convolutional layers each with 16 3x3 filters with a stride of one and ReLu activation function. On the decoder, I started with 128 filters, then 64, then 32, then 16. The final layer was a 2D convolutional layer with 1 1x1 filter. Then final activation function was sigmoid.

Physical Layer

For my physical layer, I chose to implement a trainable mask and multiply that by the input images. I made the mask using tf.Variable. The mask is 512x512 (size of the images) and I initialized it as a 512x512 matrix of ones. I also implemented gaussian noise in my physical layer.

Results

I first ran my model without the trainable mask physical layer. I ran the model for 30 epochs. I had a dataset of 134 retina fundus photographs with 134 corresponding segmentation maps by the same individual. I relegated 20 of these image-label pairs for test data leaving 114 for training. After 30 epochs my training metrics were as follows: loss was 0.2178, mean intersection over union was 0.4566, and accuracy was 0.9130. My test metrics were: loss was 0.1975, mean intersection over union was 0.4622, and accuracy was 0.9244. Figure 1 shows a representative image. Figure 2 shows the plotted loss over epochs.

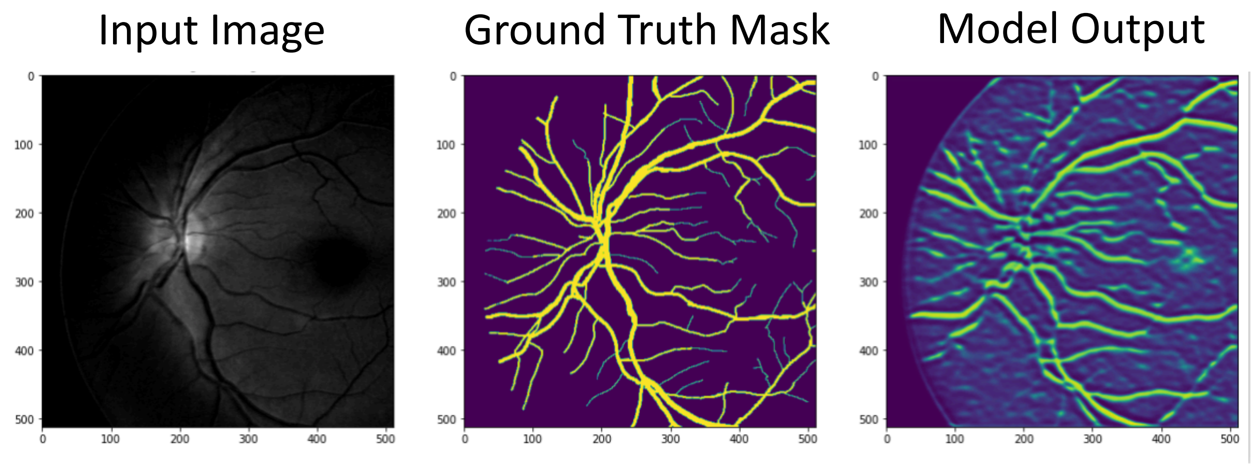

Figure 1: A representative example. The input image is the image from the ARIA dataset which has been cropped and plotted in greyscale. The Ground Truth Mask is the corresponding segmentation mask. Finally, the Model Output is the U-Net's plotted prediction after training for 30 epochs.

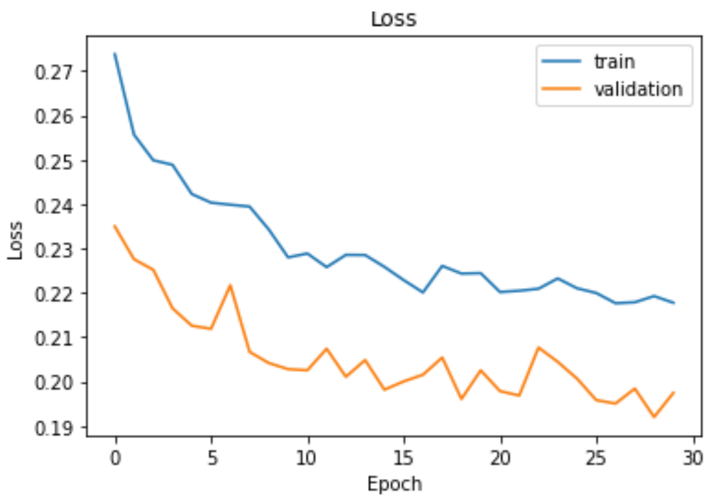

Figure 2: Training and validation loss plotted over 30 epochs.

