Plant Detection for Gardening Robot

Yuliang Zhu

Department of EECS, University of Michigan
yuliangz@umich.edu
Instructor: Prof. Dmitry Berenson
Dated: 12/18/2020

I. Abstract

Weeds can be harmful to the crops, inducing huge economic costs and irreversible ecological damages. Thus removing the weeds with automation will reduce the demands for labor in the agriculture and save the resources which can be used to improve other aspects as such crop quality. This project basically fine tunes the instance segmentation model Mask RCNN on a customized plant dataset to differentiate weeds from crops.

II. Introduction

Weeds, also known as ”invasive non-native plants”, out-compete native plants for space, nutrients and sunlight. These weed invasions can change the natural diversity and jeopardize the balance of ecological communities. In particular, weeds affect the health and sustainability of agricultural products. It is estimated by the Australian government that Australian farmers spend 1.5 billion dollars on weed control activities and suffer a loss of 2.5 billion dollars of agricultural production each year. Additionally, weeds can cause human healthy problems such as skin irritations and asthma. So weed control is vital for environmental, agricultural and human healthy purposes. In this project a pipeline is developed which takes as input a customized plant dataset, trains a Convolutional Neural Network (CNN), and produces as output a detection prediction of the weeds and crops.

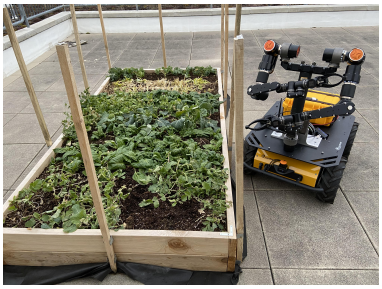


Fig. 1: Robot and Garden

In the current sate of weed detection system, a powerful and flexible classifier of soil, weeds and crops is still the greatest challenge, and another important step is to commercialize the weeding robot. Hyperspectral images provide highly accurate maps but is not profitable [1]. Thus the classifier on normal images is extremely important which allows the robots to perceive the environment. In [2], several state-of-art object detection models are compared and the Faster RCNN [3] is proved to be the best choice. Mask RCNN [4] integrates the Faster RCNN, and an extra mask prediction in parallel, so that the model can make mask predictions in addition to bounding boxes. A lot of related research works concentrate on comparing the performances of different detection models, so this project mainly implements and realizes the functionality of weed detection through fine tuning the Mask RCNN model to pave the way for future work. Another feature of this work is that a customized dataset (number of samples, species, spatial layout etc.) can be generated for the training process.

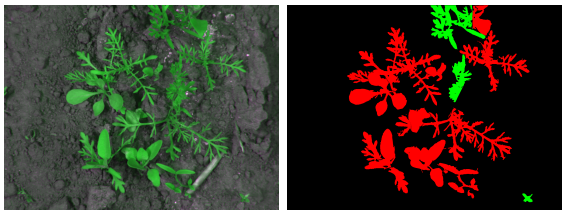
III. Methods

This section mainly talks about the model architectures, data collection and tuning training dynamics.

A. Instance Segmentation

This project is mainly developed based on the "finetune Mask RCNN" tutorial published by the Pytorch community [5]. Mask RCNN is built on top of Faster RCNN by adding an extra module for predicting segmentation masks [4], while Faster RCNN is capable of predicting bounding boxes in each RoI (Region of Interest) using RPN (Region Proposal Network) [3]. RPN is capable of predicting regions that are very likely to contain the objects of interests [6]. These deep learning models (also called "Artificial Neural Networks") are actually a huge computational graph for high-dimensional matrices. The most commonly used components for these computational structures are convolutional layers, fully-connected layers, activation layers such as ReLU, sigmoid, and batch-normalizations [7]. The convolutional layer uses small kernels to slide through the image and this idea of "shared parameters"

allows it to look for the parts of the image that are most positively correlated to the kernel, thus extracting useful feature representations. The fully-connected layers are typically used for connecting the latent space and the predicted scores. With them, we can easily adjust the output dimensions, but they would easily consume a huge amount of memory and computational resources.



