Detecting and identifying individual plants and leaves of rosette plants using YOLO, U-Net, and Watershed Algorithms

1st Ziyang Liang
University of New South Wales
Sydney, Australia
ziyang.liang@student.unsw.edu.au

2nd Anant Krishna Mahale
University of New South Wales
Sydney, Australia
a.mahale@student.unsw.edu.au

3rd Sebastian Novoa
University of New South Wales
Sydney, Australia
s.novoa@student.unsw.edu.au

4th Yuntao Zhou
University of New South Wales
Sydney, Australia
yuntao.zhou@student.unsw.edu.au

5th Hang Zhu
University of New South Wales
Sydney, Australia
hang.zhu.1@student.unsw.edu.au

***Abstract*—Image-based plant phenotyping is a growing field of computer vision in agriculture. Detecting plants, plant segmentation and individual leaf segmentation are some of the most important tasks in plant phenotyping. In this paper we have focused on 2 common rosette plants; Arabidopsis and Tobacco. For the tasks of plant detection and instance segmentation we will be using the part of the data set that contains multiple Arabidopsis plants placed in a tray. For Multi-instance segmentation task we make use of both Arabidopsis and Tobacco images that contain individual plants. Each task will be tackled by a different method. For the *Object Detection* task we implemented the famous YOLO algorithm to detect multiple plants which proved to be highly accurate. For the *Semantic Segmentation* problem we used a U-Net model which proved to be very efficient considering that it had to be trained with very few images. To tackle the *Multi-Instance Segmentation* problem, our first approach was to apply watershed segmentation, but due to background noise, we got a very poor performance on the instance segmentation. After exploring and experimenting with different approaches, we decided to use the U-Net model to extract the foreground, and then applied Watershed segmentation to achieve the multi-instance segmentation. Later we decided to compare our multi-instance segmentation results with deep learning models such as UPGen (An instance segmentation model base on Mask R-CNN) and DeepColoring (An instance segmentation model base on U-net).**

***Index Terms*—OpenCV, U-Net, DeepColoring, Mask R-CNN, UPGen**

I. Introduction

Plant phenotyping is a crucial task for many agriculture related industries and in the past few years computer vision has made it possible to automate this task to previously unsuspected levels. Phenotype is the term used to describe all the observable traits of a given organism that arise from the organism's DNA and their interaction with their environment. In the case of plants these traits include characteristics such as number of leaves, biomass, leaf shape, overall health, height, among many others. Many of these traits are of special interest for any agricultural related activity where the objective is to grow plants as quickly, cheaply, and healthy as possible. In this project we will be exploring how a computer vision approach to plant phenotyping can help us extract some interesting characteristics from plant images.

In this paper, will focus on three distinct challenges. The first one corresponds to identifying individual plants in an image where an unknown number of plants are present in different growing stages. This corresponds to the problem of *Object Detection* and drawing bounding boxes that help to locate and count the number of plants in the image. The second challenge is to separate the detected plant from the background i.e. segment plants from the background, which in computer vision is known as *Semantic Segmentation*. This segmentation is particularly helpful as it can be used to estimate the plant's biomass by obtaining the Projected Leaf Area (PLA). Finally, the third challenge is an extension of the previous one. Given the natural structure of plants that evolved to seek sunlight, an aerial perspective photo is bound to have overlap between its leaves, hence the semantic segmentation discussed above can give an inaccurate measure of the plants real biomass. For this reason, another approach is to obtain an individual segmentation for each one of the leaves. This is known in computer vision as *Multi-Instance Segmentation*.

The dataset provided for this project comes from the Plant Phenotyping Dataset available on [1] and is made of different type of pictures containing Arabidopsis and Tobacco plants.The images have been captured in 2 separate laboratories from the top view images at once or many in the scene.It consists of two distinct type of images. The first type are images of multiple plants in trays, along with the location of the bounding boxes locating each plant and the foreground segmentation. These are the images that will be used for the *Object Detection* and the *Semantic Segmentation* problem. The second type are images of individual plants, along with their respective individual leaf segmentation. These are the images that will be used for the *Multi-Instance Segmentation* problem. It is important to note that compared to other machine learning datasets, this one has a very limited number of training examples. In the same time, the image between different format have huge different on image shape and the complexity of the background, within the *Ara-2012* and *Ara-2013*, most of them plant itself have the majority part of the image, but for *tobacco* image due to the size of the plant, the plant area only have low percentage of the whole image. Within the data set it also provide the label image, leaf skeleton/centroid, plant segmentation image for us to evaluate the performance of different method or neural model.

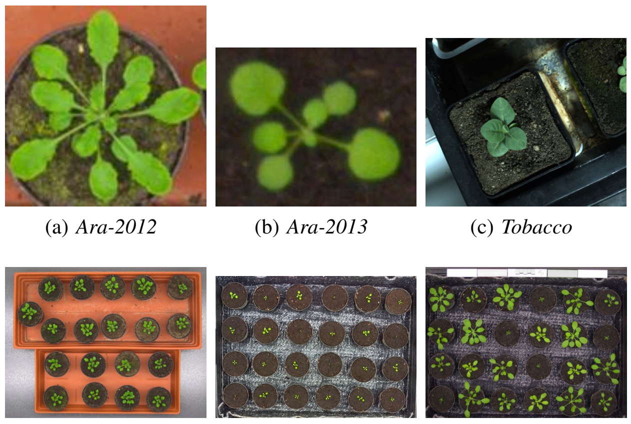

(a) *Ara-2012* (b) *Ara-2013* (c) *Tobacco*
(d) *Tray / Ara-2012* (e) *Tray / Ara-2013* (f) *Tray / Ara-2013*

Fig. 1: An overview of different data set images

II. Literature Review

A. Object Detection

Object detection is one of the most essential tasks in Image processing and Computer Vision applications. Previously, we saw rise in feature descriptors such as SIFT [2] and SURF [3] which was used as primary tool for various computer vision related tasks including object detection. Even though these algorithms are not susceptible to rotation of the image, they do not work well on the noisy images and cannot be generalized to work on a set of images belonging to a particular class.

