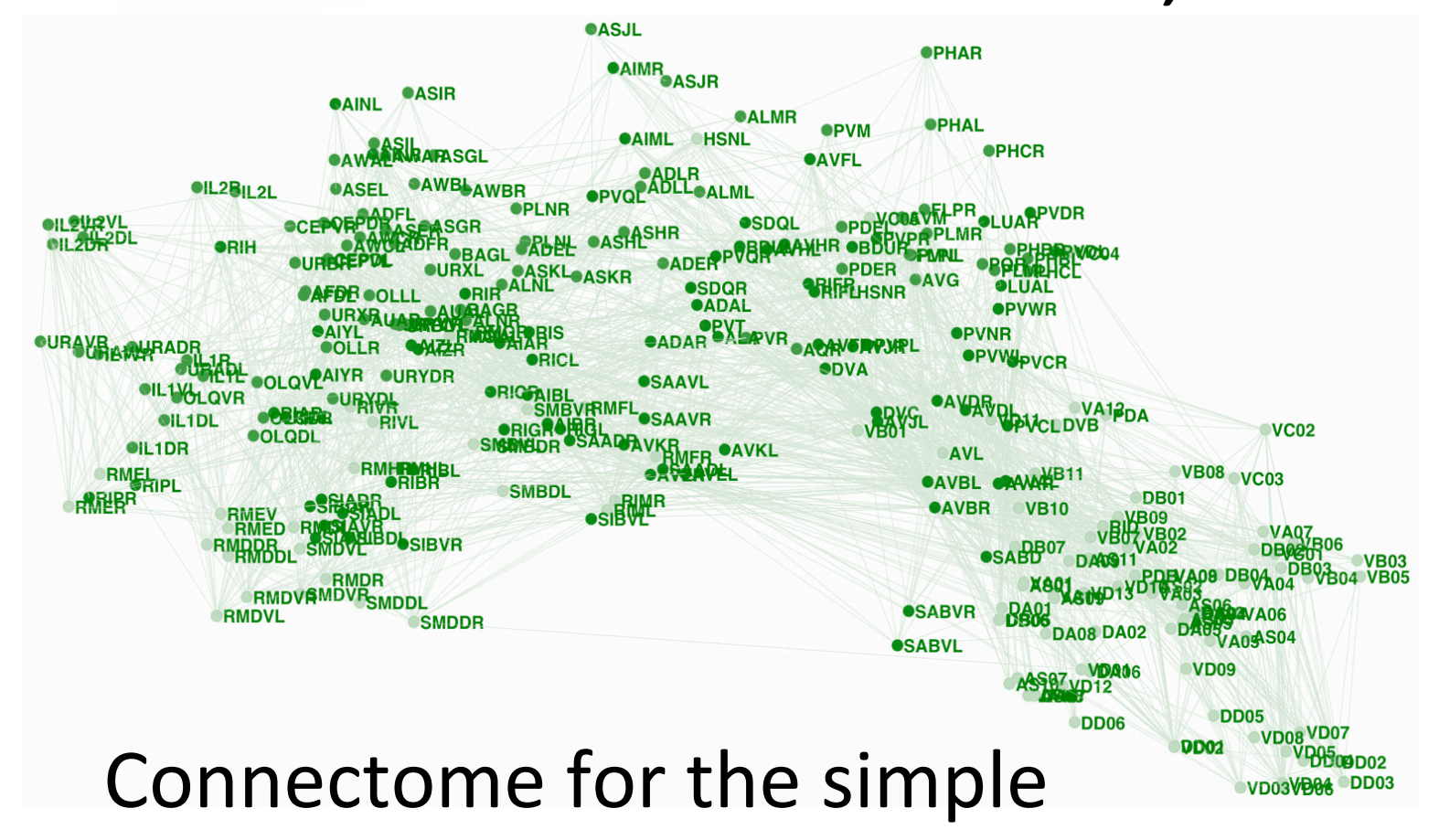
Optimizing Random Forest Image Segmentation for Connectomics

