Leaf segmentation project

Dominik Borkowicz

Faculty of Mathematics and Information Science

Contents

1 Introduction

In computer vision, image segmentation is the process of partitioning a digital image into multiple segments. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

In project, we were provided with two datasets, both consisting of 150, preselected, leaves images and theirs segmentation. Most of the pictures had white background, some of them were also blurred.

The goal of our project was to properly segment given images using established image processing techniques.

Fig. 1: Example photo

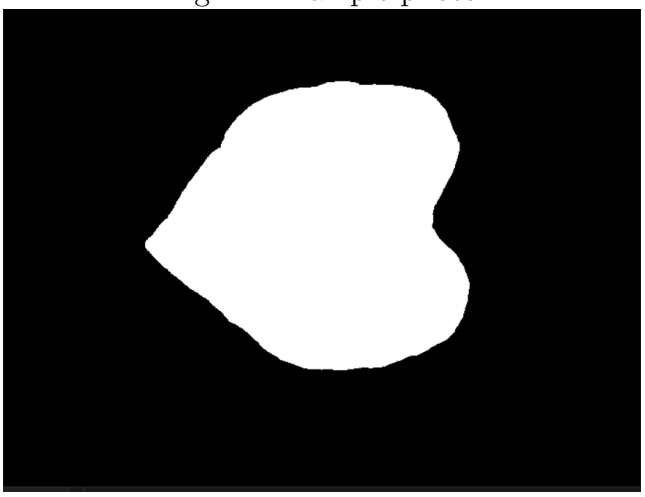

Fig. 2: Proper segmentation of example photo

2 Segmentation methods description

2.1 Binary Thresholding

The simplest thresholding methods replace each pixel in an image with a black pixel if the image intensity $I_{i,j}I_{i,j}islessthansomefixedconstantT$

In this particular case I could operate under an assumption that the background is white and the leaf had some distinct colour. Using threshold (127,255), in grayscale, we get such a segmentation:

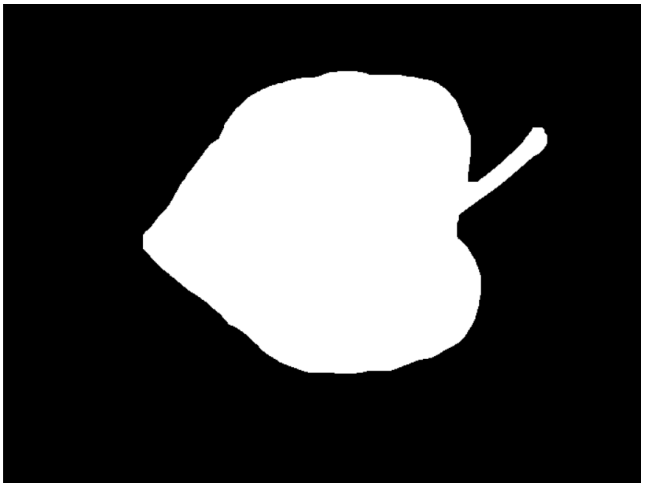

Fig. 3: Picture segmented using binary threshold

As we can, in this segmentation, there are a lot of false positives, most of which are caused by shades and a stem. To optimize above result we augmented algorithm with steps:

1. Read the image
2. Use Gaussian blur
3. Threshold image
4. Inverse threshold
5. Use median blur
6. Dilate 2 times
7. Erode
8. Threshold image

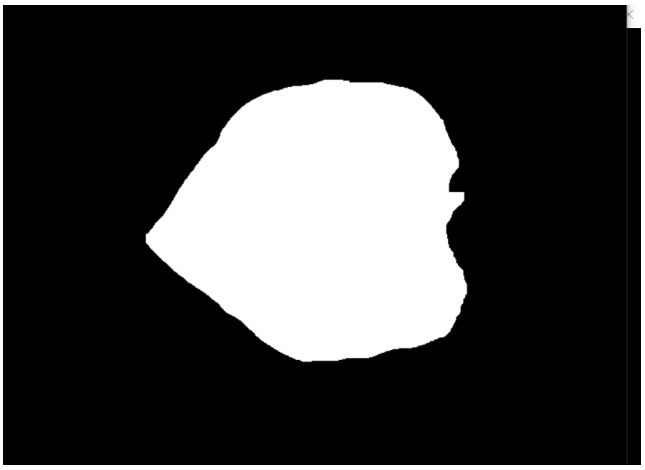

Fig. 4: Segmentation of example picture using algorithm above

The additional steps provided moderate results in getting rid of stem and shadows while using with binary thresholding.

