Automatic Classification of Plant Cell Types

Michael Scott

I. Introduction

Cells are in everything all around us. Individual cell sizes of plant species have a direct impact on numerous things in the plant. Different sizes impact how fast water can reach leaves through the vessels, the density of cell walls, and also the cells thickness to breadth ratio which determines its implosion resistance. Being able to better understand these different qualities of plant cells opens up a whole new world of research. Plant cell research has Developing a further comprehension of the plant cell world allows us to discover new unique qualities of plants that have yet to be discovered. The three primary cell types examined in this study are ray parenchyma, vessels, and fibers. The parenchyma provide sugar and water storage as well as serve to transport water short distances. In the dataset, the parenchyma appear as dark lines stretching across the image lying in between the bright yellow fibers. The vessels act as long distance water transporters and also provide mechanical support as a secondary function. In the dataset, the vessels appear as round gray colored circles located in various places throughout the image. The fibers, located around the parenchyma and vessels, simply act as mechanical support. In the dataset, the fibers appear as small bright yellow circles that take up a majority of the image that they are in. Analyzing these three different cell types allows us to better understand the plants that are being examined and how water is transported through each of these. Along with this, we are also able to learn more about the physiology and ecology of the individual plant species whose anatomy that we examine. This analysis is commonly done by hand which takes hours of work to accurately label and measure each plant cell within an image. It is an arduous process that requires a significant amount of time and concentration.

Automation of this process creates an opportunity to save an immense amount of time and labor. With this decrease in time spent manually classifying, more effort could be spent analyzing these different plant species and the properties unique to each one. Developing or discovering a method of automating this process would very well open the doors to new and exciting research within the respective field. Automation has been attempted before but with varying results. [1] Another research study shows that classification was possible using a Support Vector Machine as well as a Random Forest when looking at laser scanning confocal images. [2] One study used linear discriminant analysis in order to classify different cell types for their data. [3] These studies show that the automated classification of cells is something worth doing as the researchers doing so also see the opportunity in automating this difficult process. Lots of people want to investigate this as the days of manually measuring and classifying are quickly being realized to be inefficient and tedious.

The goal of this paper is to determine if automatic analysis of different plant cell types can be done using various scikit-learn classifiers as well as a pre-trained ResNet model. Upon discovering which of these works best, we all will have a better idea of what method to use going forward so that the automatic classification of plant cell types can be used easily in the future of plant cell research.

II. Methods

A. Dataset

The image dataset used was provided privately by Dr. Helen Holmlund and Dr. Anna Jacobsen. There are 488 images in total with the dimensions 1600x1200. 74 images are from three different Ceanothus crassifolius (CCR) roots with root 1 having 26 images, root 2 having 24 images, and root 3 having 23 images. 82 images are from three different CCR stems with stem 1 having 24 images, stem 2 having 36 images, and stem 3 having 22 images. 86 images are from three different Ceanothus oliganthus (CO) roots with root 1 having 32 images, root 2 having 24 images, and root 3 having 30 images. 74 images are from three different CO stems with stem 1 having 23 images, stem 2 having 24 images, and stem 3 having 27 images. 87 images are from three different Rhamnus californica (RCA) roots with root 1 having 25 images, root 2 having 32 images, and root 3 having 30 images. 85 images are from three different RCA stems with stem 1 having 37 images, stem 2 having 23 images, and stem 3 having 25 images. All the images were stained for starch so that it would be possible to see the dark circles. For the classification, I used the open platform labelflow.ai to label each cell type (Fiber wall, Parenchyma wall, Vessel wall) by creating a bounding box for each label. A sample of the images provided in the dataset can be seen in Figure 1.