Lately, many researchers have adopted deep-neural networks with different architectures to solve problems in computer vision field. As these networks have proven to be more efficient than conventional computer vision techniques, a few algorithms have already replaced many traditional methods.

Redmon et al. [4] proposed an algorithm called YOLO (You only look once) in CVPR2016 (IEEE conference on computer vision and pattern recognition of 2016). It is a typical one-stage target detection algorithm which combines classification and regression of target using anchor box. It is highly efficient, flexible and not only generalizable but also works on various kinds of images and datasets. YOLO V3 was proposed 2 years after the release of the original YOLO algorithm.

B. Semantic Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox [5] implemented Unet for Biomedical Image Segmentation. The UNet architecture was built to make use of the feature map obtained from the convolutional layers in the contraction path to then reconstruct the image in the expansion path. At every step of the expansion, the feature maps of the corresponding contraction layer are appended to the inputs to reconstruct the image. Using UNet we managed to achieve high accuracy with few images.

Bharath Hariharan, Ross Girshick et al [6] implemented recognition algorithms which is based on convolutional networks. These algorithms also use the result of the final layer as feature representation. The disadvantage is the output in last layer may be coarse, it is difficult to obtain precise localization. At the same time, previous layers cannot capture semantics correctly. Therefore, hypercolumn is defined as the vector of activation to improve the situation.

C. Multi-Instance Segmentation

Bernardino Romera-Paredes et al [7] developed an end-to-end model of recurrent instance segmentation by combining convolutional LSTM [8] and spatial inhibition modules as a way to keep track of spatial information within each image allowing to segment one leaf at a time. The approach also deploys a loss function that learns to sequentially segment all different instances and enables the model to learn and determine the segmentation.

Mengye Ren et al [9] proposed another neural network that uses visual attention to compute instance segmentation jointly with counting. By generating a temporal chain through an LSTM cell that outputs one instance at a time, this approach has sequential focus.

Andrei et al [10] implemented a deep learning approach for counting leaves in rosette plants. They modified ResNet50 residual neural network to learn a leaf counter from the RGB image of a rosette plant. They applied histogram stretch to all the images and resized all the images to 320x320X3.

Victor et al. [11] proposes a reduction of instance segmentation to semantic segmentation, whereas class labels are reused for non-adjacent objects. It assigns either one of the instances of the object or the background to each pixel in the image. It uses connected component analysis to retrieve the instances.

In order to achieve multi-instance segmentation by traditional method, at first we got inspired by Min Bai et al [12], building the *watershed* model combined with different architectures can highly improve multi-instance segmentation, but as for a traditional model, multi-instance segmentation is not an easy job, especially because the background in the image adds complexity. The segmentation task is also different from Min Bai's work, hence we had to explore further on how to achieve a good model combination.

Our *U-net watershed* mix model is closer to the *Wageningen* [13], inspired by combination of the model, traditional model cannot handle complex background. Instead of using traditional method to remove the background, using neural network is more efficient, inside the *Wageningen*, Hanno S. al uses simple feed-forward neural network for background pixel removing. As their results show, this approach did a great job in background removing, which inspired us to explore further into using more complex models such as CNN or R-CNN.

Compared to other datasets such as *coco* dataset, the *cvppp* dataset represents more of a challenge. The task at hand is to segment the plant and the leaf from the background perfectly, and that cause an evaluation problem. Using our previous evaluation method we cannot get a satisfactory score of the model on leaf/plant segmentation. Therefore, we will be using *Symmetric Best Dice* [14] evaluation method. Considering that when evaluating the leaf segmentation it is hard to correspond each leaf from the algorithm prediction to the ground truth, using *Symmetric Best Dice* we can have a better understanding of the model segmentation performance. Meanwhile, it reduces the post processing level of evaluating a leaf segmentation.

Since 2018, deep neural networks made huge improvements on image processing, which lead us to explore further on the performance from deep neural network such as *U-net* and *Mask R-Cnn*. Now we can have a direct comparison between the traditional method and neural network method. Even the network is *U-net*, Kulikov V. et al. [15] have modify the structure of *U-net* to achieve multi-instance segmentation, and it also get a better Symmetric Best Dice score compared to traditional method like *watershed*. Furthermore, using *Mask R-Cnn* in *cvppp* data set we got the best performance compared to previous models. [16] can achieve up to 88 symmetric best dice score, and this further inspired us to use deep neural network on leaf segmentation.

III. Methods

A. Object Detection

The Fig. 2, represents the input image. We need to detect every plant in the given image and draw bounding boxes around each of the plant and display the total number of plants in the image.

To perform this task, we propose and compare two approaches, Binarization and Edge Detection which falls under traditional approaches and YOLOV3 which is a deep learning approach.

1) Traditional method: There are mainly 3 steps involved in this method to detect the plant in a given image. They are, Binarization, Edge-detection and Marking the boundries of the detected plants.

a) Binarization: This step produces a binary image where in plants will be marked in white color and background will be marked in black color. Since the the plants are kept in a tangarine colored tray, we assume that only plants are colored green. Based on this assumption, colors in HSV range (1,1,80) to (255,100,217) are changed to 255, and those color which are not in this range is assumed to be background, changed to 0. The figure 3 shows the output of binarization.

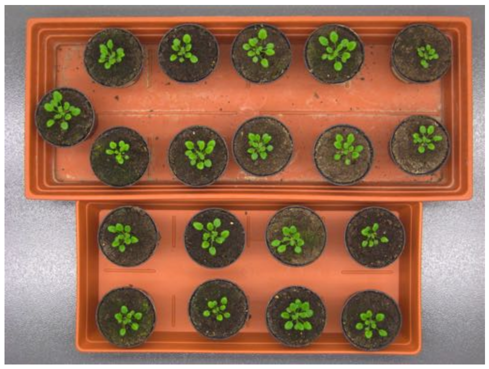


Fig. 2: Arabidopsis plants kept in a Tray. A input Image for Object(Plant) Detection and counting the number of plants in the Tray

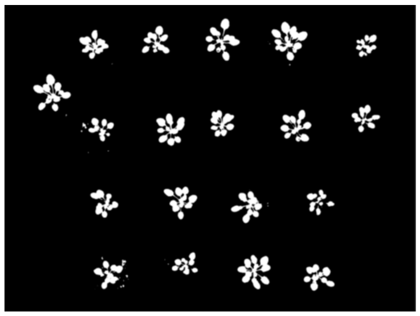


Fig. 3: Result of Binarization Method applied on the input image 2

b) Edge detection: The ouput binary image obtained from binarization method is used as input in this method. To detect the edges of the plants, we make use of Canny and findContours methods implemented in Open CV Library, which returns all the location information of the plants' edges. Using boundingRect function implementation from Open CV Library, we can obtain the bounding boxes. Better results can be achieved with as many as 7 iterations. The Figure Fig. 4 shows the improvisation of bounding box in each iteration.

c) Draw rectangles to the original image: Using cv2.retangle() to draw the bouncing box on the original image.

