University of Salento

Faculty of Engineering
Degree Course in Computer Engineering

Master Thesis
in
Image Processing

3D Machine Vision System for Plant Phenotyping

Supervisor:
Ch.mo Prof. Cosimo DISTANTE
Co-Supervisor:
Ch.mo Prof. Nome COGNOME

Candidate:
Davide Basile
Matricola 20034689

Academic Year 2020/2021

Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis.
Curabitur dictum gravida mauris.

— *Donald Ervin Knuthyu*

Acknowledgments

A conclusione di questo lavoro di tesi ringrazio...

G. M.

Abstract

Increasing attention is paid to agriculture, the constant increase in demand for food and vegetables also requires the presence of new techniques for agriculture that maximize the harvest. In this thesis we will analyze the newest techniques on machine learning for the analysis of 3D images and how they can be used in the agricultural field. Our approach will be to analyze different types of plants to obtain as much versatility of the model as possible and to subsequently extract the data through 3D analysis to obtain the main characteristics of plant height, leaf area and weight. We have noticed how methods for analyzing images based on deep learning are very versatile and are able to adapt very well to different situations and are able to compete with the various methods studied ad hoc for the specific situation. Furthermore, having our own dataset allowed us to obtain more accurate results and to validate our model even better.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Plant Phenotyping

Plant phenotyping, i.e. the visual analysis of the morphological and functional characteristics of a plant, is a fundamental part of the improvement process quantitative and qualitative of plants of agricultural interest. It is connected to genomics with the analysis of the phenotype or with the performance of plants during interaction with environmental stimuli. From the analysis of the phenotype derives a better understanding of the plant-environment system, with the possibility of setting innovative breeding programs for new varieties. The new generation DNA sequencing techniques have revolutionized the methods and timing of selection in agricultural species, facilitating the genetic investigation of the characters, the identification of allele genes responsible for phenotypic expression and the their consequent transfer into new varieties through assisted selection with the use of molecular markers and the application of genomic selection.

Phenotyping emerges as a major limiting factor in the process of understanding the genetic, physiological and biochemical basis of agronomic traits, not only because of the high cost involved and the large margin of error, but also because of the large amount of time required to collect all the necessary data. This is the reason why in recent years there has been a growing interest in the technical and scientific world in the development of high-throughput phenotyping platforms, i.e. fully automated plants in greenhouses or growth chambers, equipped with precise environmental control and remote sensing techniques to assess plant growth and performance. This has been made possible by advances in engineering and computing, and the increased availability of hardware to store data at relatively low cost. relatively low cost. In particu-

lar, the availability of new sensors based on different image acquisition systems (visible, thermal and spectral at different wavelengths, distance), as well as the development of robotics and unmanned aerial vehicles or drones, together with the development of dedicated development of dedicated processing software, facilitates the acquisition of data for features of interest on a large scale, in a precise, accurate, rapid and low-cost manner. Generally, these robotic and computerized platforms acquire and analyse images in the visible light (RGB, RGBD) to perform analyses on morphology, architecture, colour, health and phenological stage of the crop.

1.2 Instance segmentation

Computer vision problems include: image classification, object detection and object segmentation. In order to solve these problems, deep machine learning deep machine learning techniques are used to solve these problems. If in image classification, the classes to which the objects in an image belong are indicated, in object detection in an image, object detection identifies the object and indicates its position.

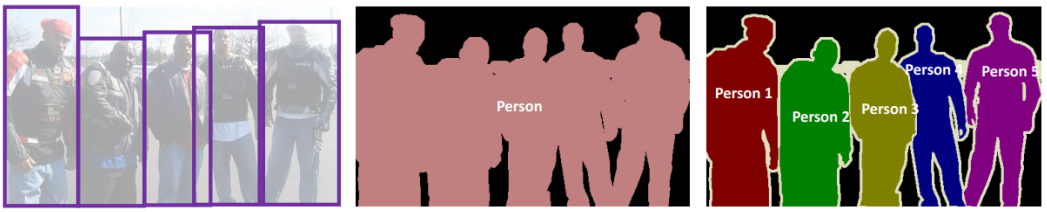


Figure 1.1: Difference between object detection, semantic segmentation and instance segmentation

On the other hand, image segmentation is the technique which, during the processing and analysis of digital images, makes it possible to divide an image into parts or regions, often on the basis of pixel characteristics. This technique was mainly developed to effectively solve problems and critical issues that require detailed information about the objects in an image. Details that cannot be provided at all by classifying the entire image or parts of it corresponding to a simple box. An important process This important process only ends when the objects, or rather the regions of interest, have been identified and we move on to the extraction and analysis of the information obtained. There are two different types of segmentation, the semantic segmentation based on the type of object present, in which all pixels belonging to a type of object are classified

in the same way and marked with a single pixel colour. In contrast, instance-based segmentation derives from the first but differs in that object detection is also applied to the elements present in the image and each object is classified and instantiated in its own class with a different label.

1.3 Thesis goal

The aim of this thesis is to apply new methods in the field of deep learning to plant phenotyping and to demonstrate how machine learning can accurately identify plant phenotypic traits and extrapolate their characteristics in order to quickly and accurately help plant breeders to identify plant phenotypes. identify precisely the phenotypic traits of plants and extrapolate their characteristics in such a way as to help the farmer, the agronomist or the technician in charge of the plant phenotyping in a fast and precise way. the farmer, agronomist or technician who will manage the plantation. The state of the art will be analysed and it will be illustrated how the data collection and comparison of the data was carried out. The state of the art will be analysed and it will be explained how to collect data and compare results:

- Analysing the state of the art and performance of neural networks with regard to plant phenotyping;
- Choosing which networks perform best, implementing them and training them on existing datasets;
- Recreating a controlled environment where measurements can be tested and extrapolated for verification;

1.4 Thesis structure

This thesis is structured in five chapters.

The first, as we have just seen, is an introduction to the problem and the techniques that will be used.

The second chapter describes the main articles related to the problem, their approach with different resolutions and their results.

In the third chapter, we present our solution to the problem, how it should be approached, what are the best neural networks that fit the problem and metrics used for comparisons. the problem and the metrics used for comparisons.

In the fourth chapter, the datasets are presented. In particular, the datasets used for training the networks are shown, followed by the calculation of the metrics and the comparison with other state-of-the-art methods. the calculation of the metrics and then the comparison with the other methods seen in the state of the art. Finally, we show our dataset, how it was acquired and its main characteristics, as well as the experimental results collected both by direct measurement and by the proposed solution. Finally, the metrics related to our dataset are shown.

In the fifth and final chapter, conclusions and possible future developments are discussed.

Chapter 2

Related Works

2.1 Plant Phenotyping

Plant phenotyping has been recognized as a bottleneck [1] for improving the efficiency of breeding programs. In recent years, image-based approaches have shown great potential for high-throughput plant phenotyping, resulting in an increased focus on image-based plant phenotyping. To meet the world's growing demand, current crop productivity must be roughly doubled by 2050, due to global population growth that may exceed 9 billion by 2050 and 11 billion by 2100. Over the past years, various HTP solutions have been developed to improve phenotyping capability and throughput, including tower systems, gantry systems, mobile ground systems, and low and high-altitude aerial systems, one obvious trend has been noted in recent HTP systems, computer vision. Cameras have been used more frequently because of their extensive ability to extract complex features. 2D images can provide spatial information of a scene plus an additional dimension of data such as spectral information. 3D images can provide a 3D structure of a scene that can be used to calculate morphological features of objects. Finally, 2.5D images retain information about the image plane structure, which is similar to that of 2D images, and capture depth information about a scene, which can be used to reconstruct the 3D structure of that scene. one of the first operations to be performed for phenotyping is to identify and define the phenotypic traits to be measured, which largely determine the use of appropriate imaging modalities for plant detection. Measuring phenotypic traits usually requires one or more computer vision tasks that can be solved by developing new algorithms or improving existing ones through conventional image/signal processing, machine learning, or a combination of them. Conventional machine-learning based approaches have

generally improved generalizability, but most still cannot meet the requirements for current phenotypic purposes. Deep learning is a subset of machine learning and enables hierarchical learning of data. The key benefit is that features will be learned automatically from the input data, thus breaking down barriers to developing intelligent solutions for different of applications. A commonly used DL architecture is deep convolutional neural networks(CNNs)[2], that were developed for image classification and eventually demonstrated better performance than humans on the same dataset.

2.2 Convolutional Neural Networks

Most imaging-based phenotyping applications essentially require solutions for one or more tasks related to image classification, object detection, and segmentation.

2.2.1 Image Classification

Image classification is one of the main tasks of computer vision and aims to assign images with predefined class labels. The development of modern CNNs for image classification can be grouped into three phases:

- emergence of modern CNNs;
- intensive development and improvement of the CNN architecture;
- reinforcement learning for the design of the CNN architecture.

After these improvements, representative CNNs can now usually outperform humans in image classification for various datasets. It should be noted that the performance improvement following the change in CNN architecture was highly dependent on human expertise and tuning efforts, which means that improving the CNN architecture could be as laborious as feature engineering in traditional ML. A reinforcement learning framework[3] was introduced to search for optimal convolutional cells on a small annotated dataset, and the resulting cells were stacked in different ways and transferred to an unknown large dataset. In addition to performance improvement, studies have been conducted to understand the mechanism of CNNs. This leads to the development of techniques towards explainable artificial intelligence that helps in developing interpretable and inclusive machine learning models. The study

also showed that learned features could be generalized to various classifiers, suggesting that CNNs could learn general representations of images rather than specific features for classification. Subsequent studies have continued in this direction and developed various gradient-based methods that can visualize the importance of features for classification results. Commonly used methods include guided backpropagation, gradient-weighted class activation mapping, and layered relevance propagation.

2.2.2 Object Detection

Object detection attempts to detect and classify all potential objects in a given image. The use of CNNs for object detection can be categorized into two groups: one-stage and two-stage CNN architecture. Two-stage models first detect candidate object regions and then classify the candidate regions into different object categories. The OverFeat framework was developed to use a single CNN to extract features for training classifiers and regressors separately[4]. The trained classifiers and regressors were used to predict class labels and bounding box coordinates, respectively, for candidate region of interest generated using a sliding window method.

Three key techniques in the CNN architecture of the RCNN family[5] were identified, including the region proposal network, the region-of-interest pooling operation, and the multitask loss function. A RPN was developed to generate ROIs of candidate objects using features extracted from CNNs, which simultaneously saved processing time and increased the accuracy of the region proposal. A ROI pooling operation was developed to extract a fixed number of features from ROIs of various sizes, thus avoiding repeated computation of features for different ROIs. A multitask loss function was used to unify the training process, which allowed for a training process for object detection.

Representative one-stage models include the you-onlylook-once family and the single-shot detector framework. A critical problem, however, has been discovered for these one-stage models: an extreme imbalance in the number of object and background regions. Most image regions contain only background information, providing little contribution to the model training process. However, if detection accuracy is the most important factor to consider, two-stage models would be the option; otherwise, single-stage models provide better computational efficiency for embedded systems and real-time applications.

