Accelerating Neuronal Genetic Research in *C. elegans* with Computer Vision & Machine Learning

Roshni Cooper
Stanford University
385 Serra Mall, Stanford, CA 94305
rccooper@stanford.edu

Abstract

This final project for CS229 implements a software system to classify C. elegans *as wild-type or mutant based on biological fluorescence images of synapses, the connections between neurons and muscles, in the worms. The project uses features of two neurons in the images, the DA8 and DA9 motorneurons, which were calculated as part of the CS231A final project. Using these features, the software trains an SVM using a Radial Basis Function kernel. The software was cross-validated on over 150 images with 75% accuracy. The input images are very noisy and blurry, so human accuracy can be as low as 60%. In comparison, the software is precise.*

1. Introduction

1.1. Motivation

For the CS231A and CS229 final project, I developed software to analyze biological fluorescence images of synapses, the connections between neurons and muscles, in the nematode *C. elegans*. Synapses appear as lines of dots, with varying intensity and spacing. Figures 1a and 1b show images of the DA8 and DA9 motorneurons, which I will be focusing on for this project. Each motorneuron signals to different muscles in the worm, and together they help the worm move. The DA9 synapses are tagged with two fluorescence proteins, GFP (green fluorescence protein) and mCherry (a red fluorescence protein). When excited, the green and red fluorophores in the DA9 synapses become visible, and the synapses appear yellow. The DA8 synapses are tagged only with one fluorescence protein, so the red dots in the image represent the DA8 synapses. Figure 1a shows a normal, wild-type worm. The yellow and red dots are distinctly separated in this worm [14]. On the other hand, Figure 1b shows a mutant worm, with overlapping DA8 and DA9 synapses. The overlap causes the same muscles to receive neurotransmitters from two different motorneurons. These potentially conflicting signals can hamper the movement of the worms. Understanding the genes and biological mechanisms that cause the motorneurons to develop incorrectly, as in in Figure 1b, can ultimately help to explain neuromuscular function and diseases.

To perform this genetic research, biologists image hundreds of individual worms, and then, based on various visual characteristics of the synapses, they decide whether the animal is wild-type or mutant. These characteristics include the intensity of the synapses, the position of synapses, etc. If the worm is a mutant, its genome is studied further to learn more about the genetics behind the nervous system. Currently, biologists look at hundreds of randomly mutagenized worms and search for mutant animals to set aside. Because the decision is made manually, the process is tedious, subjective, and error-prone. Hence, I developed software to use computer vision and machine learning to identify the mutant worms. I focused on identifying animals that have a muta-