(a) Source Image (b) Mask Image

Fig. 2: CWFID Sample Data

Aside from these two kinds of layers, the activation layers are invented to mimic the biological structure of human brains, and they usually are able to add to the non-linearity of the model. Until now, there are still no clear theory to rigorously prove the mechanism of deep neural networks. The intuition behind their learning ability is that with iterative optimization, the composition of nonlinear functions can stochastically "learn" a complex decision boundary in a very high dimensional space. Even though the learning capability of such models in theory is really strong, the prerequisites for them to succeed are really strict. Only when we have a clean, varied and large labeled dataset and learning settings are appropriate, the models will get a good performance [8]. From an application perspective, working on the data distribution might have larger marginal benefits on the model performance compared to model structure. However, this way of learning heavily relies on the data and does not generalize well. More work needs to be done to fundamentally change this learning mechanism to further improve models' ability to generalize.

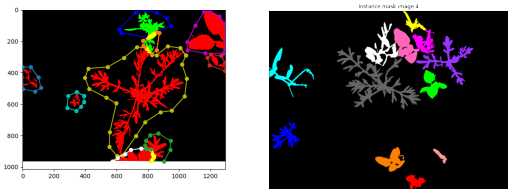
B. Data Collection

There are few open source crop-weed datasets which are specifically designed for instance segmentation. The Crop/Weed Field Image Dataset published by Huag *et al.* proposes a benchmark dataset for crop and weed discrimination, and it contains plant segmentation and manual annotations of the plant types [9]. However, CWFID has a limited size of 60. There are other weed datset such as DeepWeeds [10], which only has the classification label and thus is not suitable for this project.

To train Mask RCNN using the API provided in the tutorial [5], specific data format must be provided. A data API should be created that takes as input the image index, and produce as output the source image and targeted labels including bounding boxes, masks, categories, etc. The way the instance segmentation tasks are different from the semantic segmentation tasks is that it differentiate between each individual within the same category. The data format provided by CWFID is a semantic segmentation plus an annotation specifying the polygon boundary of each instance. Therefore the data conversion can be implemented by deciding which polygon (instance) each pixel belongs to. A simple and elegant algorithm can solve this problem as mentioned in this tutorial [11] without much computation. The following is an algorithm that completes the data format conversion.

Algorithm 1: Mask Conversion

Data: mask image M, yaml annotation Y
Result: converted maskout, labels L, boxes B
initialization: gather polygon set S from Y, get B from S, and make out a copy of M;
for *pixel* $p(i,j)$ *in* M **do**
  if *p is red or p is green* **then**
    instance_idx = check_polygon(p);
    $out[p]$ = instance_idx;
    if *type(*$S[instance\_idx]$ = *'Plant')* **then**
      $L[instance\_idx] = 1$
    if *type(*$S[instance\_idx]$ = *'Weed'* **then**
      $L[instance\_idx] = 2$
return out, L, B



(a) Instance Separation (b) Instance Mask

Fig. 3: Data Conversion

As a result, the semantic masks are converted into instance masks, while the overlap pixels are label with yellow colors in the separation result and are left unlabeled to avoid ambiguity.

Since the CWFID only has a limited number of samples, and the samples really do not help the model generalize to varied test cases, it is critical to be able to efficiently collect our customized dataset, which ideally covers the data distribution of the test cases, and has a large volume so that our model will not overfit on a

limited amount of data. [12] gives a clear instruction to complete such customized dataset generation.