2) YOLOV3: The Figure 5, represents the architecutre of YOLOV3 network. It uses a convolution neural networks as a extraction network called darknet-53. [17] From the input to the output of darknet-53, is a down-sampling process,the width and height of the image will be compression and the channel will be increased.Each channel represents one feature the image. The last 3 feature layers of darknet-53 will be output to next level network.
The last feature layer will be output to a 5 layers convolution networks which has two outputs.One of them is using as the input of the predictive network.This predictive network is show as the lowest red rectangle in Fig. 5.It will separate the

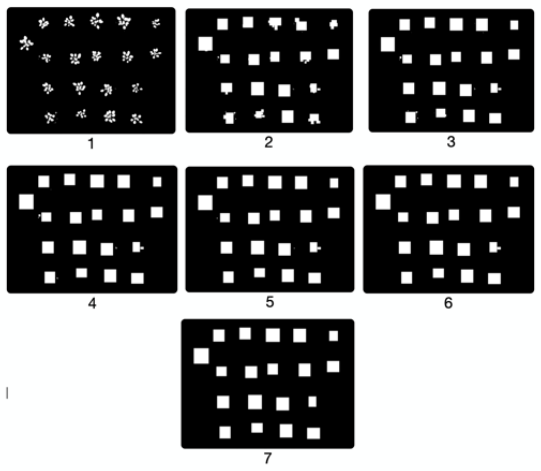


Fig. 4: The image of each step

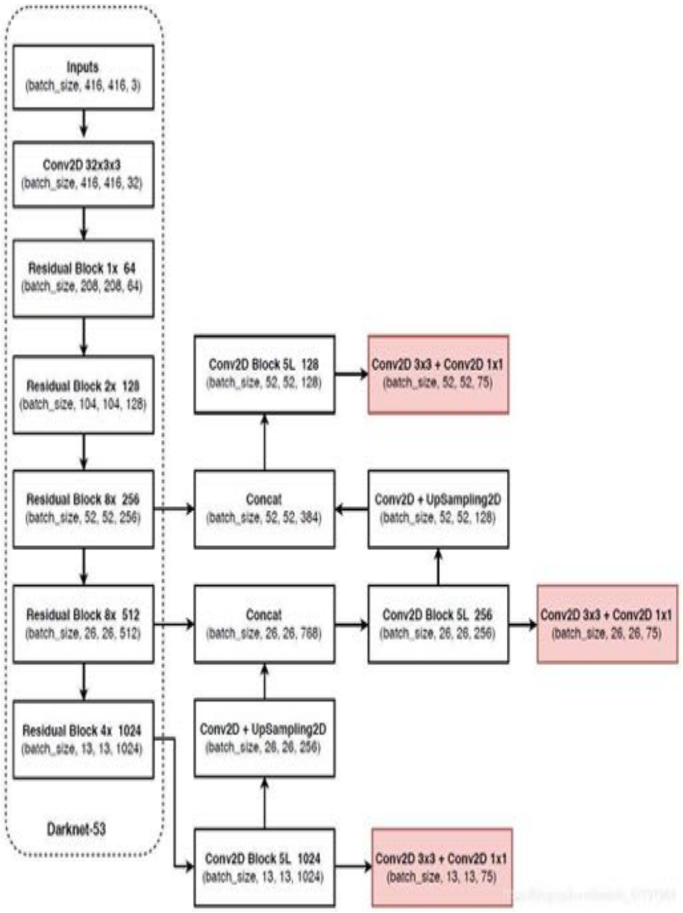

Inputs (batch_size, 416, 416, 3)
Conv2D 32x3x3 (batch_size, 416, 416, 32)
Residual Block 1x 64 (batch_size, 208, 208, 64)
Residual Block 2x 128 (batch_size, 104, 104, 128)
Residual Block 8x 256 (batch_size, 52, 52, 256)
Residual Block 8x 512 (batch_size, 26, 26, 512)
Residual Block 4x 1024 (batch_size, 13, 13, 1024)
Darknet-53
Conv2D Block 5L 128 (batch_size, 52, 52, 128)
Conv2D 3x3 + Conv2D 1x1 (batch_size, 52, 52, 75)
Concat (batch_size, 52, 52, 384)
Conv2D + UpSampling2D (batch_size, 52, 52, 128)
Concat (batch_size, 26, 26, 768)
Conv2D Block 5L 256 (batch_size, 26, 26, 256)
Conv2D 3x3 + Conv2D 1x1 (batch_size, 26, 26, 75)
Conv2D + UpSampling2D (batch_size, 26, 26, 256)
Conv2D Block 5L 1024 (batch_size, 13, 13, 1024)
Conv2D 3x3 + Conv2D 1x1 (batch_size, 13, 13, 75)

Fig. 5: The Architecture of YOLOV3 Network

image to large-size sub-image and has 3 prior boxes on each one to check if there is an target object in them.If it is ,the center and size of the the prior box will be adjusted.Another output will be up-sampling and stack with the network output of the second last layer of darknet-53.The second last layer of darknet-53 separate the image to medium-size sub-image. Then the stack output will be connected to a 5 layers convolution networks which also has two output.One of them connect to the networks show as the second last red rectangle in the Fig. 5,to detect the medium-size target objective.Another output also will be up-sampling to combinate with the third last layer.The highest red rectangle in the Fig. 5 will be use to detect the small-size target objective.

