CS679 Project Report
Leaf Segmentation and Counting in Smart Agriculture Applications using Deep Learning

Aman Kumar (*a332kuma*) [*] **Paul Ma (*p32ma*)** [*] **Peter Wu (*mqwu*)** [*]

Abstract

Recent advances in the field of agriculture are seeing increasing applications for deep learning techniques and automation, particularly in the area of plant phenotyping, where traditional techniques may be more destructive or require extensive field work. Coupled with growing research in techniques such as few-shot and transfer learning, there are many avenues for novel developments in this area. We explore the application of full and zero-shot learning coupled with transfer learning in leaf phenotyping - more specifically for the tasks of leaf segmentation and counting - on two different plant species: the Arabidopsis thaliana rosette and Tobacco (Nicotiana tabacum) plants. We explore two models: modified ResNet for regression, and Faster R-CNN for object detection. To perform zero-shot learning, we used the SAM model to predict masks for every object in an image, and send them to the ResNet Regressor as input. Our results show that Faster R-CNN outperforms ResNet Regressor with its robust detection modules, allowing additional learning opportunities, particularly on the bounding boxes of leaves.

1. Introduction

Plant phenotyping is a quantitative description of a plant's various properties. These properties help experts understand the expressed traits of a plant, leading to the ability to better control breeding and mutations, or possibly accelerate the growth and improve the quality of cultivated plants (Hati & Ranjan Singh, 2020), which is especially critical as global food and biofuel needs continue to rise with the growing global population. One important phenotype is leaf count, which, like many phenotypes, is costly and time-consuming to measure, and frequently requires expert involvement in the field (Farjon et al., 2021). Nowadays, with image sensor technologies widely applied in the agricultural field, image-based phenotyping has seen increasing interest, particularly as it has strong potential to address many of these issues - as such, leaf segmentation and counting is our primary goal for this project.

Typically, sensors periodically generate enormous amount of data, and analysis of this data, specifically images, is particularly suited to deep learning algorithms. In the case that such large quantities of data is not available however, the commonly used deep learning models are not as effective and techniques such as few-shot learning have the potential to provide better results (Ranjan et al., 2021; Chen et al., 2019). Gaps in the literature regarding few-shot learning applied to plant phenotyping (Yang et al., 2022) means that there is potential for exploration of this technique here.

Both leaf segmentation and counting are some of the most important tasks in phenotyping. The number of leaves is an important metric related to a plant's growth and provides a wealth of information about the plant's growth stages, such as flowering time and yield potential. Leaf segmentation and counting remains particularly challenging because the shape of a leaf lacks uniformity and distinguishable features from other plant segments (Dobrescu et al., 2017).

In this work, we explore two types of approaches: deep learning regression and deep learning segmentation/detection. We evaluate these approaches in the context of the Computer Vision Problems in Plant Phenotyping (CVPPP), using its Leaf Counting Challenge (LCC) dataset. Past works that tackle the same challenge show that regression networks work well for counting leaves. Giuffrida et al. 2015 developed a Support Vector Regression (SVR) model, with input being the extracted patches of the log-polar transformed images. During the testing phase, the model can predict the correct number of leaves for 25% of the images, and 57% of the cases have errors at most ± 1 from the ground truth. Image segmentation methods are also candidates for solving the challenge, but very few utilize deep learning models. We wish to expand on this front by experimenting with segmentation/detection neural networks. We choose ResNet for regression and Faster R-CNN model for object

[*]University of Waterloo, Waterloo ON, Canada.

detection, and we compare and contrast both methods. In addition, We utilize the Segment Anything Model (SAM) to automatically generate masks for all objects in images, in an attempt to simplify the features space, and feed the masks to the ResNet Regressor to see if there is any improvement in the performance.

2. Related Work

Developments in deep learning and other techniques have led to a great deal of growing interest and activity in the field of smart agriculture, particularly in applications concerning plant phenotyping. Implementations of machine learning techniques for leaf counting is relatively varied, and is still ongoing rigorous study, though much of the literature at time of writing does not involve the use of few-shot learning. Farjon et al. 2021 propose two methods: Multiple-Scale Regression, which performs leaf counting via direct regression combined with fusing images of different resolutions, and Detection with Regression Network, which combines the tasks of leaf detection with count regression in sequence, in an end-to-end architecture. Examples of direct regression approaches include Dobrescu et al. 2017 and Lu et al. 2017.

Similar direct regression approaches are taken by Dobrescu et al. 2017, whereby regression was performed using a modified ResNet-50 where the classification layers of the architecture were replaced with a series of regression layers with a single output neuron for the leaf count estimates, and by Lu et al. 2017 which took the same architectural approach but with extensions using multiple scale representation.

