1 INTRODUCTION

The goal of this component of the project is the development of an automated pipeline for the quantification of microscopy images of muscle fiber cross section. Human analysis of these images tends to be slow and prone to inconsistencies thereby introducing errors into the subsequent analysis related to number, size, shape, and other properties of the cells. Toward that end, we consider images such as that shown at the start of the pipeline in Figure 1 where healthy membranes present as bright white, the cytoplasm as gray, and the space between the cells as black. In regions of the images without highlighted membranes, the human eye can still easily detect the boundaries of cells. The diameters of cells are usually between 5 pixels and 100 pixels.

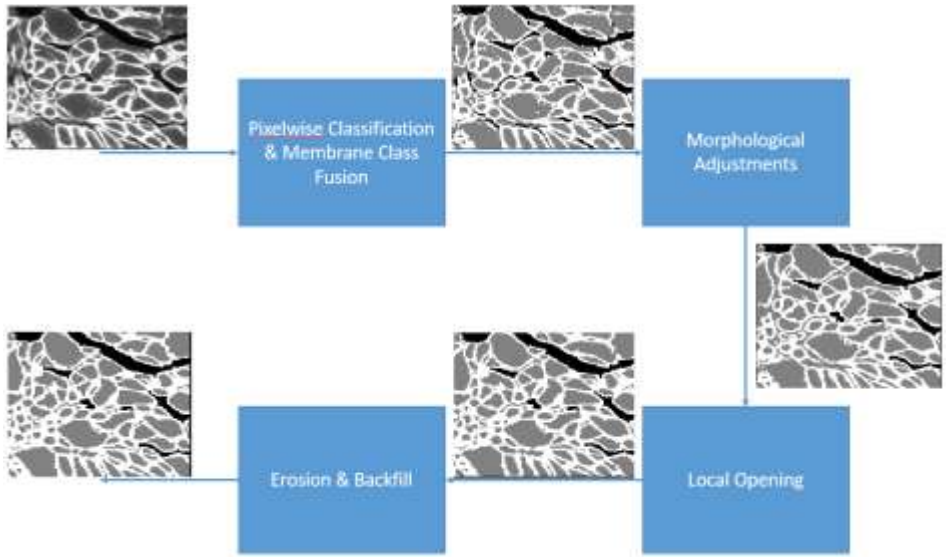


Figure 1: The image processing pipeline has three major steps: pixel-wise classification, morphological filtering, and membrane gap closure. Pixel-wise classification is achieved with a linear classifier. Each morphological filter is designed to correct mislabeled pixels. Gap detection finds faint membranes within oddly shaped components.

In order to quantify the images, the individual cells must first be identified. The difficulty of this task comes from varying qualities in the cells and membranes. If all the membranes were highlighted white in the image, this task would be simple. However, in many cases, the membrane appears only as bright as the random fluctuations within the cell interior. Cells can be perfectly round or concave, vary in diameter by an order of magnitude, and be surrounded by white membrane or black intercellular matrix.

Our pipeline for the identification (or "segmentation") of the cells is shown in Figure 1. We begin by using a machine learning method known as a *support vector machine* (SVM) (Zaki and Meira 514-546) to provide an initial prediction of the class (membrane, cytoplasm, background, etc.) of each pixel in the image. Because of the variability in the manner in which membranes appear in these images, we identify two membrane classes: weak and strong. After the SVM processing, these two classes are merged. Additionally, because the SVM processes every pixel independently of every other, the designation of a pixel's neighbors plays no role in the how that pixel is classified. As a result, we see small clusters of

manifestly mislabeled pixels. Hence, morphological process is used to remove these errors. Finally, irregularly shaped connected components of interior pixels are detected. These are typically clusters of fibers with weakly highlighted membrane between them. Local opening is used to detect and eliminate small bottlenecks, splitting the larger blob into component cells. If any irregular connected components remain, a large-structure erosion approach is used to virtually "squeeze" this blob until it is no longer connected. The area between the resulting connected components is labeled as membrane.

The remainder of this report is organized as follows. The workings of the image processing pipeline are explained in the order of the operations: pixel-wise classification is followed by morphological filters and then membrane gap closure. Next, the results of the segmentation algorithm applied to a few test images are analyzed independently.

2 Processing Pipeline Details

2.1 Histogram Equalization

Before the pixel-wise classifier is applied, the contrast of the sample is adjusted to match that of the classifier's training set. The gray-scale transformation has the form $T(x) = \min\{255, kx\}$, where k is the number between 1 and 3 for which the distribution of gray-values in the output is the closest to the distribution of gray-values in the training set.

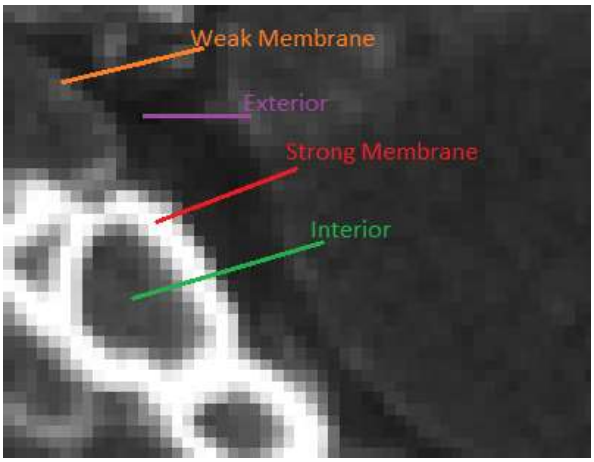


Figure 2: Illustration of four pixel classes of interest in this work

2.2 Pixel Classification

As illustrated in a magnified section from one of our images, there are four categories of pixels easily distinguishable by eye. Strong membrane pixels are generally white. Interior pixels are basically gray with a degree of somewhat random, noisy variability. They come in large, noisy clusters. Exterior pixels are black and often close to membrane pixels. Finally, what we are calling weak membrane pixels are brighter than interior and exterior pixels, and they come in thin bands.