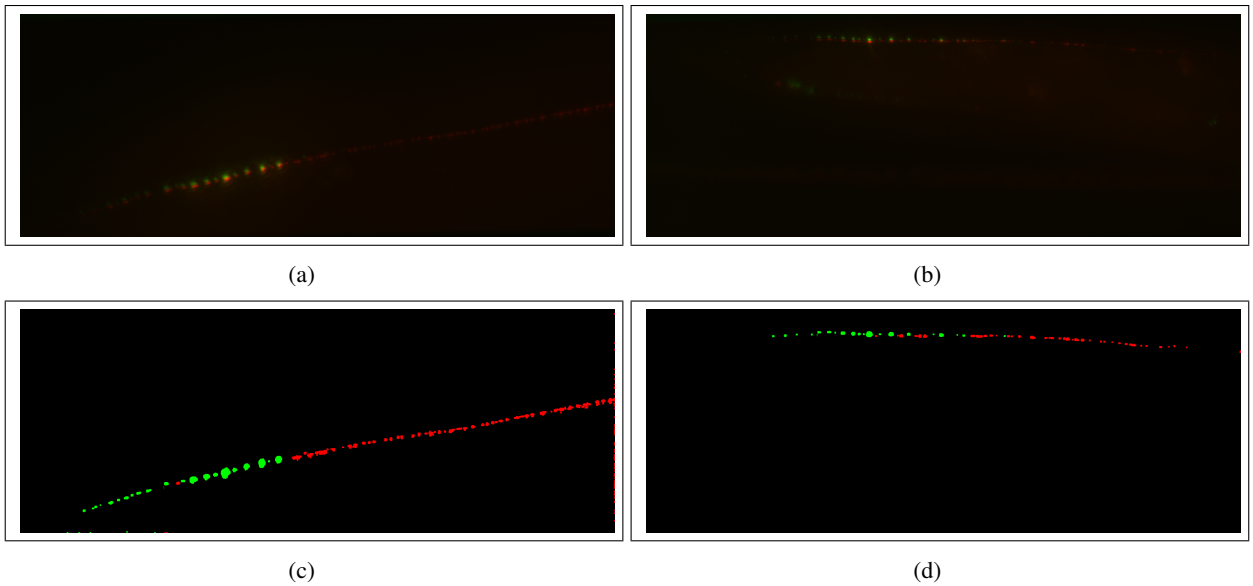

Figure 1: Example input images and corresponding output masks: (a) Wild-type input image. (b) Mutant input image. (c) Wild-type output synapse mask. (d) Mutant output synapse mask.

tion in the gene plx-1, such as the worm in Figure 1b. In the future, this software could be used to identify other mutants as well.

1.2. Problem Statement

The project comprises two main tasks: 1. identifying the synapses in an image, and 2. using the synapses to determine whether the animal is a wild-type or a mutant. Part (1) is a computer vision problem and was implemented for the CS231A final project, which will be briefly mentioned in this report. Part (2) has been completed for the CS229 final project and will be detailed in this document.

1.3. Technical Approach Overview

To solve the problem of identifying the synapses, the software first identifies which pixels belong to the synapses of the DA8 and DA9 neurons. Then it extracts various features from the images, such as the length of the synaptic region or the mean intensity of the synapses, and uses those features to train an SVM using the Radial Basis Function kernel.

1.4. Related Work

With advancements in fluorescence imaging in the last few decades, biological research has become increasing visual. Biologists are learning more and more about, for example, the nervous system by pouring over countless images. Surprisingly, computer vision and machine learning are still largely unused by most biologists [6]. Some such techniques have been applied to neuronal research, and these approaches involve using SVM to detect synapses in an image, as in [12], and using SVM and PCA to determine whether the animal is wild-type or mutant, as in [5] and [3]. These techniques have been shown to be reliable given a large set of training data.

2. Technical Approach

2.1. Synapse Extraction

Figure 2 shows a high-level block diagram of the software for synapse extraction portion of the project (the CS231A project). Synapses are found separately in the red channel, using linear filtering, and the green channel, using K-means clustering. The resulting images of the synapses in the two channels are combined to determine which

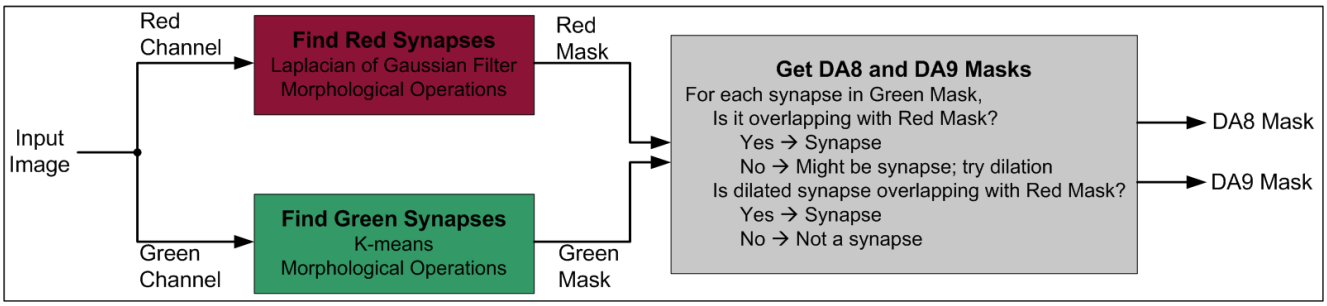

Figure 2: Final Project Block Diagram.

synapses belong to the DA8 motorneuron and which belong to DA9, yielding two mask images as output. For more details, see the final report for CS231A [4]. Figure 1 shows two input images (1a and 1b) with their corresponding output synapse masks (1c and 1d).