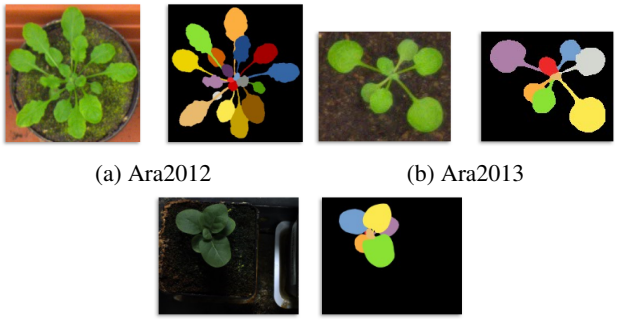
Other strategies for leaf counting take a segmentation-based approach. Hati & Ranjan Singh 2020 combine the task of leaf segmentation and leaf counting without relying on detection of leaf centers or supervised training with leaf segmentation labels. They perform segmentation by converting the training images into Hue, Saturation and Value (HSV) colour space and applying a threshold value filter to the result, after which preprocessing steps are applied prior to feeding into an Alexnet based architecture with Softmax activation for the output, which is then passed into a regression network for counting. Ren & Zemel 2016 and Romera-Paredes & Torr 2015 forego transfer learning and employ recurrent neural networks (RNN) for their leaf segmentation and counting, with the latter taking additional advantage of convolutional LSTM.

3. Methodology

3.1. Dataset Description

The primary dataset for this project is sourced from the Leaf Phenotyping dataset[1] (Minervini et al., 2016) (CVPPP-2014) which was used in the Leaf Segmentation Challenge (LSC) of the Computer Vision Problems in Plant Phenotyping workshop in 2014.

The CVPPP-2014 dataset (see Figure 1) is split into a collection of four benchmark datasets in the context of plant phenotyping (three sets of images for the Arabidopsis thaliana rosette, Ara2012, Ara2013-Canon and Ara2013-RPi, totalling 150 (120 annotated), 4186 (165 annotated) and 1951 (165 annotated) total images, respectively; one set of images for Tobacco plants (62 annotated)), which were collected and labelled for the purpose of application in plant phenotyping tasks which include classification, and leaf image segmentation and counting tasks. Annotations include bounding boxes, boundaries, centers and segmentation ground truths for all leaves.



(a) Ara2012 (b) Ara2013

(c) Tobacco

Figure 1. Samples of Arabidopsis thaliana rosette and tabacco plants from CVPPP-2014 dataset with corresponding ground truth segmentation.

3.2. Regression with ResNet

Residual Network, or ResNet (He et al., 2016), is one of the most popular CNNs in computer vision. For this work, we used a Resnet50 (50 residual layers) model to train a regressor for counting the leaves directly. It has been shown that ResNet can indeed be adapted to be a regressor (Lathuilière et al., 2020). We replaced the last classifier layer with two fully-connected layers with $(2048, 1024)$ and $(1024, 512)$ dimensions each, with an additional final fully connected layer with only one output node for predicting the number of leaves in an image. To prevent information loss, we decided to employ Mish (Misra, 2019) activation function instead of the popular Rectified Linear Unit (ReLU) (Hahnloser et al., 2000) function, which is shown to have superior performance over ReLU and LeakyReLU (Maas et al., 2013) in object detection networks. We want to try this activation in our regression network to see if there is any performance bump. The Mish activation function is described as follows:

$$Mish(x) = x \tanh(softplus(x)) \quad (1)$$

[1] https://www.plant-phenotyping.org/datasets-download

where

$$softplus(x) = \ln(1 + e^x) \tag{2}$$

We used L1 loss instead of cross-entropy loss, as the problem is now formulated as a regression problem.

3.3. Object Detection

Object detection is a vision task that localizes and classifies objects in an image. Unlike semantic segmentation, which can only locate one instance of each class of objects, object detection can find multiple instances of the same class. This is done by building bounding boxes that surround the objects, providing their locations in an image, and then classifying what is in the bounding boxes. Leaf segmentation can be formulated as an object detection problem, where each image shows a plant, which consists of multiple instances of a leaf. From there, one can simply count the number of bounding boxes that envelope leaf objects found in the image, effectively counting the number of leaves.

One of the first object detection methods is Selective Search, a technique combining both exhaustive search and segmentation using pixel-level features (Uijlings et al., 2013). The algorithm is designed to capture objects of all scales, addressing the fact that objects belonging to the same class may have different sizes and orientations. It is computationally fast, and has potential to reach towards real-time object detection. This algorithm inspired the region-based convolutional neural network, or R-CNN. It relies on Selective Search to find 2000 possible regions of objects per image, thereby building bounding boxes. The CNN architecture then extracts features from the results of Selective Search, and classifies whether the object inside a bounding box is the background, or one of the ground truth set of objects (Girshick et al., 2013). It can be summarized as a two-staged object detection method, the first consisting of combining both region proposals, which generates bounding boxes, and the second being the classification of regions.

