Multi-Class Image Classification on Leaves

Shaelin Naidoo

University Of KwaZulu-Natal, Durban KZN 4000, South Africa

Abstract. Images are segmented using an approach that includes Canny edge-detection, contouring and identification of connected components. A model for image classification is built based off the Hu moments invariant and the Haralick descriptors. Machine Learning classification models are compared for the task and Random Forest classifiers are shown to be the best equipped of all models tried.

Keywords: Computer Vision · Image Processing · Image Classification

GitHub: https://github.com/ShaelinN/Multiclass-Image-Classification-on-Leaves

1 Literature Review

Histogram equalization is an image enhancement technique that improves the contrast of objects by spreading out the gray level values in an image over the spectrum of possible values in a way that the result is roughly uniformly distributed. It makes each gray level highly distinct from the others. Global equalization considers the entire image when redistributing pixel values. Contrast Limited Adaptive Histogram Equalization (CLAHE) is a method of histogram equalization that only considers a small local region at a time when redistributing. This allows borders which would not be enhanced very much by global equalization to be made much more distinct, since they are changed specifically to contrast against nearby pixels.[13]

Edge detection is an essential part of computer vision [3]. Edge detection reduces the complexity of images by striping away unimportant details while preserving the general structure of images. Canny's algorithm for edge detection was designed to be resilient against errors. This has been backed up by comparative studies that pitted it against other algorithms such as Prewitt and Sobel [10].

Edge detection can play a large role in the initial stages of segmentation. Another simple method for segmentation is thresholding. This is the most rudimentary segmentation technique, but given the dataset in use, thresholding may be an adequate segmentation technique, given that the leaf regions are clearly contrasted against the white backgrounds. However, picking a threshold value may prove challenging, since different images, taken at different light levels, will react differently to the thresholds. Otsu's method for choosing a threshold is available

within the OpenCV library. This method is suited to bimodal images, which the leave images are. It calculates the threshold value such that the foreground is adequately separated from the background, by minimising the intraclass variance [12].

Another possible segmentation method is by adapting the related concept of contours. Contours are not guaranteed to be closed [1], which could present a problem when attempting to split up regions of an image using contours. However, based on the quality of data in the dataset, this should not be too big of an issue, as no gradual shadows or other means of blurring the leaf into the background are present.

Hu Moments Invariant are 7 features based on the concept of image moments and central moments, which when put together, describe a shape in an image. The moments invariant were designed such that changes by rotation, scale and reflection will have functionally no consequence on the values [9, 7]

The Haralick descriptors are a set of 14 features that describe the texture of an image [8].Some of the features they can be express are smoothness, coarseness and regularity [7]

2 Preprocessing and Segmentation

In the leafsnap dataset, particularly the lab subset which was used in this project, individual images are very distinct from their surroundings before preprocessing. Preprocessing before segmentation caused the capturing of segments to become much more difficult. Therefore, preprocessing for the purpose of feature enhancement was only applied once segmentation was complete.

2.1 Segmentation

To segment a leaf image, the first step was Canny edge detection. The upper and lower bounds were chosen to be one standard deviation from the mean of the image in the relevant directions. The result then underwent morphological dilation to close up as many gaps in the lines as possible. The third step was to obtain and fill in the contours of the image using the contour functions in openCV. This produced all objects in the image as white blocks, on a black background. However, at this point, the measuring scales surrounding the leaf became troublesome, as the images were of inconsistent dimensions and the equipment was irregularly placed. After noticing that leaves almost never touched the image border, while the scales almost always did, the following technique was employed:

1. Draw a thick rectangle as the bounding box of the entire image
2. Dilate the image again
3. Calculate the connected components using the function from CV2

4. Since the scales are usually touching the border, but the leaves aren't the scales count as one connected component with the border. This border + scales component was then blacked out

The idea for this came from Leafsnap's own paper, wherein they describe a similar albeit more complex mechanism that they used for segmentation, to remove regions around the edges of the images, which they found to often contain clutter [11].

With the scales and other clutter near edges removed, the next step was to perform another contouring step, this time filtering out small contours to remove small specks of noise. Once again this refined contour map was filled in. A minimum size threshold of 0.0007 was determined by experimentation. Any contours whose proportion to the image were smaller than this were discarded

The region remaining was then generally a good representation of the leaf by itself. It underwent morphological erosion to account for the dilations it underwent earlier. This is the final segmentation mask, which can be overlaid over the original image to extract the region of interest

2.2 Preprocessing

While the previously extracted segmentation captured the form of the leaf, it was not sufficient to make out the details on it. To achieve this, the original image was masked by the segmentation mask, and then underwent a local histogram equalization (more specifically Contrast Limited Adaptive Histogram Equalization) with a grid size of 8x8. Global Histogram equalization was considered at first, but the local equalization proved more effective at sharpening the images in the small set of trials run.

