Automated Stitching and Spine Detection of Spiny Projection Neurons

Albarran, E. and Sun, X.

CS 229: Machine Learning - December 9, 2015

Introduction

A complex problem in neurobiology, and microscopy in general, is image processing of neuronal processes. In particular, neuroscientists often find themselves in the position of having gigabytes of high-resolution images (e.g. confocal, 2-photon microscope images) of neuronal processes, but are then met with the burdensome task of manually analyzing their data. A common example of this is having to stitch together images of several compartments belonging to the same cell, and manually counting the number of dendritic spines (a biologically relevant protrusion) for each one of the hundreds of imaged cells. Although some software exists to help with extracting information such as dendritic length or spine densities, the process is very much still limited to manual tracing, counting and analyzing. Here we apply state of the art computer vision and machine learning techniques to approach this issue so as to process and analyze images more automatically and efficiently.

Specifically, our current project takes in as input a collection of 3D TIF stacks (raw image files of dendritic processes taken with 2-photon microscopy), processes them through filtering and feature detection, passes the processed data through a random forest classifier (*ilastik* framework) that yields probability maps, which are then used to predict the number of spine [objects] within the data set. Therefore, the entire process is a means of automatically processing and extracting the number of dendritic spines from a collection of raw 2-photon imaging files.

Related Work

In order to precisely extract spatial morphology information from a set of tiles, they need to be stitched together with incredibly high precision, therefore, many labs still rely on manually aligning their tiles to reconstruct the whole-cell morphology. For example, in Prof. Jun Ding's lab (E. Albarran), tiles are manually stitched together based on the (x, y) position data pertaining to the header files of each image stack. Dendritic spines are then manually counted by eye. The entire process is very long.

Automatic counting of dendritic spines is a classification problem, and there have been numerous attempts before ours. Herzog et al. (1997) attempted 3D reconstruction of dendritic trees using a parametric model of cylinders, but this method was not robust to the various morphologies that spines can take (i.e. great variety in spine neck width, length, etc.). Similarly, Al-Kofahi et al. (2002) used a generalized cylinder model to estimate local directions (gradients), and then dendritic backbones were directly traced using pre-estimated seed points. However, this method yielded an incomplete extraction due to the incomplete initial estimation of the seed points. Using a more geometric approach, Koh et al. (2002) constructed a median axis to locate dendrites and then detect spines as surface protrusions relative to those backbones. However, since the segmentation is achieved by a simple global threshold method, and very limited geometric information is considered for spine detection, this method detects a lot of pseudo spines. Using a more statistical approach, Zhang et al. (2007) utilized a curvilinear structure detector and linear discriminant analysis classifier, but their implementation was not very efficient

(required thousands of training sample images to yield decent results) and is again susceptible to variability in spine morphology.

In summary, previous attempts at addressing this problem have ranged from utilizing geometric fitting to machine learning approaches, but none have succeeded in terms of *both* accuracy and efficiency. The two algorithms we utilize here, SIFT and *ilastik* (i.e. random forest classification) have been used extensively before in many contexts (e.g. city panorama stitching and counting animal cells, respectively), but here we combine them in the new context of two-photon dendritic spine images.

Dataset

Our data set consists of 2-photon microscopy 3D stack images of dendritic branches of spiny projection neurons (SPNs) in the mouse dorsolateral striatum. These stacks each contain 30 frames of 512x512 pixel 'tiles' of dendritic segments (~10 micron^2). The tiles' (x, y) coordinates themselves overlap with each other to form a complete tracing of individual branches of an SPN neuron, where each neuron consists of a collection of 10-15 sets of overlapping tiles. Image data was acquired by E. Albarran and Yu-Wei Wu (postdoctoral fellow).