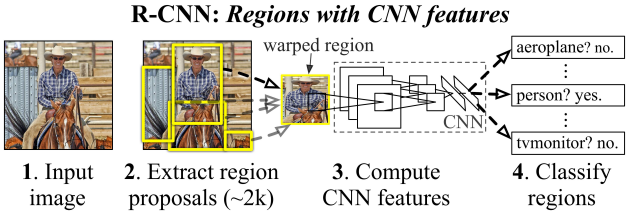


Figure 2. R-CNN architecture.

A huge limitation is that classifying 2000 regions per image is too computationally expensive, and that select search presents no learning opportunities as it is a hard-coded algorithm, unlike typical machine learning models. The same authors proposed Fast R-CNN (Girshick, 2015) to address these problems. Unlike R-CNN, it first generates feature maps using a CNN backbone, then finds regions from the feature maps instead of directly from images, decreasing the computation cost.

In an effort to gradually improve region proposal quality, another object detection model called Faster R-CNN (Ren et al., 2015) was proposed. The authors replaced selective search with a so-called Region-Based Network (RPN) to generate region proposals. RPN is a convolutional neural network sliding through the convolutional image features outputted by a CNN backbone, producing region proposals. It speeds up inferences by using shared weights for both RPN and CNN backbones. RPN consists of two components: a classifier and a regressor. The classifier determines the probability of a proposal having the target object, and the regressor regresses the coordinates of the proposals. Each component has its own loss function:

$$L(\boldsymbol{p}, \boldsymbol{t}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, t_i) + \lambda \frac{1}{N_{reg}} \sum_i p_i L_{reg}(p_i, t_i) \tag{3}$$

where L_{cls} is a classification loss function and L_{reg} is a regression loss function. In our implementation, we used $L_{cls} = CrossEntropy$ and $L_{reg} = L1Loss$. For this work, our model of interest will be Faster R-CNN.

3.4. SAM + ResNet Regressor

The Segment Anything Model (SAM) is a promptable, zero-shot segmentation model which was recently proposed by Kirillov et al. 2023. During the object detection stage, each object is localized by creating a rectangular bounding box around it. For segmentation, the aim is to classify each pixel in the image to its corresponding class, resulting in precise contours around the objects and localization.

SAM is a foundation model for image segmentation, which means that it is promptable, and is pre-trained on a broad dataset such that it has acquired strong generalization capabilities that enables it for impressive zero-shot transfer. To achieve such strong generalization capabilities, it is essential to have a dataset containing a large and diverse set of masks. Interestingly, the required dataset is also created by SAM. Since such a dataset didn't exist, the researchers created a "data engine" to create the dataset and train the model at the same time. The annotators used SAM interactively to annotate the images and, in turn, annotated images were utilized to train the model. This process was performed over many iterations to improve the dataset and the model. The resulting dataset was named SA-1B and contains more than 1 billion masks from 11 million images, with SAM itself being trained on this dataset.

The aim of the researchers was to pre-train a foundation model that can solve diverse downstream segmentation tasks

via prompt engineering. Therefore, to train the model, the researchers designed a promptable segmentation task, in which, given any prompt, the model has to return a valid segmentation mask. The researchers considered two sets of prompts: sparse (points, boxes, text) and dense (masks). Given this promptable segmentation task, the pre-training algorithm simply simulated a sequence of prompts for each training sample and compared the model's mask prediction against the ground truth.

The model architecture of SAM consists of three components: image encoder, prompt encoder and mask decoder. For the image encoder, the researchers used a minimally adapted masked autoencoder (MAE) (He et al., 2022) pre-trained Vision Transformer (ViT) (Dosovitskiy et al., 2020) to create image embeddings. For prompt encoding, different types of prompts were encoded differently. The points and boxes were represented by positional encodings (Tancik et al., 2020). The text was encoded with the off-the-shelf text encoder from the CLIP model (Radford et al., 2021). Dense prompts were encoded by performing convolutions to create embeddings, which are summed up element-wise with the image embedding. For the mask decoder, the researchers adapted a Transformer decoder block (Vaswani et al., 2017) followed by a dynamic mask prediction head. Finally, to train the model in a supervised manner, a linear combination of focal loss (Lin et al., 2017) and dice loss (Milletari et al., 2016) was used as a loss function (Carion et al., 2020).