2.2.3 Image Segmentation

Semantic segmentation seeks to provide masks for objects with the same semantic meaning, while instance segmentation seeks to provide individual objects in a given image. These in general can be divided into two categories: those encoder-decoder based, and those based on detection. The encoder-decoder-based is divided into two phases. The encoding phase uses CNNs to extract semantically meaningful feature maps from the input images, and the decoding phase uses transposed convolution for upsampling the extracted feature maps to labels. A detection-based framework is based on the CNN architecture for object detection. Several studies have explored the use of object detection models for instance segmentation, including concurrent detection and segmentation based on RCNN and DeepMask based on Faster RCNN[6]. In the last years several new architecture are developed. In order to improve object recognition and segmentation, Deformable Convolutional NNs [7] are developed. It is a combination about a different convolutional model and a new kind of RoI pooling. 2D convolution has two steps, sampling using a regular grid and sum the sampled values, instead, in the deformable convolution the regular grid is "deformed" by augmentation with offsets, by computing bilinear interpolation.

2.3 Application to Plant Phenotyping

2.3.1 Plant Morphology and Growth

Morphological changes in plant shoots are critical in describing plant development. Canopy cover and leaf area are two commonly used parameters to quantify plant growth and development. In order to segment plants many studies used color-based features, but usually had imperfect segmentation because plant color could have large variations due to illumination, shade, occlusion, and so forth. Most studies conducted on CNN in plant phenotyping consider it a semantic segmentation task and use an encoder-decoder-based CNN architecture for processing. Although these studies have shown improved segmentation accuracy, annotation of training data can be extremely difficult. To solve this problem, one study attempted to generate synthetic images with semantic annotations automatically for training the CNN model. The combination of synthetic and real images would improve the generalizability of CNNs for plant segmentation and thus the accuracy of growth analysis. The researchers also

combined CNNs with other DL methods for characterizing plant development. CNNs were used as a feature extractor to encode the spatial state of plants at individual growth stages, and RNNs were used to incorporate all spatial encodings to learn the temporal changes of plants. In this way, plant growth patterns could be fully encoded by neural networks to reveal differences between cultivars and treatment groups. This indirect phenotyping scheme could be particularly useful for selection-oriented programs, but explaining selection would be a significant challenge and barrier to many research studies that aim to understand the mechanism of many plant responses. In addition to morphological measurements, CNNs could be used to monitor some plant development events such as plant lodging. A new CNN architecture (LodgedNet)[8] was developed by integrating a custom 7-layer CNN model with handcrafted features. Compared with 10 well-established CNN architectures, LodgedNet provided comparable or better performance on differentiation between allurement and regular plots, but with a significant improvement in processing speed.

An application about this field is inside the research about Weed Growth Estimator[9]. In this research, a CNN was used to count the number of leaves on 18 different weed species or families. However, deep CNNs need to be trained on a large number of images to learn how to automatically extract general features from the input data. Overfitting makes it difficult for the network to generalize to new examples that were not in the training set. In addition, the training dataset was augmented by using horizontal tilt, rotation, zoom, width shift, height shift, and Gaussian smoothing filters. The authors selected the Inception-v3[10] architecture, which is a refinement of the GoogLeNet[11] architecture. The Inception-v3 architecture was chosen because of its good performance, ease of implementation, and relatively low computational cost, which yielded excellent results. This network with their dataset was trained for 20 epochs and the predictions from the 20 models were combined to increase the confidence in the predictions. Because the 20 models were not identical, they could predict different growth stages for the same plants. Predictions of the growth stage of the same plant by different models were more different when more plants were present in an image, resulting in a higher standard deviation for those images. In order to improve overall accuracy, the softmax outputs from all 20 models were aggregated, and the maximum value index was used as a sample class prediction. The accuracy results were 83%. Count-Diff is the average forecast bias, which is 0.07 leaves. This indicates that the model has a small tendency to overestimate the number of leaves. The Abs.

CountDiff shows that on average, the model is off by 0.51 leaves, and for 70% of the samples, it is off by zeroThus, according to the obtained results, it can be concluded that the developed model can be implemented with 70% accuracy on variable field machines, including weed control machine; because in these systems, the amount of toxic material applied to the weed in neighboring field classes does not differ much. Confidence in the estimated accuracy was calculated using Wilson confidence intervals over 10,000 iterations. The best network performance was obtained for some weed species, probably due to their large number of training samples and shared physical properties, for the others the average accuracy was less than 60% due to fewer training images. Therefore, this study presents a convolutional neural network-based method to estimate the growth stage in terms of leaf number of various weed species. Images from various camera models were collected in fields with different soil types and light conditions. Because the images were collected under field conditions, plants often overlapped each other, which this network was typically able to overcome.

Instance segmentation is used to understand single element development and different plant species can be analyzed with rich leaf shape variation. In [12], the authors aimed to implement a novel strategy to exploit the global feature of invariant leaf shapes across plant species, and create an automatic generation of leaf patterns for incorporating leaf shape knowledge. They present a novel segmentation of the leaf instance with a top-down view of plant images in high-throughput phenotyping platforms. The acquired images suffer from uneven illumination, moss, and shadows. To address the problem of scene variability, a statistical image segmentation algorithm Mean-shift Bandwidths Searching Latent Dirichlet Allocation is employed. In order to find the optimal MeanShift[13] bandwidth for stable segmentation, the authors used LUV distances between topics. However, the color of moss is very similar to plant leaves, so using LUV distance is not suitable for the studied images. To solve this problem, the authors employed the distance between texture descriptors. Plant leaves spread out from the center showing different orientations, so finding candidate regions to leaves will require examining different combinations of orientations and widths in the RGB image. To constrain the search for possible leaf candidate regions, first the segmented image is transformed to polar coordinates. The center and radius required for polar transformations are calculated based on its skeletal image. The center is extracted based on the betweenness centrality of the nodes in the skeleton image. The goal of nomi-

nating leaf candidates is to extract leaf representatives that can be processed by the model generation step. Their algorithm represents the image as a graph with a fixed number of nodes and neighboring nodes connected by weighted edges. The seed extraction strategy based on the hierarchical representation of slices and its use in the random walk results in an under-segmentation of the plant image, this is due to the connection property of vertical slices. However, this property also allows for the optimal segmentation of non-occluded leaves that exist as disconnected regions in the plant image. Given the output of the random path instance, they automatically isolate the non-occluded leaves by exploiting the symmetry property of the leaves around the medial axis. Two classes of leaf and non-leaf candidates are generated, then, multiple leaf candidates are removed from the segmented image to obtain a modified mask. If the modified mask belongs to a small area of the input segmented mask, then these leaf candidates are the final instances of the leaves extracted from the proposed picture. Otherwise, these candidates form a part of the output instance and the modified mask is used as input for the multi-leaf matching model. Leaf templates have a great influence on matching multi-leaf templates. Thus, to generate templates that accommodate large variations in the shape and size of the leaves of a plant species in its growth cycle. A post-processing pipeline is used to remove stems based on the Hough transform. The Hough transform generates a parameter space, where each pixel in an x-y image space is mapped to a line of an m-c parameter space and the peaks represent the lines. Since the canonical leaf shape is used, peaks with m centered around zero degrees correspond to stems. Since the multi-leaf template matching algorithm is sensitive to rotation and scale, the representative shapes are varied using geometric transformations such as scaling and rotation. This database of templates with different shapes, sizes, and orientations, generated through real-time image sampling, is employed in multi-leaf template matching. For each pattern, the algorithm first computes the position of the minimum matching Chamfer, then, an optimal set of leaf candidates based on an objective function with three terms:

- average CM of the selected leaves that promotes the selection of leaf patterns with the minimum CM distance;
- the number of estimated leaves selects an optimal number of leaf patterns;
- the distance of the synthesized mask from the selected leaves and the tar-

get image mask is selected and that when disposed generates the target mask accurately.

Overall, they observed that the output of the Gaussian mixture model appears noisy, while K-means, Otsu's threshold based on excess green features, and Minervini's algorithm excessively segment the foreground resulting in a disconnected vegetation region. On the other hand, significant agreement with ground truth can be observed for the modified MSBS-LDA algorithm using texture-based distance metrics. One approach was based on the orientation field map and leaf tips of the polar transformed image. For leaf tip detection, they chose the Harris-Stephens angle detector. The second approach was based on the skeletal image of the polar transformed image. Here, the path from the end nodes of the image is traversed and a minimum deviated direction is employed at the junctions. Based on the quantitative result, it was observed that due to the oversegmentation of the non-overlapping leaves, the instance output failed the symmetry threshold, however, the instances of the used approaches that labeled as leaf candidates showed good agreement with the corresponding non-overlapping leaves of the ground truth. It should be noted that the number of leaf candidates extracted plays an important role in the extraction of representative shapes, especially at later growth stages. Based on the database of plant phenotyping images, for early and middle growth stage images, multi-leaf pattern matching was not frequently used because the generated leaf candidates comprised 90% of the input segmented mask. This was due to the non-overlapping scenarios observed in these images. However, it also led to the neglect of leaves that were highly. The comparative study demonstrates the efficiency of their proposed structure in relation to different plant species. The strategy to extract non-occluded leaves involves the automatic generation of shape priors, which can be further used to segment individual leaves into overlapping regions in the image.

In High-throughput Phenotyping, the required task is to classify individual plant cultivars to assist plant breeding[14]. In this paper, the authors pre-process the images with object localization directly based on prior knowledge such as sensor fusion and experiment setup, object segmentation using green-based brightness thresholds or Otsu threshold, and finally contour extraction as an array of contour pixel coordinates. Then the authors unfold the 2-dimensional contour into a 1-dimensional radial object descriptor, which indicates the shortest distance from each pixel to the seed. Then the authors post-process these results, where they normalize the values in ROD in a range

of [0,1] and scale them from different lengths into a fixed by sampling or interpolation. To complete the ROD feature they merge them with another low-level local feature descriptor, the histogram of oriented gradients, which have local gradients at different orientations, which is powerful for differentiating objects with meaningful texture patterns. The histograms of neighboring cells are further grouped into block groups with an overlap ratio to enhance local contrast. Another way the authors used to integrate ROD is ResNet-18[15] which is a type of CNN with residual parts to skip layers so that higher accuracy can be efficiently achieved by using deeper layers. It is the representative of deep feature extraction methods because its performance has been demonstrated in plant research. To classify and calculate the loss function, the authors used Softmax regression, and learned the set of weight parameters by minimizing the loss function. The first experiment involves soybean plots grown in the field using UAS imagery, where we highlight the robustness of their approach under changing camera angle of view. The second experiment involves rosette plants in the greenhouse using a fixed camera, where they highlighted the robustness of their approach under sus change in growth phase. They evaluated the robustness of ROD and its fusion with HOG and ResNet in Softmax using seven different approaches. The first two approaches, CNN and ResNet-18, are based on CNN, Each convolution layer is followed by a batch normalization layer, a ReLu layer, and a max pooling layer. For both CNN and ResNet-18, they manually tuned the mini-batch size and learning rate based on each validation dataset before finalizing its classification result. By combining all these layers, they obtained five models, which will be trained using the Scaled Conjugate Gradient method. Using the HTP-Soy dataset, the authors found that ROD-ResNet-Softmax achieves the highest validation mA of 0.947 while ROD-HOG-Softmax achieves the highest test mA of 0.866, which slightly exceeds ROD-ResNet-Softmax. ROD-Softmax significantly outperforms Fourier-Softmax, which indicates the advantage of ROD over Fourier as a contour-based mid-level descriptor. It is even surprising to see that ROD-Softmax achieves mA comparable to HOG-Softmax even though HOG contains much richer information. ROD-HOG-Softmax outperforms ResNet-18 in inference of soybean plots for most pairs of growth stages. The advantage of ROD-HOG-Softmax becomes more apparent during the later growth stages, where the soybean canopy grows faster and its contour becomes more discriminating. In contrast, in Arabidopsis, Bean, and Komatsuna, the authors found that ROD-ResNet-Softmax outperforms almost all other approaches in

terms of both validation mA and test mA. ROD-Softmax significantly outperforms Fourier-Softmax and achieves classification accuracy comparable to HOG-Softmax.