So YOLOV3 is actually a multi-scale object detection network.The three outputs of the entire network are used to identify large, medium and small targets.The detection process of YOLOV3 is to divide the input image into grids of different sizes. The interior of each grid will contain three a prior boxes. Perform target detection inside each a prior box, adjust the a prior box and finally form a prediction bouncing box. [18]

B. Semantic Segmentation

After conducting an extensive research looking for viable methods to approach the foreground segmentation challenge, we decided to use a UNet architecture. This network architecture was designed for tackling biomedical image segmentation problems where only a limited amount of data is available for training [5]. This is indeed the case we were facing, having a dataset comprised of only 43 images. It is this, perhaps, more than any other reason, why we chose to implement this type of approach to semantic segmentation. Another reason we chose this network is that it is relatively simple to implement compared to other methods such as the one presented here [6]. The particular difficulty of image segmentation as opposed to a simpler image classification task, is that not only do we need to obtain feature mappings of the image, but we also have to later reconstruct the image. This is precisely what the UNet does. The main idea behind UNet is to take advantage of Convolutional Neural Layers that can learn the feature mapping of the image and then use these feature maps to reconstruct the image. This is accomplished by concatenating a cropped output of the downsampling part to the upsampling part. The architecture of this network is made of 2 paths, the contracting path on the left and the expansive path to the right. For a visualization of the basic architecture of the network see Figure 10. The contracting path follows a typical convolutional network architecture, where there are consecutive 3x3 convolution layers activated by ReLU and followed by 2x2 MaxPooling with stride of 2 for downsampling. The number of feature maps doubles after each downsampling step so that the architecture can effectively learn complex structures. On the other hand, the expansive path is a symmetry of the contracting path. Each block consists of an upsampling of the feature maps by an up-convolution that effectively halves the number of feature mappings. Then a concatenation of a cropping from the feature map that corresponds to the symmetric block in the contracting path. What allows the network to use the features learned previously to reconstruct the image. Then there are 3x3 convolutions followed by active functions. At the very end the resultant mapping passes through another 3X3 convolutional layer where the number of feature mappings is the number of segments desired which in this case corresponds to 2. The loss function used for this model was Binary Cross Entropy with and IoU metric commonly used for image segmentation.

C. Multi-Instance Segmentation

After our extensive literature review we found there are a number of methods to achieve Multi-Instance Segmentation. Our approach will be separated into two parts, one is using traditional method like *watershed* and *superPiexl*, and another is using *Neural network* such as CNN. Consider the *cvppp* data is a challenging dataset, we wanted to compare the performance between the traditional model and deep learning model, thus we used traditional model as the baseline model and for the deep learning model as the comparison model. Meanwhile to achieve higher performance score we have implemented different formats and used customized modifications on different models.

1) Watershed model: As a baseline model, we try to implement a multi-instance segmentation model with traditional methods including *watershed* or *superPixel*. Based on our literature review using *watershed* might be an option for instance segmentation in the traditional method [12]. As we tried to follow the steps to implement the instance segmentation, we got several problems (See Fig. 6).

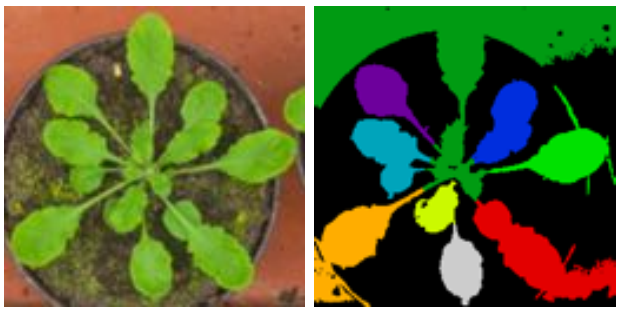

(a) *Ara2012* Original image (b) *Ara2012* Segment image

Fig. 6: Watershed segmentation result of *Ara2012*

The first was that the given data set *Plant* from *cvppp* does not have a consistent image shape. The image shape in *ara2012* and *ara2013* is around 200 $\sim$ 500, but the shape in *tobacco* is very different, around 2000. This caused the results of the segmentation to not be as expected. Because the plant area in *ara*, is much smaller than the plant area in *tobacco*, it is not suitable for the *watershed* approach for a good instance segmentation. Moreover, the images in all three datasets include complex background, like high light metal, dirt and a lot of moss. Those effect the segmentation result and cause the *watershed* model not to work as expect. This lead us to explore further the *watershed* model.

2) U-net model: After previous implementation and examination we found that the performance of the *watershed* is not good as we expected and this lead us to find out the reason and how to improve the performance of the *watershed* model. In a paper about plant instance-segmentation, which was also based on *cvppp* data set [13], it introduces using *watershed* to segment multi-instance, but before the segmentation, the author did a pre-processing of the plant image using neural network model to segment the plant itself and the background, which can highly improve the performance of the *watershed* model and reduce the effect from the background metal, dirt and moss. The best way segment the plant itself from the background is implementing a mask which only included the plant pixels. Due to the complexity of the image's background, traditional method cannot handle such a segmentation. By literature review we have a basic understanding of the *U-net* which is a powerful neural network model for medical image classification. [5]. Consider the segmentation mask itself is a binary image, which is suitable for *U-net* handle this classification task. *U-net* is a supervised learning method that does pixel level binary classification between the plant and background. After the model finished training, we use it to generate the mask of the plant image. Later we used the mask to remove the background pixel from the image and left the plant itself for further processing.(Fig. 8)

After the background segmentation, we use the new generated plant image for the *watershed* model to segment each leaf on the image, and based on previous experiment III-C1, using the best parameter of the *watershed* model to segment the plant itself. When comparing the performance of this approach versus using a raw image directly to *watershed* we see a considerable improvement on the leaf segmentation.

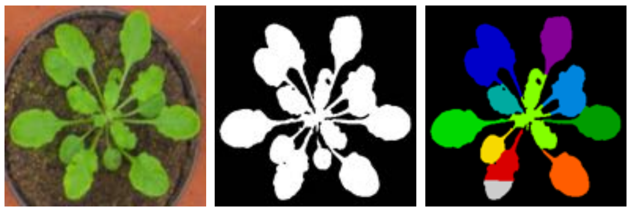

(a) Original raw image (b) U-net predict mask (c) Watershed result

Fig. 7: U-net for plant segmentation, Watershed for leaf segmentation

3) DeepColoring(U-net): From the *watershed* segmentation result (Fig. 7) we still can see some of the leaf did not get separated by the model, since the *U-net* already did a good job on plant and background segmentation, which lead us to think that the Deep neural network method might be a good choice for multi-instance segmentation.

