# Deep Learning-based Object Detection for Crop Monitoring in Soybean Fields

Muhammad Taufiq Pratama
*Graduate School of Engineering*
*Kobe University*
Kobe, Japan
pratama@stu.kobe-u.ac.jp

Sangwook Kim
*Graduate School of Engineering*
*Kobe University*
Kobe, Japan
kim@eedept.kobe-u.ac.jp

Seiichi Ozawa
*Center for Mathematical and Data Sciences*
*Kobe University*
Kobe, Japan
ozawasei@kobe-u.ac.jp

Takenao Ohkawa
*Graduate School of System Informatics*
*Kobe University*
Kobe, Japan
ohkawa@kobe-u.ac.jp

Yuya Chona, Hiroyuki Tsuji, Noriyuki Murakami
*Hokkaido Agricultural Research Center*
*NARO*
Sapporo, Japan
{y.chonan, tuzihiro, noriyuki}@affrc.go.jp

***Abstract*—In this paper, a soybean flower/seedpod detection system is built for collecting growing state information by introducing convolutional neural networks, aiming that observed plant states (e.g., #flowers and #seedpods) are used to predict the crop yields of soybeans by combining the environment information in future. To predict the crop yields (i.e., quantity of seedpods) precisely, it is considered important to know how the number of flowers are translated over time and how such flower transients can affect the final yields of soybeans. However, there has not existed a way to measure the number of flowers in real environments. For this purpose, We propose a deep learning approach to automatically detect flower and seedpod regions from images which are taken in real soybean fields without environmental control. Various object detection methods are compared, including RetinaNet, Faster R-CNN, and Cascade R-CNN. Ablation studies are provided to analyze how these methods perform on both flower and seedpod across different parameters. In our experimental results, Cascade R-CNN gives the best average precision (AP) of 89.6, while RetinaNet and Faster R-CNN give AP of 83.3 and 88.7, respectively. Cascade R-CNN also attains the highest accuracy in detecting small objects, which are not easily detected by other models. With accurate detection, the system is expected to contribute to constructing high-performance measurement for soybean flowers and seedpods, which ultimately leads to better pipeline in evaluating plant status.**

***Index Terms*—precision agriculture, object detection, deep learning, crop monitoring**

## I. Introduction

The ever-growing global demand for soybean is not followed by the increase of the annual growth rate of soybean crops as the population of skilled farmers keeps decreasing. To address this issue, precision agriculture and automation in the farming process began to be adopted with the ultimate goal of increasing the production of this highly demanded food resource. We start this effort by collecting various information from sensors, including cameras, to gather information about crop growth in understanding which kind of environment contributes to higher productivity.

Fig. 1: Dense space between soybean plants make data collection and crop monitoring become non-trivial effort.

One main factor that we observe is the number of growing flowers and seedpods on the plant over its growth period. Since manual measurement would be costly, we aim to automate this process using an object detection model based on computer vision techniques. Data collection is conducted on actual soybean fields without any environmental control, so getting an accurate quantification would not be an easy feat as it comes with multiple obstacles. The detection model needs to take into account the illumination variance and object occlusion that occurs frequently. In addition, physical limitations such as dense placement of plants as illustrated in Figure 1 makes taking the full image of each plant in one take infeasible. To address the issues, we took video of each plant and cut them into several frames. However, as a drawback, this results in unintended aftereffects such as motion blur and overlapping scene between one frame and another.