3 Feature Extraction

20 features are used in this model: the 7 Hu moments invariant[9], which describe the shape of a leaf, as well as the first 13 Haralick descriptors[8], which describe its texture, (such as orientation and shape/size of veins).

The Hu moments are obtained using the OpenCV 2 library[2], and the Haralick features make use of the Mahotas library. Mahotas claims the 14 Haralick descriptor to be unstable, and it is therefore not considered in this project[5]. The Hu moments had their logarithm taken as this turned miniscule values (some low enough to be on order of 10E-15) into larger numbers that were easier to work with.

4 Classification

In this project, only the lab subset of the Leafsnap dataset was used [11]. This dataset was chosen because it had a manageable number of classes and a high number of images, and these images were roughly of a consistent quality.

Random samples were taken to produce the train and test sets. The train to test ratio was 8:2. For both train and test sets, it was enforced that the proportion of images from each class be equal to the proportion of all other classes, in order to have a fair sample from all classes. This involved splitting the dataset by class, and splitting each class into train and test based on the proportion, before merging the test sets of all classes together, and merging the train sets of all classes together. The total train set was then shuffled. After this split was made, They were fed into a classification model.

Several alternative classification models were tried. The models used were the pre-built models from Sci-Kit Learn. The following models were included:

- Decision tree
- K nearest Neighbors
- Multilayer perceptron
- Random Forest

5 Alternative methods tried

Models with other extraction and preprocessing techniques that were constructed during the development process included models that only calculated Hu invariants (no Haralick features). The Hu-feature-only model struggled to achieve a top-1 accuracy higher than 30%. Considering Haralick features as well greatly enhanced performance. In trial runs, the Multinomial Naive Bayes classifier was tested out as well, although it required an extra data normalisation step, and this step caused poor results both for the Naive Bayes classifier and all other models. An Otsu Threshold segmentation model was attempted, where the first few steps of segmentation were performed by Otsu thresholding rather than Canny edge detection and contour filling. This method is still viable, despite not being chosen

6 Results and Discussion

6.1 Results of the segmentation Process

Despite the simplicity, the segmentation process generally achieved good results. Among the few issues were that occasionally, the contours could not be filled correctly, leading to the segmentation being a portion of the leaf outline, rather than the full leaf. additionally, leaves that touched the bounding box would have

been removed, despite being the desired segment (this problem is more prevalent with some species in the dataset than others). Additionally, the segmentations could use some extra erosion, as the current level of erosion does not balance out enough of the dilation that was performed. This project did not remove the leaf stem as an assumption was made that the stem may contain some useful information about the plant. An example of the results of segmentation is shown in the next few pages.