Considering that the original *U-net* is for binary image classification, we can use a fixed number of labels and dynamically assign to different leaves, in practice using *U-net* as a multi-instance segmentation model [15]. For each training the model will training correspond label leaf segmentation, after

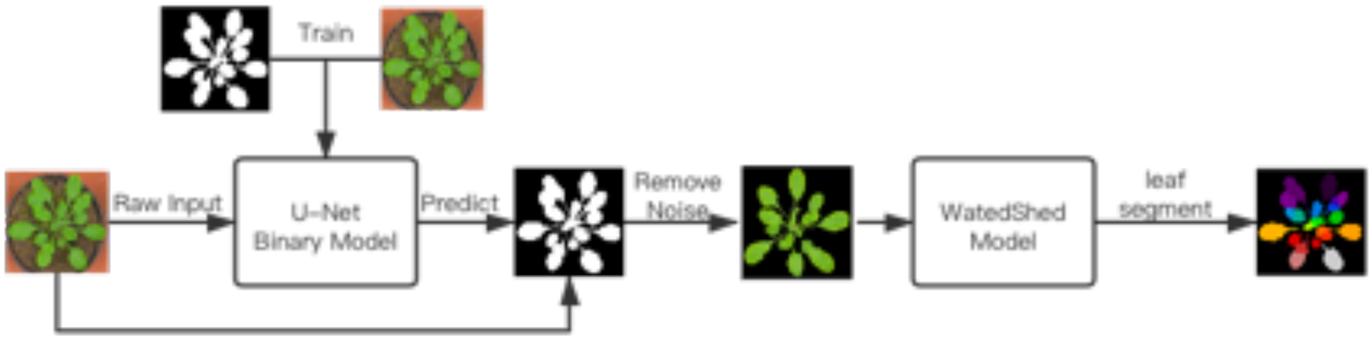
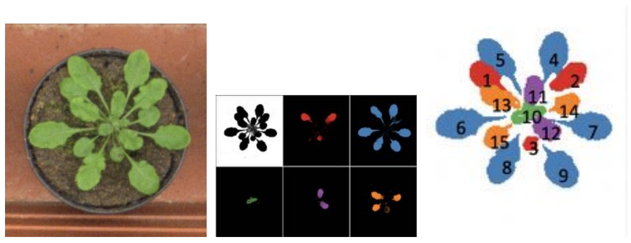


Fig. 8: U-net with watershed structure for leaf segmentation

the training process the model will combine different label prediction together, which lead the model to multi-instance segmentation, compare to large deep neural network, using *U-net* can be efficient, meanwhile because of the basic idea of *U-net* is for binary prediction, the precision cannot as much as those large deep neural network.

In order to figure out the actual performance of the *U-net* we try to implement it and have a comparison with traditional model. Because of the author provide a basic framework with this idea on GitHub [1], we just directly implement with slightly modification to fit the *cvppp* data set, then use it to evaluate the performance between the neural network and transitional method. (Fig. 9)



(a) Original raw image (b) U-net predict mask (c) Watershed result

Fig. 9: DeepColoring process procedure and the result

4) UPGen(Mask R-CNN [19]): After we implement the *DeepColoring*, we found deep neural network have better performance on multi-instance segmentation, and we want to explore the limitation of the deep neural network, thus we found using Mask R-CNN as the instancesegmentation model. [20]. It consists of a feature extractor and a Region Proposal Network (RPN). The heads of the network generate bounding box classification, bounding box regression and an object mask from each proposed region. It utilizes ResNet101 backbone with a feature pyramid network for Mask-RCNN [19]. 256 regions of interest per image were used for training. Models were trained using stochastic gradient descent optimiser with a learning rate of 0.001 and momentum and weight decay terms of 0.9 and 0.001 respectively. The batch size was fixed to four. The UPGen [20] utilizes different augmentation techniques. Namely: vertical flips; horizontal flips; cropping or padding the image by up to 25%; applying Gaussian blurring; random scaling between 80% and 120%, random rotation, translation and shearing; per channel image brightness changes and, lastly, random changes to hue and saturation between 20 and 20.

[1]https://github.com/kulikovv/DeepColoring

IV. Experimental Setup

Brief introduction to methods.

A. Object Detection

1) Traditional methods: The training of traditional method was done using GPU from Google Colab and the main Python packages used were Numpy, CV2.

2) YOLOV3: The YOLOV3 model taken from [21].The YOLOV3 is very memory intensive. It was trained on a server machine which has 40G RAM with the GPU TITAN X and CPU is Xeon E5. In order to using YOLOV3 model,the target object need to be marked in the training image. There is a tool called labelImg which is specially designed for YOLOV3.Using this tool to mark the target objects in the image could automatically generate .xml file which contain the classes labels and the location of the target objects in the image. Using voc2yolo3.py is to generate the train.txt which contain the name of training images and the .xml files. Using voc_annotaiton.py will finally generate the training set 2007_train.txt which has the information about the path of original training data and the target objects locations. Transfer Learning method could be used in this part because we need to marked lots of target plants in the images.The weights which were trained by YOLOV3 before and able to recognize plants is a ideal choice.