B. Evaluation Metric

The evaluation metric used is scikit-learn's accuracy score. The equation for this evaluation metric is as follows:

$$accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i) \quad (1)$$

For this equation, $\hat{y}_i$ is the predicted value of the i-th sample and y_i is the ground truth of the same i-th sample. We evaluated our machine learning model with this metric to see how our model's prediction does with matching to the actual label. This also uses 1(x) as the indicator function. [4]

C. Preprocessing

In order to make the data valid to input into the machine learning model so that it may work, I cropped the images so that each data point would be one individual cell type. By creating a bounding box for each cell type, labelflow.ai also allowed me to export labeled images into csv files which included the coordinates of the bounding boxes. With the coordinates of the bounding boxes, I used the Python Imaging Library (PIL) module to open and transform the images so that the images could be properly cropped. Then the images were augmented and normalized for training and just normalized for validation. To augment images to create more data for our dataset, ImageDataGenerator from Keras preprocessing was used. The parameters provided to the ImageDataGenerator are as follows: rotation range = 40, width shift range = .2, height shift range = .2, shear range = .2, zoom range = .2, horizontal flip = True, and fill mode = nearest. These parameters created images that were different enough from the initial image to properly challenge the models that were given this data. This augmentation allowed me to train the models with much more data and allowed me to create data that would be more difficult for the model to learn yet produce a better outcome. An example of a few augmented images can be seen below in Figure 2.

D. Types of Models

The types of models used in this analysis are primarily scikit-learn models as well as the pre-trained ResNet-18 model. The scikit-learn models used are: KNeighbors Classifier, Ridge Classifier, Support Vector Machine (SVM), Multi-layer Perceptron (MLP), Stochastic Gradient Descent Classifier (SGD).

III. Results

The KNeighbors Classifier achieved an accuracy score of .5891. The Ridge Classifier achieved an accuracy score of .4562. The SVM achieved an accuracy score of .6098. The MLP achieved an accuracy score of .5891. The SGD achieved an accuracy score of .5106. Each of the runt times for the models from scikit-learn were negligible as the runs were completed instantaneously upon running the code. The pre-trained ResNet-18 model achieved an accuracy score of .9071. The runtime of the ResNet-18 model had a runtime of 47 minutes and 1 second.

IV. Discussion

The pre-trained ResNet-18 model greatly surpassed the scores of the scikit-learn models as it achieved the highest accuracy score. The best scikit-learn model was the Support Vector Machine. The second best scikit-learn model was the KNeighbors Classifier. The third best scikit-learn model was the Multi-layer Perceptron. The fourth best scikit-learn model was the Stochastic Gradient Descent Classifier. The worst performing model was the Ridge Classifier. The reason why ResNet-18 severely outperforms these other models is likely due to ResNet-18 being pre-trained on over a million images while the scikit-learn models are not pre-trained. The varying scores among the scikit-learn models likely is just because of the differences in each classifier. This would explain why the Ridge Classifier performed so poorly while the Support Vector Machine was able to perform considerably better.

V. Conclusion

The results of these models are quite promising for the future of being able to automatically detect and measure different plant cell types. The results from ResNet-18 showed that models pre-trained on millions of images should be used in future studies to accurately predict the cells. The only downside to using pre-trained models is the considerable increase in runtime. With the success of this automation process, it is possible for people to work on further projects that would help to reduce significant amount of time and labor for other types of labor; such as calculating the distance of the walls for each cell type, and annotating each cell type within the image. Researching further into automatic methods of measurement within cells would certainly be a massive step in being able to better understand changes in plant cells. Something I could have examined more is other pre-trained models such as other variations of ResNet. These other models may have lower run times as well as better accuracy scores compared to the ResNet-18 network. It would be interesting to attempt to use other networks such as EfficientNet or DenseNet to see what their times and accuracy scores would be. These models have varying numbers of trainable parameters so each would definitely produce noticeable changes. A greater dataset size may also produce better results for both the pre-trained and scikit-learn models. A Random Forest classifier from scikit-learn would also be worth looking at as it appeared to perform better than the Support Vector Machine in another study.[2] Along with a greater dataset, I could have augmented more images in order to possibly further improve each of the models' results. I may also could have taken different approaches to preprocessing in order to create better results by augmenting the data in different ways.