

Joelle Frye and Beatrice Madison

For this assignment we started out by familiarizing ourselves with how to work with the images and how to handle the formatting for submitting to the website. We looked at different tutorials made by those on kaggle for the competition who had very useful insights into how to go about solving the problem and handling the necessary formatting. The most useful tutorial we found was the second most highly rated (<https://www.kaggle.com/stkbailey/teaching-notebook-for-total-imaging-newbies>). This tutorial went through the process of getting an extremely basic functioning solution. Additionally, this solution included a function pre-made that could encode the final product of each image into the RLE format required for the competition. We kept this RLE function and solution form for all the work we did which helped us immensely as we therefore did not need to worry about how to make the encoding work.

In the end we ended up altering the creation of the masks which was located in a function in order to enable quick changes between runs. This tutorial states at the beginning that using a purely otsu thresholding method, like is done for the tutorial, will achieve levels of .22 correctness. Originally, we attempted to improve the already existing method, since it was observed that the otsu method left a lot of very small pieces of junk in the images. For this we implemented the `remove_small_objects` function from the morphology section of skimage. This seemed to improve the results enough at first glance to justify attempting a submission to see how it fared. This submission scored a .387 correctness which was surprising as we anticipated it being better but did not expect this much improvement.

At this point, we messed around with a lot of other methods to see if they would achieve a better looking mask. None of the basic methods seemed to show improved performance though. Because of this we looked into different threshold functions. From the different methods we compared, the yen thresholding seemed to show improved function when dealing with the colored pictures, by grabbing slightly less of the large objects that were not actually cells. We knew this was probably not going to lead to improved performance though since it could also cut out cells in other images but we tested it anyway just to confirm. This gave us a score of .29 correctness which was disappointing because we were hoping it would improve the colored images enough to the point that this was a valid solution.

From here we were stumped on how to improve, from what we could see other users on kaggle ended up using some robust methods would be the only way to improve this which seemed to be mostly machine learning with many different moving parts in these sections. So instead we went back to doing more basic solutions to see if there was a way to make outlines around the cells and then fill them. However, the canny and sobel methods couldn't find the edges of the objects with a good performance.

We did in the end test many different functions that we found were complete garbage such as the blob in ski-kit image which would sometimes just make the entire picture the blob. We did not feel that this was a powerful enough solution to turn this in as it performed so terribly on the few tests we ran. Then we also tried the solutions that opened and closed the image to try and clean it up. Unfortunately this solution cut out a lot of the cells and made the solution worse overall, in the end we decided after testing to not submit it.