Chandan Singh - Turaga Lab
Luke Hewitt – UCL
Srini Turaga - HHMI

Background

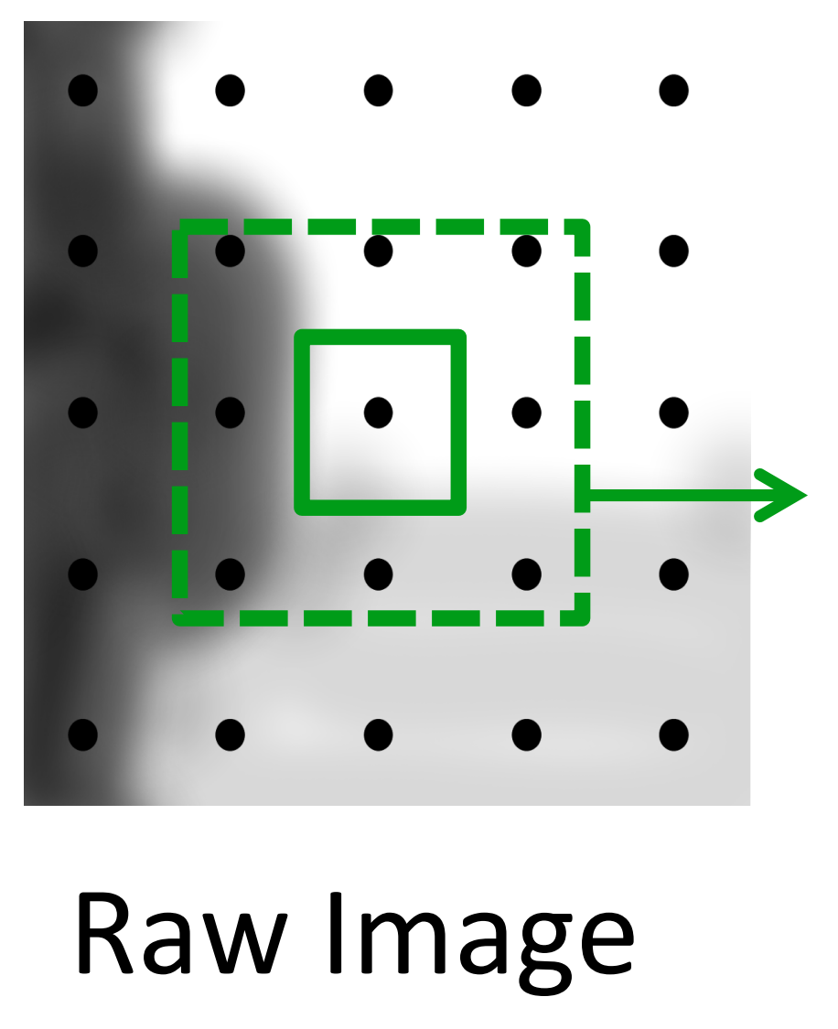
Recent advancements in neuroscience have allowed scientists to start creating maps of neural connections in the brain known as connectomes. However, to construct a connectome, brain tissue must be imaged,


Connectome for the simple network of *C. Elegans [4]*

the images must be segmented to identify neurons, and then neural connections must be determined. Segmenting neurons is a slow and arduous process when done by humans, and this project focuses on segmenting neurons efficiently and robustly with machine learning. There are challenges segmenting neurons with computers because segments in images can be miniscule and small errors can lead to entire neurons being merged or disconnected.