In this work, we propose a novel methodology to perform leaf counting using SAM. Since SAM can perform zero-shot segmentation well, the images will be first segmented using SAM. The idea is that we want to provide a model with a better representation of the data; we can look at segmentation as an intelligent preprocessing step that will create a simpler and cleaner representation of the raw, noisy image. In a way, the resultant segmented image is like an interpretable encoding or a better representation of the original image. After passing images through SAM, the segmented images will be used to train a ResNet Regressor, as described in Section 3.2, to perform leaf counting.

3.5. LIME Explanations

Due to limitations in our ability to discern how the ResNet Regressor model described in Section 3.2 generates count predictions, we turn to the Local Interpretable Model-Agnostic Explanations (LIME) explainability method proposed by (Ribeiro et al., 2016) to attempt to gain some relevant insight. LIME is a model agnostic local post-hoc explainability method that provides explanations for predictions made by machine learning models, and which can be applied in the areas of either tabular or image data.

Regarding images, LIME provides explanations for a particular image instance in the form of highlighted regions in the image that are deemed to have contributed most to the model's decision. Broadly, this is achieved in a four-step process. First, the image is broken up into what are referred to as superpixels, which consist of various patches of the image made up of contiguous pixels in a region with similar characteristics (i.e., brightness and color). Second, a sample of new perturbed versions of the original image is generated by randomly removing these superpixels, replacing them with solid colors or random noise. Third, predictions are produced on these perturbed images using the chosen model, to which LIME fits a linear model whose coefficients can be interpreted as the relative importance of each superpixel in the original image in making the final prediction. Finally, a visual explanation of the chosen model's decision is generated by producing a heatmap based on the linear model coefficients and overlaying it on the original image.

For our experimental setup, the number of perturbed image samples per image explanation is chosen to be 10,000, with visual explanations showing the top 10 superpixels whose coefficients in the corresponding LIME linear model were of the highest magnitude (green for positive, red for negative).

4. Experiments

4.1. Training setup

Before training, we first perform data augmentations on the training data to artificially increase the number of samples and variations of the samples. Augmentations include same-size crops, random flips, random rotation, random color and brightness adjustments, and random blurs.

In our experiments, all the selected models are pre-trained, with ResNet Regresssor pre-trained on the ImageNet dataset (Deng et al., 2009), Faster R-CNN pre-trained on the COCO dataset (Lin et al., 2014), with fine-tuning performed on our leaf dataset. We trained the ResNet Regressor and Faster R-CNN models on an NVIDIA RTX 3080 GPU for 100 epochs, with a total training time of about 40 minutes for ResNet Regressor, and about 70 minutes for Faster R-CNN. Both models are trained with the $AdamW$ optimizer, with weight decay set to 0.005, and a cosine annealing learning rate scheduler, with initial learning rate set to 0.0001. The setup is summarized in table 1

4.2. Evaluation

To validate the performance of our selected models, we use L1 Error (L1), Mean-Squared Error (MSE) for regression, and mean Average Precision for segmentation.

	Epoch	LR	WD	Scheduler	Optimizer	Loss
ResNet Regressor	100	0.0001	0.005	CosineAnnealing	AdamW	L1
Faster R-CNN	100	0.0001	0.005	CosineAnnealing	AdamW	L1 + CrossEntropy

Table 1. Training Setup.

4.2.1. Regression Measures

To measure the performance of the ResNet Regressor, we compare the predicted leaf count with the ground truth leaf count. We look at both the L1 and MSE metrics:

$$L1(\boldsymbol{p}, \boldsymbol{t}) = \frac{1}{n}\sum_{i}^{n} |t_i - p_i| \tag{4}$$

$$MSE(\boldsymbol{p}, \boldsymbol{t}) = \frac{1}{n}\sum_{i}^{n} (t_i - p_i)^2 \tag{5}$$

4.2.2. Mean Average Precision

Mean Average Precision (mAP) is a popular metric used for the evaluation of object detection and segmentation models. The precision is calculated based on a confusion matrix generated for each class of object. The confusion matrix is a 2×2 grid with the horizontal axis being the actual labels, and the vertical axis being the predicted labels. It shows the number of True Positives (TP) and True Negatives (TN) where the prediction matches the actual label, as well as False Positives (FP) and False Negatives (FN) where the prediction and actual labels are mismatched (i.e., positive prediction and negative ground truth for FP, negative prediction and positive ground truth for FN). Positive and negative classifications refer to whether an object truly belongs to the class the confusion matrix is built for (Positive) or it belongs to some other class (Negative). The precision is calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

Thus, precision measures how well the model can find objects with the correct classifications for each class.