The goal of the initial stage of the processing pipeline is to classify each pixel in an image as one of these four types. Toward that end, we employ a support vector machine approach. This method for classification requires a collection of hand labels pixels along with a set of features for each pixel capturing information about the neighborhood in which the pixel is located. Based on this training data, a set of parameters defining the SVM is determined. For each pixel in a newly acquired image, the features are computed and the SVM (along with the pre-determined parameters) are used to determine the class. In our work to date, the feature vector is comprised of gray-level of the specific the pixel, as well as the mean, standard deviation, minimum, and maximum in 3x3 and 5x5 neighborhoods around the pixel.

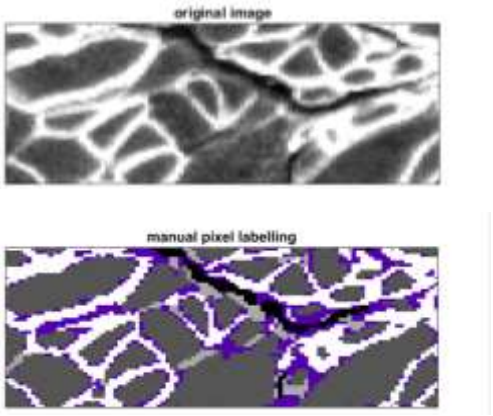


Figure 3: Manual labeled pixels. White is for strong membrane, silver is for weak membrane, gray is for interior, and black is for exterior. Purple pixels are unlabeled.

In Figure 3, we show a portion of the training data used for the results in Section 3. The original grayscale image is provided in the top panel and the hand-labeled classes are given in the bottom with white being strong membrane, silver for weak membrane, gray for cell interior, and black for background. Finally, purple pixels are those which we do not use in the training set as, to our eye, it is not immediately clear to which class they should belong.

Before the classifier is applied, the contrast of the test image is adjusted using histogram equalization.

The classifier we used is a linear multi-class Support Vector Machine implemented in Matlab's machine learning toolbox (Mathworks). We used the linear kernel and the one-versus-all scheme for reducing the multiclass classification problem to several binary classification problems. A support vector machine is a generalization of the maximum margin problem; it finds an optimal separating hyperplane between two clouds of data.

When there are more than two classes of data, the one-versus-all scheme creates one SVM for each class. In machine 'X,' class 'X' is the positive class and all other classes are fused into the negative class. If only machine 'Y' gives a positive prediction for a test datum, then the datum is labeled as belonging to class 'Y.' Cases where there is no unique positive prediction are more complicated.

2.3 Class Fusion

First, the two categories representing membrane are combined. The classifier considered them separate only because this was found to improve the accuracy. Though having an extra category increases the complexity of the multiclass SVM, strong membrane and weak membrane look different to both the human eye and the feature extractor. Since strength of the membrane plays no role in any of the following steps, the distinction between strong membrane and weak membrane can be safely erased. As a result, the output of this stage of processing is comprised of three classes: membrane, cell interior, and background.

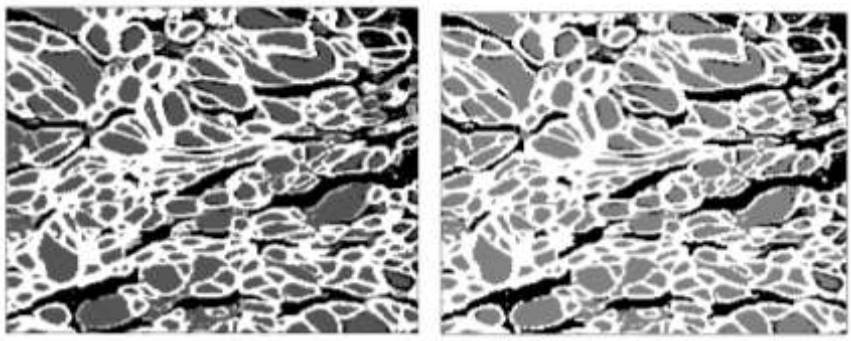


Figure 4: Initial 4-class segmentation in the left panel and class fusion on the right

The result of this portion of our pipeline are shown in Figure 4 In the left panel, we see the four class output of the SVM while the results after merging are provided in right subplot.

2.4 Morphological Filtering

Based on our initial evaluation, there are four types of error that tend to be made by the SVM processing:

1. Pixels labeled as cell interior which are really membrane.
2. Pixels labeled as background which are really membrane.
3. Pixels labeled as background which are really cell interior.
4. Pixels labeled as membrane which are really cell interior.

To correct these errors, which tend to manifest themselves in small sized regions of misclassification, we have implemented a number of morphological filtering operations, as described in the following sections.

Figure 5: First type of labeling error: interior pixels that should be membrane

2.4.1 Mislabeled Interior Pixels

We have observed that no fibers have cross sectional area of less than four pixels. Thus, small components of interior pixels such as those shown in Figure 5, can be considered gaps in the membrane. The mislabeling is corrected with the following method: Consider the set of pixels belonging to the category of interior. For each connected component of this set, calculate the area. If the area is less than four pixels, re-categorize each pixel in the connected component as membrane.