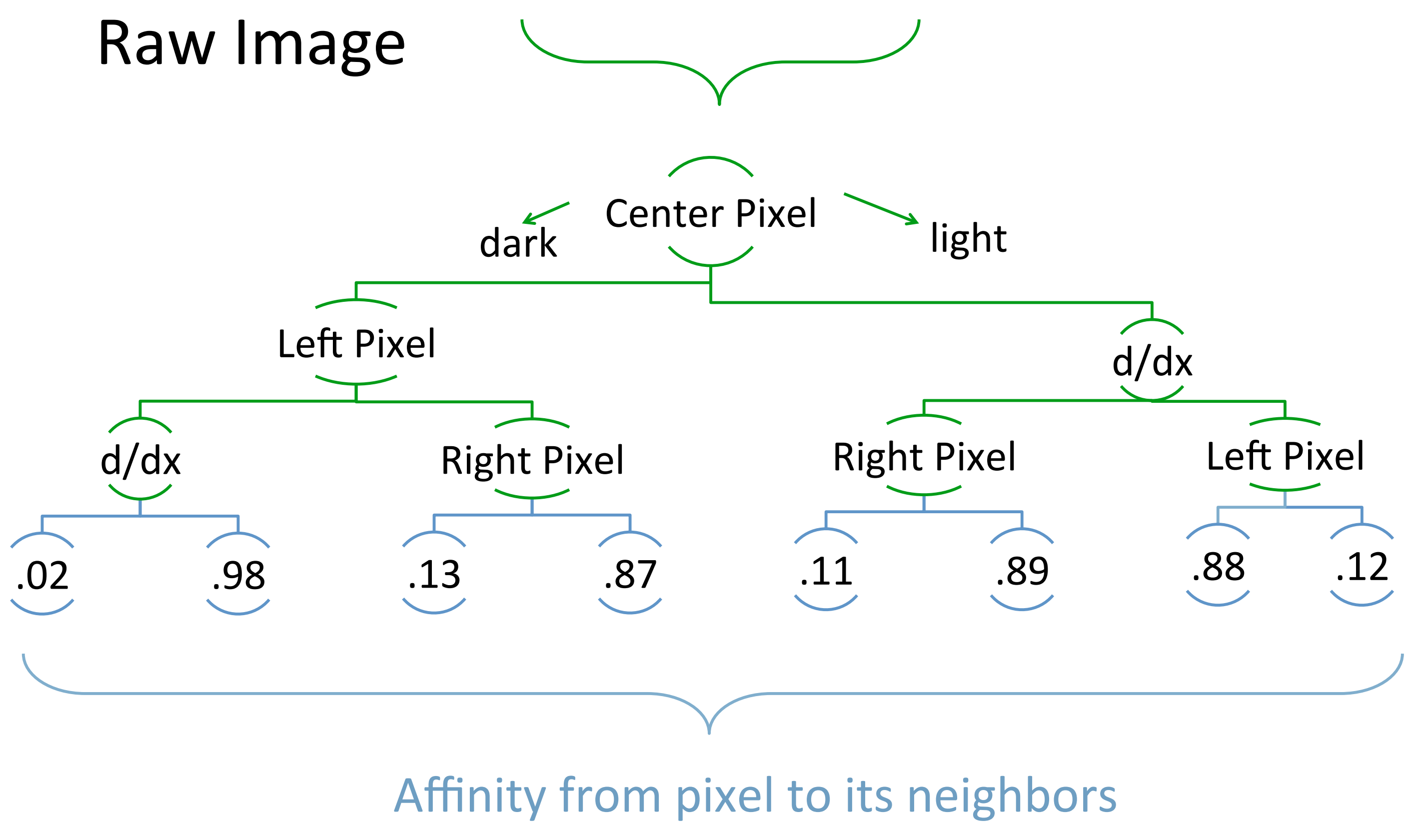
Algorithm Part I: Random Forests

The core algorithm used was a machine learning technique known as decision trees. In this case, the decision trees used features extracted from images in order to predict whether a specific pixel was connected to its neighbors.