We adopted deep-learning-based object detection models in this work for their well-known higher performance compared to traditional computer vision techniques. Two-stage object

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

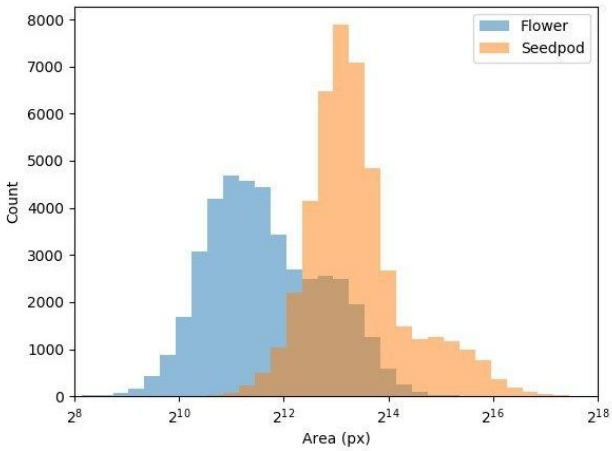



Fig. 2: Data distribution of bounding box area in the whole dataset, represented in pixel.

(a) Flower sample (b) Seedpod sample

Fig. 3: Sample data of both flower and seedpod. Flower object has smaller bounding boxes in average, compared to the seedpod.

detection models such as Faster R-CNN [9] and Cascade R-CNN [15] especially have high detection accuracy on difficult datasets like MS COCO [16]. We created our own dataset based on the images we have collected on the field. The dataset contains two class labels, *flower* and *seedpod*. The dataset functions as a means of benchmarking in considering which detection model suits our case.

The size distribution of bounding boxes is presented in Figure 2. Large objects are relatively rare in flower class, and smaller objects occurred more frequently. In complement with this, seedpod class lacks smaller object, and larger object occurs more instead. We can see the comparison example between the two in Figure 3. Thus, we would like to confirm if deep learning-based detection models can perform well on such distributions as well. To make the comparison more general, we include not only two-stage object detection models such as Faster R-CNN and Cascade R-CNN, but also one-stage detection models like RetinaNet in our study to figure out how much the result may differ.

## II. Related Work

The pursuit of agricultural productivity, especially on soybean plant, has been featured on prior works, though with different objectives and methods from us. Simple Linear Iterative Clustering (SLIC) superpixels segmentation were used by [1] in complement with CNN classifier to detect weeds that possibly hinder the growth of soybean crops. [2] also used the method to identify soybean leaf diseases. While object detector based on SLIC has its own advantage of being fast, it often has inferior accuracy compared to detectors derived from R-CNN model [3].

Deep-learning-based object detectors have been widely used on other crops. [4] adopted Faster R-CNN model to detect fruits in the orchard field. We will show the performance of Faster R-CNN on our dataset as well as the base comparison. [5] compared the performance of R-CNN, Fast R-CNN, and Faster R-CNN models in detecting strawberry flowers in the outdoor field. The challenges in detecting objects outdoor such as illumination variance are similar to our case, though the data gathering process in our case may relatively more complex due to the dense planting of soybeans. [6] applied a modified YOLOv3 model for apple detection in orchards. It is one of very few works that utilized one-stage detection in such case. Recent progress of one-stage detection is even claimed to be able to par with two-stage object detection models in accuracy while having faster inference speed.

The rapid development of deep-learning-based object detection models arguably has its root from the appearance of Region-based Convolutional Neural Network (R-CNN) [7]. R-CNN appends a region proposal stage to the CNN object recognition pipeline as an object localization mechanism. It proposes a specific location of objects in an image, which later classified by the object recognition stage. Such kind of model is known as the two-stage model, as it consists of region proposal stage and recognition stage.

Numerous improvements to the R-CNN detector have been proposed, resulting in better-performed models such as Fast R-CNN [8] and Faster R-CNN [9]. State-of-the-art model such as Cascade R-CNN [15] even includes multiple region proposal stages in its pipeline to further boost the detection performance. The needs of a more efficient object detection pipeline also give birth in the one-stage model, beginning from YOLO [10] [11] [12], Single Shot Detector (SSD) [13], until the appearance of RetinaNet with its competitive accuracy compared to the two-stage detector while maintaining low latency on MS COCO dataset [14].