To select which bounding boxes we should calculate precision for, we use another measure called the Intersection-over-Union (IoU). It captures the overlap of predicted bounding box coordinates to the ground truth bounding box. Higher IoU indicates the predicted bounding box coordinates closely align with the ground truth bounding box coordinates:

$$IoU(B', B) = \frac{B' \cap B}{B' \cup B} \tag{7}$$

where B' is the predicted bounding box coordinates and B is the ground truth coordinates. We use IoU as a threshold to select bounding boxes. For example, we can choose to select bounding boxes with $0.5 < IoU < 0.95$, indicating that if the IoU of a bounding box is within this range, we select it as a candidate bounding box to perform classification. Then we calculate the precision for each class over all selected bounding boxes, and then compute the average:

$$mAP = \frac{1}{N}\sum_{i}^{N} AP_i \tag{8}$$

where N is the total number of classes, and AP is the average precision of class i. Since we only have one class in our problem, the calculation is simplified.

4.3. SAM experimental setup

To perform the segmentation using SAM, no preprocessing was applied except resizing all the images to 320×320. Given an input image, SAM generates a collection of several masks and their respective properties, including the area of the mask in pixels, the bounding box of the mask, a measure of the mask quality, etc, where each generated mask corresponds to a single segmented object/region. In essence, the whole segmented image is split into individual masks of different objects in the image; but for our purposes, a single, complete mask per image was required. Therefore, all the individual (sub)-masks were combined by superimposing them onto each other, to create a single mask. Furthermore, in an attempt to remove irrelevant regions from the mask, we experimented with dropping the two (sub)-masks with the largest areas. As seen in Figure 6c, the zero-shot segmentation performed by SAM is quite impressive. However, there are some cases where segmentation results are noisy and inaccurate (Figure 6f). These segmented images (masks) were then used as input images to train and evaluate a ResNet Regressor, as described in Section 4.1 and Section 4.2.

5. Results

5.1. Regression

By directly performing regression, with no segmentation masks or bounding boxes as part of the input, the ResNet Regressor can already achieve good results. Table 2 shows that the ResNet Regressor achieves an average L1 loss of 2.07, and MSE of 6.08. This means that the model mistakes the number of leaves within a margin of 2. In the case of SAM + ResNet Regressor, contrary to our expectations, performance did not improve compared to applying ResNet Regressor on raw images. It achieved an average L1 loss of

3.09 and MSE loss of 12.91, which is much higher (poorer) than solely applying ResNet Regressor. Even dropping the two largest masks did not improve performance, instead harming it significantly.

Regarding model explanations, LIME was applied to generate explanations for ResNetCount predictions on a large number of instances, of which four are presented here, as shown in Figure 8. Figure 8a shows the generated explanation for a rosette from the Ara2012 dataset, with an actual leaf count of 18 and a predicted leaf count of approximately 12.97, while Figure 8b shows the same for a rosette from the Ara2013 dataset, with an actual leaf count of 8 and a predicted leaf count of approximately 10.43. Figures 8c and 8d show generated explanations for two selected tobacco plants, with actual leaf counts of 6 and 8 and predicted leaf counts of approximately 6.64 and 10.44, respectively.

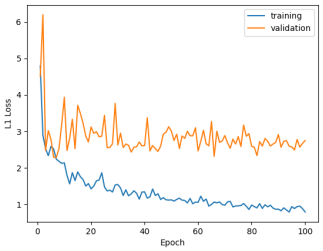


Figure 3. Loss over Epoch for ResNet Regressor.

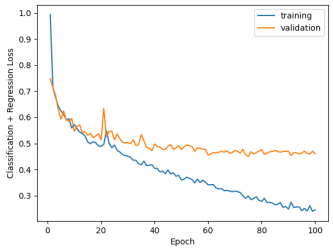
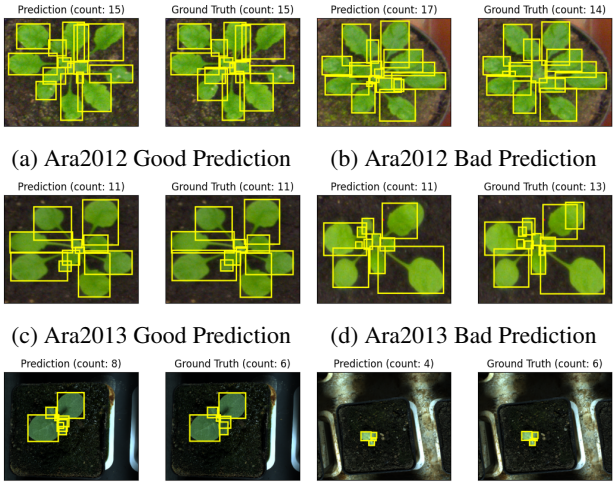


Figure 4. Loss over Epoch for Faster R-CNN model.