2.4.2 Mislabeled Exterior Pixels

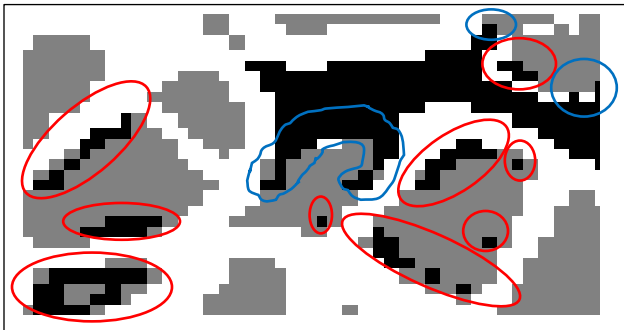


Figure 6: Stray exterior pixels should be interior; interior components should be surrounded by membrane

It has been observed that true exterior pixels come in large clusters. As shown in Figure 6 the classification process can mislabel the darkest interior pixels as exterior. Also, each cell must be enclosed by membrane; some exterior pixels should actually be membrane. The mislabeling is corrected with the following algorithm: Consider the set of pixels belonging to the category of exterior. For each connected component of this set, calculate the area. If the area is less than fifteen pixels, re-categorize each pixel in the connected component as interior. Otherwise, consider the pixels in the component adjacent to interior pixels. These are re-labeled as membrane.

Connected components of exterior which have the following two properties are also relabeled as interior: area less than 1.25 times the perimeter and a convex hull with area greater than 1.25 times the component's area.

2.4.3 Mislabeled Membrane Pixels

It has been observed that a cell membrane encloses a cell. As shown in Figure 7, the classification process can mislabel the lightest interior pixels as membrane, even in the middle of a cell. The mislabeling is corrected by removing any connected components of membrane pixels which do not enclose any other region.

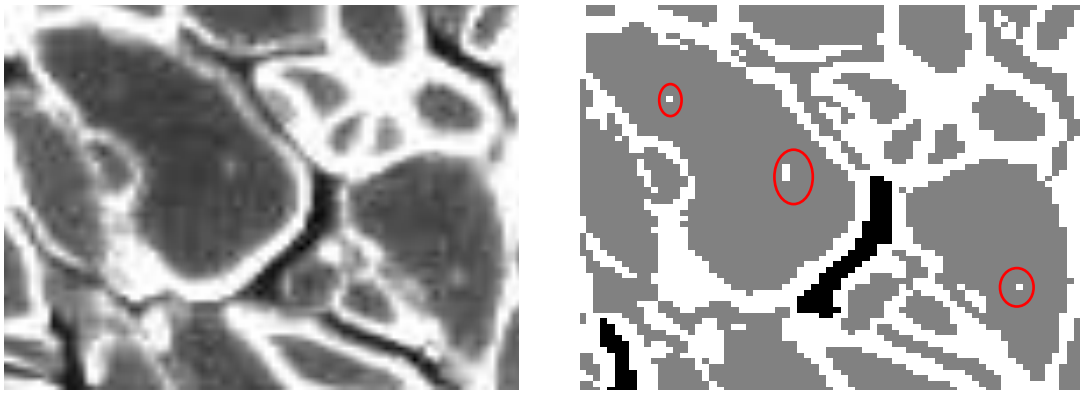


Figure 7: Membrane components inappropriately labeled

2.5 Membrane Gap Closure

As illustrated in Figure 8, there remains the need to close "gaps" in membranes left by the SVM processing. Indeed, even with the initial separation of the membrane into two classes, we see that the distinction between membrane and cell interior can still be is still rather subtle resulting in regions that need to be closed so as to properly identify individual cells.

To address this issue, a two-step approach has been developed. First, a second classifier has been developed to identify each connected component of interior pixels as either a cell or not a cell. Second, a local morphology approach is applied to those regions not cells in order to identify the individual cell components.

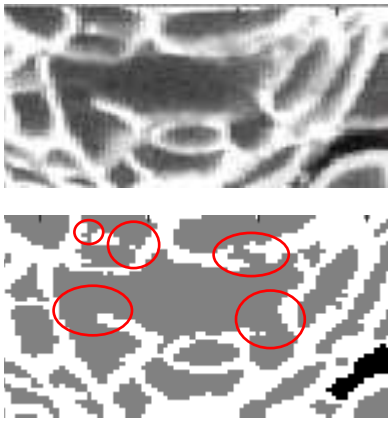


Figure 8: Bottom) after the morphological filters, the membrane has gaps

2.5.1 Cell or Not a Cell Classifier

The goal of this component of the processing is to determine whether or not a given connected component of pixels labeled as interior constitute a likely cell based on the following features:

1. Cell diameter defined to be the maximal Euclidean distance between two points on the boundary of the connected component.
2. Total boundary curvature which is estimated by representing the connected component using a signed distance level set function, $F(x,y)$, and computing the curvature as $\kappa = \frac{|F_{xx}F_y^2 - 2F_xF_yF_{xy} + F_{yy}F_x^2|}{|\nabla F|^3}$ (Kimmel 53).
3. Cell perimeter, estimated as the number of pixels on the boundary.
4. Cell area, calculated by counting the pixels in the region.
5. Ratio of the area of the region's convex hull to the area of the cell itself, which is calculated using Matlab's image processing toolbox (Mathworks).
6. Mean brightness-value of the original image in the cell.
7. Variance of brightness of the original image in the cell.

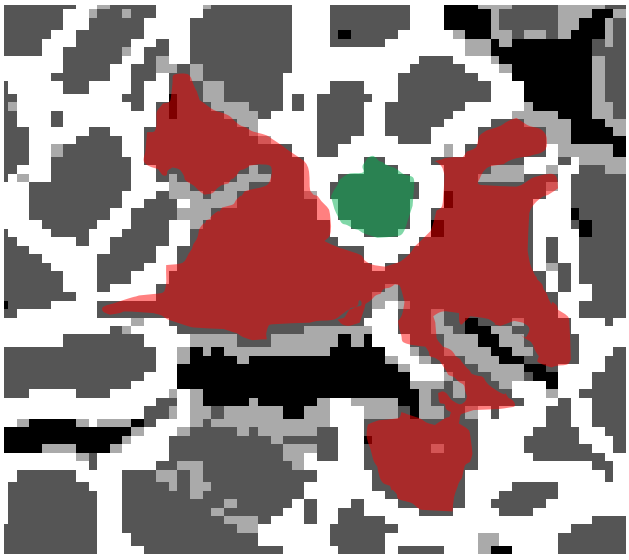


Figure 9: The component highlighted in red needs to be split, the component highlighted in green does not, and the random forest classifier distinguishes between these two cases.