2.2 Otsu's method

Otsu's method is used to perform automatic image thresholding. In the simplest form, the algorithm returns a single intensity threshold that separate pixels into two classes, foreground, and background. This threshold is determined by minimizing intra-class intensity variance, or equivalently, by maximizing inter-class variance .

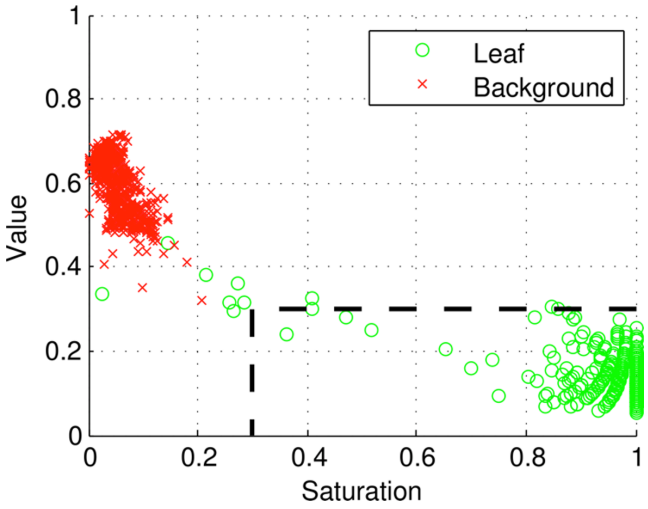

Fig. 5: Expectation maximalization clustering

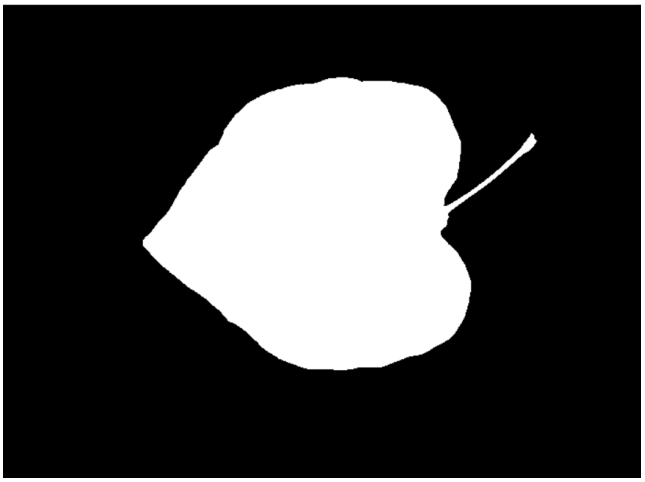

Fig. 5: Segmantation of example image with Otsu's method

To get rid of the stem we used morphological closing operation

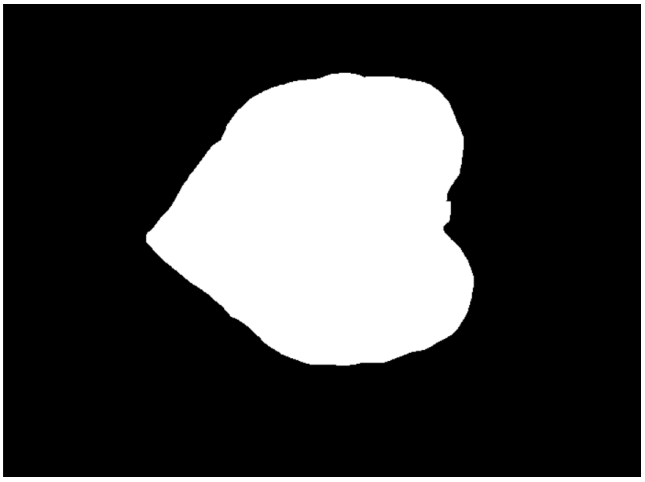

Fig. 6: Segmantation of example image with Otsu's method enhanced with closing algorithm

The other method to get rid of the stem was to use median blur, this option separates area near stem more aggressively, on the other hand it sacrifices leaf's edges segmentation sharpness.