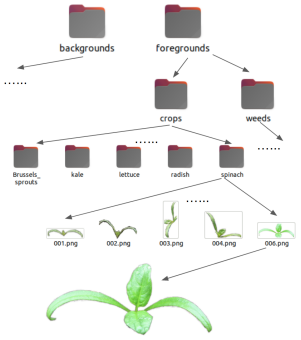


Fig. 4: Data Augmentation

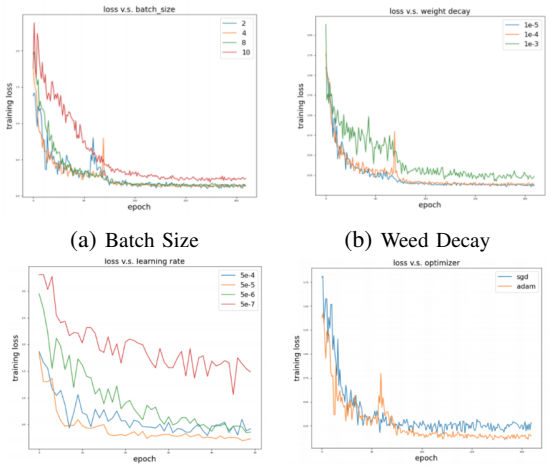
Hyperparameter	Value / Type
batch_size	4
optimizer	Adam
learning_rate	5e-3 5e-5
betas (for adam)	0.9, 0.999
eps	1e-8
weight_decay	5e-4
epoch	210
Lr scheduler	0.1 per 70 epochs
train_val_split	True

TABLE I: Hyperparameter table for training

The basic idea is to cut out the foreground images of interests, and then stochastically paste them on background images (soil). In this process, we have control of the number of generated training samples, the species, the definition of weeds, the spatial layout of the synthesized images, etc.

C. Hyperparameters Tuning

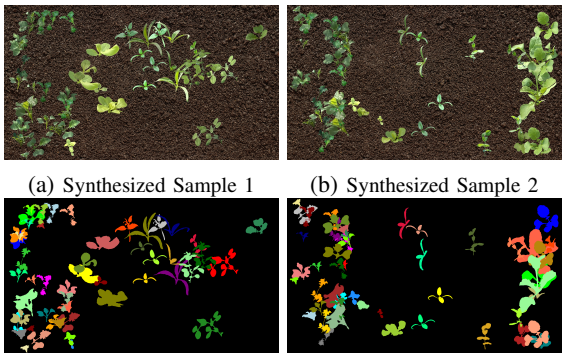
Training settings are important for the model performance. For example, either a too large or too small learning rate for the optimization step would cause the model to diverge or not to make any significant progress. Other hyperparameters also matter. Adam is usually default choice for the optimizer as it performs bias correction for the initialization step and also incorporates momentum terms that both accelerates the optimization and also rescale the learning rate adaptively. An appropriate batch size can accelerate the training and help the model generalize better. Weight decay is essentially a weight for the regularizer, which helps avoid overfitting. A learning rate scheduler would help the model to further reduce the training loss in the middle or late stage of training. An empirical set of hyperparameters are settled after experiments.



(a) Batch Size (b) Weed Decay

(c) Learning Rate (d) Optimizer

Fig. 5: Data Conversion



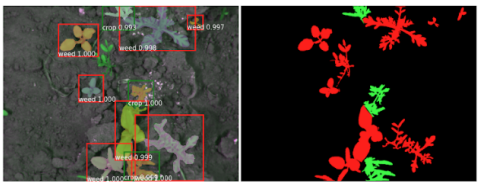
(a) Synthesized Sample 1 (b) Synthesized Sample 2

(c) Synthesized Mask 1 (d) Synthesized Mask 2

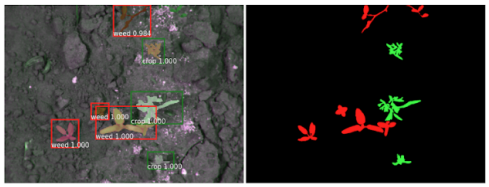
Fig. 6: Data Augmentation

IV. Results

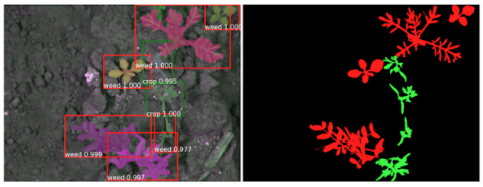
Right now decent detection results on be obtained using the above mentioned pipeline. The predictions made from the model trained on the CWFID in the figure 7. To test if the model trained on CWFID can generalize to other test cases, real garden images are fed into the model, and the predictions are shown in the figure 9. Using the above training setting, we can get an average precision of 0.608 for IoU of 0.50 in terms of bounding box, an average precision of 0.550 for IoU of 0.50 in terms of segmentation. For larger IoU, e.g. 0.75, the average precisions are relatively lower, which are 0.342 and 0.308 respectively.