5.2. Object Detection

We show the loss progression of Faster R-CNN in Figure 4 and inference results in Figure 5. The model with the least loss value at an epoch was chosen to perform inference. We selected bounding boxes with $IoU > 0.7$ as the final predictions. We can see that even though the model appears to overfit the training dataset, the results are still satisfactory. Figure 5 shows some examples of good and bad bounding box predictions for each plant type. For good predictions, the locations and sizes of bounding boxes are nearly identical to the ground truth. We also observed some



(a) Ara2012 Good Prediction (b) Ara2012 Bad Prediction

(c) Ara2013 Good Prediction (d) Ara2013 Bad Prediction

(e) Tobacco Good Prediction (f) Tobacco Bad Prediction

Figure 5. Faster R-CNN Bounding Box Predictions vs Ground Truths.

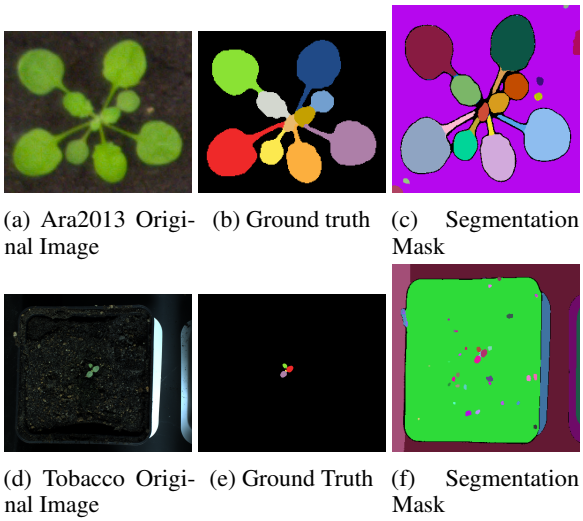
cases where the number of bounding boxes are the same for both the prediction and ground truth, but the prediction has a few bounding boxes with incorrect locations. This can be seen for all types of plants. Furthermore, we observed the model struggles to find bounding boxes for leaves with significantly smaller sizes relative to others, especially those which also overlap with other leaves, as seen in the bad prediction examples. There are also cases where a leaf looks like two leaves merged into one, which can confuse the model.

6. Discussion

From the results, Faster R-CNN is clearly superior than ResNet Regressor in terms of L1 and MSE errors. It can learn much more about the image as the model is composed of several CNN blocks, and additional inputs such as bounding boxes of objects can be provided during training. These conditions enable the model to learn the properties of the leaves. Since we only need to segment leaves, we need not worry about any other objects in the image, so it can be formulated as a binary classification problem. After predicting the coordinates of the bounding boxes, it suffices to assign a binary label to the object inside each bounding box: 1 for leaf and 0 for background. This simplifies our problem and learning becomes easier as a result. Despite the test set loss being larger than training set loss as depicted in Figure 4, it does not matter much for leaf counting, as the objective of counting is mutually exclusive to the objective of object detection. Thus, it suffices to predict a reasonably decent bounding box, which envelopes a leaf, and keeps the classifier loss low. Figure 7 shows the comparison between regression and classifier loss over epochs for our trained

	L1	MSE	mAP	mAP@.5	mAP@.5:.95	mAP@.75
ResNet Regressor	2.07	6.08	-	-	-	-
ResNet Regressor w/ SAM masks (full)	3.09	12.91	-	-	-	-
ResNet Regressor w/ SAM masks (2 dropped)	3.40	18.83	-	-	-	-
Faster R-CNN	0.68	1.27	0.816	0.951	0.705	0.792

Table 2. Evaluation Measures.



(a) Ara2013 Original Image (b) Ground truth (c) Segmentation Mask

(d) Tobacco Original Image (e) Ground Truth (f) Segmentation Mask

Figure 6. SAM Segmentation Masks vs Ground truth.

Faster R-CNN model. We can see that classifier loss is much lower than the regression loss, meaning that as long as our ultimate goal is to count the number of leaves, the regression loss has less importance than classifier loss, since classifier performance directly relates to the accuracy of leaf counts.

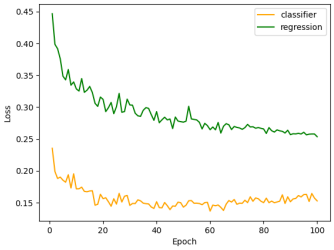


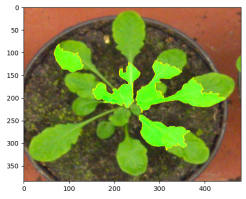
Figure 7. Regression and Classifier Loss over Epoch for Faster R-CNN.

The mAP scores show that the model performs the best when using $IoU >= 0.5$ as a threshold to select bounding boxes for the training set, which helped us decide what threshold to use during inference. The model achieved decent average precision for all observed thresholds, which means the model can effectively locate the general area of the individual leaves after fine-tuning. There is potential for the model to achieve even better performance with a larger dataset with more plant types.