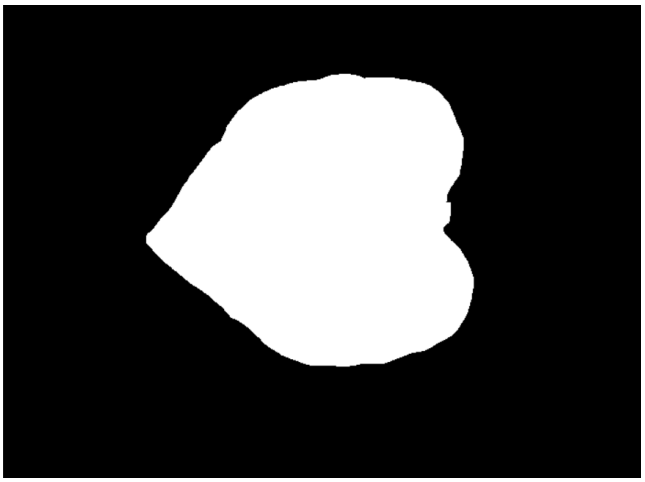

Fig. 7: Segmantation of example image with Otsu's method enhanced with median blur

2.3 Canny edge detection

Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. The optimal function in Canny's detector is described by the sum of four exponential terms, but it can be approximated by the first derivative of a Gaussian.

Applying the algorithm yields such a mask

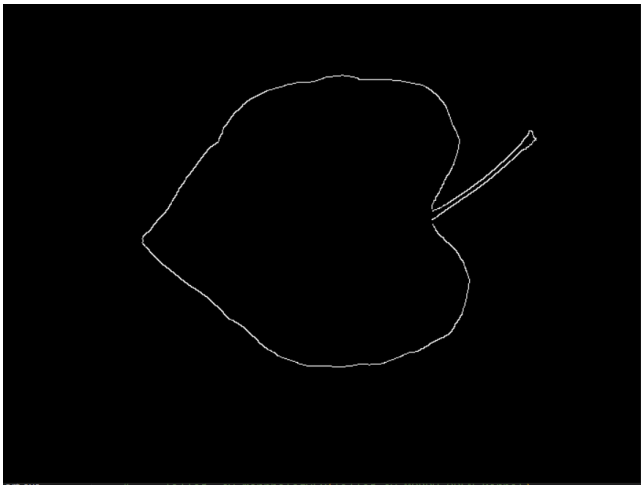

Fig. 8: Edges of example image

From this stage leads pretty simple algorithm:

1. dilate edges 2 times
2. erode edges
3. fill edges from the outside
4. apply closing
5. invert the mask

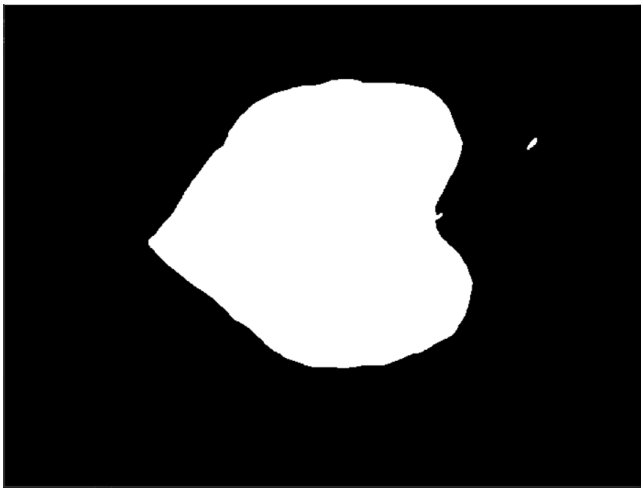

Fig. 9: Segmentation using above algorithm

The presented algorithm was great at conserving edges of leaf, however it was very sensitive to blur. To combat this we tried two approaches, first subtracting Gaussian blur from the image, second filtering using sharpening kernel. The latter solution did not yield significant improvement. However unsharpening in some cases did help improve segmentation significantly, which helped to increase IQR of Dice coefficient from 0.56 to 0.73