The first step of pre-processing of the data consists of denoising the image stacks. For each frame in the stack, we apply 2D median filtering (Lim, 1989) to remove the "salt and pepper" noise common in 2-photon microscopy, but maintain boundary information for the dendrites and spines. The next step involves transforming each 3D stack into one single 512x512 2D tile. This is done by taking the max projection (the max intensity value at each pixel location) across all 30 frames in the stack. The result is a single 2D tile image (referred to from here on out as a 'tile'). In total, the dataset consisted of 1092 of these tiled projections, spanning across a total of 12 cells (imaged from 4 mice). Larger, full-cell images were then created by stitching together the 10-12 tiles corresponding to the cell (see methods section). 15 tiles were used for training the classifier, and testing was done on the 12 fully-reconstructed images. Additionally, the 12 full-cell images were previously manually stitched and their spines manually counted to use as a comparison for the spine numbers predicted by our classifier.

Methods

The first step was to reconstruct the whole-neuron images from the tiles. In order to do this, we implemented the Scale-Invariant Feature Transformation in order to extract features to use for tile matching. SIFT combines a feature detector and descriptor. The algorithm consists of the following: for each tile, (1) construction of gaussian scale space, (2) taking the difference of gaussians and locating the extrema, (3) subsampling potential feature points, (4) filtering out edge and low contrast, (5) and assigning keypoint orientations / building the keypoint descriptors. The last step yields a set of descriptor objects (i.e. features), which are then used to determine matching points between tiles.

For matching, we utilized a RANSAC-based algorithm to determine homology of features (Fischler and Bolles, 1981). Similar feature descriptors (based on RANSAC least-squares estimate of homography) between the two tiles are paired up, and the relative distances between the selected feature pairs are computed. Of these pairs, we then select the subset of distance vectors that meet the threshold criteria (we use avg. slope std < .05 and avg. length std < 100px), and using those distance vectors, we translate one tile onto the other. The entire matching algorithm consists of the following algorithm: (1) place the center tile (pertaining to the soma), (2) stitch the next tile, (3) the stitched tile is now the new

center tile, (4) loop to step 2 until there are no more tiles left. This algorithm results in a complete, feature-based reconstruction of the neuronal image.

After the whole-cell images were stitched (12 total), we then aimed at detecting and counting dendritic spines from the stitched images. We first used the random forest classifier algorithm implemented in by the *ilastik* framework (Sommer et al., 2011) for object identification. This process involved three steps: pixel classification, object segmentation, and density counting.

For pixel classification (fig.1a,b), we first did manual annotations (labeling) of dendritic spines, dendrite processes and background, on 15 of our unstitched tiles. Automatic classification could be done in batch mode with test images, where the classification rule is learned from the training images. Each pixel in the image has a feature vector based on what features are selected for classification. The weight of each feature in the feature vector is learned from the labeling of the training images. *ilastik* includes a convenient annotating GUI that can be used to label arbitrary numbers of classes in the images. These labels, along with a set of generic (nonlinear) image features, are then used to train a the random forest classifier for both object classification and counting.

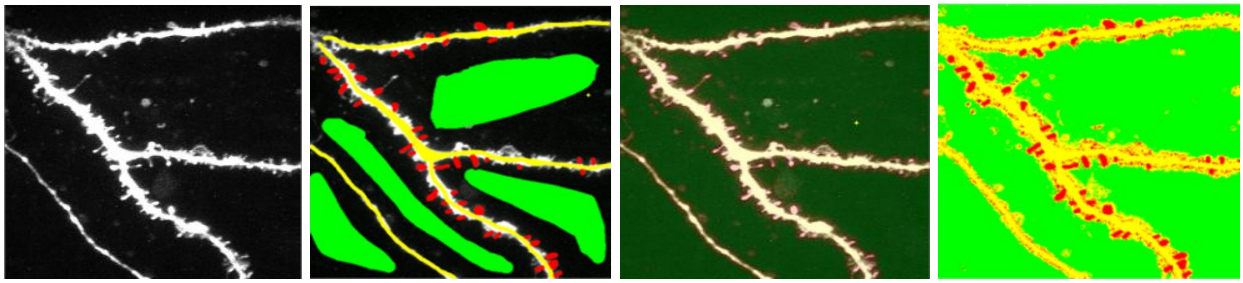
More specifically, when training the pixel classifier, we used the following features for feature selection: for *color features*, gaussian smoothing to get rid of as much background noise as possible, but preserve the dendritic spine. *Edge features:* Gradient Magnitude of Gaussian, euclidean norm of the first derivative in each direction. High value assigned to edges and small value assigned to non edges. *Laplacian of Gaussian (approximated by difference of Gaussian):* second derivative in each direction. High value assigned to edges and small value assigned to non edges. One value per pixel. □So the number of features equals the length of the feature vector of each pixel.