## III. Methods

### A. Data Collection

We collected the soybean plant image data from soybean plantation fields across several regions in Japan, using a GoPro Hero 7 camera in natural daylight. Soybeans are often planted

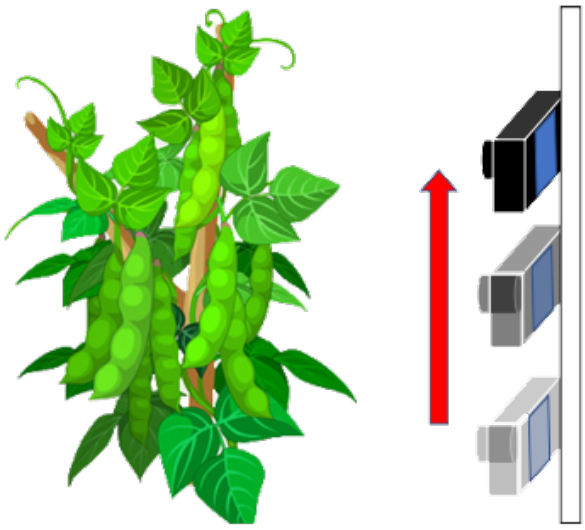


Fig. 4: Illustration of data collection process by capturing video of each soybean plant in particular distance.

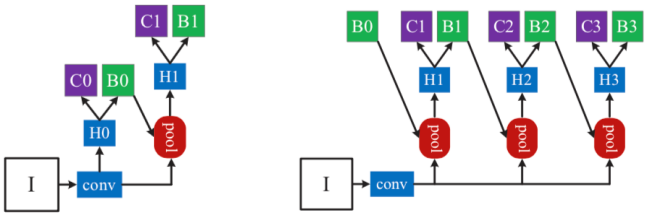
with a dense space between plants, so there are limitations in the distance we can take between the camera and the plant. Thus, it is challenging to make the camera capture the whole picture of a soybean plant in one take. Taking aerial picture like [1] [2] is also not an optimal solution in our case, since it would be harder to capture the whole image of flowers and seedpods we want to detect.

As an alternative, we took video of each plant by sliding the camera from bottom to top of the plant in an acceptable distance [21]. This process is illustrated in Figure 4. We conducted this process once every few days, starting from when the soybean starts to grow flower until the full seed period. This way, we can quantify the change of number in flower and seedpod within each growth period.

Images were generated from the videos by cutting them into several frames. Resulting images were hand-annotated using labelImg annotation software [20]. We annotated every flower and seedpod visible in the image regardless of which plant these objects belong to. While our primary purpose is calculating the number of flowers and seedpods within each plant, there is a difficulty in differentiating which object belongs to which plant because of the limited distance between each plant.

### *B. Object Detection*

Faster R-CNN is one of the state-of-the-art object detection methods that has undergone several modifications. It was originally introduced as an improvement to R-CNN model by adding the Region Proposal Network (RPN) as a mechanism to reduce the number of region proposals while increasing the proposal quality. Later on, Feature Pyramid Network (FPN) [18] was introduced as a solution in detecting object across different sizes by utilizing multiple feature scales. FPN is also adopted in RetinaNet in spite of the difference in




(a) Faster R-CNN (b) Cascade R-CNN

Fig. 5: Architecture difference between Faster R-CNN and Cascade R-CNN, as explained in [15]. ”I” represents input image, ”conv” is backbone convolution, ”pool” is region-wise feature extraction, ”H” is network head, ”B” is bounding box, and ”C” is classification.