Extracted Features

Center Pixel	Light
Left Pixel	Dark
Right Pixel	Light
d/dx	Negative

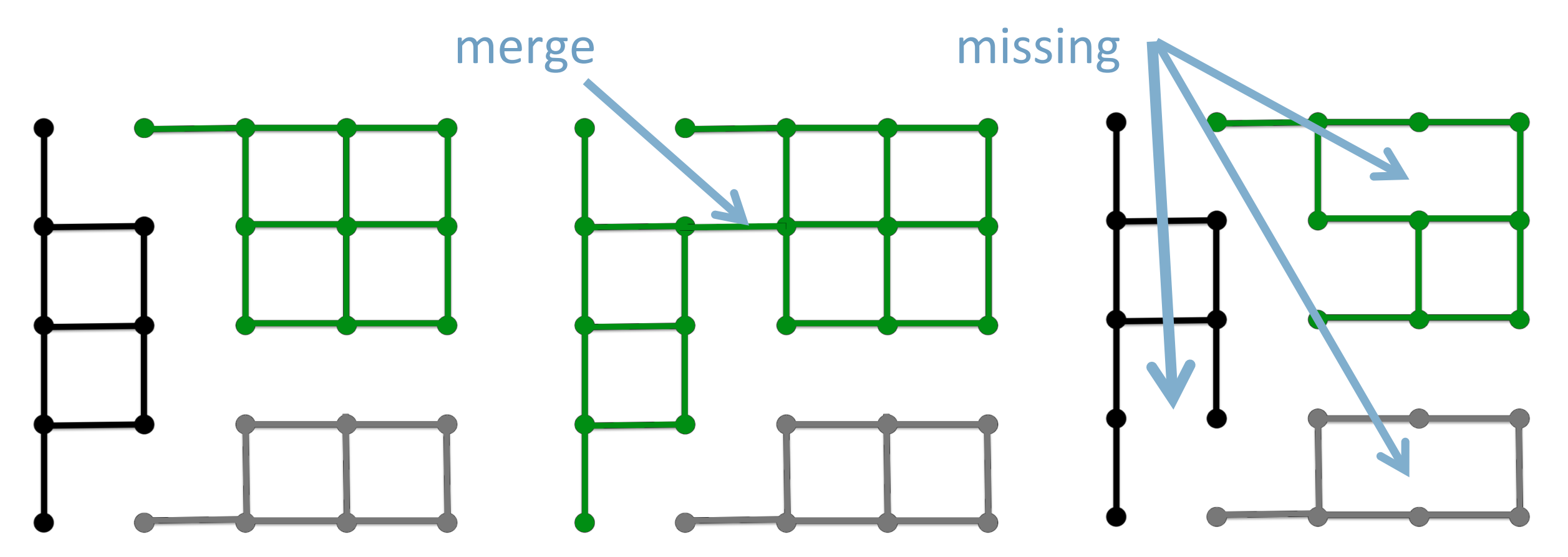


For each pixel, a local set of pixels, and their derivatives in different directions is fed into the decision tree which outputs whether or not the pixel is connected to its neighbors.

The decision trees split based on the values of the features to derive probabilities of different outputs. They pass through the training data and sort the possible outputs based on the input features. Then they prioritize features as splits in the tree in order to minimize the uncertainty of the output (as measured by a loss function such as entropy.) Many decision trees were trained and their predictions were averaged in order to give the final output; this setup is known as a random forest.

Algorithm Part II: MALIS Gradient Boosting

When looking at segmentations, small errors can lead to large mistakes in segmentation. The standard way of training random forests is to minimize the pixel-wise error produced in the segmentations. However, this is not always the best way to produce accurate segmentations. Take the following example:

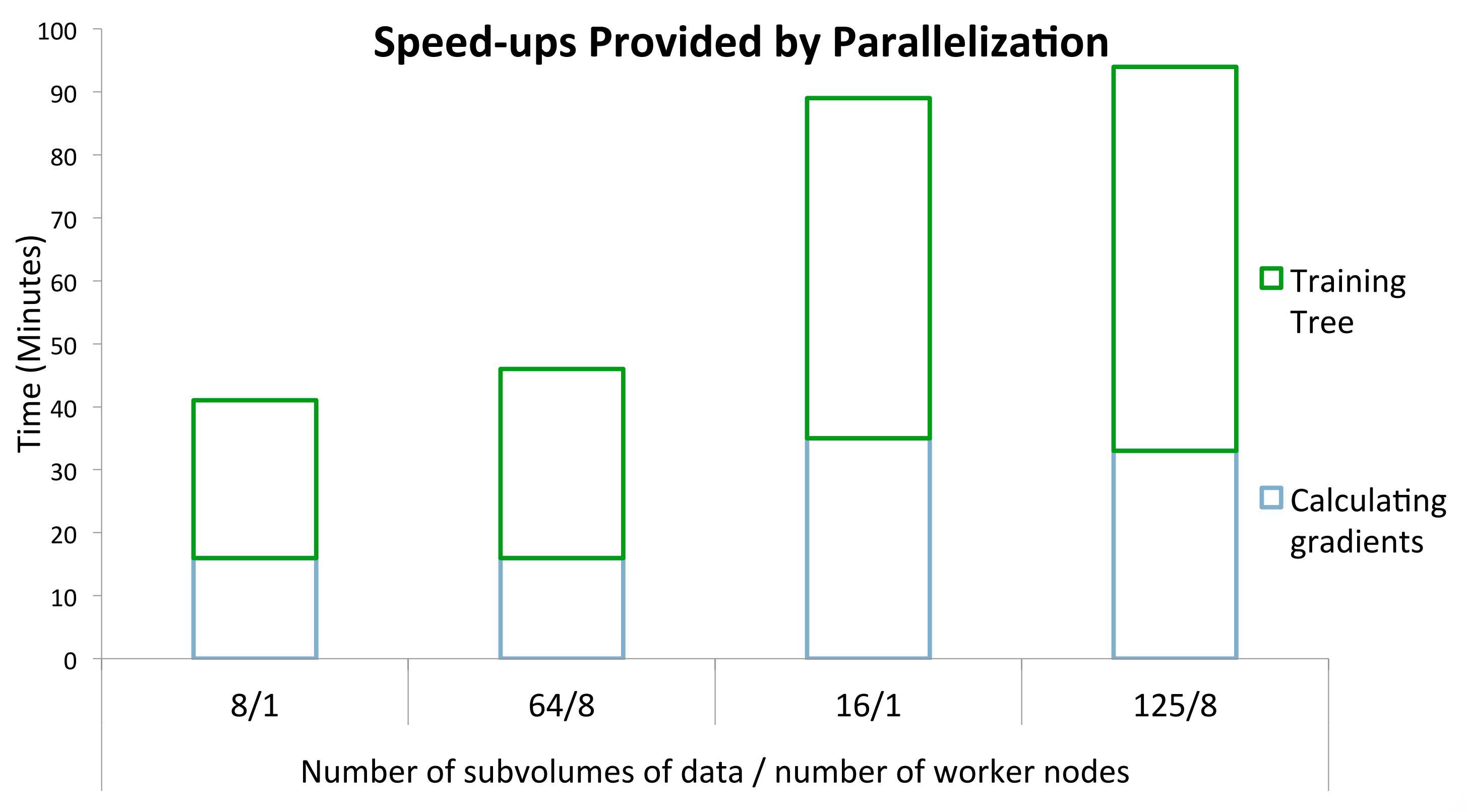


Comparison of correct segmentation (left) to segmentations with different pixel errors (Based on [1]). Having less pixel errors doesn't necessarily lead to a better segmentation.

In order to adjust for this, we aim to minimize the Rand Index, which measures the proportion of pixel pairs which are correctly labelled as being in the same segment [1]. To minimize the Rand Index, we use MALIS (maximin affinity learning of segmentation), an algorithm which specifically targets pixels that cause errors in segmentation [2]. Generally, the algorithm looks at the paths between pixels and tries to find the path that maximizes the minimum affinity. The edge on this path with the minimum affinity is altered in order to produce a more accurate segmentation.