For this problem, we use a random forest classifier (Liaw and Wiener). Figure 9 shows an example of a component likely to be a cell and an example of a components unlikely to be a cell.

2.5.2 Conditional Local Opening

Figure 10: Gaps lie between points on the boundary which have high curvature and are close together

Components which contain multiple fibers tend to have separate lobes between which the undetected membranes end. The key to solving the membrane completion problem is finding the spaces between the lobes, as illustrated in Figure 10. The points on the boundary of the region between the lobes of individual cells usually have high curvature.

The approach used to selectively close the bottlenecks between true cells is to examine neighborhoods of each point in the component. If using morphological opening on the part of the image in the neighborhood does close a bottleneck, then the pixels removed from the component by the opening are re-labeled as membrane pixels.

This is done in the neighborhood of each pixel in parallel. The process is performed three times, using as neighborhoods disks of radius $\sqrt{2}$, 2, and $\sqrt{8}$.

2.5.3 Erosion & Backfilling

To detect larger gaps in the membrane, a component which is classified as not a cell is eroded with a large structuring element. If the result of the erosion has less than two connected components, then nothing happens. Otherwise, the original component is split into cells containing the results of erosion.

Figure 11: The erosion method for bottleneck detection shrinks the original region (black outline) into multiple shrunken components (red), which must be restored (green) to fill most of the original region.

After the separation, the split components of the blob must be restored: only the space between them becomes membrane. This is accomplished by incrementally increasing the sizes of the new components. Where this would cause the two to intersect, cell membrane is inserted and the component no longer grows in that direction. Figure 11 illustrates how a component is split by erosion and restoration.

3 Results

3.1 Qualitative

Qualitatively, the output can appear to be a reasonable segmentation of the image. Figure 13 shows three such examples. However, if the test image has cell interiors significantly darker than in the training image, parts of cells may be mistaken for extracellular matrix, as is the case in the examples in Figure 12. Furthermore, the process consistently over-segments.

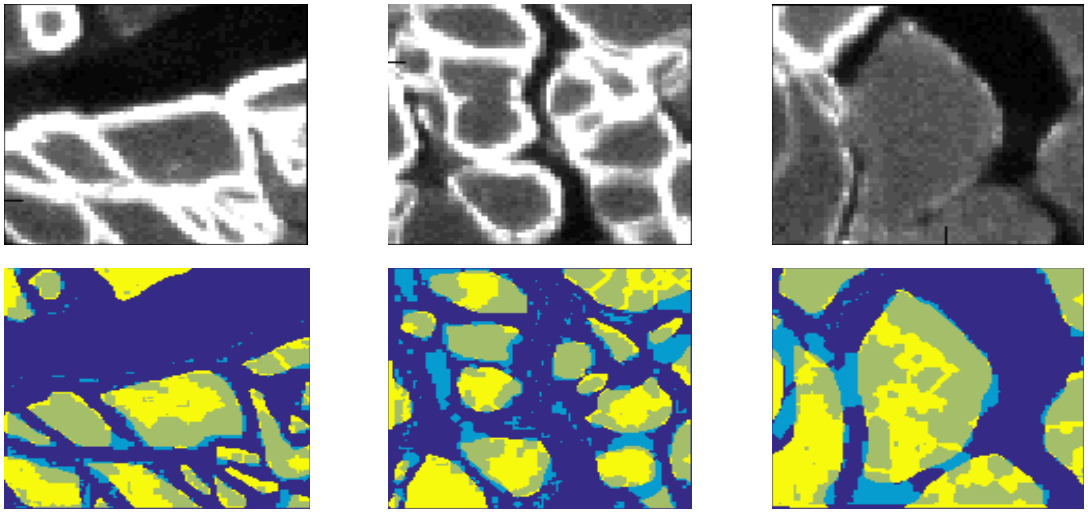


Figure 12: Results in bad cases. Top) Histogram-adjusted inputs. Bottom) Comparison of output and manual labelling--dark blue represents membrane and exterior, light blue represents pixels mislabeled as interior, dull yellow represents cell interior, and bright yellow represents pixels mislabeled as exterior or membrane.

The results depend strongly on the quality of the initial classification of pixels. If the test image after the gray-scale transformation is still darker or of higher contrast than the training data, then the algorithm does not work well. Even in the best cases, it may split fibers where it shouldn't—the source of error may be the "cell or not" classifier. Another approach, besides improving the "cell or not" classifier, would be to develop a method to merge neighboring components where appropriate.

There are some cases where the algorithm fails to identify neighboring cells as distinct. A possible remedy for this problem would be to use the gap closure techniques before applying the filter which removes membrane. Another approach would be to use an initial classifier which does not mislabel large, faint regions of membranes.