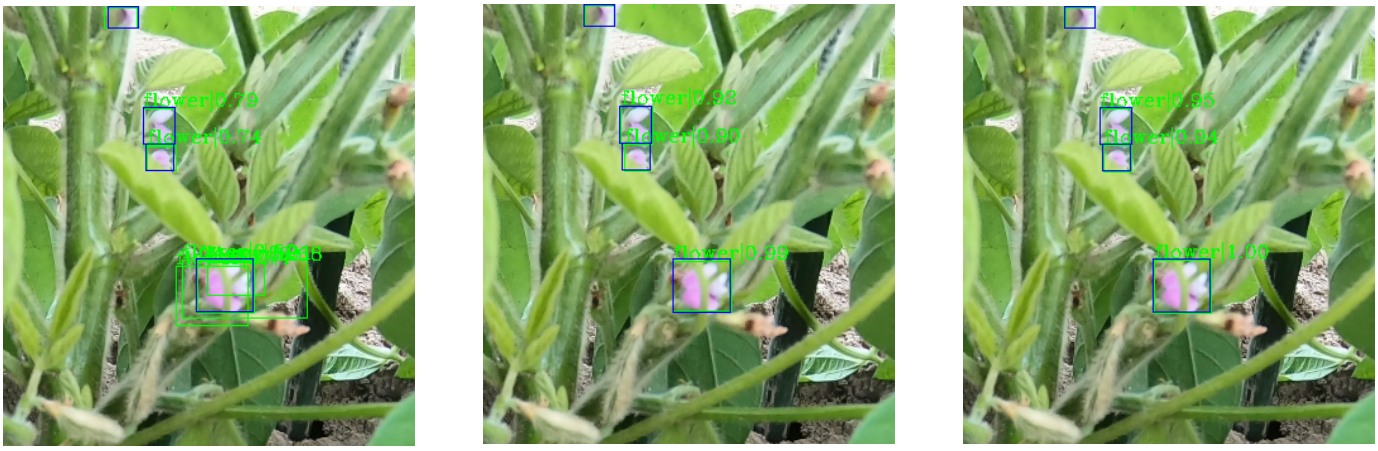
approach, in which RetinaNet directly applies detection to the image without region proposal stage. In addition with focal loss proposed by the original paper, its detection accuracy can compete with Faster R-CNN with less overhead in the architecture.

Object detection model such as Faster R-CNN usually choose exactly one IoU threshold and stick with it in the whole pipeline. This may result in a model that only good on particular IoU. As an example, detectors trained on lower IoU may results in diversified bounding boxes, but noisy detection would appear often. In opposite, detectors trained on higher IoU, while only output few detections because of its strict threshold, mostly left true positive bounding boxes. Cascade R-CNN addresses this issue by appending additional region proposal stages. Moreover, it utilizes different IoU thresholds on each stage with an increasing value (e.g. 0.5; 0.6; 0.7), which acts as a resampling mechanism. The architecture difference between Faster R-CNN and Cascade R-CNN is illustrated in Figure 5. Cascade R-CNN has multiple ”network heads” that function as an additional region-of-interest detector with a particular IoU threshold being set.

The actual process can be seen in Figure 6. Lower stage with a lower IoU threshold can detect many ‘rough’ bounding boxes. There are two possibilities on how the proposed bounding boxes is handled within particular stage. Bounding boxes that pass the IoU threshold on the current stage will be passed to the next stage to be refined further. On the other hand, false positive bounding box that does not pass the threshold will be ignored. The repeating process results in a better quality of bounding boxes on each stage iteration.

### *C. Object Counting Mechanism*

Object counts across frames are aggregated to get the final quantity of objects in each video. As a note, this should be done in a way such that the overlapping scenes will not be counted twice. Though this process is not the focus of our current work, the mentioned condition can be fulfilled by adopting object tracking techniques, which can be done in several approaches, such as correlation filter-based methods [17].



(a) $1^{st}$ stage with IoU=0.5 (b) $2^{nd}$ stage with IoU=0.6 (c) $3^{rd}$ stage with IoU=0.7

Fig. 6: Illustration of cascading process in bounding box regression. Bounding boxes are either be ignored or passed to the next stage, depends on whether they can pass the IoU threshold in the current stage. The blue box represents the ground truth, while the green box represents the detection given by the model.