2.3.2 Plant Stress Phenotyping

Plant stress phenotyping aims to identify and evaluate plant responses to abiotic and biotic stresses, providing information for selection of accession lines with high stress resistance and tolerance in breeding programs and understanding of intrinsic mechanisms in genetics/genomics studies. Plant stress phenotyping can be categorized into four stages:

- identification (presence of stress);
- classification (type of stress);
- quantification (severity of stress);
- prediction (possibility of stress occurrence).

The development of image classification-based approaches can be divided into two phases. In the first phase, studies have intensively investigated known and customized CNN architectures because of the availability of annotated datasets and the simplicity of implementing and training CNN for image classification. Data annotation for image classification is also relatively easy, so a large number of images in a newly collected dataset can be annotated in reasonable time and cost, especially when a proper data collection procedure is used. As a result, studies related to plant stress detection typically have a sufficient number of images annotated for model training. In the second phase, pioneering studies sought to understand the reasons leading to the high performance of CNNs for stress identification and classification, because understanding would not only help to improve CNNs but also ensure the biological correctness of the results obtained. Although some studies have adopted deconvolution layers to visualize activated pixels in different convolution layers, the visualization results have not been used to compare with human evaluation or correlate with biological knowledge. Compared with the first phase studies, the two pioneering studies demonstrated the importance of understanding the mechanism of CNNs for stress phenotyping, as well as the potential for quantifying stress severity. Image annotation is still recognized as a limiting factor for the use of many DL algorithms, so the researchers investigated the use of generative

adversarial networks to generate synthetic images for training CNN models for plant stress detection and classification[16]. A very recent study explored the use of a custom CNN architecture to detect plant diseases in hyperspectral images. The novelty of the custom architecture is the use of a 3D convolution operation that can directly convolve spatial and spectral information into hypercubes. This would not only inspire future studies related to plant stresses, but also allow the reanalysis of many previous hyperspectral images collected for plant stress analysis. With improved detection accuracy, subtle differences in stress between cultivars/treatments can be revealed to improve their understanding of plant responses to stresses.

An example about drought stress classification is [17], where the authors modify the standard SfM pipeline[18] as proposed in to use learned keypoints and descriptors based on deep networks. They employed Learned Invariant Feature Transform (LIFT)[19] for keypoint detector and descriptor learning instead of Scale-Invariant Feature Transform[20]. The key points in LIFT correspond to distinctive regions, where the conditions that define distinctiveness are learned with a Siamese deep network on the dataset for the target domain. Thus, the authors selected LIFT for its ability to learn key points and the corresponding descriptor based on the specific characteristics of the dataset. Learning descriptors and patch detectors based on deep networks requires training the detector-describer with specific examples. This allows the network to adapt to the complexities of plant structure, especially curvature and color variation. Therefore they perform LIFT detector-orientation estimator-descriptor pipeline training with a leaf matching dataset. For segmentation, they used and Voxel Cloud Connectivity Segmentation to segment the plant canopy from the reconstructed point cloud. It works directly on 3D point clouds. While leaf segmentation in 2D images is an active area of research, due to the availability of depth information the problem in 3D can be looked at from a pure computer vision perspective. The method consists of converting the input point cloud into a voxelized point cloud and construct an adjacency graph. To construct the adjacency graph, a grid of voxels is formed and seeded voxels are selected and initialized. Isolated voxels are filtered considering a small search volume around the seeded voxels. The voxels are then clustered, conditioned by the smallest gradient within the search volume. The clusters are further aggregated into supervoxels by comparing a feature vector 39 a dimensional feature vector derived using XYZ, RGB etc. The authors considered two types of descriptors Local Descriptors and Deep Descriptors.

For the former, they evaluated and Signature of Histograms of Oriented Gradients, Rotational Projection Statistics, and Fast Point Feature Histograms. The applicability and comparison of these descriptors in characterizing various plant-related tasks, and in particular the identification of drought stress, have not been studied before. The main advantage of local descriptors lies in their ability to encode geometric properties of the model. This feature makes them suitable for quantifying various phenotypic traits involving structural changes. For the second, there are two types of deep architectures to process 3 data:

- 3D Convolutional Neural Networks (3D-CNN);
- PointNet.

Due to the inherent nature of the convolution operation, 3D-CNNs work on structured data, i.e., voxelised clouds. However, PointNet[21] works directly on unstructured 3D PointCloud data. The voxelization of the point cloud introduces an approximation to the model as it is essentially a quantization process. PointNet generates a global feature on the point cloud point cloud. This is done by learning a permutation invariant representation of points from the input point cloud that is encoded into a vector using a symmetric function. However, by design, the global feature produced by PointNet does not capture local geometric information. They observed that this is the reason for the relatively poor performance of PointNet global features trained on traditional objects. To overcome this limitation, the authors aggregates the local and global information in a manner similar to PointNet's segmentation network, i.e., the global descriptor is fed back into the network along with the keypoint descriptor to generate a more robust keypoint descriptor. Next, they quantize the resulting local descriptors for the detected keypoints on the point cloud and concatenate them with the global descriptor. The pipeline starts by extracting 3D features from the segmented 3D point cloud. This is followed by a training and testing phase. It should be noted that the number of key points is different for each point cloud. Therefore, during the training phase, they learned a quantized representation of the 3D features. Quantization is necessary to obtain a single uniform length descriptor for each point cloud. In this work, they directly quantize the features using Fisher Vector. As will be evident in the experimental section, Fisher Vectors are capable of discriminatively encoding features in point clouds. Since the size of the codebook from Fisher Vector is small, they use a linear classifier, Support Vector Machine for

further classification. This approach allows us to reduce the overall computational complexity while maintaining robustness. For PointNet, they report results on both the pre-trained model and after tuning PointNet with 3D Point Cloud models from the training dataset. The fine-tuning is performed by initializing the weights from the pre-trained PointNet for object classification and then continuing the training process with the 3D point cloud of the wheat plants. The reconstruction of such fine details can be attributed to the quality of matches on the leaf surfaces from the key points and descriptors learned for 3D reconstruction. Accuracy is calculated as the percentage of the number of correct classifications over the total number of test inputs for the respective drought and health classes. However, it is interesting to note that PointNet pre-trained on rigid objects performs poorly against all compared descriptors. This could be due to two reasons: the default architecture of PointNet is not easily generalized, and plants have a smooth, textureless surface and are usually heavily occluded, which is not usually found in rigid bodies. In addition, RoPS outperforms all other local descriptors followed by SHOT (FV). It can be seen that descriptors quantified with Fisher Vector consistently perform better than the corresponding coding with Bag of Visual Words with an average gap of 1.6%. This shows that as with 2D descriptors, Fisher Vector is able to encode a more discriminative representation of local descriptors than compared to compared to BoVW. Therefore, in aggregating features in fine-tuned PointNet, Fisher Vector was used as a feature quantization technique.

Segmentation networks can be used for plant related applications such as plant trait estimation or in quantification of plant stress. Shrikrishna Kolhar and Jayant Jagtap[22] have used, SegNet[23] models based on CNN, U-Net[24] and residual U-Net are used for semantic segmentation of plant images. For data images are involved with scaling, normalization and data augmentation. image pixel values are normalized between 0 and 1 by dividing each pixel with 255. Normalization ensures that each pixel has a similar distribution which helps fast convergence in the process of training deep neural networks. SegNet is an encoder-decoder type model with 13 convolutional layers in the encoder/contraction and decoder/expansion part. Two fully connected layers are used between the encoding and decoding networks, as shown in. Each encoder in the encoding network performs 3×3 convolution with a filter bank to produce a set of feature maps. Batch normalization is performed after each convolution layer followed by a rectified linear unit. There is a decoding layer corresponding to each encoding layer. At each decoding layer, the number

of feature channels is reduced by half. The decoder samples its input feature map by a factor of 2. These feature maps are applied by repeated 3×3 convolution with multiple filter banks to produce dense feature maps. After each convolution layer, a batch normalization is used along with the ReLU activation function. The output with the multidimensional feature map is finally applied to the sigmoid function since they want to separate the plant pixels from the background, which is a two-class classification problem. The U-Net architecture includes an encoding network and a decoding network. The encoding network resembles the convolutional neural network. The encoding network includes repeated implementations of two 3×3 convolutions, rectified linear unit activation function, and a 2×2 max pooling with step 2 for down sampling. Each down sampling step doubles the number of feature channels. Max pooling helps capture a large context of input images with a compact feature map. Decoding helps to obtain more complex features at the loss of location information. At the final layer each 64-dimensional feature vector is mapped to the desired number of classes using 1×1 convolutions. Finally, the sigmoid function is used to classify image pixels as plant and non-plant pixels. In total, the network has 23 convolution layers. The modified residual U-Net is constructed using residual blocks and has a similar structure to the U-Net. The residual blocks make training easy, while the skipped connections help the information flow without degradation. The combination of batch normalization layer, ReLU activation, and convolution layer is repeated twice in each residual block. At each coding layer, instead of max-pooling, a stride of 2 is used for the first convolution operation to down-sample the feature map by 2. In the decoder, before each residual block, the lower level feature maps are up-sampled and combined with the feature maps from the corresponding coding blocks. After the last decoding layer, a 1×1 convolution and a sigmoid activation function is used for the segmentation task. Data augmentation is used to increase the variations in the dataset. The ReLU activation function is used in both networks in all layers except the output layer, where the sigmoid activation function is used. In this paper, a total of 8 evaluation metrics are used to analyze and compare the performance of residual U-Net with SegNet, U-Net and existing algorithms available in the literature. The metrics used for the analysis are dice coefficient, precision, recall, f1-measure, specificity, negative predictive value, mean absolute error, and accuracy. In these equations, the true positives are the correctly predicted plant pixels, the false positives are the predicted plant pixels that actually belong to the background, the false