Fig. 10: Example image, whose canny segmentation yielded empty mask

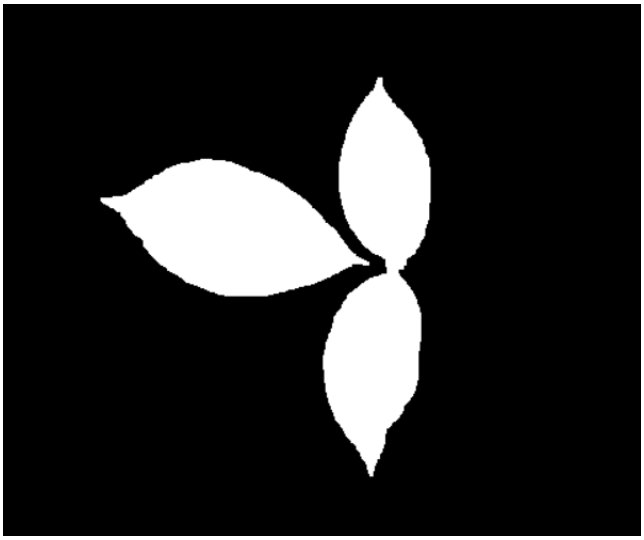

Fig. 11: Augmented canny segmentation

2.4 Dealing with false negatives

My stem detection algorithm, because of similar geometry, treated, leaves from conifer trees as a stems and marked them as negatives. To counter this issue I developed algorithm labeling all items in foreground, calculating maximal and minimal axis ratio for each label. Then based on greatest ratio across all labels I decided whether given leaf is from conifer tree. In case of positive response, stem reduction algorithms were bypassed.

However this algorithm did not made

3 Assessment methods

To assess our segmentation we used ground truths provided with dataset.

3.1 Intersection over union

Mean Intersection-Over-Union is a common evaluation metric for semantic image segmentation. IOU is defined as follows:

$$IoU \equiv \frac{target \bigcap prediction}{target \bigcup prediction}$$

The predictions are accumulated in a confusion matrix, weighted by sample weight and the metric is then calculated from it.

3.2 Dice coefficient

Dice coefficient is F1-score's equivalent for image segmentation and is defined as follows:

$$Dice \equiv \frac{2 * target \bigcap prediction}{target + prediction}$$

It values ranges from 0 to 1 for perfect segmentation

4 Segmentation Evaluation

4.1 Evaluation for dataset 1

https://www.overleaf.com/project/60c7124eab307802c84ca3e3

index	I…	Dice_binary	I…	Dice_otsu	I…	Dice_canny…
count	150	150	150	150	1…	150
mean	0.…	0.6640725433	0.…	0.80788745…	0…	0.6424815056
std	0.…	0.3605413862	0.…	0.34156231…	0…	0.4092305521
min	0.…	0.0041929518	0.…	0.01087336…	0	0
25%	0.…	0.3005322351	0.…	0.90656119…	0…	0.1091495929
50%	0.…	0.8551119822	0.…	0.98029836…	0…	0.9022593574
75%	0.…	0.9839594259	0.…	0.98890156…	0…	0.9646304745
max	0.…	0.997749811	0.…	0.99642839…	0…	0.9912213291