TABLE I: Dataset configuration in training process.

| Train Set | | Test Set | |
|---|---|---|---|
| ***Flower*** | ***Seedpod*** | ***Flower*** | ***Seedpod*** |
| 30,729 | 40,898 | 1,711 | 3,186 |

TABLE II: Number of bounding boxes grouped by area.

| Area | Flower | Seedpod |
|---|---|---|
| **Small** | 1,513 | 157 |
| **Medium** | 176 | 1,134 |
| **Large** | 22 | 1,895 |

## IV. Experiment

### A. Experiment Setup

We split each video into seven frames on average. Our consideration is fewer frame results in the loss of scenes, which possibly contains any objects that would be precious for training – more frame results in the higher appearance rate of scenes that highly overlap with another image. Though the influence is not yet confirmed, we tried to avoid this as there is a possibility that the model would overfit particular scenes.

We have collected 3,082 videos in total, resulting in about 39,583 frames. Several frames do not contain any object, so we omit them from the dataset. As a result, only 12,659 images remain that contain either flower or seedpod. After splitting the data into the train set and test set, the number of flower and seedpod bounding boxes are presented in Table I.

Since we especially want to observe the performance of these models on particular sizes, we also grouped the objects' bounding boxes in the test set based on their area. Previously, MS COCO dataset did this by grouping the bounding boxes into three groups: small, medium, and large. It was done by these rules:

- $AP^{small} : area < 32^2$
- $AP^{medium} : 32 < area < 96^2$
- $AP^{large} : area > 96^2$

In addition, the longer axis of each image in MS COCO dataset does not exceed 640 pixels. It means that an object is grouped as small if it has a 1:20 length ratio from the maximum axis length. For the large group, the length ratio would be 1:6.67 by the same calculation. In our case, we decided to resize the images in our dataset such that the longest axis does not exceed 1333 pixels. By following similar ratio to that of MS COCO as a guideline, we group an object as small if the area is less than $67^2$, and an object is grouped as large if the area exceeds $200^2$. The number of objects on each group are summarized in Table II. The smaller objects are dominated by the *flower* class, while medium and larger objects are dominated by *seedpod* class.

### B. Result

In this section, we provide an ablation study regarding the result of RetinaNet, Faster R-CNN, and Cascade R-CNN. We also did a specific analysis in evaluating the performance of models according to the bounding box area in the ground truth, as we need to confirm the models' performance in detecting small objects especially.

We trained each model with two different backbone, the ResNet50 and ResNet101. Training was done using SGD with momentum of 0.9. Detectors are set to handle three classes: flower, seedpod, and background. Images were resized into 1333 x 800 pixels in the preprocessing step. We believe smaller resolution will impact the performance of models in detecting smaller objects, especially in the case when convolution network with deep enough layer are used. It will make smaller bounding box area in the ground truth lost its

TABLE III: Model performance over whole test data.

| Model | $AP^{50}$ | $AP^{75}$ |
|---|---|---|
| RetinaNet50 | 81.6 | 50.2 |
| RetinaNet101 | 83.3 | 51.6 |
| Faster R-CNN w/ ResNet50 | 87.1 | 53.7 |
| Faster R-CNN w/ ResNet101 | 88.7 | 57.1 |
| Cascade R-CNN w/ ResNet50 | 87.6 | 60.0 |
| Cascade R-CNN w/ ResNet101 | **89.6** | **62.6** |

fine details, and lead into less chance for the detector to find meaningful features [19].

Detection performance is evaluated by the average precision (AP), which is the average score of the area below precision-recall curve in each class (except the background). Table III summarizes the comparison between RetinaNet and Cascade R-CNN on IoU = 0.5 ($AP^{50}$) and IoU = 0.75 ($AP^{75}$).

Cascade R-CNN with ResNet101 backbone tops other models in both metrics. In fact, even with the ResNet50 backbone, Cascade R-CNN exceeds the performance of Faster R-CNN model with ResNet101 backbone on $AP^{75}$ score. We could say that the strong point of the cascading stage architecture is that its performance degrades less on the stricter IoU, compared to other models.

