Neuronal Shape Classification In Calcium Imaging Data

Shangying Jiang[1], Yang Sun[1], Yidi Zhang[1], Yiran Xu[1], Zhaopeng Liu[1], Andrea Giovannucci[2], Eftychios A Pnevmatikakis[2]

[1]Center of Data Science, New York University, New York, NY, USA, 10011; [2]Simons Foundation, New York, NY, USA, 10010

Abstract

In this project, three convolutional neural network (CNN) based approaches were applied to classify spatial components of mice's neuron cells based on the calcium transient images. Our base model achieves 80% accuracy, VGG-like model achieves 90% accuracy, Resnet model achieved 89% accuracy.

Introduction

High-resolution monitoring of brain activity in behaving animals has recently been made possible by calcium imaging. Analyzing such data entails the identification of the location and the activity of the neurons and processes within the images [1]. Our major task would be to build models to classify calcium images into four classes: neurons, doubtful neurons, processes and noises. We first relabeled components from two classes (neuron and non-neurons) into four classes; then trained various models to do classification. By analyzing incorrectly classified components, we improved our labeling and retrained our models.

Data

We obtained 9028 50×50 pixels gray-scale images from calcium imaging videos that monitored large neuron populations. Each image contains one component that we wish our classifier can recognize correctly. We generated a label and a 50×50 numpy array that contains pixel values for each component. For our project, the labels are target variables. Originally the components were labeled into one of two classes: neurons and non-neurons. For better practicality, we divide non-neurons class into two classes: processes and noises since we want to train a classifier that can also recognize dendrites. To prevent confusing classifier, we labeled all dubious components as doubtful neuron. Figure 1 shows some examples of the four classes (neuron, process, doubtful neurons and noises).

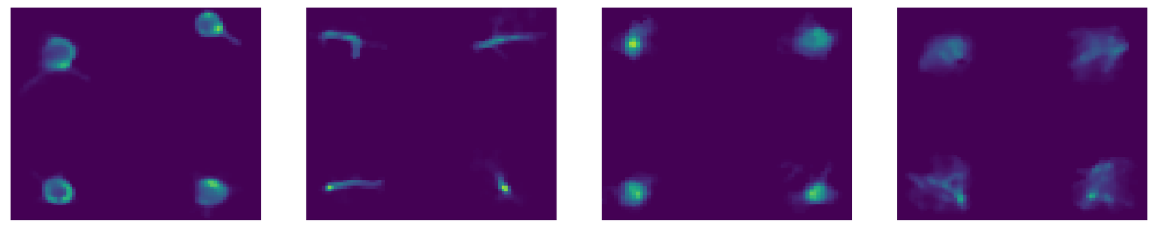

Figure 1: Example for components (neurons, processes, doubtful neurons and noises from left to right)

Existing Work

Previous to our work, our mentors have trained a CNN based model to classify components into two classes (Neuron, and Non-neuron). The accuracy for two-class classification is 95%.

Baseline Model

CNN has been proved very useful in image classification and thus, CNN is chosen as our baseline model. Like regular neural nets, CNN consists of layers of neurons that have learnable weights and biases. The difference is that neurons of CNN have three dimensions: width, height and depth. In terms of layers, CNN has three main types of layers: Convolutional Layer, Pooling Layer, and Fully-Connected Layer. These layers are stacked to form a CNN architecture. Figure 2 shows the architecture of our baseline model.

Very Deep Convolutional Neural Network (VGG-like Architectures)

By adding layers to our baseline model, VGG-like models have gained higher validation accuracies from different architectures. The specific configurations are shown in figure 3. The constructed architectures were inspired by [2], but the differences lie in the sizes and the number of channels of the input images. We used grayscale smaller-size images so that the features within them are much fewer than those used in [2]. We added layers until the filter number reaches 256, and 4 pooling layers were used in the VGG-like models. The optimal model achieved the best result with 16 layers, and the accuracy started to drop with more layers.

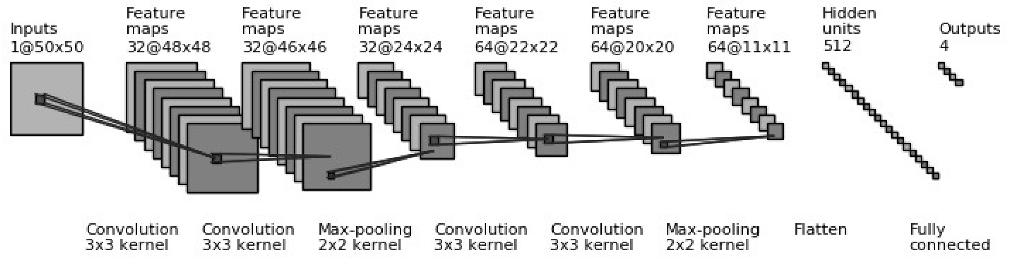

Figure 2: CNN architecture used in the baseline model

ConvNet Configuration		
13 weight layers	16 weight layers	19 weight layers
input: 50 * 50 image		
conv - 32	conv - 32	conv - 32
conv - 32	conv - 32	conv - 32
conv - 64	conv - 64	conv - 64
conv - 64	conv - 64	conv - 64
maxpool & dropout		
conv -128	conv -128	conv -128
conv -128	conv -128	conv -128
conv -128	conv -128	conv -128
		conv -128
maxpool & dropout		
conv - 256	conv - 256	conv - 256
conv - 256	conv - 256	conv - 256
	conv - 256	conv - 256
		conv - 256
maxpool & dropout		
conv - 256	conv - 256	conv - 256
conv - 256	conv - 256	conv - 256
	conv - 256	conv - 256
		conv - 256
maxpool & dropout		
FC - 512	FC - 512	FC - 1024
FC - 4	FC - 512	FC - 1024
	FC - 4	FC - 4

Figure 3: Configurations of VGG

Residual Neural Network (ResNet) Architecture

Although VGG-like models can produce very good results for large scale image classification, training time increases significantly with increased depth. This problem can be addressed using a deep residual learning framework. The core idea of ResNet is providing shortcut connection between layers, which make it safe to train very deep network to gain maximal representation power without degradation problem.