Figure 3 shows an image where I believe the model did not perform well. I have highlighted an artifact in the input image. The algorithm mistakenly segmented that artifact as a vessel. I believe that training the algorithm with more data, and particularly data that either naturally or via augmentation had more noise and artifacts, would make the model more robust so it would learn to distinguish artifacts from vessels.

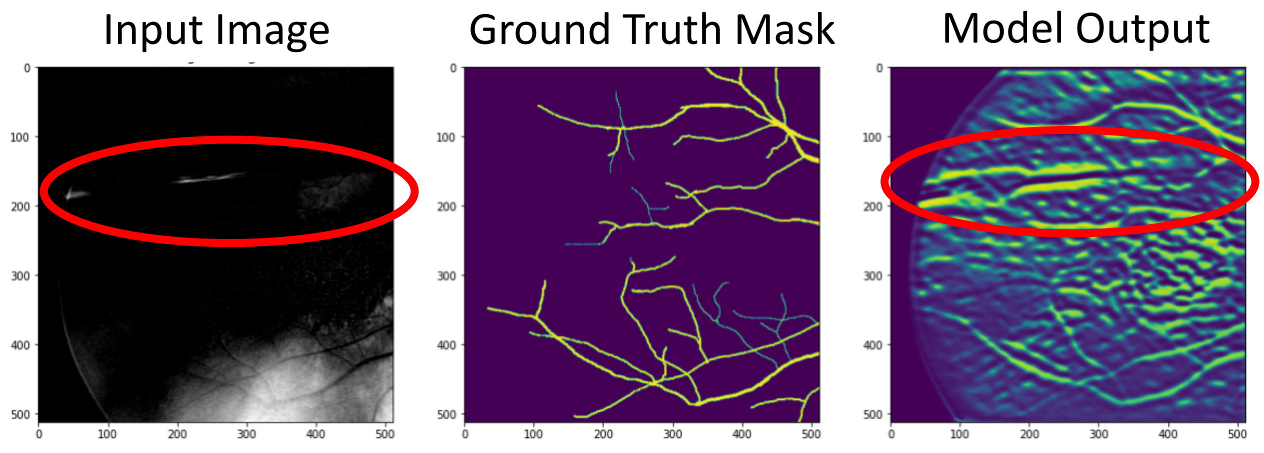

Figure 3: An example where I believe the model failed. I have added a red circle to an area where an artifact in the input image exists and where the model predicted a vessel there where one did not exist in the ground truth mask.

I next implemented a physical layer, a mask multiplied by input images and learned by the algorithm. First as a control I made my mask untrainable (by setting Trainable=False in my tf.Variable command). For the sake of time I ran the control with 20 images. I verified that the mask was the same (a 512x512 matrix of ones) before and after training. The training loss was 0.567 and the validation loss was 0.495.

Next, I made the mast trainable (by setting Trainable=True) and ran the algorithm again with only 20 images. This time the training loss was 0.392 and the validation loss was 0.393. This was an improvement! Figure 3 shows the output of the mask after training.