Fig. 12: summary table for dataset1

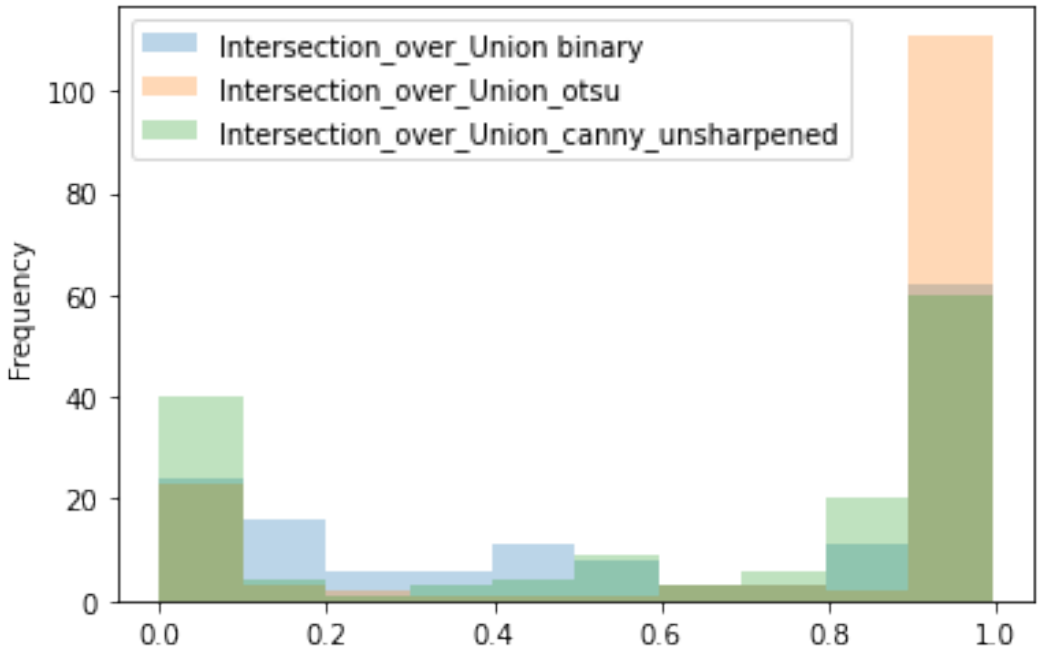

Fig. 13: histogram of accuracies for dataset1

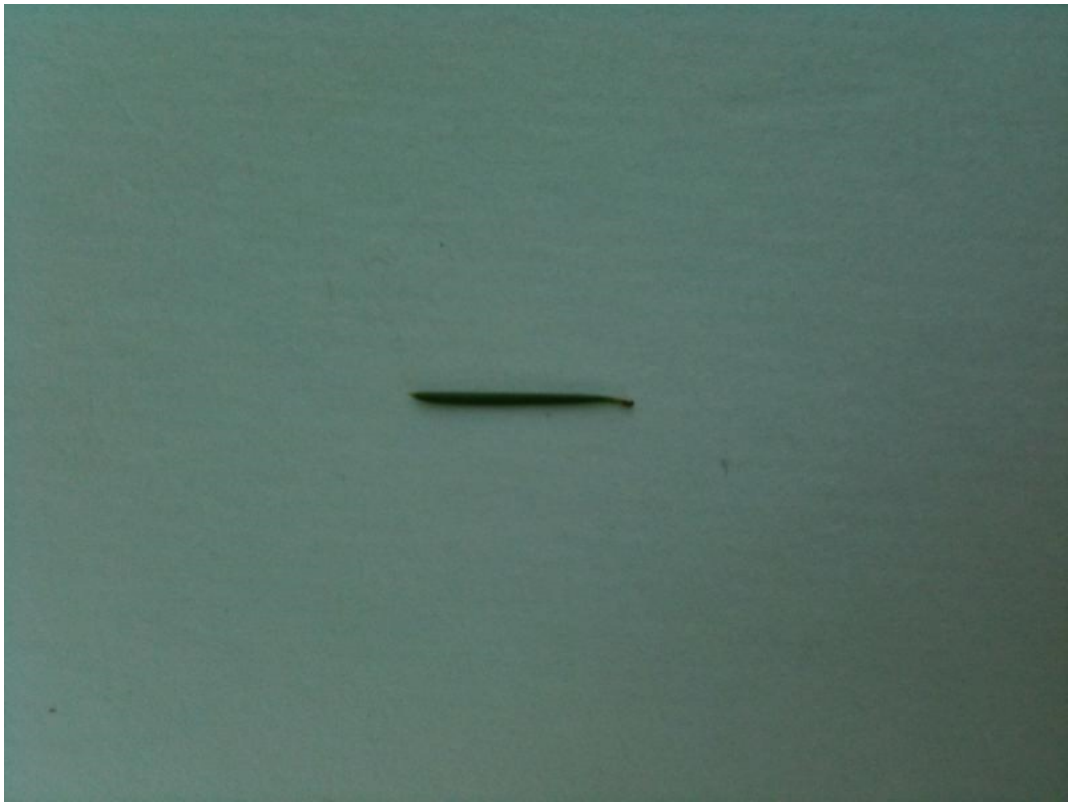

Fig. 14: example of false negative segmentation

Dataset 1 was relatively easy to segment, the photos were taken in good lighting conditions and