(a) Prediction 1

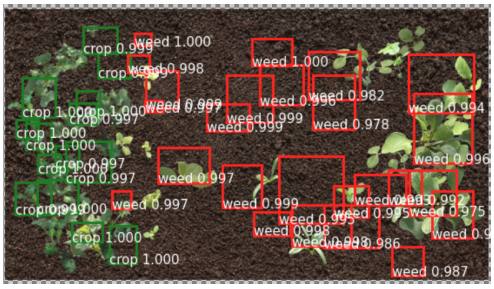


(b) Prediction 2

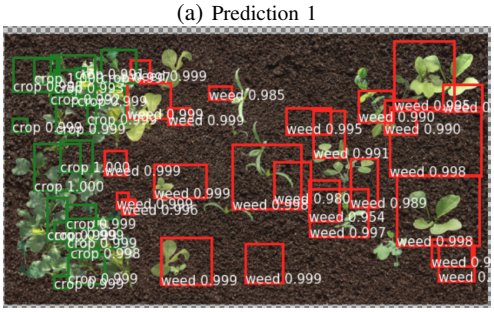


(c) Prediction 3

Fig. 7: Bounding Box and Mask Prediction (left: prediction, right:GT)



(a) Prediction 1



(b) Prediction 2

Fig. 8: Prediction on the customized dataset, showing the trend of separating weeds from crops (the left column, given that the crops are intentionally put in the left column)

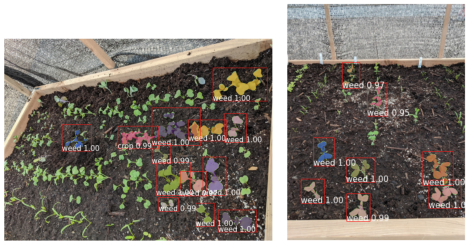
V. Discussion

A. Model and Data

As in the figure 9, the model can predict a clear and precise mask for only some of the plants, and the prediction works best when the plants are far separated from each other, and when the plants have the top-down orientation as in the training set. As in the figure 8, This is the prediction result I got after training the Mask RCNN for 20 epochs on the customized dataset (160 training samples and 40 test samples) due to limited computational resources. Even though the training is short and insufficient and the model is not doing well predicting the plant boundaries, we can still observe the trend that the model is able to differentiate weeds from plants, given the fact that I intentionally locate the kales on the left column and define them as "crops". With more computational resources, training iterations and more clearly manipulated data, this networks will surely perform better.

When it comes to further improve the model performance on the test set (real garden images), the optimization problems fall on two subjects - model structure and the training data. From the application perspective, collecting more data that would cover the distribution of the potential test cases will be an effective way, but the brute-force nature of this approach requires huge effort and it is usually very expensive. From the theoretic perspective, the current deep learning technologies usually reply heavily on the quality and quantity of the training data, and do not generalize well on the cases that are very different from the samples that the models has seen. Throughout the development of deep learning models, there are already many really mature models that are good at specific tasks. However,they learn in a fundamentally different way from the way human learns. Humans have strong capabilities to apply the learned knowledge to new stuff and extent them to related areas.

To beat the heavy dependency on the data, there are works trying to achieve good performance with very few training samples. However this will still require a lot of effort to label data. A new concept of self-supervised learning has been studied, such as image inpainting tasks [13] and contrastive learning tasks [14].