negatives are the plant pixels present in the ground truth but not predicted by the algorithm, and the true negatives are the correctly predicted background pixels. Although the LSC and fig datasets are completely different, residual U-Net and simple U-Net work well on both datasets. Dice coefficient is one of the important metrics for image segmentation evaluation as it provides the spatial overlap between the ground truth and the output segmented image. All three networks have better segmentation performance for the LSC dataset than the Fig dataset also in terms of other metrics such as precision, recall, F1-score. The output masks predicted by U-Net and UNet residuals closely match the truth images, while the masks predicted by SegNet have subtle changes that can be observed through visual inspection. These results show that the UNet and U-Net residuals are able to draw accurate boundaries for plant leaves and perform better than SegNet in overlapping leaf segmentation. From these results, the authors can conclude that, the residues of U-Net and U-Net have comparable segmentation results segmentation results while both networks (U-Net residual and U-Net) perform significantly better than SegNet in predicting leaf shape and boundary areas of fig plants. U-Net and U-Net residual have better segmentation results than the state-of-the-art model for fig plant segmentation, while SegNet achieves comparable performance. Although there is a less significant difference in segmentation accuracy values, other metrics such as specificity, precision, recall, NPV, and F1 score are significantly higher for SegNet, U-Net, and residual U-Net than the state-of-the-art. Since U-Net residual and U-Net have better performance, both these networks are also tested on 10 high-resolution images of the whole fig field provided in the fig dataset. From the above results and discussion, it is clear that U-Net residual achieves better performance than other methods for the LSC dataset and comparable results with U-Net for the fig dataset. U-Net residual achieves this segmentation performance with about half of the trainable parameters compared to SegNet or U-Net. As they can see in Table 6, U-Net residual has the lowest, about 15.3 million trainable parameters followed by U-Net with 31.03 million and SegNet with 33.3 million parameters. This indicates that residual U-Net, although computationally very efficient, achieves better results than SegNet and comparable results to U-Net.

Effective segmentation of plant leaves is very necessary for non-contact extraction of plant leaf phenotype, especially leaf phenotype under environmental stress. Xuan Liu, Chunhua Hu and Pingping Li[25] proposed a combination of deep learning and point cloud clustering methods to detect individual leaves.

Point cloud processing algorithms are very time consuming. By combining the point cloud processing algorithm with deep learning, the proposed method clusters only the point clouds in the area detected by Mask R-CNN[26], which greatly reduces the processing time. The depth information is important to improve the detection accuracy, and can provide complementary information to the color image and can help separate overlapping leaves. In Their study the authors introduced the feature mapping approach to fuse the features of color and depth data into the FPN[27] structure during the R-CNN Mask formation process to improve the detection performance. The authors used Mask R-CNN to train the RGB-D data and the resulting features to the FPN structure, to improve the fusion of RGB-D information. The Mask R-CNN algorithm was used by the authors to detect pixel-level state-changing leaves in their study. First, the input data are transformed into feature maps through the backbone. Then, a network of region proposals is employed on the feature maps to generate region proposals. Then, through the RoIAlign process, the extracted features with the input image are aligned correctly. Finally, the features obtained by RoIAlign are fed to the fully connected layers for classification and bounding box regression, and also fed to the convolution layer for segmentation to obtain binary masks. Unlike simply overlaying the color and depth data for training, they train them separately. To combine the information from the input color and depth data into the Mask R-CNN backbone, we merge the feature pyramid of the depth data into the feature pyramid of the RGB data. The fusion operation is implemented as a summation of the features. Thus, each feature layer after fusion contains both color and depth features. When deep learning is used for leaf detection, the detected leaf areas may be inaccurate, may contain neighboring leaves or background pixels. To separate target leaves from overlapping leaves in R-CNN Mask detection results, the clustering method based on spatial information is commonly adopted. Among the common clustering algorithms in leaf detection studies, DBSCAN is always used. During training, the authors observed that after a number of epochs, the validation set loss begins to show an increasing trend while the training loss is still decreasing, indicating over-fitting of the model. Therefore, the number of training epochs is set as 20, selecting the last epoch that did not exhibit overfit. To analyze the quality of segmentation results, we adopted the Precision-Recall curve, average precision (AP), and F1 score as evaluation metrics. The detected leaf areas are more than 70% coincident with the ground truth. In a similar study, an AP of 0.899 was acquired. While their data set

is quite different from theirs. The performance of Mask R-CNN was evaluated in terms of AP and F1-score. A confidence threshold of 0.7 was selected to obtain their performance values. Comparing the results, our network with preprocessing gave the best performance with an F1-score of 0.825 and an AP of 0.912, followed by the network without preprocessing with an F1-score of 0.803 and an AP of 0.892. However, the classical Mask R-CNN network only achieved an F1-score of 0.791 and an AP of 0.877. Thus, the performance of our network is superior to that of the classical Mask R-CNN network in our dataset. In addition, the use of preprocessing on the depth information also helps to improve the accuracy of the model. The DBSCAN clustering results of normal leaves based on Euclidean distance and manifold distance are similar. This is because the normal leaf is relatively flat, so the difference between the Euclidean distance and the manifold distance is small. While the Euclidean clustering algorithm, and can only detect the points with a greater distance from the leaf instead of the whole noise clusters, with the noise points closer to the leaf not detected. DBSCAN based on manifold distance achieves good results when segmenting neighboring leaves that overlap. Because of the overlap, the authors used the depth map obtained from the RGB-D camera and the camera parameters to convert the detected leaf area into a 3-D point cloud. Then, using the proposed clustering algorithm, the target leaf was successfully separated from the overlapping leaf. In object detection, on the other hand, Intersection over Union (IoU) is often used as a metric to evaluate the accuracy of bounding box location, but the accuracy of pixel segmentation cannot be indicated. In leaf phenotype research, the accuracy of leaf pixel detection is more important than leaf pixel localization. In conclusion, the proposed method could facilitate the study of 3-D lead phenotype search and improve the accuracy of automatic leaf segmentation.

2.3.3 Plant Counting

Counting plants and plant organs is critical to characterizing plant development. Regression or image classification is the simplest and most straightforward way for fruit/organ counting from the perspective of technical development. For regression-based methods, a major modification was made that replaced the softmax layer of a CNN with a single neuron for regression of numerical values. This simple end-to-end counting solution provided high accuracy for counting fruit and leaves from plants. A particular challenge

of regression-based solutions is the limited availability of annotated images, which leads to many potential problems such as poor generalizability of the model. To address this issue, one study attempted to generate synthetic data of tomatoes to increase data availability and diversity. Although the trained CNNs achieved 91% counting accuracy on real images, the study tested only red tomatoes, which have distinctive color characteristics from the background. The generalizability of this approach should be further validated for challenging situations such as detecting green tomatoes from leaves. GANs were also used to generate synthetic data for model training.

An alternative approach was to use patch-based training[28]. TasselNet was developed to count corn tassels in two phases. In the first phase, a local CNN regression model was established to predict the number of tassels in each patch of an image. In the second step, the estimated count in each image patch was averaged over the individual pixels in that patch to create a count map with the same spatial dimension as the original image. The sum of all pixel intensities in the count map represents the final count of tassels in that image. Experimental results showed that TasselNet achieved counting accuracies from 74.8% to 97.1%, which were 2 to 5 times higher than those of conventional methods. TasselNet uses the patch-based training method, which substantially increases the number of training images. For classification-based methods, plant/organ counts were treated as a discrete counting problem and, therefore, a predefined score or grade was assigned to a given image rather than an exact count. An example of a classification-based method is WheatNet, which was developed to predict the percentage of flowering in wheat images. Multiple images were acquired for each plot. A total of 11 classes were annotated for each plot, corresponding to 11 visual scores with a percentage header from 0 to 100% with a 10% interval. The average prediction of all images in a plot was the final percent header for that plot, which reduced counting errors due to inaccurate classification.

Object detection is an intuitive approach to counting plants and plant organs in still images: accurate object detection ensures accurate object counts. DeepFruits was the first study to explore the use of modern CNN architecture for fruit detection. Several key contributions were recognized in this study. First, transfer learning was used to train a Faster RCNN[6] model with 100 labeled images, demonstrating the potential of using limited labeled images to train the CNN architecture. Second, when using RGB images, the trained Faster RCNN model provided a 1% improvement in F1 score over that of the

CRF model. Third, data fusion was conducted at the level of the raw data and at the decision level for the Faster RCNN models. A custom two-step framework was proposed using the superpixels generated by the simple iterative linear clustering algorithm as proposed regions. A CNN model was used to classify each superpixel as a flower or a non-flower object. While this approach showed higher performance than conventional ML methods, it has a potential limitation in region proposal. The advantage of the end-to-end CNN architecture is that they are able to use richer features for accurate localization, especially when the images vary greatly. However, superpixels are subject to image variation and may not provide optimal region proposals. The generalizability of this approach, therefore, is most likely lower than that of end-to-end methods. Many studies have also investigated semantic segmentation-based approaches for plant/plant organ counting. CNN architectures for semantic segmentation were first used to obtain plant/plant-organ masks. Then, the obtained masks were post-processed using conventional computer vision methods to isolate individual plants/plant organs so that the objects could be counted. An obvious concern is that although CNNs could provide accurate semantic masks, counting accuracy may still suffer from inaccurate post-processing. To address this concern, studies have explored the use of instance segmentation CNNs that can directly segment individual objects in images. These studies faced the same challenge in the lack of training data. To overcome this limitation, most of these studies developed algorithms to generate synthetic images for model training. Two types of image synthesizing methods have been proposed: rule-based and GAN[29]. Rule-based methods use a predefined leaf model to generate a plant based on predefined growth rules. GAN-based approaches,however, could generate synthetic images without sacrificing leaf structure. Thus, a method combining rule-based and GAN-based methods was developed for image synthesizing.