The pixel classification step generated probability maps (fig.1c) as inputs for the following steps, namely, object segmentation and classification (fig.1d)**:** this step transformed a probability map (generated by pixel classification) into a segmentation by Gaussian smoothing and thresholding by a size filter. With object segmented, we then selected the features for object classification, assigned the feature vector to each object and then classified the object. Because we don't need to classify dendritic spines, here we classified objects as either dendritic spines or non-spine circles, rather than different subtypes of dendritic spines. The classification was done by random forest algorithm.

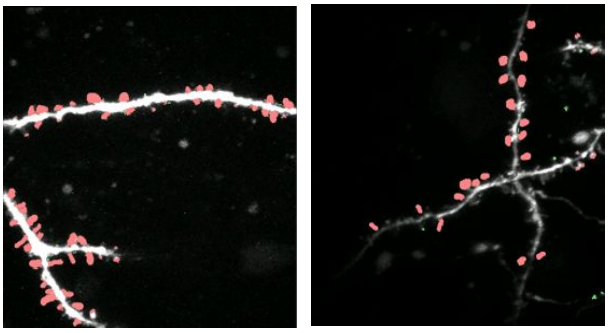
The last step is density counting which is done with the segmentation images as inputs. First we labeled the segmentized dendritic spines as the object for counting. Then we used random regression forest (based on sklearn, with number of trees and depth of each individual tree adjustable) to compute the density (number) of objects (here it is dendritic spine).

The RF algorithm we used is as follows (Breiman, 2001): RFs grow many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest). Each tree is grown as follows:

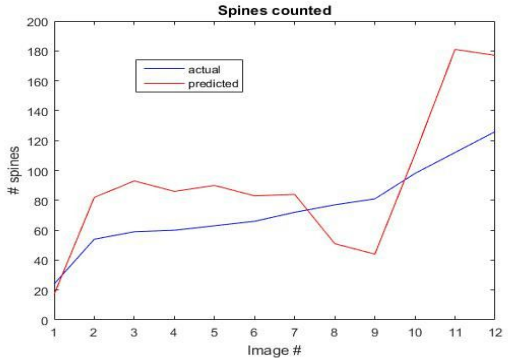
1. If the number of cases in the training set is N, sample N cases at random - but *with replacement*, from the original data. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible.

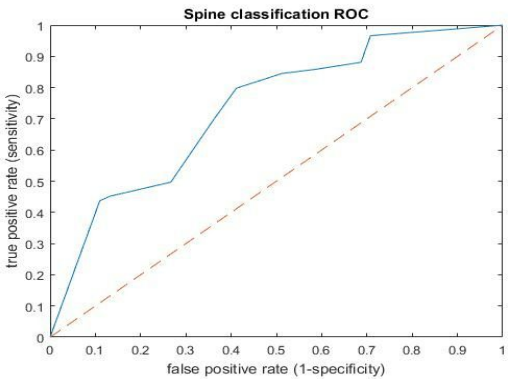


Results and Discussion



Classification resulted in a predicted labeling of dendritic spines (fig.2). These labelings were extracted via transforming the classified images into binary images and then using the Matlab function *bwboundaries* which returns 'pixel blob' object handles. The number of objects returned then corresponds to the number of spines detected. Comparing the number of predicted spines to the manually-counted spine images yields a performance measure of our classification. We first order the 12 test images by number of manually ('actual') spines, and plot the number of predicted spines yielded by our classifier (fig.3). It can be appreciated that the general trend is captured by our classifier (more actual spines -> more predicted spines). However, due to the small sample size, we are unable to perform more sophisticated validation such as LOOCV. Collecting more cell images will allow us to perform stronger validation.