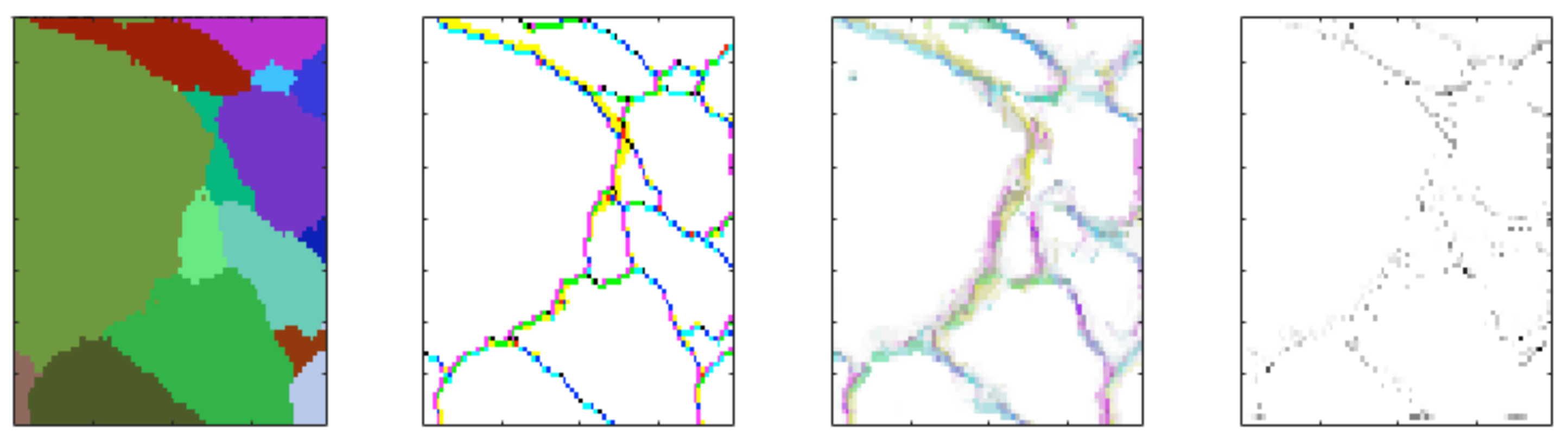
Implementation

A large part of this project was parallelizing code so it could train faster. Decision trees can train independently of one another and thus can be trained simultaneously. Parallelizing the algorithm allows training on massive data sets without increasing the training time. This is done by splitting the data into subvolumes which are processed independently and then aggregated. Additionally, speed was obtained by making calls to compiled C++ functions. This algorithm was implemented using Scala and Apache Spark [3]. Using Spark, the program was run in parallel on the Janelia cluster, which provided a linear speed-up in time as shown below.



Increasing the amount of data and nodes proportionally doesn't slow down program.