(a) Prediction 1 (b) Prediction 2

Fig. 9: Prediction on the real images

B. Detection Challenges

Typical challenges for instance detection are scale variation, viewpoint variation, intra-class variation, background clutter, illumination changes, deformation and occlusion. One of the most difficult challenges among these is the occlusion. During the early stage of plant growth, they are sparsely distributed and this is relatively easier for classifiers to detect the weeds. Once the plants become bigger, their leaves are occluded by each other and it is impossible for the model to tell apart every instances, even human have a hard time finishing this task simply by vision. There might be alternative problem statement that can avoid tackling occlusion challenges. We might try to track the root location of each leaf instead of looking for the entire plant. As is the case of many weeding systems, quadcopters are used for imaging the large field to detect weeds. I such cases, detection targets can be an large area instead of single plants.

C. Weeds Localization

Even if we could get a precise prediction of bounding boxes and masks, it remains challenging to find the grasp point for the weeding robots, as it would require 3D information of the scene. A naive approach would be to train a classifier to predict the grasp point in the image and use the RGB-D camera sensor to get the depth information of the pixel labeled as weeds. Another method to acquire the 3D data is to use a 3D lidar to get point clouds. There are works related to lidar-based detection algorithms. However 3D lidars are expensive, and the 3D inference algorithms are usually computationally expensive. Another approach would be to perform a scene construction using multiple camera views, or we can use one camera to slide and capture continuous frames from different angles.

VI. Conclusion

The project focuses on the implementation of transfer learning based on the Mask RCNN model to differentiate weeds from crops, and tries to explore how to improve the detector performance from an application perspective. This work is more into the details instead of being experimental. This work gives an insight about what to expect from the state-of-art instance segmentation models and the prerequisites for it to succeed. However, this work is limited in that it does not put forward any novel, practical and concrete solutions to the above mentioned bottlenecks.

To further enhance the accuracy, two approaches can be considered: one is working on collecting good training set, which is the labor-intense and unclever way, and the other is to come up with learning mechanism that can quickly converge and generalize way better. As for the occlusion challenge, an alternative problem statement should be formulated to bypass the difficulty to detect occlusion. Finally, a computationally inexpensive and accurate way should be established to inference the grasp point for the weeding robot.

This project is a naive exploration to use deep neural networks to tackle weed detection problems. It to some extent reveals the capabilities as well as the limitations of the deep neural networks.

Acknowledgment

I would like to thank Prof. Dmitry, the members in the ARM Lab for a great research experience despite the difficult circumstances presented to us all.

References

[1] F. LÓPEZ-GRANADOS, "Weed detection for site-specific weed management: mapping and real-time approaches," *Weed Research*, vol. 51, no. 1, pp. 1–11, 2011.

[2] A. N. V. Veeranampalayam Sivakumar, J. Li, S. Scott, E. Psota, A. J Jhala, J. D. Luck, and Y. Shi, "Comparison of object detection and patch-based classification deep learning models on mid-to late-season weed detection in uav imagery," *Remote Sensing*, vol. 12, no. 13, p. 2136, 2020.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

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

[5]

[6] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8971–8980.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[8] [Online]. Available: http://karpathy.github.io/2019/04/25/recipe/

[9] S. Haug and J. Ostermann, "A crop/weed field image dataset for the evaluation of computer vision based precision agriculture tasks," in *European Conference on Computer Vision*. Springer, 2014, pp. 105–116.

[10] A. Olsen, D. A. Konovalov, B. Philippa, P. Ridd, J. C. Wood, J. Johns, W. Banks, B. Girgenti, O. Kenny, J. Whinney *et al.*, "Deepweeds: A multiclass weed species image dataset for deep learning," *Scientific reports*, vol. 9, no. 1, pp. 1–12, 2019.

[11]

[12] akTwelve, "aktwelve/cocosynth." [Online]. Available: https://github.com/akTwelve/cocosynth

[13] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.

[14] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[15] [Online]. Available: http://karpathy.github.io/2019/04/25/recipe/