In opposite, RetinaNet model seems to struggle in our dataset compared to the two-stage detectors. The performance even gets worse on $AP^{75}$ score, where higher IoU threshold is used. It is often said that the performance of one-stage detectors often be inferior as it needs to take the whole background image into consideration by its pipeline, which results in imbalance between background class and other classes in the training phase. The case would be different from the two-stage detectors, as they only consider regions proposed by the region proposal layer for the classification. Even with class imbalance countermeasures such as focal loss as proposed in RetinaNet architecture, it still could not perform well compared to Faster R-CNN, at least in our dataset. In contrast with RetinaNet, Cascade R-CNN exceeds Faster R-CNN performance with its region proposal resampling mechanism.

The performance difference of each model can also be estimated from their precision-recall curve, as shown in Figure 7. Larger area under curve means better performance the model gives. While the performance gain of Cascade R-CNN does not look significant in $AP^{50}$, the model shines on $AP^{75}$ as we can see larger difference of area under curve. This means that the resulting bounding boxes of Cascade R-CNN resembles more to the ground truth compared to other models.

Performance measurement of the models on each object is summarized in Table IV. It reflects the previous result, where Cascade R-CNN with ResNet101 backbone gets the highest score on both flower and seedpod detection.

The overall accuracy in flower detection is comparatively lower than the seedpod detection, with the highest AP in seedpod detection is 92.5 while the highest AP in flower detection only reaches 86.6. Flower class dominated by small-sized objects may be the main factor in why it is harder to get high performance in flower detection. To see how the object-specific performance and object size correlates, we provide the performance of models in a particular size, represented in Table V.

TABLE IV: Model performance on specific objects, represented in $AP^{50}$.

| Model | Flower | Seedpod |
|---|---|---|
| RetinaNet50 | 78.5 | 84.7 |
| RetinaNet101 | 81.2 | 85.5 |
| Faster R-CNN w/ ResNet50 | 84.0 | 90.1 |
| Faster R-CNN w/ ResNet101 | 85.8 | 91.7 |
| Cascade R-CNN w/ ResNet50 | 83.6 | 91.7 |
| Cascade R-CNN w/ ResNet101 | **86.6** | **92.5** |

TABLE V: Model performance on each bounding box size, represented in $AP^{50}$.

| Model | Small | Medium | Large |
|---|---|---|---|
| RetinaNet50 | 41.3 | 73.7 | 79.0 |
| RetinaNet101 | 44.9 | 78.2 | 86.7 |
| Faster R-CNN w/ ResNet50 | 49.8 | 80.4 | 81.0 |
| Faster R-CNN w/ ResNet101 | 51.9 | 83.0 | 77.2 |
| Cascade R-CNN w/ ResNet50 | 49.2 | 81.9 | 81.5 |
| Cascade R-CNN w/ ResNet101 | **53.2** | **86.1** | **85.8** |

While each model performs relatively fine on detecting objects with medium and large sizes, we can see a significant performance drop in each model when detecting small objects. Particularly in flower detection, several fail cases should be put into consideration to create a more robust model.