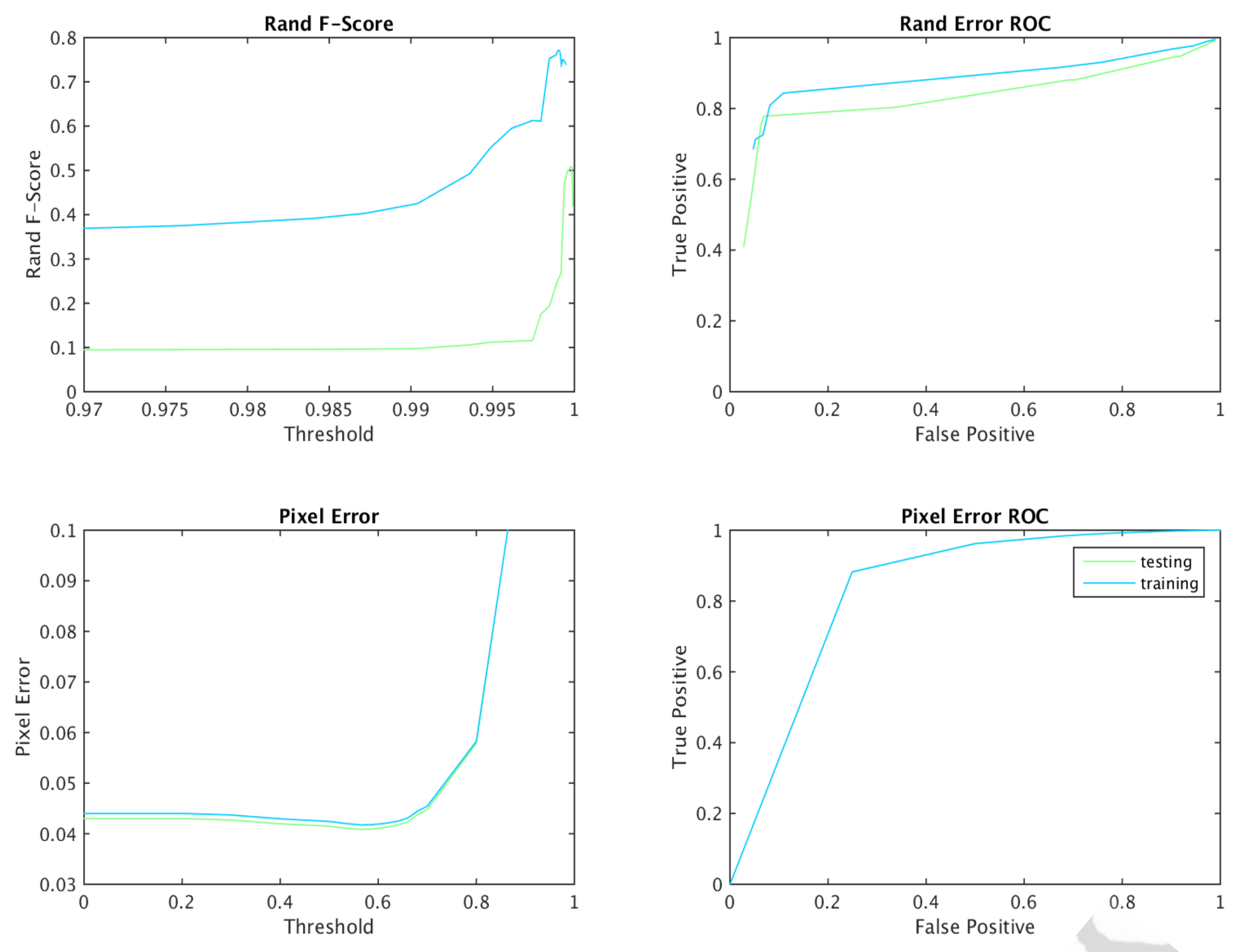
Results



Key differences in the segmentation (left), labels (second), predictions (third), MALIS changes (fourth)

Several parameters were manipulated in order to train the most accurate forests. For example, the number of features and the locations of the features extracted for each pixel were changed. The parameters for MALIS were tested until they were working as shown above.

Finally, after training a forest and gradient boosting, the following performance was achieved:



Key statistics on the segmentation performance.

These preliminary results show that the algorithm could be very effective at quickly producing high-quality image segmentations.

References

[1] W. Rand. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association, pages 846–850, 1971.

[2] Briggman, K., Denk, W., Seung, S., Helmstaedter, M. N., & Turaga, S. C. (2009). Maximin affinity learning of image segmentation. In *Advances in Neural Information Processing Systems* (pp. 1865-1873).

[3] Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., ... & Talwalkar, A. (2015). MLlib: Machine Learning in Apache Spark. *arXiv preprint arXiv:1505.06807*.

[4] http://connectomethebook.com/wp-content/uploads/2011/11/Brainforest16_1119-640x356.jpg