2.2. Image Classification

2.2.1 Features

The first step in performing the image classification is selecting which features to use. This portion of the code uses information from both the original image and the mask generated in the previous section. The features of interest are shown in Table 1. Each of these features were calculated for four synaptic regions The most obvious three regions are all of the synapses (both DA8 and DA9), only the DA8 synapses, and only the DA9 synapses. The features were also calculated for the overlapping region at the border of the DA8 and DA9 synaptic regions. Because of variations in worm size, imaging intensity, etc., the ratios of each of the features from the overlap to the three first regions were also calculated. For example, the ratio of `totalArea` of the overlapping region to `totalArea` of the red synaptic region was another feature. Taking the ratio helped to normalize the overlap from image to image, giving a better sense of "how much" overlap a worm's neurons actually exhibited.

Once the features were selected, each feature was scaled to the range $[-1, +1]$ in order to give each feature equal weighting in the training process [1]. The feature scaling was performed on the entire data set as a whole, since there was no specific training data versus test data.

The data set, which is described in greater detail in later sections, contained 153 images. Since 42 features seemed like a large number of features given the data set [8], I tried to reduce the number of features to avoid any overfitting of the data. I tried two methods for reducing the number of features. First, I ran cross-validation with each feature individually. Then I sorted the features based on accuracy, i.e., $\mathcal{F} = \{f_1, f_2, \ldots f_{42}\}$, where the accuracy with f_i is greater than the accuracy with feature f_{i+1}. Next, I ran cross-validation with both $\{f_1, f_2\}$, then with $\{f_1, f_2, f_3\}$, etc., and kept adding features. The intention was to stop adding features when the accuracy of the cross-validation was no longer increasing. Adding features kept increasing the accuracy, though, so I tried another method of feature reduction. Instead of sorting the features based on the cross-validation accuracy they yielded, I sorted them based on the variance of the feature vector. Again, adding more features into the mix only improved the accuracy, so I kept all 42 features for training the SVM.

2.2.2 Model Parameters

The SVM to classify the images was created using LIBSVM [2] and LIBLINEAR [7]. The linear and polynomial kernels yielded worse accuracy than the Radial Bias Function kernel (essentially a Gaussian kernel). Furthermore, it has been shown that the RBF kernel can emulate the performance

Feature	Description	Calculated on...	Using ...
`num`	Number of Synapses	Output Mask	MATLAB's `bwlabel`
`totalArea`	Total Area Occupied by Synapses	Output Mask	MATLAB's `regionprops`'s "Area" property
`meanArea`	Mean Area Occupied per Synapse	Output Mask	`totalArea` divided by `num`
`totalIntensity`	Total Intensity of All Synapses	Input Image & Output Mask	MATLAB's `regionprops`'s "MeanIntensity" property multiplied by `num`
`meanIntensity`	Mean Intensity Per Pixel of All Synapses	Input Image & Output Mask	`totalIntensity` divided by `totalArea`
`len`	Distance Between Terminal Synapses	Output Mask	Euclidean distance between leftmost synapse and rightmost synapse

Table 1: Features

of both linear and sigmoidal kernels, so it encompasses multiple models [9], [11].

The parameters of the RBF kernel are C and γ. A grid search was used to try 1600 combinations of C (logarithmically spaced from 1 to 10^{10}) and γ (logarithmically spaced from 10^{-10} to 1). Since there was no separate training set from the test set of data, three-fold cross-validation was used to calculate the accuracy of the SVM. Hence the options string input to the function `svmtrain` in MATLAB was `'-c 16237.7674 -g 0.0023357 -v 3'`.

3. Experiment

3.1. Data Set

The software was evaluated on images of both wild-type and mutant worms captured in my research lab. I grew wild-type and *plx-1* mutant worms, and captured those images using a microfluidics system developed at Georgia Tech [3]. This system allows the user to image multiple worms per minute, which is faster than the current process of selecting individual worms and placing them on slides.

Due to the size of the microfluidics system imaging chamber, the worms need to be the same age when imaged. *C. elegans* grow very quickly (their lifespan is less than a week), so special precautions need to be taken to synchronize the worms' ages. Worms are bleached to isolate their eggs, which are placed on Petri dishes with bacteria to begin growing at the same time. Then the worms are sent through the microfluidics system to be imaged. I imaged over 150 worms over multiple imaging sessions (see Table 2). Then I used those images to train and test my software.