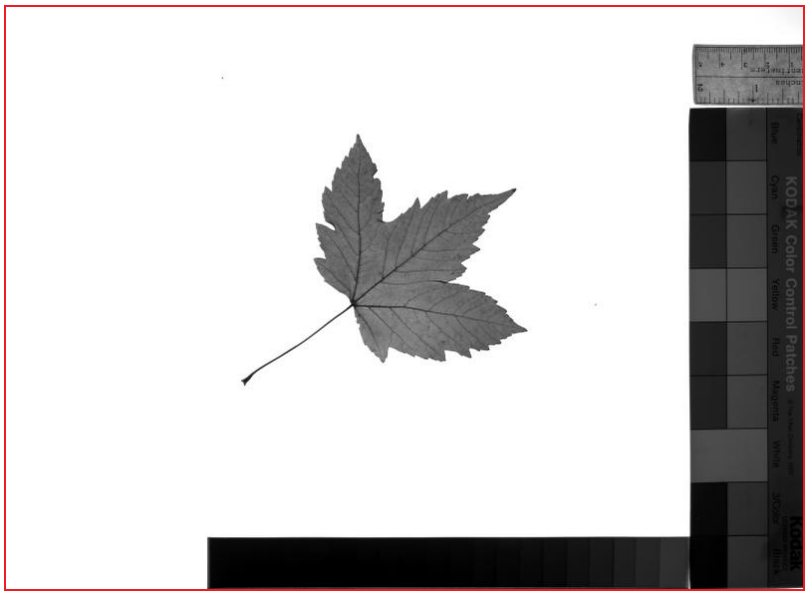

Fig. 1. original image

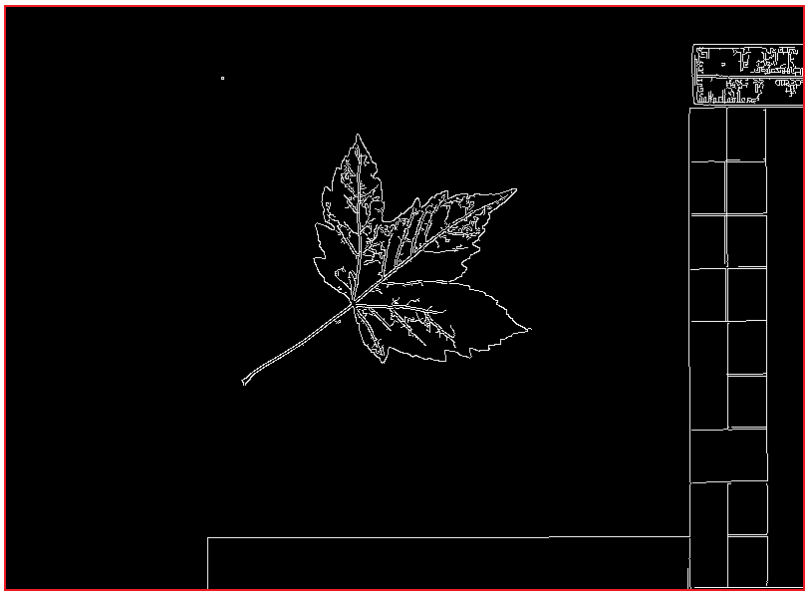

Fig. 2. after Canny edge detection

Fig. 3. Some dilation is applied

Fig. 4. contours of the image are filled in

Fig. 5. draw a bounding box around the whole image

Fig. 6. determine the connected components

Fig. 7. remove the component which touches the bounding box

Fig. 8. refine contours, removing small specks

Fig. 9. final segmentation mask obtained by eroding

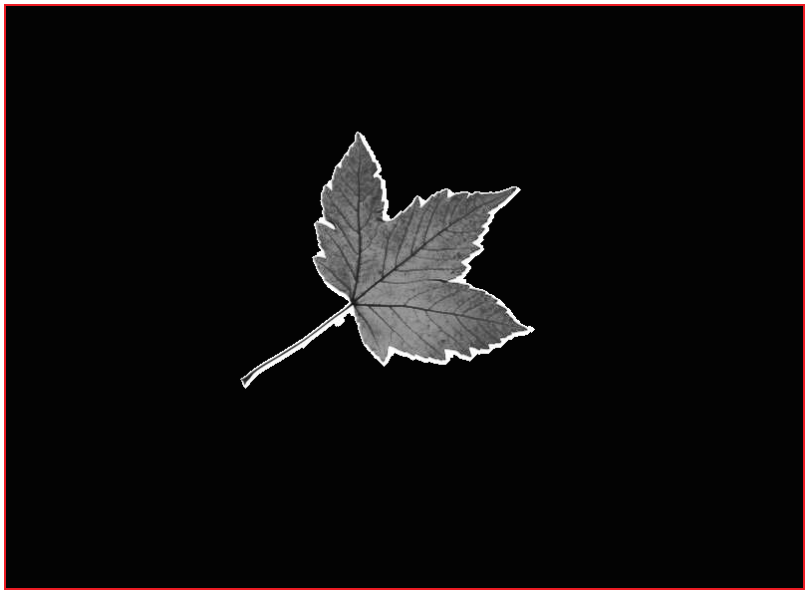

Fig. 10. Apply bitwise AND between the mask in fig. 8. and the original image in fig. 1. and perform CLAHE enhancement on the result

6.2 Results of the Classification Process

The final chosen classification model (Random Forest, with 200 estimators) has strongly identified a trend in the features of images, however, the accuracy results, are mediocre, with a 52% top-1 accuracy, and 81% top-5 accuracy. The confusion matrix of the model is shown on the next page (note that there are the occasional mis-classifications in the dark sections, although they may be difficult to see on the page. the full size image has been providedin the GitHub repository)