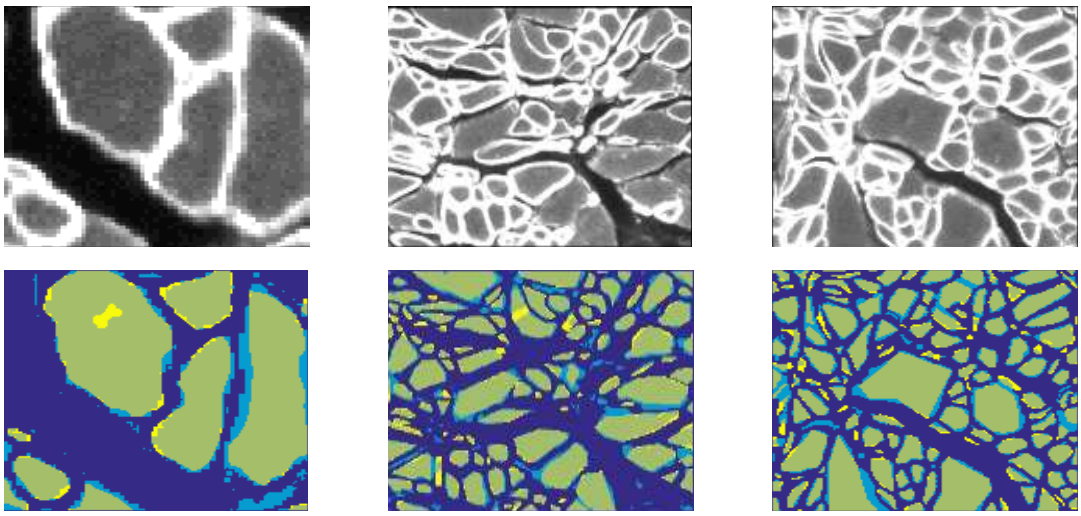


Figure 13: Results in good cases. Top) Histogram-adjusted inputs. Bottom) Comparison of output and manual labelling--dark blue represents membrane and exterior, light blue represents pixels mislabeled as interior, dull yellow represents cell interior, and bright yellow represents pixels mislabeled as exterior or membrane.

3.2 QUANTITATIVE

To quantify the performance of the algorithm, it is necessary to examine the distribution of areas of fibers identified. To this end, images were segmented by hand while the automatic segmentation ran. The manual labeling was performed on forty-two small test images, twenty-seven of which yielded qualitatively reasonable results.

Overall, the algorithm does over-segment, even when the last step, which involves splitting blobs, is omitted. Figure 14 shows by how much the algorithm over-segments, both before and after the last step. As the figure shows, after the last step, the computed number of cells can be as much as 2.5 times the correct value. This shows the algorithm's tendency to mis-categorize small regions where cell membrane and exterior intersect as parts of cell interior, creating many tiny false cells.

Despite the problem of over-segmentation, the last step does improve the distribution of cell sizes. The Kolmogorov-Smirnov statistic, a measure of difference between distributions, is defined as the maximum of the difference between two cumulative distributions. Figure 15 shows the distribution of Kolmogorov-Smirnov distances between the data segmented by the pipeline and the data segmented manually. Both on average and in the worst cases, the final splitting steps make the distributions of the automatic and manual segementations more similar.

The ACC metric measures how often the pipeline's output agrees with the manual labeling. Let TP be the number of pixels that both ways label as cell interior, FP be the number of pixels that the pipeline incorrectly labels as cell interior, TN be the number of pixels that both label as not cell interior, and FN be the number of pixels that the pipeline incorrectly labels as other than cell interior. Note that the nature of the human error in the manual segmentation makes

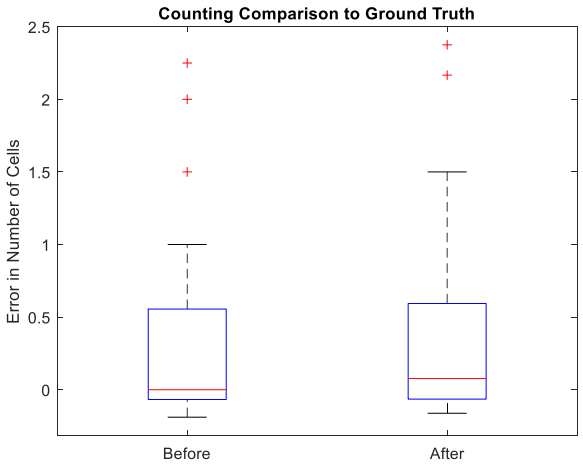


Figure 14: The distribution of error in the number of computer-segmented cells with respect to the numbers of hand-counted cells.

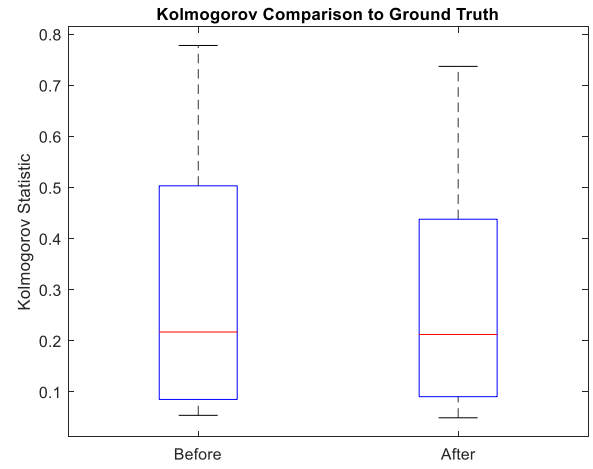


Figure 15: Distribution of errors in the CDF of the data

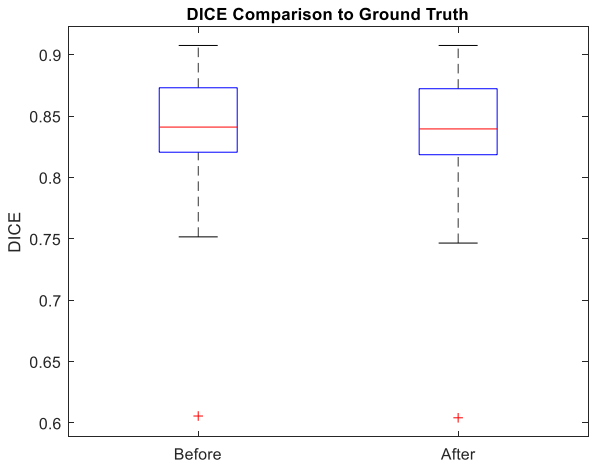


Figure 16: Distribution of DICE values

FP artificially high. The definition is $ACC = \frac{TP+TN}{TP+FP+TN+FN}$. (Pang, Ozkucur and Ren)

In most cases, the last step improves the ACC slightly. Figure 17 shows the distribution of ACC values before and after the last step. On average, about 90 percent of the pixels are labeled correctly. Much of this 10 percent disagreement comes from the boundaries of cells, where the manual segmentation and automatic segmentation disagree on where the membrane begins and the cell interior ends.