Evaluation

We begin with evaluating the performance of individual models at a single scale. The test set is of size 500. The evaluation metric we use is accuracy, which shows how often predictions have maximum in the same spot as true values. According to the previous work, local response normalization does not improve model performance so we do not employ any normalization in the architectures.[3]

Result & Conclusion

Table 1: Accuracy Result Table

Model	3 classes	4 classes
Baseline Model	0.9	0.80
13-layer VGG	NA	0.895
16-layer VGG	NA	0.903
19-layer VGG	NA	0.903

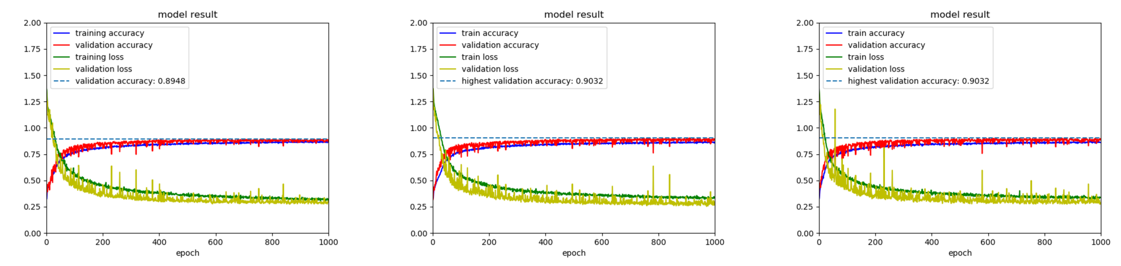

Figure 4: Learning curves of VGG-like models (from left to right: 13 layers-VGG like, 16 layers-VGG like, 19 layers-VGG like)

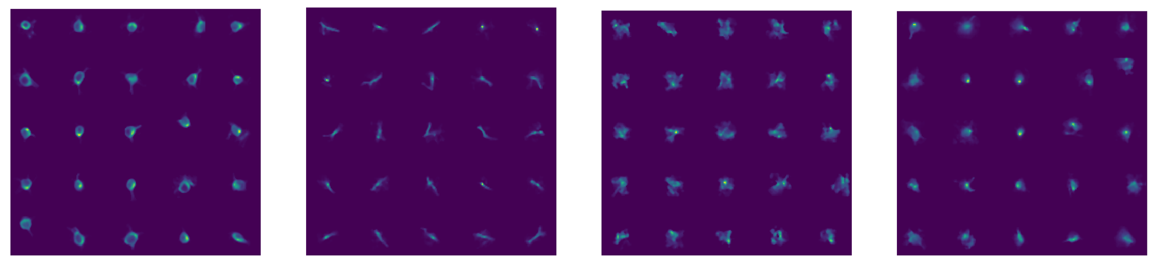

Figure 5: Example of well-classified neurons, processes, noises and doubtful neurons(from left to right)

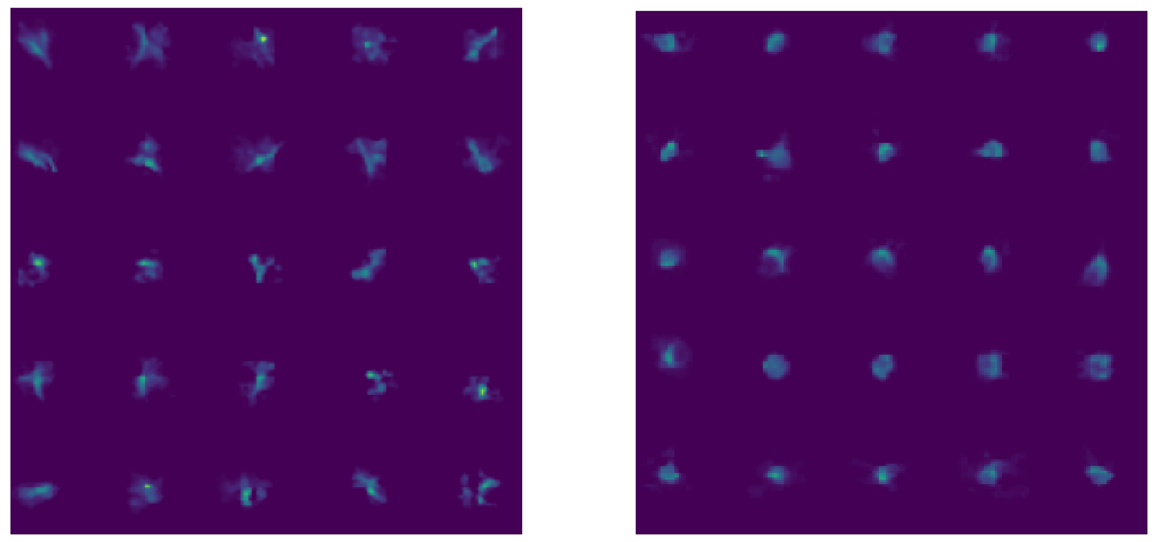

Figure 6: process classified as noise & doubtful neuron classified as clear neuron (from left to right)

Reference

[1] Pnevmatikakis, E. A., et al., Simultaneous denoising, deconvolution, and demixing of calcium imaging data. 2016

[2] Simonyan, K. and Zisserman, A., Very deep convolutional networks for large-scale image recognition. 2014

[3] He, K., et al., Deep residual learning for image recognition. 2016