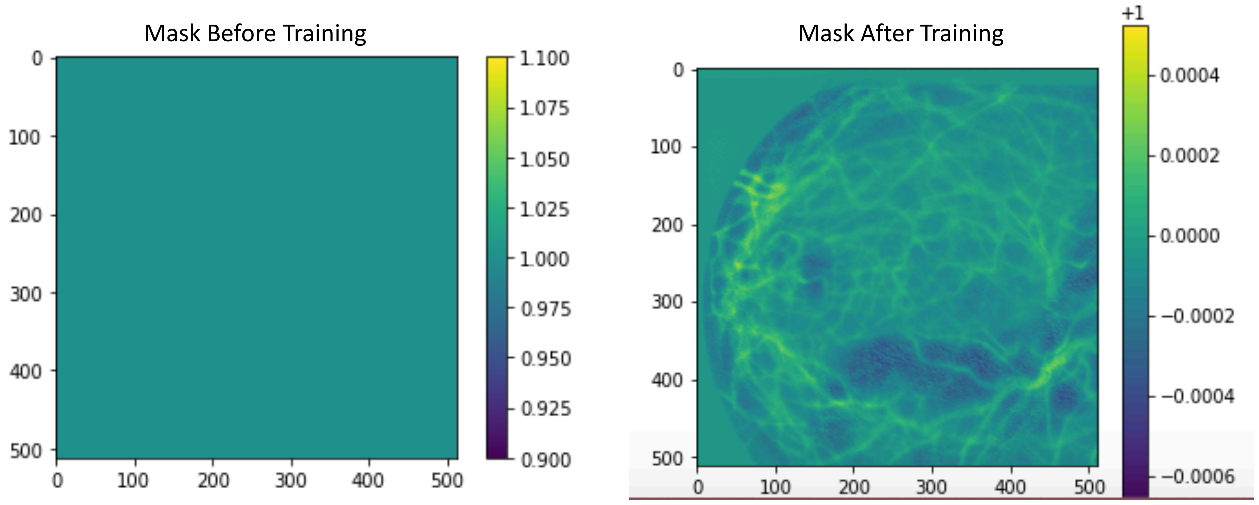

Figure 3: Mask Before Training is the 512x512 matrix of ones that I initialized. Mask After Training is the mask that the algorithm learned after training for 30 epochs with 20 images.

Next I made my mask trainable and ran with all 134 training images for 10 epochs. The training loss after 10 epochs was 0.297 and the validation loss was 0.265. Figure 4 shows the mask after training. Regrettably I did not have the time to run for 30 epochs to match the model without the physical layer, which I ran for 30 epochs. Despite the fact that I lack a comparison of the model trained with all 134 images, the results using only 20 images appear promising. I will be increasing and optimizing the number of epochs training with all 134 images in my future work.

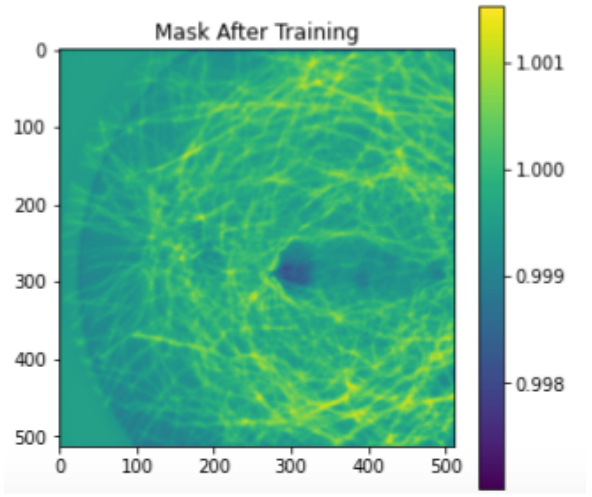

Figure 4: The mask that the algorithm learned after training for 10 epochs with 134 images.

<u>Discussion</u>

I acknowledge that my project has limited scope. Here I will discuss a few challenges and limitations to this project. One major limitation was my personal inexperience. This course was my first introduction to coding in Python (I had only coded in MATLAB previously) and my first introduction to the field of machine learning. Therefore, although I started work on this project early, I spent a significant amount of time exploring and learning commands and syntax and debugging errors. I also spent a majority of my time working on writing the code for the U-Net and debugging, validating, training, and testing. I found this learning process to be an incredibly valuable and I did very much enjoy the journey. Furthermore, the resources provided by the course in particular assistance from Professor Horstmeyer and both TA's were of great help. Another limitation was time. By the time I had achieved success in implementing my model it was nearly the deadline, so unfortunately I did not have time to train for more epochs or with a larger dataset.