Another measure of accuracy is DICE. The definition is $DICE = \frac{2*TP}{2*TP+FP+FN}$. (Pang, Ozkucur and Ren)

The last step has an insignificant impact on the DICE score of the image. Figure 16 shows the distribution of DICE values. The average DICE value is almost 85 percent. Disagreement in where cell ends and membrane begins contributes to this relatively low score.

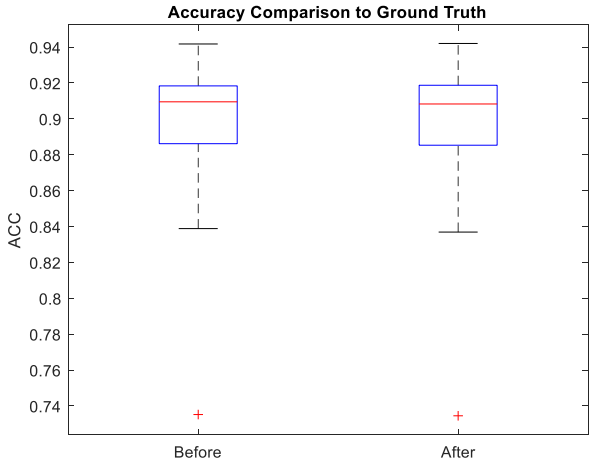


Figure 17: Distribution of ACC values

3.3 A Good Case

In the case of Figure 19, the results are good. Automatic segmentation yields a mean cross-sectional area of 86 pixels while manual labeling yields a mean of 74 pixels. What should be 135 cells are under-segmented into 125 components. The ACC is 0.9223, and the DICE is 0.8699. The histograms are shown in Figure 18—the Kolmogorov-Smirnov statistic is 0.0708. As shown in Figure 19, the results of automatic and manual segmentation are quantitatively comparable in this case, though there are a few errors.

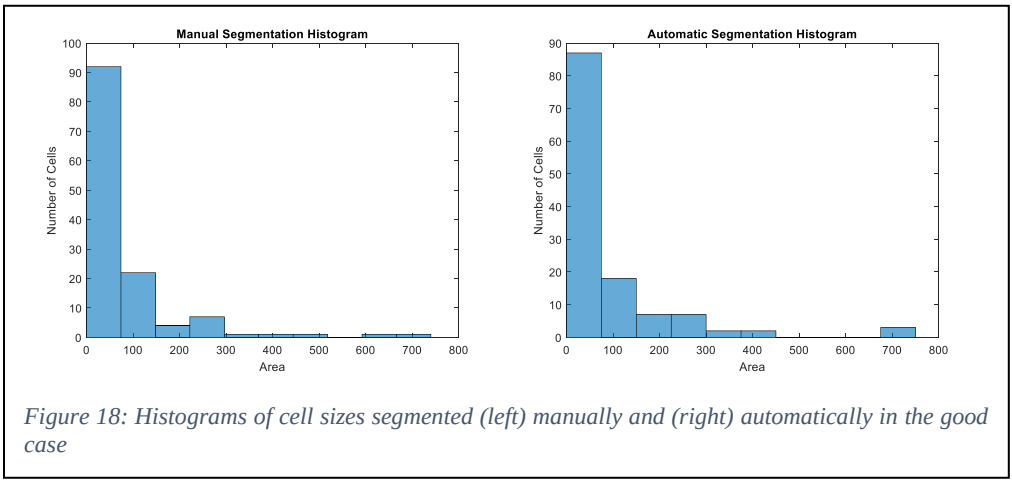
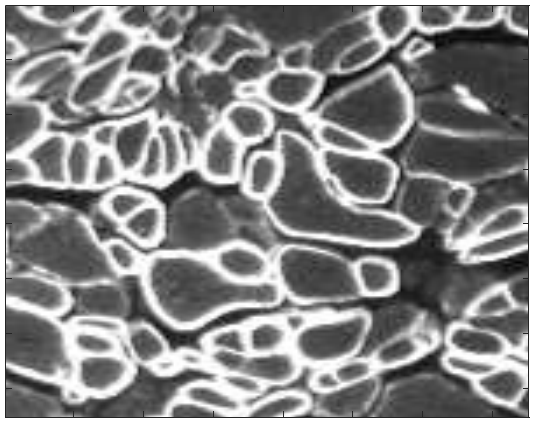
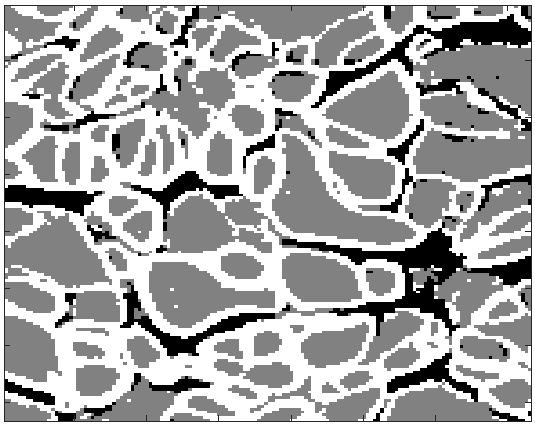


Figure 18: Histograms of cell sizes segmented (left) manually and (right) automatically in the good case

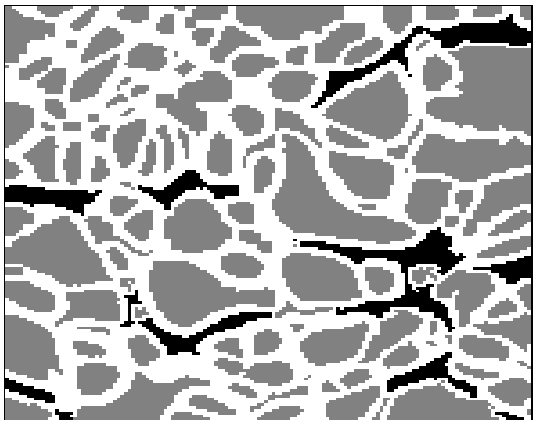
Near the top-right corner of the image, there is a cluster of cells with dim membranes. The initial classification recognized some of the membrane, but that section of membrane was removed because it was disconnected from the rest of the cell membrane. This caused two separate cells to be labeled as one.