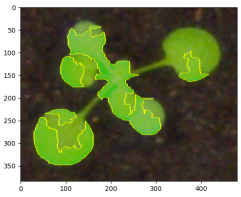
Contrary to our expectations, SAM + ResNet Regressor did not perform well. However, after observing segmentation masks generated by SAM, the reason was quite apparent. First of all, SAM was automatically generating masks without any prompts. This resulted in segmentation of each and every object that the model could identify in the image, introducing a great deal of noise. For example, in Figure 6d, the tobacco leaves are so small as to be barely visible, yet segmentation of these leaves was quite successful, as can be seen in Figure 6f. However, along with the leaves, the model also segmented many soil particles, causing difficulty in differentiating noisy patches from leaves. In future, when text prompt engineering is made available in SAM, the ability of the model to adjust according to the task may improve and possibly cut down noise. Furthermore, a majority of the poor segmentation results concern the tobacco plant class, likely due to its overall small size (including leaves), which already raises difficulties for the task of leaf counting.

We next turn to LIME to glean additional insight into how ResNetCount is making its predictions. The highlighted superpixel regions for the rosette plants in both the Ara 2012 and Ara2013 datasets (as shown in Figures 8a and 8b, respectively) appear to very reasonably identify the leaves of the plant, with more focus placed on the leaf petioles and bottom parts of the leaves. Bearing in mind that the chosen number of superpixels to be included in the LIME explanations was set to 10, this seems generally indicative that LIME is correctly identifying regions the ResNetCount model is using for its count predictions for the rosettes. It should be noted that in both cases, there is some discrepancy between the predicted counts and the actual counts, with the prediction for the rosette in Figure 8a being undercounted by around 5 leaves and overcounted by over 2 leaves in Figure 8b. This is likely due to the difficulty in counting overlapping leaves, particularly those close to the plant stem; this is further exemplified by the highlighted superpixel in that region of both instances.

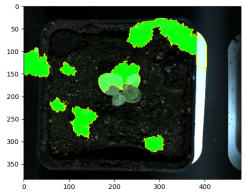
This issue of overlapping leaves is also readily apparent



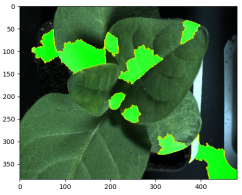
(a) Rosette from Ara2012 with 18 leaves, 12.96 predicted.



(b) Rosette from Ara2013 with 8 leaves, 10.43 predicted.



(c) Tobacco with 6 leaves, 6.64 predicted.



(d) Tobacco with 8 leaves, 10.44 predicted.

Figure 8. Generated LIME explanations for ResNetCount predictions on images of Arabidopsis thaliana rosettes from the Ara2012 and Ara2013 datasets, and of two Tobacco plants.

in the tobacco leaf ResNetCount predictions shown in Figures 8c and 8d, though there clearly appear to be additional shortcomings in the ResNetCount performance. The smaller tobacco plant shown in Figure 8c received the most accurate predicted count out of the four chosen instances, with a predicted count of approximately 6.64 and an actual leaf count of 6, though its explanation suggests that many irrelevant areas in the image are being used by ResNetCount towards this prediction. Indeed, superpixels overlaying the pot border and beyond, as well as areas of plain soil appear to be contributing more to the prediction than the bottom unhighlighted area of the plant. A similar occurrence is visible in the larger tobacco plant in Figure 8d despite its significantly larger size, with superpixels overlaying soil and table areas being highlighted. This suggests that ResNetCount may be struggling with accurately identifying tobacco plant leaves compared to rosette plant leaves, though this may be unsurprising given that the original unaugmented data contains only 62 tobacco plant images versus the rosette plants' 285 images. In order to corroborate this, exploring other explainability methods could provide additional insight, however this is left as an avenue for future exploration.

7. Future Work

Unfortunately, due to time and resource constraint, we were unable finish some parts of our experiments as originally planned, so we leave them as future work:

- Train with more datasets.
  - We aimed to include two datasets to train and test in our work, but the other dataset was excessively large in size and too different from the CVPPP-2014 LCC dataset. We did not succeed in finding a preprocessing pipeline to extract useful labels (e.g., bounding boxes) in time, as these were not provided.
  - It may be interesting to see how training and testing performance is affected if more training data is available.
- Train with limited data.
  - Few-shot learning was the original goal of this work. However, we could not find a suitable method to tackle the few-shot aspect of this challenge, and decided to go with the traditional learning task instead.
  - It may be worth to explore the distribution of pixel values in the images, since leaves are mostly green. Potentially, one can leverage the color distribution to augment or even generate adversarial samples to have more variation in the training dataset.
- Explore more models
  - Other state-of-the-art object detection methods such as YOLOv6 (Li et al., 2023), Cascade R-CNN (Cai & Vasconcelos, 2018), and other models with ViT (Dosovitskiy et al., 2020) backbones, with transformers being the current state-of-the-art sequence model.