Through ROC analysis (fig.4) we were able to get a sense of the sensitivity-specificity tradeoff of our classification, as well as fine-tune it. Although random forest is based on a "largest vote" classification process, the ROC curve was calculated by placing a threshold (0.1 - 0.9) on the posterior probabilities (classifications) yielded from the random forest and determining the true positive and false positive classifications. From the ROC curve it is clear that the classifier yielded mostly weak (< 0.5) or moderate-strength (< 0.75) classifications. We attribute the low of high TP:FP to the sample size of our training. Using the ROC curve, we set our optimal threshold to be 0.4, which still results in a TP rate of ~0.8 and a FP rate of ~0.4.

For spine identification and counting, we trained our classifier with a dataset of 15 images and tested with 12 images. Comparing actual spine numbers and the predicted numbers, we concluded that the prediction result is reliable, though not precise enough. At the very least, the number of predicted spines is in positive proportion to the real number of spines. If we increase the size of training set in the future, we expect that this automatic counting algorithm can work much better.

Conclusion / Future Work

Our stitching process worked very well to reconstruct the tiled two-photon images of neurons, and this automatic stitching algorithm is much faster than current nearly-manual method. It currently takes ~30mins-1hr to manually stitch the images and count spines. Our automated performance takes ~2mins from start to finish. In the future, we plan to optimize the image pre-processing parameters (e.g. the parameters for Gaussian smoothing, image erosion and dilation, etc.) to better 'de-noise' the images and thus achieve better performance. We expect that both this combination of SIFT and classification, after being adjusted to specific datasets for specific purposes, can be applied to larger-scale two-photon image data processing in our own labs, as well as other labs doing similar analysis.

As a future direction, we plan to do functional analysis of the dataset. Based on the stitched images and identified objects, we will use accompanying Ca2+ imaging data to analyze the locations/distributions of the upstream inputs (from sensory or motor cortical neurons, and thalamic inputs) that synapse onto the identified dendritic spines, the morphology of the spines themselves, and even the functional dynamic properties (by applying PCA/FA to get low-dimensional state trajectories of the high-dimensional neural activity of spines). We are planning to characterize and compare the structural and functional features of motor, sensory, and thalamic inputs from cortical cells and thalamic nuclei onto striatal spiny projection neurons.

References:

- VLFeat. http://www.vlfeat.org/index.html
- A. Herzog, G. Krell, B. Michaelis, J. Wang, W. Zuschratter, A.K. Braun (1997). Restoration of three-dimensional quasi-binary images from confocal microscopy and its application to dendritic trees. Proc. SPIE, 2984: 146-157, Three-Dimensional Microscopy: Image Acquisition and Processing IV Bellingham, WA.
- K.A. Al-Kofahi, S. Lasek, D.H. Szarowski, C.J. Pace, G. Nagy, J.N. Turner, B. Roysam (2002). Rapid automated three-dimensional tracing of neurons from confocal image stacks. IEEE Trans. Inf. Technol. Biomed., 6: 171-187.
- I.Y.Y. Koh, W.B. Lindquist, K. Zito, E.A. Nimchinsky, K. Svoboda (2002). An image analysis algorithm for dendritic spines. Neural Comput., 14: 1283-1310.
- Zhang et al. Dendritic spine detection using curvilinear structure detector and LDA classifier (2007). NeuroImage, 36(2), 346-360.

- C. Sommer, C. Strähle, U. Köthe, F. A. Hamprecht (2011). ilastik: Interactive Learning and Segmentation Toolkit in: Eighth IEEE International Symposium on Biomedical Imaging (ISBI).
- J.S. Lim. Two-Dimensional Signal and Image Processing (1989). Prentice Hall PTR.
- L. Breiman. Random forests. Machine Learning(1989), 45(1): 5–32.

Acknowledgment:
We thank **Yu-Wei Yu** (lab of Jun Ding) for help in data acquisition, **Saurahb Vyas** (lab of Scott Delp) for suggestions on mosaic stitching, **Irene Kaplow** (TA) for advice with image processing and project direction, and Prof. **Andrew Ng**.