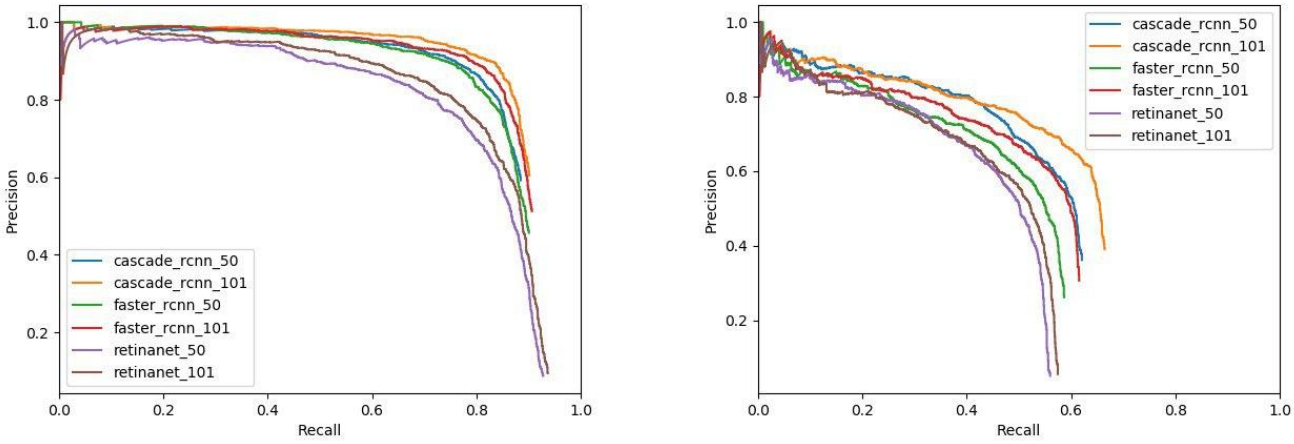
Detection would especially get tricky when the flowers make a cluster, as illustrated in Figure 8. It is a common problem in object detection, as overlapping bounding boxes often be filtered in the process.

The flowers in our dataset also has two variants of petals colors, the purple one and the white one. Flower with purple petals dominates in the dataset. Flowers with white petal, as a minority in the dataset, is more challenging to be detected, especially in images with high exposure in the surrounding illumination. Data imbalance in the petal color may also be one of the factors in why it is harder to detect in our dataset. Figure 9 is one example of undetected flower in accordance with the aforementioned problem.

### C. Parameter Tuning

Default configuration of Cascade R-CNN architecture was set with three stages, with the corresponding IoU threshold in each stage is $\{0.5, 0.6, 0.7\}$. Table VI shows how the stage configuration affects the model performance.

Setting the first stage with a lower IoU gives a slight accuracy increase on $AP^{50}$. Nevertheless, it could not handle the higher the IoU that well, as we can see the decrease of the result in $AP^{75}$ compared to the default setting. The performance reduction only gets worse as more stage were added, which means that more stage does not make better model in this scenario.




(a) $AP^{50}$ (b) $AP^{75}$

Fig. 7: Precision-recall curve for both $AP^{50}$ and $AP^{75}$ evaluation metrics.


Fig. 8: Accurate detection is hard to obtain on cluster of flowers

Fig. 9: High exposure on the background influences the detection performance, especially on flower with white petals.

TABLE VI: Performance of Cascade R-CNN with ResNet50 over several stage configurations.

| **IoU Configuration** | $AP^50$ | $AP^{75}$ |
|---|---|---|
| {0.5, 0.6, 0.7} (default) | 87.6 | **60.0** |
| {0.4, 0.5, 0.6} | **87.7** | 57.7 |
| {0.6, 0.7, 0.8} | 86.2 | 56.3 |
| {0.4, 0.5, 0.6, 0.7} | 87.2 | 57.2 |

## V. Conclusion

We presented the end-to-end process of flower and seed-pod detection of soybean plants in an actual soybean field environment. Data were collected by capturing video of each plant and cutting it into several frames. The annotated frames were then used for training the object detection model. We also studied the performance comparison of the state-of-the-art object detection model in our case, including RetinaNet, Faster R-CNN, and Cascade R-CNN. The evaluation shows that Cascade R-CNN gives the highest performance on both $AP^{50}$ (89.6) and $AP^{75}$ (62.6). Ablation studies were conducted on the performance of object detectors in particular object sizes. The result indicates that flower detection is comparatively harder as smaller objects occurs more frequently in our dataset. Edge cases such as flower clusters and overexposure in flower with white petals were also discussed to make a more robust detector in the future works.

## Acknowledgment

This work is supported by FY2005 Ministry of Agriculture, Forestry and Fisheries contract research project of "Development of diagnostic method and countermeasure technology for high yield inhibitory factor". We would also like to thank Shunsuke Higuchi (Fukuoka Agriculture and Forestry Research Center), Hidenori Asami (Western Region Agricultural Research Center, NARO), and Tomiya Maekawa and Hiroko

Sawada (Central Region Agricultural Research Center, NARO) for their valuable contribution in our research.