Inside Leaf Counting with deep Convolutional-Deconvolution Networks [30], the authors studied an architecture used for segmentation and counting are trained separately, but not independently, since the binary mask generated by the segmentation model is used to train the counting model along with the RGB channels. The idea of deconvolutionary networks is to construct a compact and informative set of feature maps or vectors from a set of input images. Usually, the design of a deconvolutionary network contains fully connected layers in between to generate the feature vector from the clustered feature maps. However, they propose that the features in the semi-global context

should be sufficient to segment the leaf regions from the background in the color images, and thus the FC layers could be omitted for their application. One advantage of eliminating FC layers is that it greatly reduces the number of trainable parameters without sacrificing performance. For these reasons, they adopt the SegNet[23] architecture. Instead, the convolutional front-end substructure of the network is the VGG[31] architecture with batch normalization followed by each convolutional layer. In the convolutional front-end of SegNet, there are five 2×2 pooling operations with zero overlap following multiple convolutional layers and rectification each time. Then, the convolutional feature maps are compressed 32 times before starting decompression through the deconvolutional back-end. The logic behind providing the counting module with the segmentation mask and the original RGB image instead of providing the segmented region in the RGB image or the binary mask alone. Although the segmentation results generated by SegNet are sufficiently accurate for the counting step for many images in the dataset, their network generates spurious segmentations for some of them. The design of their regression leaf counting network takes inspiration from the VGG architecture, which reinforces the idea of deeper architectures with a long list of convolutional and rectification layers stacked one after the other with several pooling layers in between and then the classification layer follows a pair of fully connected layers. The authors trained their model from scratch without using any pre-trained weights for initialization. Also in SegNet, the authors used different learning rates for different modules, while a fixed learning rate was used for all layers in their training. They augmented the data and trained the network in 3 steps. First, for each image, they extracted the union of the first 20 object proposals, flipped from top to bottom and left to right, rotated with an angular step of 4 degrees, cropped the largest square from the central position to avoid dark regions due to rotation, and created a pair of blurred versions with Gaussian and the corresponding sharp images. Second, they took the proposed images and their flipped versions and generated nearly 0.3M subsamples of size 224×224 deterministically with a fixed step and train the network for 8 more epochs. Finally, they generated another 0.19M samples in a similar manner to the second step, but this time from the original images instead of the proposals. Spatial cross-entropy was used as an error criterion. The ratio of foreground to background weights in the cross-entropy calculation for the first step was 2.0 and 1.2 for the later steps. In the test phase, they took dense 224 × 224 samples deterministically with fixed step from each of the test images and classified each

pixel according to the aggregate probability on the samples. During the design of the network architecture, they experimented with adaptive operations to handle images of varying sizes, but they did not seem to work any better than resizing the images to a fixed size. In addition, they had to be cautious in choosing the size for the resize operation so that for larger images with resolution such as 2000 × 2500, the properties of the small leaf regions would not deteriorate much. Considering this fact, they chose the size of the modified image to be 448×448 while preserving the aspect ratio. After performing the resizing operation, each of the images was augmented 8 times using intensity saturation, Gaussian blur and sharpening, and additive Gaussian noise. Each image was also flipped from top to bottom and left to right and rotated. After data generation, the counting or regression network was trained for 40 epochs using Adam with a fixed learning rate. The authors found that their predictions can be accurate for easier samples with no leaf overlap or moderate leaf size or both, but deteriorate for more difficult cases with smaller or overlapping leaves. In this sense, their generalized framework is able to model and infer leaf shapes under deformation and partial occlusion better than their main competitor given a few hundred images for a particular species.

2.4 Available Dataset

2.4.1 3D Phenotyping for Komatsuna dataset

Komatsuna dataset[32] has built by a platform to capture overhead views of a plant and measure the environmental conditions around it. They first selected a plant species, Komatsuna, which is a Japanese mustard spinach and a leafy vegetable. It can be grown indoors, and is known to be resistant to pests and grows very fast. Hydroponic culture is an alternative method. It is clean, irrigation and fertilization are automated so that plant growth is accelerated, fewer agricultural chemicals are needed than soil culture, and replanting failures do not normally occur. Another advantage is that plant root systems can also be measured because roots growing in water can be captured through cameras. For lighting, they used LED lights, with lighting durations and colors programmable using the software provided. To measure ambient temperature, humidity, and light intensity, they used a Sony MESH, where light intensity is measured, which can be measured in lux. However, lux may not be appropriate in a precise sense because it is based on the property of

human eyes. To measure the pure energy of lights, photosynthetic photon flux density (PPFD) is more appropriate in biology. In their platform, lux is used as a simplified index in environmental information. An RGB-D camera consists of an RGB camera and a structured light or time-of-flight depth camera based on infrared lights. Since the leaf size of plants in the early stages of growth is small, depth images usually need to be captured at a closer distance from the plant to magnify the size in the captured images. However, some RGB-D cameras are not designed for such short depth ranges. In their platform, they used Intel RealSense SR300 which is specifically optimized for short-range acquisition. They created two types of datasets using an RGB-D camera and a multiple RGB camera at approximately 2400 lux, 28 °C, and 30% humidity.

RGB-D Dataset

Since one RGB-D camera was attached to capture the entire part of the toolkit, each plant region was manually segmented as a plant image and labeled as a label image. In order to use the dataset for temporal leaf tracking, the same leaf label was assigned to the same leaf in images captured at different times. The original camera resolution was 640 × 480 and the resolution of the plant images was, for example, 166 × 190 pixels. Because the viewpoints of the RGB and depth images were aligned by the camera library, the labels were valid for both images. Due to the mismatch between RGB and depth camera, the resulting images were not aligned, so the depth image was aligned into an RGB image in their platform. For this reason, the authors provided the original depth image and a transformation matrix from the depth image viewpoint to the RGB image viewpoint so that the point cloud could also be transformed as needed without any interpolation.

Multi-View Dataset

For the multiview dataset, they captured five plants from three different viewpoints and manually segmented them into an individual plant like images. To use the dataset for spatiotemporal tracking of leaves, the same leaf label was assigned to the same leaf in images captured with different cameras at different times. The ground truth of the 3D shape was not measured because it was not easy to capture more accurate 3D shapes than when using multiple high-resolution images. This dataset will be useful for evaluating the segmentation of spatiotemporal instances for multiple views. In the relevant literature, in-

stance segmentation has typically been performed using a single view image, and may be more difficult than using multiple views.

2.4.2 Multi-Modality Plant Imagery Database

Two main species are in this dataset[33], first one is Arabidopsis thalian plants were grown at 20°C, in a 16 h/8 h day-night cycle with daylight intensity set at $100\frac{\mu \times mol \times photons}{m \times s}$. The second is black bean plants of the cultivar Jaguar, were grown in a day-night cycle of 14 h/10 h with day and night temperatures of 24 and 18°C, respectively, and a daylight intensity set at $200\frac{\mu \times mol \times photons}{m \times s}$. In all cases, seeds were planted in soil covered with a black foam mask to minimize the fluorescence background of algal growth. Two-week-old plants (Arabidopsis and bean) were transferred to imaging chambers and allowed to acclimate for 24 hours to LED illumination before beginning data collection. Different types of images were collected; fluorescence images were captured once every hour during the daylight period in a growth chamber. A series of five images were captured using a Hitachi KP-F145GV CCD camera, during a short period of bright light saturating photosynthesis provided by an array of white Cree LEDs. Fluorescence was excited using monochromatic red LEDs, collimated using Ledil reflector optics, and pulsed for 50 μ s during a short window when the white LEDs were electronically turned off. Infrared images were collected once every hour with the same camera and filter used for chlorophyll a fluorescence. RGB color and depth images were collected using a Creative Senz3D sensor. The sensor contains both a 1280×720 color camera oriented parallel to, and separated approximately 25 mm from, a depth camera that has a resolution of 320×240 pixels. The depth sensor uses a near IR flash illuminator and measures the time-of-flight of the beam in each pixel to obtain depth estimates in each pixel. Low-reflectivity surfaces, such as foam under plant leaves, are accurate only at close range. Image data, including fluorescence, IR reflectance, RGB color, and 3D depth images, were collected once every hour. Five minutes before the end of each hour, the 3D depth images and color image were captured using the Creative Senz3D sensor, followed by fluorescence and IR reflectance images collected sequentially at 2-minute intervals from the CCD camera with IR filter. A planar checkerboard model was used to calibrate all three cameras to obtain both intrinsic and extrinsic parameters. While the grid pattern is not visible in the depth image, it is still observable in the reflected IR image whose pixels correspond to the depth

pixels. The intrinsic and extrinsic parameters are stored as text files, and a Matlab function is provided that reads the parameters and can plot the camera positions. Time-of-flight depth measurements can have significant noise, and it is useful to both model and quantify it. Depth noise ϵ is modeled as the sum of an image-dependent term ϵ_I and a sensor-dependent term $\epsilon_S : \epsilon = \epsilon_I + \epsilon_S$. The term ϵ_I is a random variable for each pixel with a value that varies between successive images taken from a fixed pose of a static scene. On the other hand, ϵ_S is a random variable for each pixel that models its depth offset, and its value changes only when the scene changes. The variance of ϵ_I is estimated for each pixel of a fixed scene observed over multiple images. In their experiments, they observed flat albedo and uniform albedo. MSU-PID includes two subsets, one for each plant type: Arabidopsis and bean. Since there is no light at night, the plants cannot be captured by the RGB fluorescence and color sensors, while the IR and depth cameras can still perform capture at night. To make sure that all four modes are present at the same time, they release the portion of images captured only during daylight hours, which is 15 images per day for Arabidopsis and 13 for bean for all four modes. The two subsets differ in image resolutions. For Arabidopsis images, they labeled four frames per day, while for bean images, then they labeled seven frames per day due to the fast and spontaneous leaf movements of the plant. Leaf labeling can be propagated from fluorescence images to each of the other modes for Arabidopsis sequences. To quantify the accuracy of label propagation, they manually labeled three images for each of the three Arabidopsis plants on the fluorescence and RGB modes. The label in each mode is propagated on other modes. label propagation to provide labels for all four modes for Arabidopsis sequences. There are two benefits:

- decreases the need for manual labels,
- the labeling is consistent for all four modes.

In the case of bean images, pixel association between modalities is more difficult because the depth variations within the plant are large. The authors found that a homograph-based mapping worked poorly, and thus the manual annotations they provide apply only to infrared and fluorescence images using the same camera. Leaf segmentation, leaf counting, leaf alignment, and leaf tracking. To facilitate future research, they separate the database into training sets and test sets. Forty percent of the data is used for training and 60% for testing. Specifically, six Arabidopsis plants and two bean plants are selected

for training. The user can decide to use one or more modes of the plant images for training and testing, respectively. Using RGB and depth modes for training and RGB for testing can take advantage of additional information during learning without incurring additional sensor costs during testing, which can be implemented either by learning with side information or transferring learning with the missing mode. To evaluate the performance of leaf segmentation, alignment, tracking, and counting, we use four performance metrics, whose Matlab implementations will be provided along with the data. Three of them are based on tip-based error, which is defined as the average distance of a pair of estimated leaf tips with a pair of labeled leaf tips, normalized by the length of the labeled leaves.

The authors defined a threshold τ to operate on the corresponding tip-based errors, from this τ three metrics are calculated:

- Unmatched leaf rate, the percentage of unmatched leaves relative to the total number of labeled leaves. This can be attributed to two sources. The first is missed detections and false alarms. The second is the matched leaves with tip-based errors greater than τ. When τ is large enough, this value is equal to the leaf count error,

- Landmark error, the average of the tip-based errors smaller than τ of all matched leaves frame-to-frame. It is used to measure the alignment error of the leaf tips,

- Tracking consistency, the percentage of matching leaves from video to video whose tip-based errors are smaller than τ. This is used to measure leaf tracking.

In order to evaluate the accuracy of leaf segmentation, an additional metric based on the Dice score of estimated segmentation results and ground truth labels was adopted: Symmetric best Dice, the average of the best symmetric Dice among all labeled leaves. The authors treated all images over nine days as a video from the first image on the first day to the last image on the last day. To generate the template set, they first selected 12 templates with different aspect ratios from the labeled images in the training set along with the corresponding tip positions. For each template, the images are scaled to ten different sizes to cover the full range of leaf sizes in the database. For plant segmentation, they use a simple threshold and edge detection process to generate an image mask and an edge map. The best threshold is learned from the training set,

which is done by tuning the threshold in a certain range and they find the best one by evaluating the overlap of the segmented masks with the ground truth label masks. Firstly, they find the best position of each model in the edge map which has the minimum CM distance, secondly, the authors applied multi-leaf alignment, thirdly, they applied multi-leaf tracking. In the tracing process, the authors delete a leaf when it becomes too small. A new leaf is generated when there is a relatively large portion of the mask that has not been covered by the current leaf candidates.