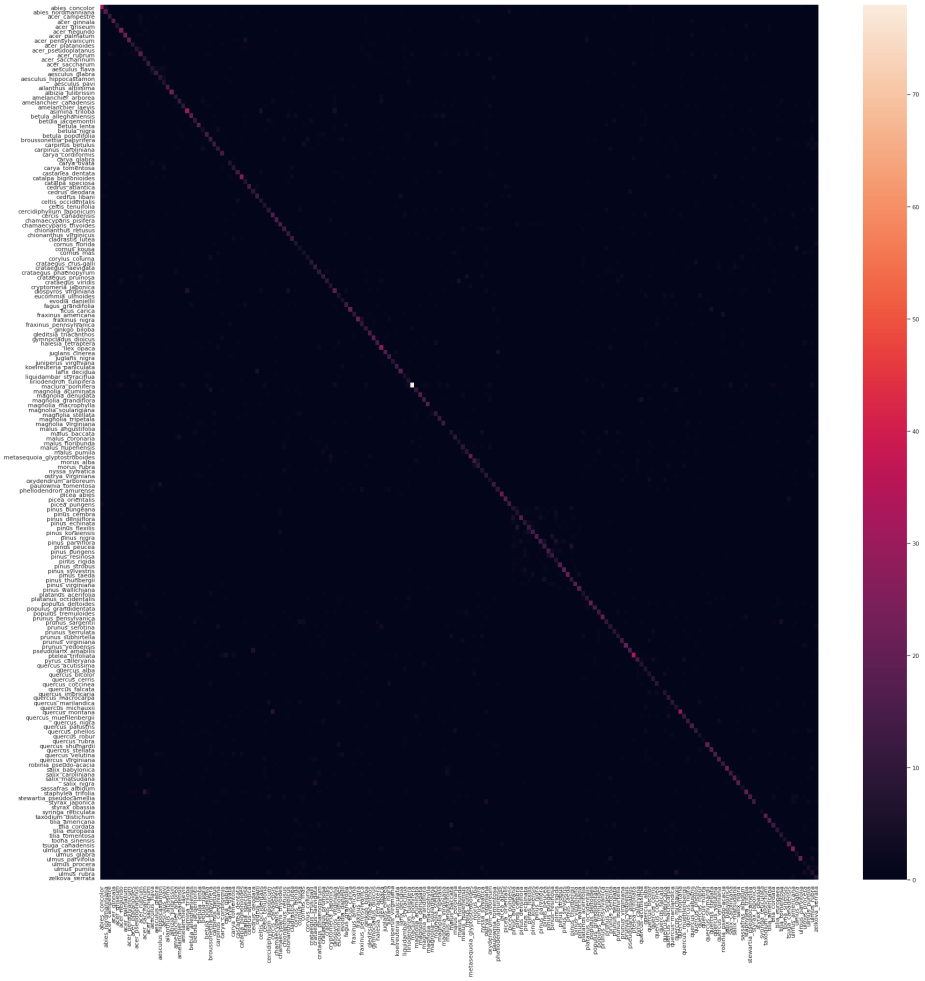

Fig. 11. Confusion matrix of the classifier

References

1. Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. (2010). Contour detection and hierarchical image segmentation. IEEE transactions on pattern analysis and machine intelligence, 33(5), 898-916.
2. Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.
3. Canny, J. (1986). A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence, (6), 679-698.
4. Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg,

Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke & Travis E. Oliphant. Array programming with NumPy, Nature, 585, 357–362 (2020), DOI:10.1038/s41586-020-2649-2 https://www.nature.com/articles/s41586-020-2649-2

5. Coelho, L.P. 2013. Mahotas: Open source software for scriptable computer vision. Journal of Open Research Software 1(1):e3, DOI: http://dx.doi.org/10.5334/jors.ac
6. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 12, 2825-2830 (2011) https://jmlr.org/papers/v12/pedregosa11a.html
7. Gonzalez, R. C., Woods, R. E.: Digital Image Processing. Third Edition. Pearson Education, Upper Saddle River, New Jersey
8. Haralick, R. M., Shanmugam, K., & Dinstein, I. H. (1973). Textural features for image classification. IEEE Transactions on systems, man, and cybernetics, (6), 610-621.
9. Hu, M. K. (1962). Visual pattern recognition by moment invariants. IRE transactions on information theory, 8(2), 179-187.
10. Juneja, M., & Sandhu, P. S. (2009). Performance evaluation of edge detection techniques for images in spatial domain. International journal of computer theory and Engineering, 1(5), 614.
11. Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I. C., & Soares, J. V. (2012, October). Leafsnap: A computer vision system for automatic plant species identification. In European conference on computer vision (pp. 502-516). Springer, Berlin, Heidelberg.
12. Otsu, N. (1979). A threshold selection method from gray-level histograms. IEEE transactions on systems, man, and cybernetics, 9(1), 62-66.
13. Reza, A. M. (2004). Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement. Journal of VLSI signal processing systems for signal, image and video technology, 38(1), 35-44.
14. Shepherd, B. A. (1983, August). An Appraisal of a Decision Tree approach to Image Classification. In IJCAI (pp. 473-475).
15. Wes McKinney. Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 51-56 (2010)(http://conference.scipy.org/proceedings/scipy2010/mckinney.html)
16. StackOverflow discussion on automating thresholds for Canny Edge detection, https://stackoverflow.com/questions/4292249/automatic-calculation-of-low-and-high-thresholds-for-the-canny-operation-in-open. Last accessed 20 Jun 2021
17. StackOverflow discussion on Hu Moment extraction, https://stackoverflow.com/questions/24580263/shape-recognition-using-hu-moments-from-opencv-in-python. Last accessed 20 Jun 2021