Another limitation was that the dataset I used had only 134 images. I hypothesize that with more training data, the performance of the algorithm would improve. One way I attempted to improve upon that situation was to combine multiple datasets of segmented fundus retinal images. However, this proved to be difficult because images from different sources were different sizes and file types. As I learned Python I came up with fixes as to how to deal with not being able to stack images of different sizes and other work-arounds. However, two of the datasets besides ARIA that I proposed adding each contained only 20 images, therefore not significantly increasing the amount of training data. Another way to circumvent the limited data availability is to use data augmentation. I experimented with Keras' ImageDataGenerator and data augmentations such as rotation, random cropping, and adding noise. Ultimately I did not implement these techniques for my final test of my algorithm because I wanted to first ensure that it was running and working appropriately before adding a layer of complexity. However, when I do implement augmentations, I plan to add blur, noise, and image rotations.

Another limitation to my results was that I only ran the algorithm for 30 epochs and I believe that the loss could have continued decreasing if I ran it longer. Like other challenges I faced, this came down to a lack of time to run training, but I do have a plan for working on this further. I will run my algorithm for 50 epochs and evaluate if the loss continues to decrease, and increase the number of epochs if it does. Alternatively, I can use the EarlyStopping callback (keras.callbacks.callbacks.EarlyStopping(monitor='val_loss'…) to automatically stop at the epoch where loss ceases to decrease.

Besides what I've discussed already, I have plans for future work. I will experiment with using a dice loss instead of a binary cross-entropy loss, as dice loss is another established and favorable loss function for segmentation that I have found in literature. I will also binarize my physical layer to truly understand what areas of my input images are deemed necessary and evaluate the improvement based on that. Finally, I will further explore the physical layer component of this project. I am interested in plotting filters and feature maps from the first layer and observe how and if these change for experiments with and without the physical layer, and with and without a binarized physical layer, and making direct comparison across metrics of loss, accuracy, and intersection over union.

Conclusion

In conclusion, I am motivated to use machine learning to segment vessels by the goals of my lab. I have implemented a U-Net with a trainable mask to work toward this goal. I have achieved metrics of 0.1975 loss, 0.4622 mean intersection over union, and 0.9244 accuracy after 30 epochs training with 134 images. Preliminarily I have shown some promising results when comparing the addition of a trainable mask to a model without the trainable mask. Training with only 20 images, I achieved training loss of 0.392 and the validation loss of 0.393, which was an improvement over the same model with only 20 images without the trainable mask, which yielded training loss was 0.567 and the test loss was 0.495. However, I will improve upon these conclusions in my future work. Moving forward I will use data augmentation and train for more epochs, binarize my mask and

<u>References</u>

1. Optical Imaging of Glucose Uptake and Mitochondrial Membrane Potential to Characterize Her2 Breast Tumor Metabolic Phenotypes. Megan C. Madonna, Douglas B. Fox, Brian T. Crouch, Jihong Lee, Caigang Zhu, Amy F. Martinez, James V. Alvarez and Nirmala Ramanujam. Mol Cancer Res July 1 2019 (17) (7) 1545-1555; DOI: 10.1158/1541-7786.MCR-18-0618
2. Hu, Fangyao et al. "Dark field optical imaging reveals vascular changes in an inducible hamster cheek pouch model during carcinogenesis." Biomedical optics express vol. 7,9 3247-3261. 5 Aug. 2016, doi:10.1364/BOE.7.003247
3. Bankhead P, Scholfield CN, McGeown JG, Curtis TM (2012) Fast Retinal Vessel Detection and Measurement Using Wavelets and Edge Location Refinement. PLoS ONE 7(3): e32435. doi:10.1371/journal.pone.0032435
4. Guo, Changlu & Szemenyei, Márton & Yi, Yugen & Wang, Wenle & Chen, Buer & Fan, Changqi. (2020). SA-UNet: Spatial Attention U-Net for Retinal Vessel Segmentation.
5. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234– 241. Springer, Cham, 2015.