2.4.3 Fine Grained Annotated Dataset

Arabidopsis' images[34] are collected in 2 different time June 2012 and between September and October 2013. The imaging setup consisted of a growth rack and an affordable automatic detection system to capture and send images of the scene via a wireless connection to a receiving computer. Plants were illuminated artificially and in a controlled manner to emulate daylight on a fixed daily cycle; camera sensors were placed between the lights. Images were captured only during the day for a period of time, two cameras were used: a consumer grade 7 Mpixel camera, and abbreviated as Canon, and an even cheaper Raspberry Pi based system. The tobacco images were acquired as part of the European "Gardening with a cognitive system" project, the GARNICS project, which aims to 3D detect plant growth and build perceptual representations for learning links to the actions of a robotic gardener. The setup allowed plants to be viewed from different poses. Below the robot and environmental sensors, the implemented system included:

- a water and nutrient solution delivery system for treatment application;
- high-power white LED lighting;
- low-power white fluorescent lighting used for imaging.

Tobacco plants were grown in 7×7cm pots under constant light conditions with a day/night rhythm of 16h/8h. Images show growth stages from germination to leaf development, early observations at the growth stage. A significant number of object-based annotations, bounding boxes, can be computationally obtained based on pixel-level segmentation masks of plants and leaves, respectively, that were manually annotated by experts. The annotation consisted of three steps. First, they obtained a binary segmentation of plant objects in the scene in

a computer-assisted manner. For Arabidopsis, they used the active contour-based approach described by Minervini et al. While for tobacco, a simple color-based approach was used for plant segmentation. The result of this segmentation was manually refined using raster graphic editing software to ensure that all visible part of the shoot is included in the plant mask and the background is excluded. Next, within the binary mask of each plant, The authors outlined the individual leaves completely manually. To reduce observer variability and increase accuracy, the labeling process always involved two annotators: one annotating the dataset and one checking the other. To reduce observer variability and increase accuracy, the labeling process always involved two annotators: one annotating the dataset and one checking the other. On secondary inspection of the data, annotators recorded additional categorical qualitative annotations such as: estimated difficulty of segmentation, plant appears in focus, leaves appear in vertical positions which is typical in tobacco, plant is occluded by another, and scene contains complexity. The authors divided the acquired images into different datasets according to the type of task:

- For plant segmentation, the dataset consists of 15 tray images from Ara2012, 27 from Ara2013, and 27 from Ara2013. For each image a corresponding black and white mask coded as an indexed image provides pixel-level information about the location of plant objects. For evaluation criteria they suggest Dice coefficient, precision, and recall, as they have been used throughout image analysis and are also common in plant imaging;

- For leaf segmentation, they used a leaf mask but without temporal label consistency. 120 from Ara2012, 165 from Ara2013, and 62 from Tobacco, images of individual plants that appear centered are used. Instead for the evaluation criteria they suggest SymmetricBestDice, the average symmetric Dice score across all objects;

- In leaf detection, the dataset consists of single plant images, 120 from Ara2012, 165 from Ara2013, and 62 from Tobacco, and for each image, a CSV file that stores per row the leaf index and coordinates of each bounding box, with as many rows as there are leaves, the number of accurate detections, and their accuracy assessed with overlap measures;

- For leaf counts they extract for each leaf the weighted transformed dis-

tance of the center of mass and also the center of mass. When these discord significantly, and if any of the centers are outside the binary shape, it indicates a highly asymmetric leaf and the annotator has been asked to select a center. The dataset consists of individual plant images and accompanying binary images that contain the centroids of each leaf as a single pixel. In the evaluation criteria suggest:

  - the difference between the number of leaves in the algorithm result and the ground truth $DiffFGLabels = \#Lar - \#Lgt$,
  - AbsDiffFGLabels, the absolute value of DiffFGLabels.

- For leaf tracking they provide 4 stacks of 13 images each from Ara2012 and 8 stacks of 17 images each from Ara2013. Leaf-level segmentations are provided with leaves that have the same label index throughout the sequence to ensure temporal consistency, with the Nawaz and Cavallaro protocol, which is based on overlap criteria, and code available from the authors. When leaves are vertical in the image axis, the overlap criteria may not be able to assign the correct matches;

- In boundary estimation they used leaf masks where each leaf labels and found its boundary, to produce a labeled indexed image where '0' is the background, '1' denotes a boundary between plant and background, and '2' denotes a boundary between overlapping leaves. This separation can facilitate the training of specialized boundary detectors. Typical criteria such as precision and recall are suggested, as well as those that do not penalize small local misalignment, which are suitable for evaluating the performance of boundaries between leaves, the MHD, and Minervini;

- In classification and regression For mutant and treatment recognition the authors release individual plant images and a text list denoting for each line image name, genotype, treatment type. For treatment type classification, 62 data are currently available from Tobacco. For mutant classification, 165 are currently available from Ara2013 and 165 from Ara2013. Because of the diversity of problems considered in this category, for classification problems, precision and recall criteria are recommended, and for regression problems, mean absolute error and root-mean-square error between predicted measures and truth are recommended. and ground-truth measures are encouraged.

2.4.4 Deep Learning for Multi-task Plant Phenotyping

The ACID dataset[35] consisting of a doubled haploid population of spring wheat plants was obtained from the Nottingham/BBSRC Wheat Research Centre and grown in 2l pots in a greenhouse. Imaging was conducted using a 12MP consumer grade camera attached to a custom built imaging system that provides a consistent black background containing multiple high resolution peaks. Instead of predicting location directly, the authors performs a pixel-wise regression, identifying the high probability areas of each target. Heatmap regression has been successfully used in, among other things human pose estimation. The network architecture is based on an encoding/decoding framework, in which a series of convolutional and spatial downsampling operations begin by computing a fixed-dimensional feature representation of the image. This feature space is then resampled to the original resolution. During training, random augmentation was applied with random rotation and scaling extracted from normal distributions with standard deviations. The network was trained end-to-end, from scratch, using RMSProp. The authors used a mean square error loss function with an initial learning rate of $2.5x10^{-4}$, and reduced by a factor of 10 every 200 epochs. Spike and spikelet positions are computed from each output heatmap using non-maximum suppression. For all output pixels, any pixel with an intensity greater than its four neighbors is classified as a feature. The number of additional false positives generated by the non-maximum suppression component of their approach is negligible. Accuracy represents the fraction of detections that are true positives. They applied a distance threshold for successful detection, then varied this threshold to explore the effectiveness of the approach at different tolerances. A true positive for a spike or spikelet is any predicted location that is within this normalized distance of a ground truth point. Similarly, a false negative is any ground truth point that is not within the normalized distance of a predicted feature. To measure counting accuracy, they compared only the number of predicted features against the number of ground truth points and calculated a percentage error, to simulate a counting-based phenotyping task. Given the lower recall performance on the tips for the 256-pixel input size, one should expect a higher error. they also found a negative value for the error of the tips indicates that the network tends to underestimate, rather than overestimate the number of spikes.

Chapter 3

Proposed Solution and Experimental Setup

Deep learning is showing very effective and promising results in a number of areas, often surpassing those obtained with the old methods. In this chapter we will show how we apply deep learning techniques to plant phenotyping, in particular we will show how the network used is adapted to different plants and how it can extrapolate individual leaves from the images.

3.1 Our Method

Segment each instance of a plant is an hard work, images can have different dimension, and each plant can have different position in an image. For this reason, we decided to identify each plant separately and then go on to identify each instance. In order to classify each plant, we made use of different technologies and we needed to modify the datasets at our disposal. To do this we used an online tool called Make Sense, which allowed us to create the position annotations of the boxes for object detection.

3.1.1 Object Detection

For the development of object detection of individual plants we used one of the best known and most powerful neural networks in the field of object detection YOLO [36]. This network, taken in version 5, in contrast to the sliding windows, sees the entire image during training and test time so it implicitly encodes contextual information about classes as well as their appearance. Inside YOLO the separate components of object detection are unified into a single

neural network. It divides the input image into an $S \times S$ grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. Each of this consists of 5 predictions: x, y, w, h, and confidence. The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell. The width and height are predicted relative to the whole image. Finally the confidence prediction represents the IOU between the predicted box and any ground truth box. YOLO is inspired by GoogLeNet [37] model

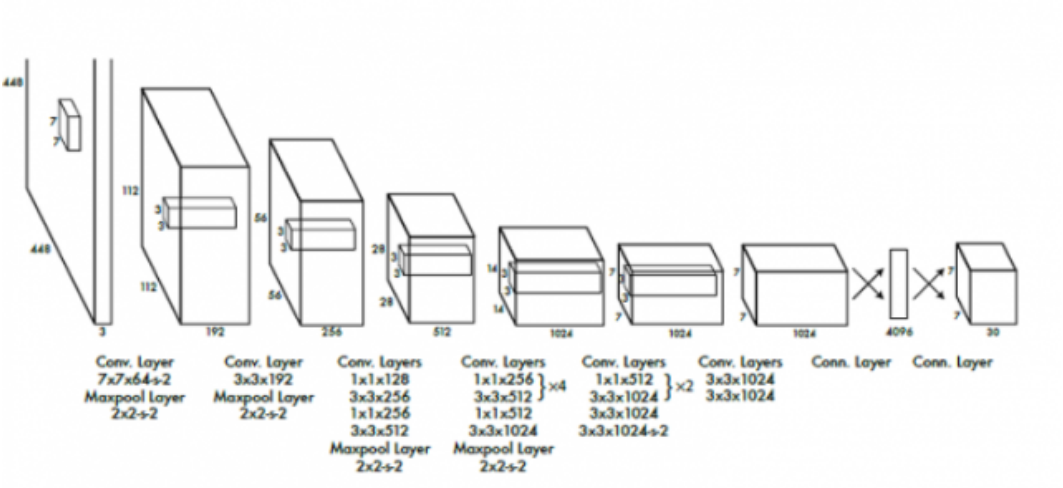


Figure 3.1: YOLO architecture

for image classification, it has 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules used by GoogLeNet, YOLO simply use 1×1 reduction layers followed by 3×3 convolutional layers.

At the end of the training, we used YOLO in such a way as to obtain the individual plants from each image containing different quantities from our dataset. Several frames are then created, one for each plant identified by the network, which are then placed in a folder and catalogued in two files, one containing the data in COCO [38] format, where all the images created are listed, while the second file contains information regarding the position (x, y) in the original image of the plant just cut out. All this has been created to facilitate the adaptation to the segmentation network and to recompose the data.