3) evaluation method: Average Precision(AP)

Precision = TP/(TP+FP)
Recall = TP/(TP+FN)
In the target detection algorithm, there is a very important concept is confidence. If the confidence is set high, the

predicted results and the actual situation are very consistent. If the confidence level is low, there will be a lot of false detection.Assuming that there are a total of 4 positive samples in a picture, there are 10 prediction results for this picture by target detection, of which 4 are actually positive samples and 6 are actually negative samples. The corresponding confidence is in Table I. If we set the acceptable confidence level to 0.95, then the target detection algorithm will regard the sample with sequence number 1 as a positive sample, and the others are negative samples. At this time TP = 1, FP = 0, FN = 3.
Precision = TP/(TP+FP)=1
Recall = TP/(TP+FN)=$\frac{1}{4}$
At this time, the precision is very high, but in fact, we only detect one positive sample, and three are not detected, so it is not appropriate to use precision only. If we set the acceptable confidence level to 0.35, at this time TP = 4, FP = 3, FN = 0.
Precision = TP/(TP+FP)=$\frac{4}{7}$
Recall = TP/(TP+FN)=1
In this case, recall is very high, but in fact, among the samples that the target detection algorithm considers to be positive samples, 4 samples are indeed positive samples, but 3 are negative samples. There is a very serious false detection, so it is not appropriate to use recall only.The combination of the two is the correct method of evaluation.AP actually combinations of Precision and Recall.When we take different confidence levels, we can get different Precision and different Recall. When we get dense enough confidence, we can get a lot of Precision and Recall.Precision and Recall can draw a line on the picture, and the area under this line is the AP value of a certain class. [22]

B. Semantic Segmentation

The data available for this task is the one on the "Tray" folder from the Plant Phenotyping Dataset. This dataset consists of two different sets corresponding to trays of many Arabidopsis or Tobacco plants and their respective ground truths that will be used for the model training. The UNet model implementation was taken from [23], and after performing some code changes, it was trained Google Colab on 38 out of the 43 images available. The training was done using TPU from Google Colab and the main Python packages used were Numpy, CV2, Keras and TensorFlow. Once the model was trained, the resulting weights were stored to be used later in the prediction stage. To test the model, the 5 images set aside from the training stage were passed on to the model loaded with the weights obtained from the training stage. The evaluation methods used for this task were the ones under specification corresponding to Dice Similarity Coefficient (DSC) and Intersection over Union (IoU). DSC is defined as:

		Actual Class	
		P	N
Predicted Class	P	TP	FP
	N	FN	TN

TABLE I: confidence table

index	label	confidence
1	positive	0.99
2	positive	0.9
3	positive	0.89
4	positive	0.85
5	negative	0.65
6	negative	0.55
7	negative	0.45
8	negative	0.35
9	negative	0.30
10	negative	0.25

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|} \tag{1}$$

While IoU is defined as:

$$IoU = \frac{|X| \cap |Y|}{|X| \cup |Y|} \tag{2}$$

C. Multi-Instance Segmentation

To evaluate the performance of different model, all the model will use *rgb* image as the training data set and using *label* image as the output image, for the traditional method such as *watershed* is not required the output image, thus previous set up is majority for deep learning model. All the data set was randomly split and divide into train/test set with the percentage of test set is 30%. Consider the deep network requirement, all the image including RGB image and label image will resize as 256×256 to format the input of the model.

For the evaluation we go through the literature review and found that using normal evaluation method on *cvppp* data set is not efficient, because the leaf position and the segmentation label problem, using normal evaluation method need more post processing to calculate the accuracy of the model. Instead of using normal evaluation method, using Symmetric Best Dice [14] score is better for performance evaluation in this data set, suppose L^{ar} is the prediction of the model, and L^{gt} is the actual value of the image, as the definition of the Symmetric Best Dice (SBD) as follow:

$$SBD(L^{ar}, L^{gt}) = min\{BD(L^{ar}, L^{gt}), BD(L^{gt}, L^{ar})\}$$
$$BD(L^{ar}, L^{gt}) = \frac{1}{M}\sum_{i=1}^{M} max_{1 \le j \le N} \frac{2|L_i^a \wedge L_j^b|}{|L_i^a| + |L_j^b|} \tag{3}$$

Where M is the numbers of L_i^a and N is the numbers of L_j^b.

Moreover, to increase the efficiency of calculate SBD, before the evaluation if possible the evaluate image and the

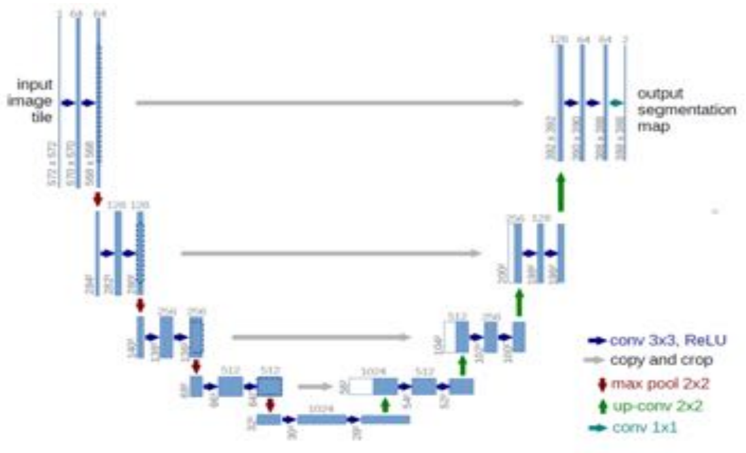


Fig. 10: UNet Architecture representation.

label image will all convert to gray scale image for higher compute rate.

V. Results and Discussion

Brief introduction to Results and Discussion.

A. Object Detection

1) traditional method: The traditional method could generate result show as Fig.11 However,some results are performed

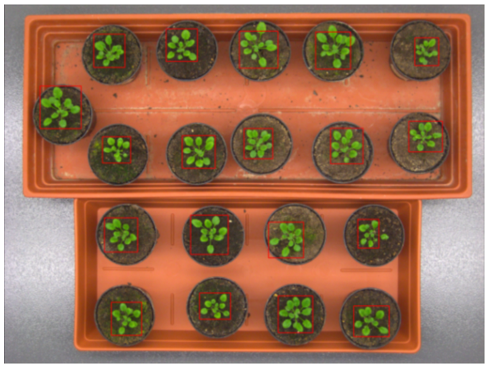


Fig. 11

well by using traditional method because the plants in some images are very close or some leaves are separate.