there were not much of needle leafs, which gave a lot of false negatives.
Algorithms failed to properly segment image above, mostly due to similar background, shape of leaf, blur and poor lighting conditions.

4.2 Evaluation for dataset 2

index	I…	Dice_binary	I…	Dice_otsu	I…	Dice_canny_u…
min	0…	0.0024269027	0…	0.0044984348	0	0
25%	0…	0.1395354846	0…	0.6561159622	0	0
50%	0…	0.5145530777	0…	0.9376303018	0…	0.1073929758
std	0…	0.3674346511	0…	0.3018867767	0…	0.3871020444
mean	0…	0.5144891584	0…	0.7770260397	0…	0.3388878633
75%	0…	0.9226321039	0…	0.9718777832	0…	0.7317027291
max	0…	0.9923686133	0…	0.9967153899	0…	0.989007055
count	1…	150	1…	150	1…	150

Fig. 15: summary table for dataset2

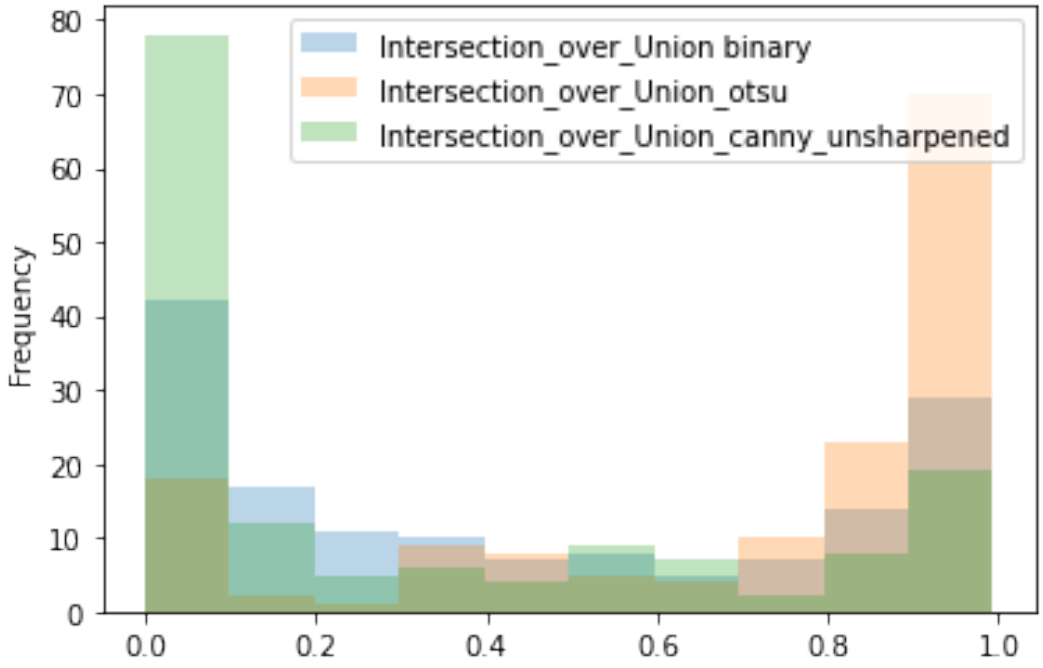

Fig. 16: histogram of accuracies for dataset2

Fig. 17: example of false negative segmentation by binary and Otsu's algorithms

Dataset 2 was particularly hard for Canny augmented algorithm. Most of the photos were blurred, with missed focus, which made it impossible to correctly segment them.
The example above were segmented correctly only by edge algorithm. It proved to be resistant to lighting and background on assumption that the photo was sharp.