Wild-type Images	78
Mutant Images	75
Total Images	**153**

Table 2: Data Set

Cross-Validation Accuracy	75%
Human Accuracy	58%
Accuracy Improvement	**29%**

Table 3: Accuracy

3.2. Evaluation

3.2.1 Accuracy

According to biologists who are familiar with the microfluidics system, approximately 30%-50% of the images captured with the system are considered usable for various reasons [10]. Sometimes the worms are rotated, or other tissues block the view of the synapses. Sometimes the worms are still moving during imaging, so the output images are too blurry to use. The human accuracy of 58% for classifying these images corroborates this statistic (see Table 3). In comparison, the 75% cross-validation accuracy of my software is nearly a 30% improvement over the status quo.

3.2.2 Speed

A key benefit of my software over the current manual method of classifying *C. elegans* is the increase in speed. The fluorescence molecules that are imaged are susceptible to photobleaching when they are excited by the fluorescent light in the microscope. The photobleaching causes an exponential decay in intensity [13], so it is critical to make decisions about the worm quickly. In particular, when the biologists are classifying the worms, if they are unable to classify them before the fluorescence molecules fade, my software can still make those decisions accurately.

To find the synapse masks manually, the user needs to click on each pixel in synapse and classify which neuron it belongs to, which is an incredibly tedious process. Then, even if the biologist is not manually calculating all of the features mentioned above, the user still needs to take time to consider such a high-dimensional problem. Conversely, using my software, synapses can be identified in an image, on average, within 4 seconds. I used MATLAB's Profiler to track the amount of time spent on processing each image in my data set. Training the SVM takes time, but obviously that step only needs to be completed once, and does not increase the time to classify a worm at the time of imaging.

4. Conclusion

For the CS229 final project, I implemented software to classify fluorescent biological images of synapses in *C. elegans*. The algorithm trained an SVM with a Radial Basis Function kernel and has 75% cross-validation accuracy, which is very high given that biologists only consider about half of the images captured with the microfluidics system useful. The software also incorporates work completed for the CS231A final project, which identifies synapses in the noisy input images. The entire software package can classify an image quickly (on average within four seconds), making it not only more accurate, but also much faster, than a biologist.

References

[1] C.-C. C. C.-W. Hsu and C.-J. Lin. A practical guide to support vector classification. 2010.

[2] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[3] K. Chung, M. Crane, and H. Lu. Automated on-chip rapid microscopy, phenotyping and sorting of *C. elegans*. *Nat Meth*, 5(7):638–643, 2008.

[4] R. Cooper. Accelerating neuronal genetic research in *C. elegans* with computer vision & machine learning. CS231A Final Project Report, 2012.

[5] M. M. Crane, J. N. Stirman, C. Ou, P. T. Kurshan, J. M. Rehg, and K. S. H. Lu. Autonomous screening of c. elegans identifies genes implicated in synaptogenesis. *Nat Meth*, 9(10):977–80, 2012.

[6] G. Danuser. Computer vision in cell biology. *Cell*, 147(5):973–978, 2011.

[7] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[8] J. Hua, Z. Xiong, J. Lowey, E. Suh, and E. R. Dougherty. Optimal number of features as a function of sample size for various classification rules. *Bioinformatics*, 21(8):1509–1515, 2005.

[9] S. Keerthi and C. Lin. Asymptotic behaviors of support vector machines with gaussian kernel. *Neural computation*, 15(7):1667–1689, 2003.

[10] P. Kurshan. Personal communication, 2012.

[11] H. Lin and C. Lin. A study on sigmoid kernels for svm and the training of non-psd kernels by smo-type methods, 2003.

[12] A. S. Miguel. Personal communication, 2012.

[13] N. B. Vicente, J. E. D. Zamboni, J. F. Adur, E. V. Paravani, and V. H. Casco. Photobleaching correction in fluorescence microscopy images. *Journal of Physics: Conference Series*, 90(1):012068, 2007.

[14] J. G. White, E. Southgate, J. N. Thomson, and S. Brenner. The structure of the ventral nerve cord of caenorhabditis elegans. *Philos Trans R Soc Lond B Biol Sci*, 275(938):327–48, 1976.

Honor Code Acknowledgements

The software developed for the CS231A and CS229 final projects will be used in my research lab at Stanford (Prof. Kang Shen's lab in Biology). The synapses were extracted for the CS231A final project, and then the images were classified for the CS229 final project. I worked by myself so all of the completed code was implemented by only me.