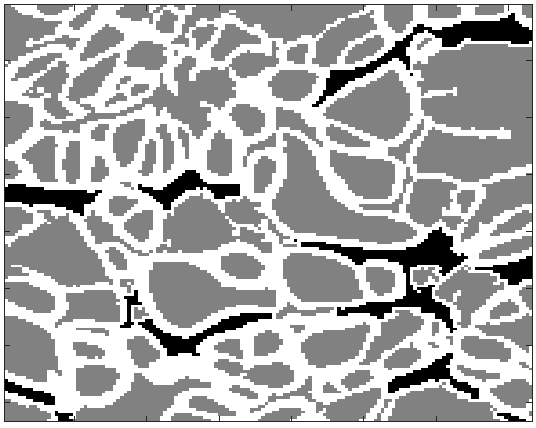
A



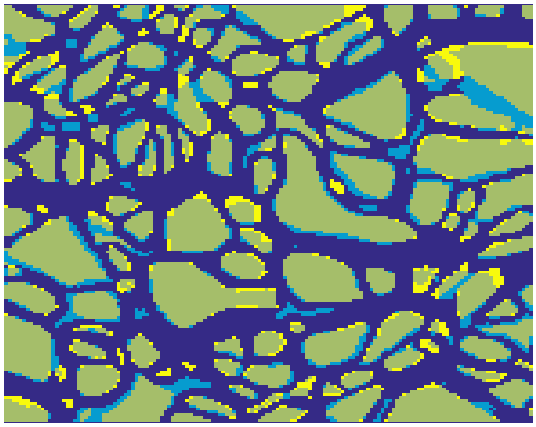
B



D



C



E

Figure 19: Pipeline summary in a good case. A) Original Image. B) Result of pixel-wise classification. C) Output of morphological filters. D) Output of gap closure methods. E) Comparison of ground truth and algorithm output—dark blue represents pixels which are labeled both ways as other than cell interior, light blue represents pixels which are incorrectly labeled as cell interior by the algorithm, dull yellow pixels are labeled both ways as cell interior, and bright yellow pixels are incorrectly labeled as cell interior by the algorithm.

3.4 A BAD CASE

In the case of Figure 21, the results are poor. While the automatic generation yields a mean cross-sectional area of 86 pixels, the manual labeling yields a mean of 619 pixels. Figure 20 shows the histograms of cell sizes generated by both manual and automatic segmentation—the Kolmogorov-Smirnov statistic is 0.7750. The algorithm over-segments what should be 13 cells into 80 components. The DICE is 0.6812, and the ACC is 0.8624.

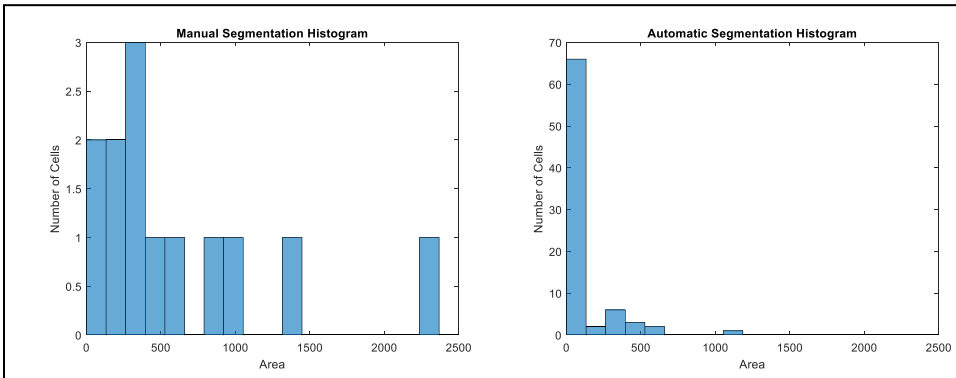
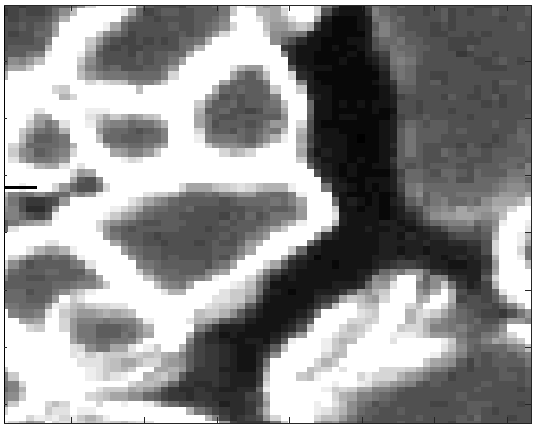
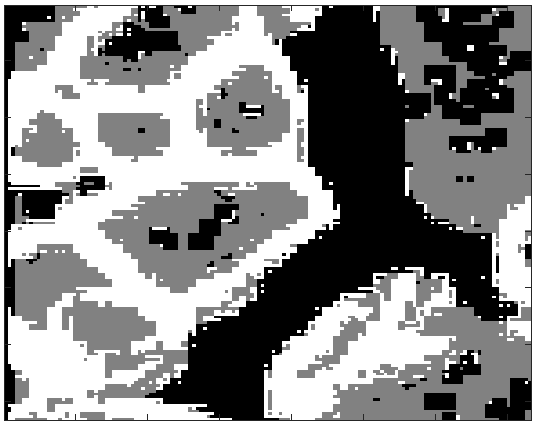