## REFERENCES


[1] A. S. Ferreira, D. M. Freitas, G. C. da Silva, H. Pistori, and M. T. Folhes, "Weed detection in soybean crops using ConvNets," Computers and Electronics in Agriculture, vol. 143, pp. 314–324, December 2017.

[2] Tetila et al., "Automatic Recognition of Soybean Leaf Diseases Using UAV Images and Deep Convolutional Neural Networks," IEEE Geoscience and Remote Sensing Letters, pp. 1–5, August 2019.

[3] J. Zhong, T. Lei, and G. Yao, "Robust Vehicle Detection in Aerial Images Based on Cascaded Convolutional Neural Networks." Sensors (Basel) vol. 17(12), pp 2720, November 2017.

[4] S. Bargoti and J. Underwood, "Deep fruit detection in orchards," in IEEE International Conference on Robotics and Automation (ICRA), 29 May - 3 June 2017, Singapore, Singapore [Online]. Available: IEEE Explore, http://www.ieee.org. [Accessed 23 Jan. 2020].

[5] P. Lin, W. S. Lee, Y. M. Chen, N. Peres, and C. Fraisse, "A deep-level region-based visual representation architecture for detecting strawberry flowers in an outdoor field," Precision Agriculture, June 2019.

[6] Tian et al., "Apple detection during different growth stages in orchards using the improved YOLO-V3 model," Computers and Electronics in Agriculture, vol. 157, pp. 417–426, February 2019.

[7] R. Girshick, J. Donahue, T. Darell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 24-27 June 2014, Columbus, US.

[8] Ross Girschick, "Fast R-CNN," IEEE International Conference on Computer Vision (ICCV), 7-13 December 2015, Santiago, Chile [Online]. Available: IEEE Explore, http://www.ieee.org. [Accessed 25 Jan. 2020].

[9] S. Ren, K. He, R. Girschick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection," Advances in Neural Information Processing Systems (NIPS), 7-12 December 2015, Montréal, Canada [Online]. Available: NIPS, http://www.nips.cc. [Accessed 25 Jan. 2020].

[10] J. Redmon, S. Divvala, and R. Girshick "You Only Look Once: Unified, Real-Time Object Detection," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 26 June - 1 July 2016, Las Vegas, US.

[11] J. Redmon and A. Farhadi "YOLO9000: Better, Faster, Stronger," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 21-26 July 2017, Honolulu, US.

[12] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1506.02640 [cs.CV], May 2016.

[13] Liu et al., "SSD: Single Shot MultiBox Detector," European Conference on Computer Vision (ECCV), 8-16 October 2016, Amsterdam, Netherlands.

[14] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 21-26 July 2017, Honolulu, US.

[15] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving Into High Quality Object Detection," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 18-23 June 2018, Salt Lake City, US.

[16] T. Lin et al., "Microsoft COCO: Common Objects in Context," European Conference on Computer Vision (ECCV), 6-12 September 2014, Zurich, Switzerland.

[17] B. V. K. V. Kumar and A. Mahalanobis, The Technical Writer's Handbook. Cambridge University Press, 2005.

[18] T. Lin et al., "Feature Pyramid Networks for Object Detection," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 21-26 July 2017, Honolulu, US.

[19] L. Cui et al., "MDSSD: multi-scale deconvolutional single shot detector for small objects," Science China Information Sciences. vol. 63, January 2020.

[20] Tzutalin, "LabelImg," MIT License, 2015 [Online]. Available: https://github.com/charlespwd/project-title [Accessed 29 Jan. 2020].

[21] K. Omura et al., "An Image Sensing Method to Capture Soybean Growth State for Smart Agriculture Using Single Shot MultiBox Detector," IEEE International Conference on Systems, Man, and Cybernetics (SMC), 7-10 October 2018, Miyazaki, Japan.