2) YOLOV3: The predict result of YOLOV3 show in Fig.12 And as Fig.13 the Average Precision is 99.53

B. Semantic Segmentation

Given the limited number of images available we were only able to test the UNet with 5 images, which were not used for the training stage. For each of these images we obtained a segmented outcome which we can evaluate visually and quantitatively by the DSC and IoU methods. The visual output for the semantic segmentation task can be seen in Figure 14. Intuitively we can see that the results are very promising, showing a high similarity to the ground truth which the model is trying to achieve. To evaluate the results in a

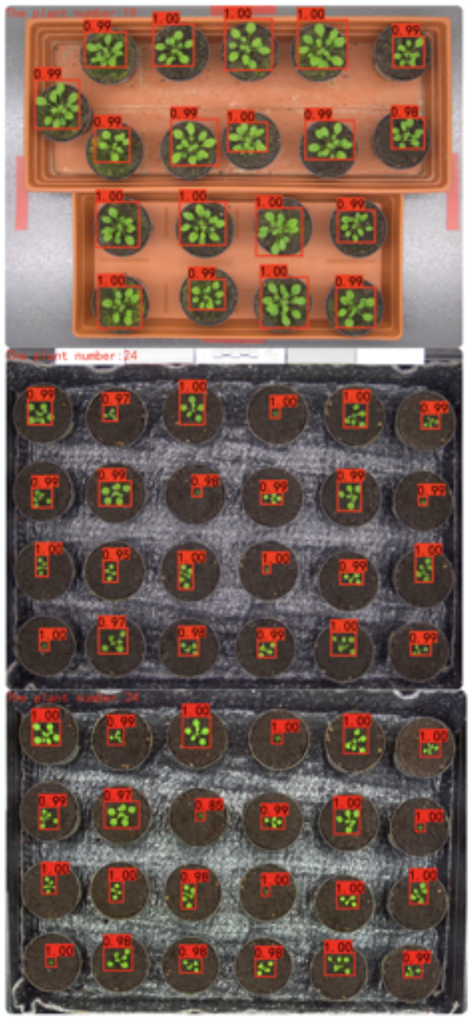


Fig. 12: The predict result of YOLOV3

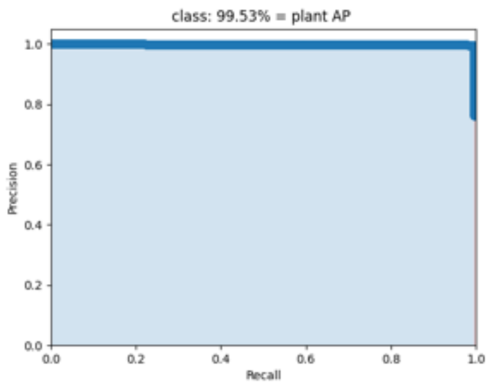


Fig. 13: Average Precision

Image	DSC	IoU
ara2012 tray06	92.87%	86.68%
ara2012 tray16	89.88%	81.61%
ara2013 tray07	90.19%	82.14%
ara2013 tray17	92.20%	85.52%
ara2013 tray27	91.35%	84.07%
Average Performance	91.30%	84.01%

TABLE II: Results obtained by UNet model when tested on 5 images that were set aside for training. Performance is measured by the Dice Similarity Coefficient (DSC) and Intersection over Union (IoU).

more precise way we use the DSC and IoU methods. The performance evaluation for the images presented in Figure 14 are displayed in table III. The methodology chosen to approach this complex problem proves to be appropriate and the results are satisfactory. The UNet, as promised by their authors, is indeed capable of learning from a very limited training dataset and the segmentations obtained are highly accurate, reaching a 92% of similarity according to the DSC metric. But in the method, the Relu active function is used during the downsampling part. Considering Relu active funcion, its gradient has no relationship with variable. If there is an error in prediction, the changes made by back progagation is constant and not depending on the change in input data. In order to avoid the situation, tanh active funcion is used. After using tanh, better accuracy can be reached.

C. Multi-Instance Segmentation

Base on previous experimental setup, for each model performance we will using symmetric best dice as the evaluate method. For comparison of the model, we will use *watershed* without pre-processing as the baseline model, following using *U-net* as background segmentation, using *U-net* as multi-instance segmentation and implement *Mask R-CNN* as the segmentation model.

Image	DSC	IoU
ara2012 tray06	93.20%	87.26%
ara2012 tray16	91.19%	83.81%
ara2013 tray07	90.68%	82.95%
ara2013 tray17	93.19%	87.24%
ara2013 tray27	93.29%	87.43%
Average Performance	92.31%	85.74%

TABLE III: Results obtained by UNet model with tanh active funcion when tested on 5 images that were set aside for training. Performance is measured by the Dice Similarity Coefficient (DSC) and Intersection over Union (IoU).

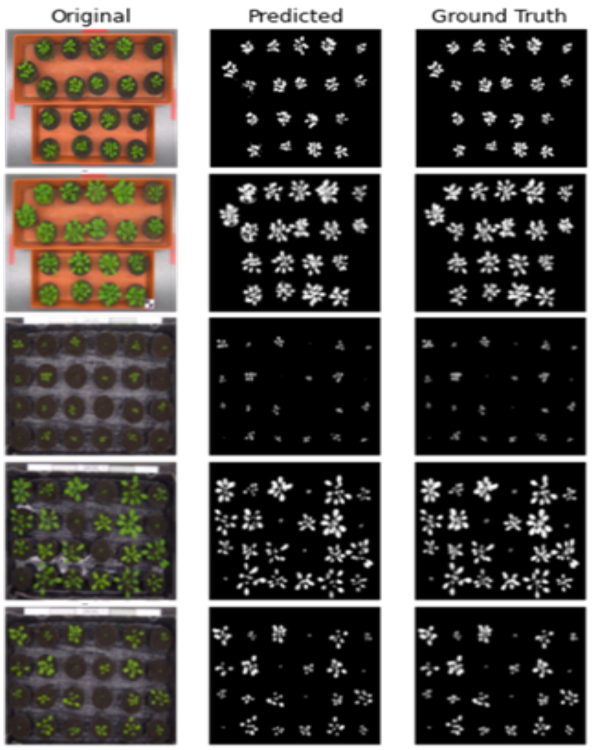


Fig. 14: Output from UNet model implemented for Semantic Segmentation. Leftmost image shows the raw input, center image shows the prediction made by our model and rightmost image shows the ground truth.