3.1.2 Instance Segmentation

The instance segmentation part is performed by BlendMask [39]. Here we use this convnet in order to segment each instance of leaves. All the leaves

found are saved into a file with the segmentation map still in COCO format. This convnet is a hybrid of top-down and bottom-up approaches, BlendMask consists of a detector network and a mask branch. The mask branch has three parts, a bottom module to predict the score maps, a top layer to predict the instance attentions, and a blender module to merge the scores with attentions.

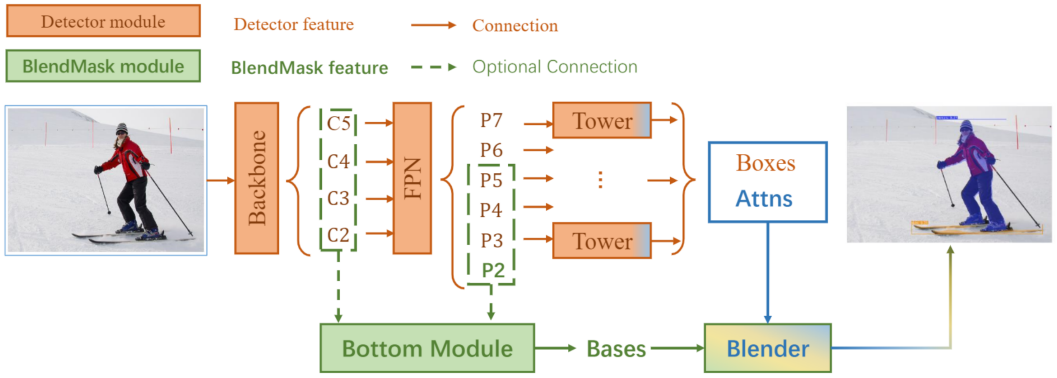


Figure 3.2: Blendmask pipeline overview

The bottom module predicting score maps which that are called bases. The input for the bottom module could be backbone features like YOLACT or Panoptic FPN. Into the top layer the authors append a single convolution layer on each of the detection towers to predict top-level attentions. Then the blender module get as input the bottom-level bases, where with a RoIPooler in Mask R-CNN crops the bases with each proposal, and the it resize the region to a fixed size. During training, the net simply use ground truth boxes as the proposals. During inference, FCOS [40] predict the results.

Thanks to this network, we achieved good results in leaf detection and segmentation, so we figured out how to implement the next section that extrapolate the information about the leaves.

3.1.3 Feature extraction

After segmentation we collect the information about the extracted leaf and derive only those with a probability of not being covered greater than 80%. Next, we save the predictions coming out of the blendmask with the information about both the leaf identifier and the name of the image to which the leaf belongs. With this, we are able to find out where the leaf is present in the image, so that in the next step we can extrapolate it from the depth image. In this way, we will finally obtain the 3D image with an rgb part and the fourth dimension, which is the depth. By means of a simple remapping we

then obtain the xyz position on the camera plane.

$$\begin{cases} Z = z/depth_scale \\ X = (x - cx) * Z/fx \\ Y = (y - cy) * Z/fy \end{cases}$$

Here we have two camera intrinsics which can be used to compute this functions. Those are related to the image which we are refferring to. cx, cy are the center of camera image while fx and fy are the focal length of the lens. Now with the xyz positions of the image we can finally calculate the various properties. The first most interesting one will be the area of the leaf for which we will rely on the pyvista [41] library, computed on the mesh surface obtained by the library itself. This method does not consider the part of the leaf that are overlapped if it is curved, so the surface area could be underestimated. For this reason we have selected only the leaves which are non overlapped and we have trained the networks with this categories of leaves. Another metrics which we have computed is the calculation of the major and minor axes, where with scikit-learn [42] metrics we have computed the centroids of the leaf and then with the orientation vector we have computed the length of the axis and the direction. We also calculated the height of the intrinsic leaf with just subtracting the highest value on the Z axis with the lowest one, and the height of the leaf from the ground, where in a first time we have computed with pyransac3d the plane of the base surface and then with the points of the leaf we have computed the relative distance from the plane and obtaining the leaf height from the ground.

3.2 Used Metrics

In this section the two evaluation techniques which are used by the referenced paper [43] the DICE score and the Symmetric Best DICE (SBD).

3.2.1 DICE Score

For semantic and instance segmentation is used the F1 score, or DICE score which is obtained by the fraction of the true positive prediction by the sum of

the true positive with false negative and false positive prediction:

$$F_1 = \frac{(2) \cdot \mathrm{TP}}{(2) \cdot \mathrm{TP} + 1 \cdot \mathrm{FN} + \mathrm{FP}}$$

which TP are the true positive in the image, FP are the false positive detected pixel and FN are the false negative found in the image.

3.2.2 SBD Score

The other metrics is the Symmetric Best Dice (SBD) [43], which is computed by the symmetric average Dice among all objects where for each input label the ground truth label yielding maximum Dice is used for averaging, to estimate average leaf segmentation accuracy:

$$BD(L^a, L^b) = \frac{1}{M} \sum_{i=1}^{M} \max_{1 \leq j \leq N} \frac{2|L_i^a \cap L_j^b|}{|L_i^a| + |L_j^b|} \tag{3.1}$$

where the $|\cdot|$ is the predicted mask area in pixel and L_i^a for $1 \leq i \leq M$ and L_j^b for $i \leq j \leq N$ are sets of leaf object segments belonging to segmentations of leaves for its corresponding. to subsequently calculate the symmetric, a minimum between predicted and gt and vice versa must be done:

$$SBD(L^{ar}, L^{gt}) = \min\{BD(L^{ar}, L^{gt}), BD(L^{gt}, L^{ar})\} \tag{3.2}$$

.

3.3 Other Networks

In order to understand which network was best for our purpose, we used different CNNs, so that we could understand which one was best suited to different types of leaves in different contexts.

3.3.1 YOLACT

The first network used was YOLACT (You Only Look At CoefficienTs) [44]. The network proposes to split instance segmentation into two simpler tasks, first it generates a non-local prototype mask dictionary for the entire image, then it predicts a linear combination of coefficients per instance. Thus, producing a full-image instance segmentation from these two components is straight-

forward: for each instance, linearly combine the prototypes using the corresponding predicted coefficients and then crop with a predicted bounding box. We show that by segmenting in this way, the network learns to locate instance masks on its own, where visually, spatially and semantically similar instances appear different in the prototype. This approach also has several practical advantages. First and foremost, it's fast: because of its parallel structure and extremely lightweight assembly process, YOLACT adds only a marginal amount of computational overhead to a one-stage backbone detector, making it easy to reach 30 fps even when using ResNet-101. Secondly, the masks are of high quality: since the masks use the full extent of the image space without any loss of quality from repooling, these masks for large objects turn out to be of significantly higher quality than those of other methods.

YOLACT is composed of different components, the backbone formed by the resnet, is the one used for the classification of the objects present in the scene. It is flanked by the Feature Pyramid Network (FPN) suitably modified to obtain a more precise classification and, where the softmax cross entropy is used to refine the prediction process.

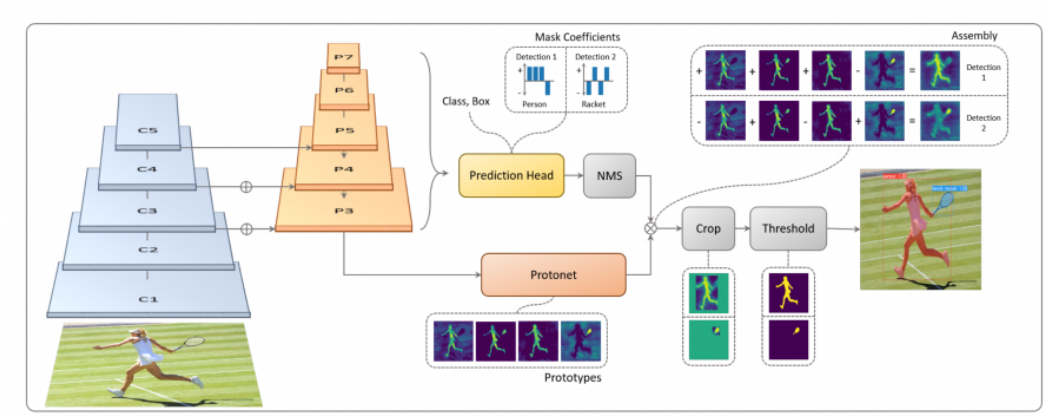


Figure 3.3: YOLACT model using ResNet-101 + FPN

The prototype generation branch (protonet) predicts a set of k prototype masks for the entire image. The authors implemented this protonet as an FCN whose last layer has k channels and attach it to a backbone feature layer. In addition, taking protonets from the deeper features in the backbone results in more robust masks, and the higher resolution prototypes produce both higher quality masks and better performance on smaller objects. Thus, we use FPN because its largest feature layers are the deepest.

For mask coefficient prediction, simply the third branch is added in parallel that predicts k mask coefficients, one corresponding to each prototype. Thus,

tanh is added to the k mask coefficients, which produces more stable outputs over no nonlinearity.

Finally mask are recombined, with prototype mask and mask coefficient branch, using a linear combination of the former with the latter as coefficients.

3.3.2 YOLACT++

A similar approach is taken with YOLACT++ [45]. It is based on the use of the deformed convolutional networks. Deformable Convolution Networks (DCNs)[7], [46] have proven to be effective for object detection, semantic segmentation, and instance segmentation due to its replacement of the rigid grid sampling used in conventional convnets with free-form sampling. YOLACT++ was designes following DCNv2, and replace the 3x3 convolution layer in each ResNet block with a 3x3 deformable convolution layer. Adding deformable convolution layers into the backbone of YOLACT, leads to a +1.8 mask mAP gain. The boost is due to DCN can strengthen the network's capability of handling instances with different scales, rotations, and aspect ratios by aligning to the target instances, YOLACT does not have a re-sampling process.

3.3.3 SOLO

Instance categories, is the quantized center locations and object sizes, which enables to Segment Objects by LOcations (SOLO) [47]. An image can be divided into squared number of cells, with the same numbers of center location classes. Each output channel is responsible for one of the central location categories and the corresponding channel map should predict the instance mask of the object belonging to that location. In essence, an instance position category approximates the position of the object center of an instance. Thus, classifying each pixel into its instance position category is equivalent to predicting the object center of each pixel in latent space.

With the SOLO framework, the network optimizes an end-to-end way for the instance segmentation task using mask annotations exclusively, and perform pixel-level instance segmentation outside the restrictions of local box detection and pixel grouping. For each grid SOLO predicts the C-dimensional output to indicate the semantic class probabilities, where C is the number of classes. The design is based on the assumption that each cell of the $S \times S$ grid must belong to one individual instance, thus only belonging to one semantic category.