Figure 20: Histograms of cell sizes segmented (left) manually and (right) automatically in the bad case

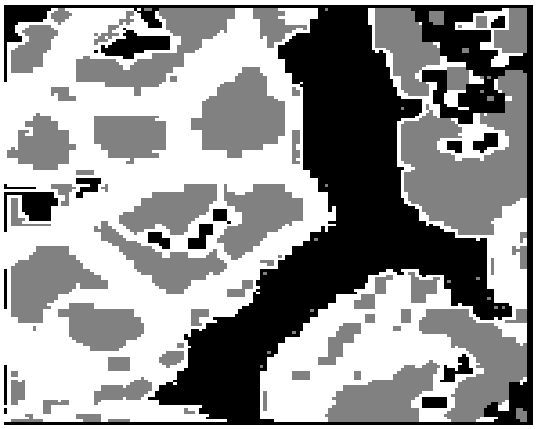
In this case, as displayed in Figure 21 part B, the pixel-wise classifier mistakes large regions of cell interior for cell exterior. In later steps, membrane is inserted around these regions of error, causing single cells to be interpreted as clusters of cells interspersed with exterior regions. Furthermore, parts of the interface between bright membrane and true exterior are misinterpreted as cells. These two factors lead to high error.



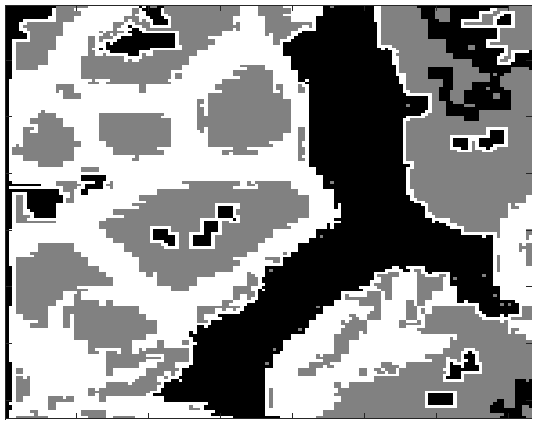
A



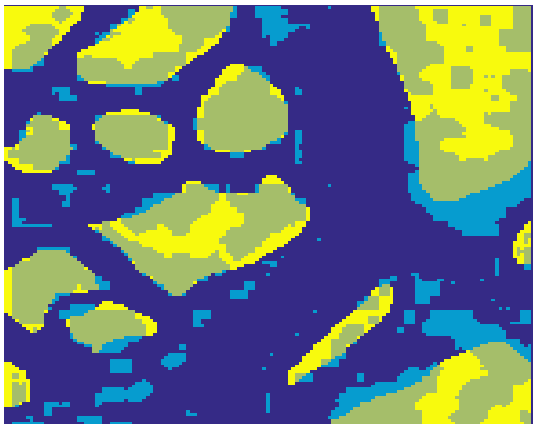
B



D



C



E

Figure 21: Pipeline summary in a bad case. A) Original Image. B) Result of pixel-wise classification. C) Output of morphological filters. D) Output of gap closure methods. E) Comparison of ground truth and algorithm output—dark blue represents pixels which are labeled both ways as other than cell interior, light blue represents pixels which are incorrectly labeled as cell interior by the algorithm, dull yellow pixels are labeled both ways as cell interior, and bright yellow pixels are incorrectly labeled as cell interior by the algorithm.

4 CONCLUSION

An algorithm for the fully automatic detection of fibers has been developed. It has mixed results. In some cases, the output of algorithm agrees almost completely with manual labelling. In other cases, the statistics of the output are unacceptably far from the truth.

The method used to detect the cells in the input image can be summarized in three parts. The first part is to categorize each pixel independently, given various characteristics of the image in the neighborhood of said pixel. Next, this initial segmentation is refined by applying morphological filters: removing small connected components and making sure each cell is enclosed by membrane. The last step is to close any narrow bottlenecks in components which are deemed unlikely to be correctly segmented.

4.1 FUTURE DIRECTIONS

Firstly, the gap closure methods do not detect every missing cell membrane, even when the true membrane is relatively bright. A curve evolution approach would have the advantage of incorporating gray-level information into the way the gaps are closed, but this has been implemented and found to be unreliable.

Secondly, the cell-or-not classifier is insufficiently accurate. Furthermore, it does not distinguish between clusters of falsely conjoined cells and other types of non-cell components. A better way to categorize the components would improve the performance.

Thirdly, the initial categorization by independently considering each pixel can lead to egregious error. It may be worth investigating applications of region-based methods instead.

5 BIBLIOGRAPHY

Kimmel, Ron. *Numerical Geometry of Images*. New York: Springer Science + Business Media, 2004. Print.

Liaw, Andy and Matthew Wiener. "Classification and Regression by randomForest." December 2002. *Northwestern.edu.* 26 June 2016. <cogns.northwestern.edu/cbmg/LiawAndWiener2002.pdf>.

Mathworks. *MATLAB Documentation: bwconncomp*. n.d. 28 June 2016. <www.mathworks.com/help/images/ref/bwconncomp.html>.

—. *MATLAB Documentation: bwconvhull*. n.d. 28 June 2016. <www.mathworks.com/help/images/ref/bwconvhull.html>.

—. *MATLAB Documentation: fitcecoc*. n.d. 28 June 2016. <www.mathworks.com/help/stats/fitcecoc.html>.

Pang, Jincheng, et al. "Automatic neuron segmentation and neural network analysis method for phase contrast microscopy images." *Biomedical Optics Express* 6.11 (2015).

Zaki, Mohammed and Wagner Meira. *Data Mining and Analysis*. New York: Cambridge University Press, 2014. <http://www.cs.rpi.edu/~zaki/PaperDir/DMABOOK.pdf>.