4.3 Evaluation for the whole dataset

index	]	Dice_binary	]	Dice_otsu	]	Dice_canny_unsharpened
min	0…	0.0024269027	0…	0.0044984348	0	0
25%	0…	0.2075014556	0…	0.7739049283	0…	0.0002829932
std	0…	0.3710370892	0…	0.3221670838	0…	0.4257317544
50%	0…	0.665409037	0…	0.9658784532	0…	0.5886416452
mean	0…	0.5892808509	0…	0.7924567463	0…	0.4906846845
75%	0…	0.9662882799	0…	0.9860858287	0…	0.9479511594
max	0…	0.997749811	0…	0.9967153899	0…	0.9912213291
count	3…	300	3…	300	3…	300

Fig. 18: summary table for both datasets

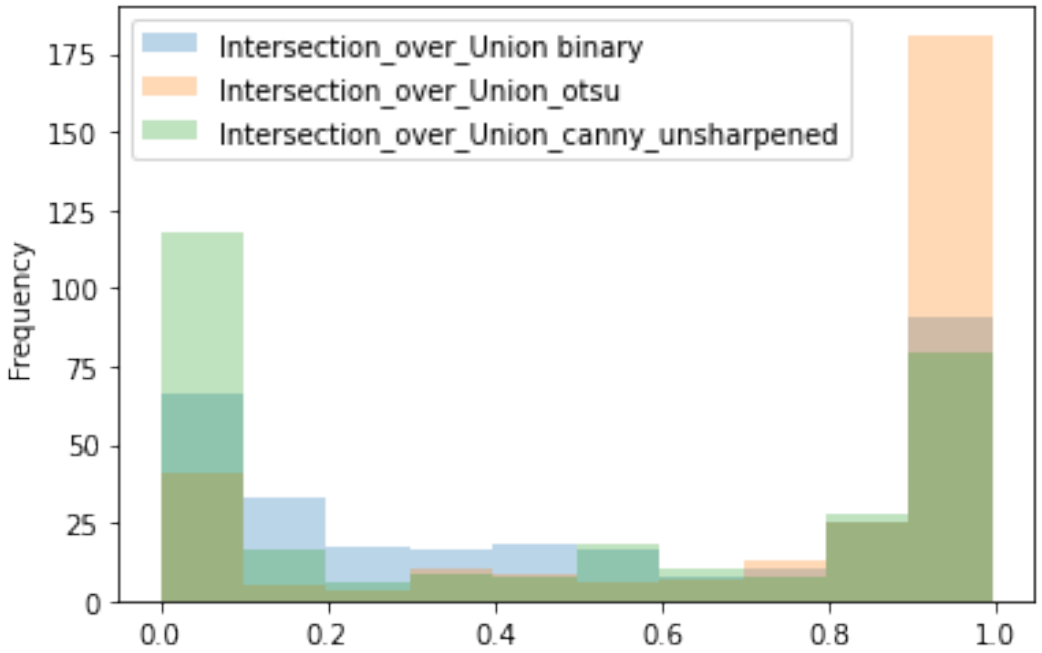

Fig. 19: histogram of accuracies for both datasets

5 Conclusions

Each of presented methods had different strengths. Binary segmentation was easy to implement and fast to compute, however it was sensitive to lighting conditions and foreground color. It could be improved using histogram equalization to extend dynamic range.

Edge algorithm was the best at extracting fine details of leaf, however it was hard to use due to it's sensitivity to blur.

Otsu's segmentation proved itself to be most robust and having overall good scores. There is still room for improvement, especially misclassification of needle leaves could be improved.

6 References

- "Leafsnap: A Computer Vision System for Automatic Plant Species Identification," Neeraj Kumar, Peter N. Belhumeur, Arijit Biswas, David W. Jacobs, W. John Kress, Ida C. Lopez, João V. B. Soares, Proceedings of the 12th European Conference on Computer Vision (ECCV), October 2012

- .https://docs.opencv.org/4.x/d7/dbd/group$_{imgproc.html}$

- https://scikit-image.org/docs/dev/api/skimage.measure.html