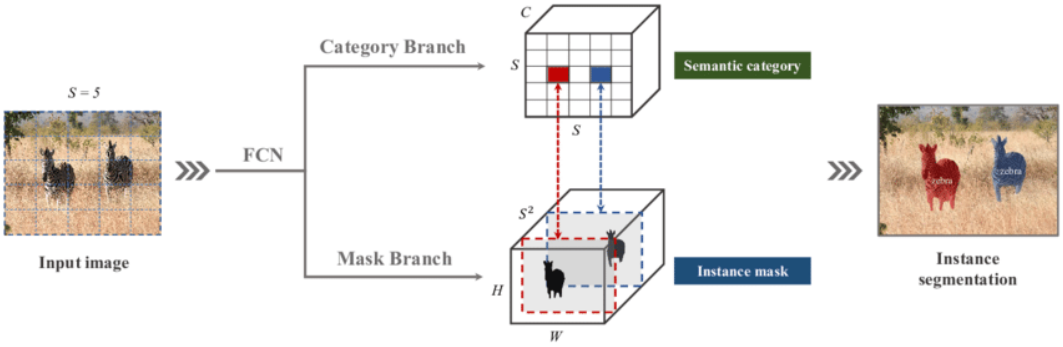


Figure 3.4: SOLO framework, semantic and mask task

In parallel with the semantic category prediction, each positive grid cell will also generate the corresponding instance mask. A straightforward approach to predicting the instance mask is to adopt fully convolutional networks, such as FCNs in semantic segmentation.

SOLO use FPN which generates a pyramid of feature maps with different sizes with a fixed number of channels for each level. These maps are used as input for each prediction head: semantic category and instance mask.

Chapter 4

Experimental Results

In plant phenotyping, several datasets were used, one being the Multi-Modality Imagery Database for Plant Phenotyping, and the Komatsuna dataset. these two datasets were chosen for leaf variety and type. Both have similar "broad" leaves. To get the most out of neural networks we tested and adapted our datasets to the most famous dataset used in object detection and image segmentation, COCO. To do this we used coco annotator [48], where through a script written by us we have adapted the annotations within the dataset in COCO format, and once imported into the annotator, we have finished and fixed the masks. This allowed us to make our dataset common to all the networks used so that we could compare the results obtained for both YOLACT and SOLO and Blendmask. In this chapter, we will describe what are the datasets used and show how we used them and the results obtained

4.1 Datasets

4.1.1 Multi-Modality Imagery Database

Multi-Modality Imagery Database is provided by Michigan State University. It contains two different types of plants, that of the bean and Arabidopsis thaliana. The images were taken at different times of the day at regular intervals. Regarding Arabidopsis the dataset is composed of a total of 2160 acquisitions with only 576 annotated images with a size of 116×119, each acquisition is composed of four images one RGB, one depth, one infrared and one fluorescence. The bean images instead is composed of 325 images of which 175 annotated with a dimension of 481×491, but unlike the bean as annotated images are reported only those fluorescent, then they will be inserted

these within the network for training.

(a) Arabidopsis image from MMI database

(b) Bean image from MMI database

Figure 4.1: Example of Multi-Modality Imagery Database images for training and validation

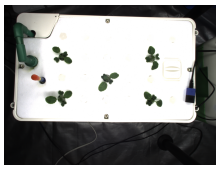
The total of the 751 images are divided into 2 sets with a split ratio of 60% 40%, the training set with a total of 451 images and the validation set with a total of 300 images. This can be performed by a script which randomly split the entire dataset into two or three sets and then recombine them into two or three files which contains training, validation and optionally test set of images.

4.1.2 Komatsuna Dataset

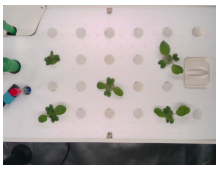
Komatsuna, is a Japanese leaf vegetable, it belongs to the Brassica rapa family. This dataset consist of a 3d setup with two different kind of images. The 3D images are divided into whole images containing the entire set of plants linked to a 3d image, these were collected on different days every four hours. These images, both RGB and depth images, are 640×480 in size, which will later be cropped according to what the network, in our solution, considers most appropriate. These images have been placed at a distance of about 50cm from the plane, and have the following intrinsic matrix:

$$\begin{pmatrix} 382.641 & 0 & 223.248 \\ 0 & 382.641 & 233.523 \\ 0 & 0 & 1 \end{pmatrix}$$

We used these images as test images to validate the network trained specifically to recognize komatsuna leaves, and through the depth images and the intrinsic matrix we sized the found leaves.



(a) Training image for Komatsuna dataset



(b) Test image for Komatsuna dataset

Figure 4.2: Komatsuna Dataset training, test images

The other images, concerning the multiview RGB, start from a higher dimension of 2048×1536, without having the depth images. These have been divided plant by plant by the authors obtaining a total of 900 images for the dataset with dimension 480×480. to improve even more the model obtained from the neural network we have increased the dataset up to 4500 images, through the different techniques of rotation flip and blur, where we used 2700 images for training and 900 for validation, finally to test the metrics we used the remaining 900 images.

4.1.3 Our Dataset

In our dataset we collect the most of images in a single day at a local farm, consisting of a set of images of basil seedlings. Some plants were subsequently collected from which a second set of images was extrapolated over the next five days. The first set designed for training has a total of 128 images of size 640×480 containing a total of 892 seedlings. All of these images have appropriately aligned depth images attached to them, so that 3D networks can also be trained. The second set designed for testing consists of a set of images divided by days for a total of five days with 20 daily images. Their size is 1280×720 with the depth images also aligned. Each image consists of four seedlings rotated and swapped positions at each acquisition. On the last day, the dimensional information of the leaves that are fully visible not covered by other leaves was also included.

(a) training image for our dataset

(b) Test image for our dataset

Figure 4.3: Our Dataset training, test images

To be able then to do the dimensional calculation we made use of the intrinsic matrix provided by the camera used, an Intel RealSense D435:

$$\begin{pmatrix} 637.735 & 0 & 645.414 \\ 0 & 637.735 & 349.205 \\ 0 & 0 & 1 \end{pmatrix}$$

All images were acquired at a ground clearance of 92cm under varying but high light conditions. Placing a cloth in the background allowed us to adapt

the network trained via yolo for object detection to our dataset as well in order to automate the calculation of measurement validation.

4.2 State of The art Comparison

Chapter 5

Conclusion

Bibliography

[1] L. C. Jiang Yu, "Convolutional neural networks for image-based high-throughput plant phenotyping: A review," *Plant Phenomics*, vol. 2020, 2020.

[2] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and cooperation in neural nets*, pp. 267–285, Springer, 1982.

[3] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[4] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[7] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," 2017.

[8] S. Mardanisamani, F. Maleki, S. Hosseinzadeh Kassani, S. Rajapaksa, H. Duddu, M. Wang, S. Shirtliffe, S. Ryu, A. Josuttes, T. Zhang, *et al.*, "Crop lodging prediction from uav-acquired images of wheat and canola using a dcnn augmented with handcrafted texture features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

[9] N. Teimouri, M. Dyrmann, P. R. Nielsen, S. K. Mathiassen, G. J. Somerville, and R. N. Jørgensen, "Weed growth stage estimator using deep convolutional neural networks," *Sensors*, vol. 18, no. 5, 2018.

[10] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[12] S. Bhugra, K. Garg, S. Chaudhury, and B. Lall, "A hierarchical framework for leaf instance segmentation: Application to plant phenotyping," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 10173–10179, 2021.

[13] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[14] B. Lyu, S. D. Smith, and K. A. Cherkauer, "Fine-grained recognition in high-throughput phenotyping," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 320–329, June 2020.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[16] H. Nazki, S. Yoon, A. Fuentes, and D. S. Park, "Unsupervised image translation using adversarial networks for improved plant disease recognition," *Computers and Electronics in Agriculture*, vol. 168, p. 105117, 2020.

[17] S. Srivastava, S. Bhugra, B. Lall, and S. Chaudhury, "Drought stress classification using 3d plant models," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 2046–2054, 2017.

[18] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3d," in *ACM siggraph 2006 papers*, pp. 835–846, 2006.

[19] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," in *European conference on computer vision*, pp. 467–483, Springer, 2016.

[20] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2, pp. 1150–1157, Ieee, 1999.

[21] A. Garcia-Garcia, F. Gomez-Donoso, J. Garcia-Rodriguez, S. Orts-Escolano, M. Cazorla, and J. Azorin-Lopez, "Pointnet: A 3d convolutional neural network for real-time object class recognition," in *2016 International joint conference on neural networks (IJCNN)*, pp. 1578–1584, IEEE, 2016.

[22] S. Kolhar and J. Jagtap, "Convolutional neural network based encoder-decoder architectures for semantic segmentation of plants," *Ecological Informatics*, vol. 64, p. 101373, 2021.

[23] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[24] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[25] X. Liu, C. Hu, and P. Li, "Automatic segmentation of overlapped poplar seedling leaves combining mask r-cnn and dbscan," *Computers and Electronics in Agriculture*, vol. 178, p. 105753, 2020.

[26] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[27] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[28] H. Lu, Z. Cao, Y. Xiao, B. Zhuang, and C. Shen, "Tasselnet: counting maize tassels in the wild via local counts regression network," *Plant methods*, vol. 13, no. 1, pp. 1–17, 2017.

[29] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[30] S. Aich and I. Stavness, "Leaf counting with deep convolutional and deconvolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[32] H. Uchiyama, S. Sakurai, M. Mishima, D. Arita, T. Okayasu, A. Shimada, and R.-i. Taniguchi, "An easy-to-setup 3d phenotyping platform for komatsuna dataset," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[33] J. A. Cruz, X. Yin, X. Liu, S. M. Imran, D. D. Morris, D. M. Kramer, and J. Chen, "Multi-modality imagery database for plant phenotyping," *Machine Vision and Applications*, vol. 27, no. 5, pp. 735–749, 2016.

[34] M. Minervini, A. Fischbach, H. Scharr, and S. A. Tsaftaris, "Finely-grained annotated datasets for image-based plant phenotyping," *Pattern Recognition Letters*, vol. 81, pp. 80–89, 2016.

[35] M. Pound, J. Atkinson, D. Wells, T. Pridmore, and A. French, "Deep learning for multi-task plant phenotyping," 10 2017.

[36] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[38] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014.

[39] H. Chen, K. Sun, Z. Tian, C. Shen, Y. Huang, and Y. Yan, "Blendmask: Top-down meets bottom-up for instance segmentation," 2020.

[40] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9627–9636, 2019.

[41] C. B. Sullivan and A. Kaszynski, "PyVista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (VTK)," *Journal of Open Source Software*, vol. 4, p. 1450, may 2019.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[43] H. Scharr, M. Minervini, A. P. French, C. Klukas, D. M. Kramer, X. Liu, I. Luengo, J.-M. Pape, G. Polder, D. Vukadinovic, *et al.*, "Leaf segmentation in plant phenotyping: a collation study," *Machine vision and applications*, vol. 27, no. 4, pp. 585–606, 2016.

[44] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: Real-time instance segmentation," 2019.

[45] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact++: Better real-time instance segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1–1, 2020.

[46] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convnets v2: More deformable, better results," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9308–9316, 2019.

[47] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "SOLOv2: Dynamic and fast instance segmentation," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[48] J. Brooks, "COCO Annotator." `https://github.com/jsbroks/coco-annotator/`, 2019.