Based on the result of these four models, as *U-net* did help *watershed* improve the performance from leaf segmentation, but compare to the DeepColoring and UPGen, deep neural network have huge advantage in leaf segmentation, although the model UPGen originally have up to 88 SBD score in the paper. [20] During this evaluation it already the stable and better model on leaf segmentation. As the performance of the *watershed*, we think we still can push it up, consider the performance in *Wageningen* of SBD are around 89% $\sim$ 55% [13], thus it would be the pre-processing is not as perfect as we expect, moreover, *watershed* model itself might not the best

	Ara2012	Ara2013	Tobacco
Raw+Watershed	0.0403	0.0313	0.0176
Unet+Watershed	0.0462	0.0292	0.051
DeepColoring	0.84	0.67	0
UPGen	0.77		0.74

TABLE IV: Evaluation result of four models

parameter setting inside, consider we can use more different method such as *superPixel* to generate perfect leaf centroid of the plant, we still have a lot to do on *watershed* model.

VI. Conclusion

In this paper we address three problems in plant phenotyping, namely, plant detection, plant segmentation and individual leaf segmentation. In image detection and segmentation, the YOLOV3 and U-Net outperformed Image binary classification and Edge Detection techniques. Similarly while performing individual leaf segmentation, UPGen and DeepColoring methods outperformed traditional Watershed algorithm. The overall performance shows that deep neural networks have a better performance than the traditional methods.

Compared to traditional method such as *watershed* and *superPixel*, those deep neural model have huge advantage on image processing, but it does not mean traditional method is outdated, in some special task using traditional methods might be more efficient and more reliable, using deep neural network usually cost a large numbers of the data set with preprocessing and high computing power, consider the cost of the deep neural network, if we are implementing small task and that is not worth to use deep neural network, although it have high performance but it also required a lot of cost.

VII. Contribution of Group Members

- Ziyang Liang: Responsible for Task 3, watershed/u-net model implementation/training. Wrote Literature review, method, experiment and result of Multi instance segmentation and Conclusion of the report.
- Anant Krishna Mahale: Responsible for Task 3. Trained UPGen model with different combinations of parameters. Compared the results with deepcoloring model. Prepared the final presentation slides. Wrote Abstract, Literature review, Experimental set-up and Conclusion of the report.
- Sebastian Novoa: Responsible for Task 2. Research into different approaches for Semantic Segmentation, Training of UNet. Wrote Introduction and Semantic Segmentation part of the report. Proofreading the report.
- Yuntao Zhou:Responsible for Task 1. Using traditional method and YOLOV3 to do object detection. Experimental setup, Results and Discussions .
- Hang Zhu: Responsible for Task 2. Implementing many experiments on Unet algorithm to select appropriate active function, writing literature review of Semantic Segmentation part.

Acknowledgement

We would like to thank our professors Prof. Arcot Sowmya, Dr. Wafa Johal and Dr. Gelareh Mohammadi for giving us an opportunity to work on this project.
We also acknowledge and thank our tutor and mentor Hongfei Ao for guiding us in every part of the project.

References

[1] M. Minervini, A. Fischbach, H. Scharr, and S. Tsaftaris, "Plant phenotyping datasets," 2015. [Online]. Available: http://www.plant-phenotyping.org/datasets

[2] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–, 11 2004.

[3] C. Liu, J. Yang, and H. Huang, "P-surf: A robust local image descriptor," *J. Inf. Sci. Eng.*, vol. 27, pp. 2001–2015, 11 2011.

[4] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157 vol.2.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, ser. Lecture Notes in Computer Science, vol. 9351. Cham: Springer International Publishing, 2015, pp. 234–241.

[6] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," 2014.

[7] B. Romera-Paredes and P. H. S. Torr, "Recurrent instance segmentation," 2016.

[8] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," 2016.

[9] M. Ren and R. S. Zemel, "End-to-end instance segmentation and counting with recurrent attention," *CoRR*, vol. abs/1605.09410, 2016. [Online]. Available: http://arxiv.org/abs/1605.09410

[10] A. Dobrescu, M. V. Giuffrida, and S. A. Tsaftaris, "Leveraging multiple datasets for deep leaf counting," *CoRR*, vol. abs/1709.01472, 2017. [Online]. Available: http://arxiv.org/abs/1709.01472

[11] V. Kulikov, V. Yurchenko, and V. S. Lempitsky, "Instance segmentation by deep coloring," *CoRR*, vol. abs/1807.10007, 2018. [Online]. Available: http://arxiv.org/abs/1807.10007

[12] M. Bai and R. Urtasun, "Deep watershed transform for instance segmentation," 2016.

[13] H. Scharr, M. Minervini, A. P. French, C. Klukas, D. M. Kramer, X. Liu, I. Luengo, J.-M. Pape, G. Polder, D. Vukadinovic, X. Yin, and S. A. Tsaftaris, "Leaf segmentation in plant phenotyping: a collation study," *Machine vision and applications*, vol. 27, no. 4, pp. 585–606, 2015.

[14] M. Minervini, A. Fischbach, H. Scharr, and S. A. Tsaftaris, "Finely-grained annotated datasets for image-based plant phenotyping," *Pattern Recognition Letters*, pp. –, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865515003645

[15] V. Kulikov, V. Yurchenko, and V. Lempitsky, "Instance segmentation by deep coloring," *arXiv preprint arXiv:1807.10007*, 2018.

[16] D. Ward and P. Moghadam, "Scalable learning for bridging the species gap in image-based plant phenotyping," 2020.

[17] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[18] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *CoRR*, vol. abs/1506.02640, 2015. [Online]. Available: http://arxiv.org/abs/1506.02640

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[20] D. Ward and P. Moghadam, "Scalable learning for bridging the species gap in image-based plant phenotyping," *Computer vision and image understanding*, vol. 197-198, p. 103009, 2020.

[21] bubbliiiing, "The realization of you only look once target detection model in pytorch," 2020. [Online]. Available: https://github.com/bubbliiiing/yolo3-pytorch

[22] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2009.

[23] H. K. P. Nikhil Vijay S, “Unet implementation,” 2015. [Online]. Available: https://github.com/LeadingIndiaAI/Leaf-Segmentation-Challenge-Using-UNET