8. Conclusion

In this paper, we aim to evaluate regression and object detection approaches in the context of leaf counting. We trained and tested our models on the CVPPP-2014 LCC dataset. From our results, Faster R-CNN model outperforms the ResNet Regressor due to its more structured and supervised learning. It can also tackle both leaf segmentation and leaf counting tasks, as the number of leaves are inferred from the number of the predicted bounding boxes. We also experimented with a novel methodology where we first transformed images into their respective segmentation masks using SAM, then applying ResNet Regressor to perform leaf counting. However, this approach did not perform better as compared to applying ResNet Regressor on raw images.

References

Cai, Z. and Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6154–6162, 2018. doi: 10.1109/CVPR.2018.00644.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 213–229. Springer, 2020.

Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., and Huang, J.-B. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Dobrescu, A., Giuffrida, M. V., and Tsaftaris, S. A. Leveraging multiple datasets for deep leaf counting. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 2072–2079, 2017. doi: 10.1109/ICCVW.2017.243.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Farjon, G., Itzhaky, Y., Khoroshevsky, F., and Bar-Hillel, A. Leaf counting: Fusing network components for improved accuracy. *Frontiers in Plant Science*, 12:575751, 2021.

Girshick, R. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, 2015. doi: 10.1109/ICCV.2015.169.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 11 2013. doi: 10.1109/CVPR.2014.81.

Giuffrida, M. V., Minervini, M., and Tsaftaris, S. Learning to count leaves in rosette plants. In *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP)*, pp. 1.1–1.13. BMVA Press, 2015. ISBN 1-901725-55-3. doi: 10.5244/C.29.CVPPP.1. URL https://dx.doi.org/10.5244/C.29.CVPPP.1.

Hahnloser, R., Sarpeshkar, R., Mahowald, M., Douglas, R., and Seung, H. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405:947–51, 07 2000. doi: 10.1038/35016072.

Hati, A. J. and Ranjan Singh, R. Towards smart agriculture: A deep learning based phenotyping scheme for leaf counting. In *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, pp. 510–514, 2020. doi: 10.1109/ICSTCEE49637.2020.9277402.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

Lathuilière, S., Mesejo, P., Alameda-Pineda, X., and Horaud, R. A comprehensive analysis of deep regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(9):2065–2081, 2020. doi: 10.1109/TPAMI.2019.2910523.

Li, C., Li, L., Geng, Y., Jiang, H., Cheng, M., Zhang, B., Ke, Z., Xu, X., and Chu, X. Yolov6 v3.0: A full-scale reloading, 2023.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T. (eds.), *Computer Vision – ECCV 2014*, pp. 740–755. Springer International Publishing, 2014. ISBN 978-3-319-10602-1.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Lu, H., Cao, Z.-G., Xiao, Y., Zhuang, B., and Shen, C. Tasselnet: counting maize tassels in the wild via local counts regression network. *Plant Methods*, 13, 2017. doi: 10.1186/s13007-017-0224-0.

Maas, A., Hannun, A., and Ng, A. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the International Conference on Machine Learning*, 2013.

Milletari, F., Navab, N., and Ahmadi, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571. Ieee, 2016.

Minervini, M., Fischbach, A., Scharr, H., and Tsaftaris, S. A. Finely-grained annotated datasets for image-based plant phenotyping. *Pattern recognition letters*, 81:80–89, 2016.

Misra, D. Mish: A Self Regularized Non-Monotonic Activation Function. *arXiv e-prints*, art. arXiv:1908.08681, 2019. doi: 10.48550/arXiv.1908.08681.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Ranjan, V., Sharma, U., Nguyen, T., and Hoai, M. Learning to count everything. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3394–3403, 2021.

Ren, M. and Zemel, R. S. End-to-end instance segmentation with recurrent attention. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 293–301, 2016.

Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 1135–1144. Association for Computing Machinery, 2016. ISBN 9781450342322. doi: 10.1145/2939672.2939778. URL https://doi.org/10.1145/2939672.2939778.

Romera-Paredes, B. and Torr, P. H. S. Recurrent instance segmentation. *CoRR*, abs/1511.08250, 2015. URL http://arxiv.org/abs/1511.08250.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33: 7537–7547, 2020.

Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., and Smeulders, A. W. M. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013. URL https://ivi.fnwi.uva.nl/isis/publications/2013/UijlingsIJCV2013.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yang, J., Guo, X., Li, Y., Marinello, F., Ercisli, S., and Zhang, Z. A survey of few-shot learning in smart agriculture: developments, applications, and challenges. *Plant Methods*, 18(1):1–12, 2022.