Please do not post this final writeup to the course website.

**Neurosieve**

***Introduction***

It is common that researchers would like to trace neuron images to gain an idea and understanding of the neurons’ morphology. For example, understanding the morphology of neurons in the brain is necessary to understanding their functional role. In the past, researchers would have to trace and label each neuron by hand - a tedious and time consuming task. Recently, there have been several attempts to automate this tracing process through various software and tracing algorithms in hopes of reducing tracing time from several weeks to only a few hours. Unfortunately, due to variation in microscope quality, sample quality, and other factors, neuron images obtained using optical microscopy are often marred by background noise, reducing the performance of many automatic tracing algorithms. Neurosieve is an attempt to solve this issue by using neural networks to perform segmentation on the neuron images, removing background noise, and then running the tracing algorithms for better performance and accuracy.

***Group members and work division***

* Hasan Khan
  + Converted files .vp3d files to .tif image files and then was converted to .swc files (trace coordinate files)
  + Worked on image segmentation
* Crystal Gong
  + Wrote the architecture of the program using U-Net
  + Worked on image segmentation
* Puja Soni
  + Traced images using Vaa3d plugin (APP2)
  + Worked on determining accuracy
* Charu Mishra
  + Worked on tracing algorithms—SmartTracing and APP2
  + Worked on determining accuracy

***Milestones & Challenges***

Hasan retrieved all of the data that was used, which were originally in v3dpbd and swc files. He converted them into npy arrays, making it easier for us to use and manipulate. Converting the files was a challenge since it took a long time to figure out what format the swc files were actually in. The people who submitted the data to BigNeuron did not follow the formatting standards that the online documentation stated.

Crystal found a usable architecture that was simple, especially for medical image segmentation and converted it to be usable with 3d images. She used U-Net which is a fully convolutional neural network which skips connections of residual learning. Charu launched a P2 Instance on Amazon Web Services to run the project on because P2 was the best for machine learning and deep-learning. However, AWS charged much more money than what expected to run the GPU, and it went beyond the amazon educate amount. So, we had to stick to using the 2D algorithm instead. One way we tried to minimize the amount of time and power it took per run was by saving the numpy arrays created in the first part of the program in a separate file using Pickle so it didn’t have to be repeated run during development. Unfortunately this gave some more errors because Pickle doesn’t work well with large file sizes in Python 2.7. Hasan and Crystal then worked on getting the model to learn what pixels of each image are a part of the neuron and what pixels are a part of the background.

Puja and Charu researched and compared the tracing algorithms that were used in the original research paper to figure out which of the two algorithms (SmartTracing or APP2) should be implemented in the project. Originally SmartTracing seemed like a better idea because many reserachers had said that SmartTracing was able to perform better than APP2. APP2 was said to miss tracing in certain areas where the neuron seemed to be extremely thin. When it was time to implement on the neurons from the gold166 data set however, there didn’t seem to be a difference between the outputs of the two algorithms. They ended up choosing APP2 because it’s a built in plugin to Vaa3d (which is a program written by the owners of BigNeuron) rather than SmartTracing which was a third party plugin that had to be installed and run separately. There was a lot of data that had to be run with the APP2 algorithm, so they tried figuring out a way to get APP2 to work in batch. So in order to trace with APP2 for this project, we had to use the Vaa3D GUI and manually run the plugin on an image one at a time. In the paper, accuracy was computed based on averages using the 3D layers. Since we worked with 2D images in our project, there was no need to compute averages this way. Instead, the npy array values were compared and the accuracy of their matching was computed. Another challenge Charu faced was that on Windows 10 computers, Keras did not work on Python 2.7 and then on 3.5 running our program would cause a “Memory Error” on Python due to there being an insufficient amount of RAM available on her laptop for the files. This led to all coding to be done on her partner’s laptops as peer programming.

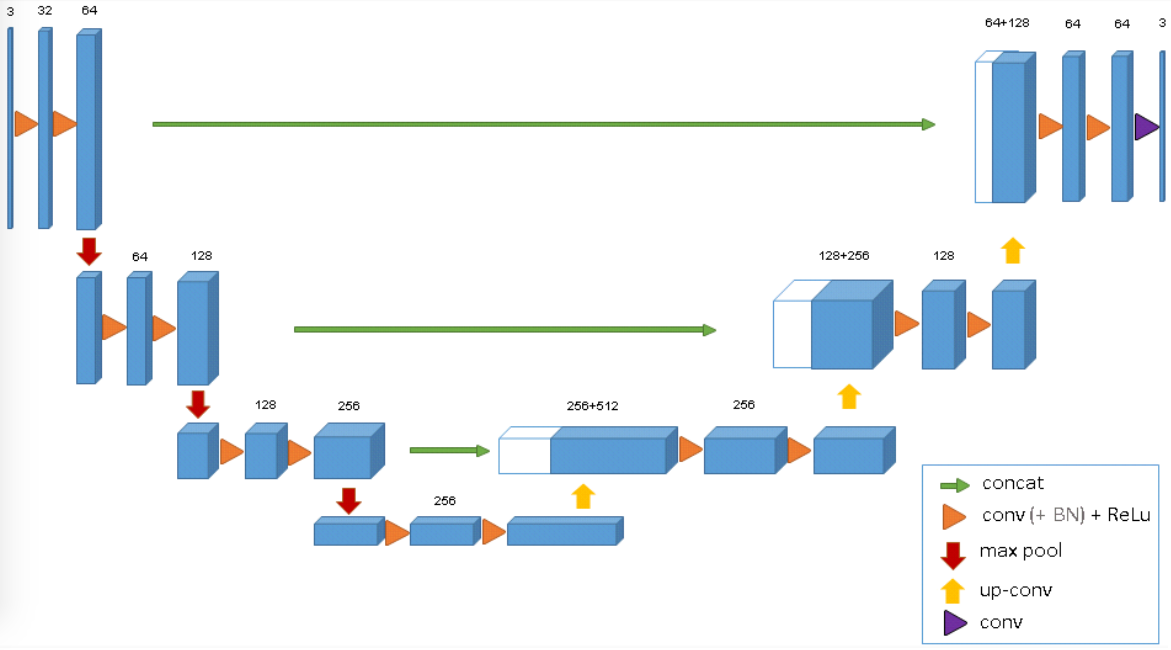
***Results***

* Figures for results, e.g. performance and confusion matrices of classifiers, images produced for graphics projects, other visualizations, etc.



* Explanations or figures for the architectures of neural networks used (if appropriate)





***Conclusions and Future Work***

* Conclusions

For future directions, we hope to implement data augmentation such as flipping, shifting, and recoloring the neuron images. Also, it would be interesting to use different tracing algorithms other than just APP2, such as SmartTracing, to gain more insight about the performance of these tracers. Also, rather than using the BigNeuron datasets, we would like to test Neurosieve on our own data, such as data from Crystal’s lab.

***References***

U-net 2d architecture from O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in MICCAI, pp. 234–241, Springer, 2015.

U-net 3D architecture from O. Cicek, A. Abdulkadir, S.S. Lienkamp, T. Brox, O. Ronneberger, *3D U- Net: Learning Dense Volumetric Segmentation from Sparse Annotation*, 2016, [online] Available: .