Computational comparison of image preprocessing techniques for plant diseases detection

1st Emmanouil Karantoumanis
Department of Electrical & Computer Engineering
University of Western Macedonia
Kozani, Greece
e.karantoumanis@uowm.gr

2nd Vasileios Balafas
Department of Electrical & Computer Engineering
University of Western Macedonia
Kozani, Greece
v.balafas@uowm.gr

3rd Malamati Louta
Department of Electrical & Computer Engineering
University of Western Macedonia
Kozani, Greece
louta@uowm.gr

4th Nikolaos Ploskas
Department of Electrical & Computer Engineering
University of Western Macedonia
Kozani, Greece
nploskas@uowm.gr

2022 7th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM) | 979-8-3503-9858-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/SEEDA-CECNSM57760.2022.9932972

***Abstract*—There is an abundance of deep neural network models for plant disease detection. Prior to applying these models, image preprocessing techniques are applied in order to improve the detection results. However, there is a lack of computational comparisons on the application of different image preprocessing techniques before applying object detection algorithms for plant disease detection. This paper aims to fill this gap by presenting a computational comparison of seven different image preprocessing techniques (auto-orientation, object isolation, resizing, grayscale conversion, static crop, contrast adjustment, tiling) applied prior to the execution of two state-of-the-art object detection algorithms, one single-stage detector, YOLOV5, and one two-stage detector, Faster-RCNN. We investigate whether or not these preprocessing techniques improve the accuracy, training time, and inference time, of plant disease detection. Apart from comparing these techniques solely, we also perform combinations of the preprocessing techniques. The PlantDoc dataset was used for this experimental study. Computational results show that the best method improves the mean average precision by 9% and 3% for YOLOv5 and Faster-RCNN, respectively. Finally, the combination of all seven preprocessing techniques yields an improvement of about 13% in the mean average precision of both object detectors.**

***Index Terms*—Object recognition, neural nets, model classification.**

I. INTRODUCTION

CRop diseases contribute to production loss. The Food and Agriculture Organization of the United Nations estimates that 20–40% of the annual global crop production is lost to diseases and pests [1]. Therefore, plant diseases and pests detection is a field that has attracted much attention in the recent literature. Advances in image processing and neural network models have been applied for the detection of diseases and pests in crops.

In the vast literature of precision agriculture, there are a lot of deep learning models for identifying diseases and pests [2] [3] [4] [5]. However, there is a lack of works dealing with the application of novel image preprocessing techniques prior to the execution of object detectors. Most papers that propose a new deep learning model for plant diseases and pests detection either do not apply any preprocessing method or apply simple preprocessing techniques without any justification of their effect in the detection accuracy. Therefore, it is of great significance to assess the performance of different preprocessing techniques before applying deep learning methods for the detection of diseases and pests in crops.

Image quality is one of the most important factors that affect object detection and image classification accuracy. A high-quality image has a higher rate of classification or recognition than any unprocessed noisy image. In precision agriculture, we acquire images from high altitudes using Unmanned Aerial Vehicles (UAVs) or from difficult terrains using Unmanned Ground Vehicles (UGVs). Because of the air turbulence and the speed of the UAV, many captured images are blurry and shaky [6]. The same applies for UGVs when they are used to capture images on difficult terrains. Thus, it is harder to extract features from such images, which reduces the object recognition and localization accuracy.

There is a large gap in the literature regarding the computational comparison of image preprocessing techniques for plant diseases and pests recognition. The aim of this paper is to study the effect that image preprocessing techniques have on the recognition of crop diseases. In the experimental study of this paper, we run experiments detecting crop diseases using two state-of-the-art object detection models either by using image preprocessing techniques or without preprocessing. Seven image preprocessing techniques are utilized in this paper, namely auto-orientation, object isolation, resizing, grayscale conversion, static crop, contrast adjustment, and tiling. In order to study the effect of preprocessing on different types of object detection models, we have selected one single-stage detector, YOLOV5 [7], and one two-stage detector, Faster-RCNN [8]. Different metrics are used to assess the efficiency of the preprocessing techniques like the accuracy, the training time, and the inference time of the object detection models.

The rest of this paper is structured as follows. Section II includes related work. In Section III, we present the image preprocessing techniques that were utilized in the experimental study. The computational comparison of the preprocessing techniques is presented in Section IV. Finally, conclusions are provided in Section V.

II. Related Work

Various works have studied image preprocessing techniques in different image types. Image preprocessing techniques have been applied in applications for face recognition, medical imagery, object detection, and agriculture. Dharavath et al. [9] used image preprocessing techniques to improve the face recognition rate. They used techniques such as face detection and cropping, image denoising, image resizing, image normalization, and filtering. In addition, they used three different feature extraction techniques, to illustrate the effect of preprocessing techniques. The impact on the recognition rate for the three methods was 66.32%, 62.12%, and 48% respectively.

Vasuki et. al. [10] used image preprocessing techniques for various areas of medical imaging. They used denoising and image enhancement techniques (i.e., frequency and spatial domain enhancement methods) to make images more clear for human perception or machine analysis. Schmidt et. al. [11] proposed an image preprocessing technique for object recognition of aerial images. They compared three different methods, no segmentation, segmentation to equal size, and the proposed AVIS method. They concluded that the AVIS method surpassed by 8.72% the average precision of the previous better method with a confidence score threshold of 20 and surpassed by 10.5% the average precision of the previous better method with a confidence score threshold of 50.

There is a large gap in the literature regarding the computational comparison of image preprocessing techniques for plant diseases and pests recognition. Most papers use image preprocessing techniques to improve the accuracy of predictions in plant disease detection without an in-depth analysis of the effect of different preprocessing techniques. Belay et. al. [12] developed a disease detection model using deep learning to classify diseases on a chickpea. They used a combination of the gaussian and median filters to conduct noise filtering, picture scaling, normalization, and other image preprocessing processes. They achieved an accuracy of 92.55%. Huang et al. [13] presented a method to make decisions about dumping chemicals in Helminthosporium Leaf Blotch (HLB) disease through a system that provides decision support information for spraying machines. For HLB identification, they used remote sensing from an UAV. They divided the data into four classes of disease, normal, light, medium, and heavy, and they used a Convolutional Neural Network (CNN) to identify them. They used batch normalization as a preprocessing technique. The model achieved an accuracy rate of 91.43%.

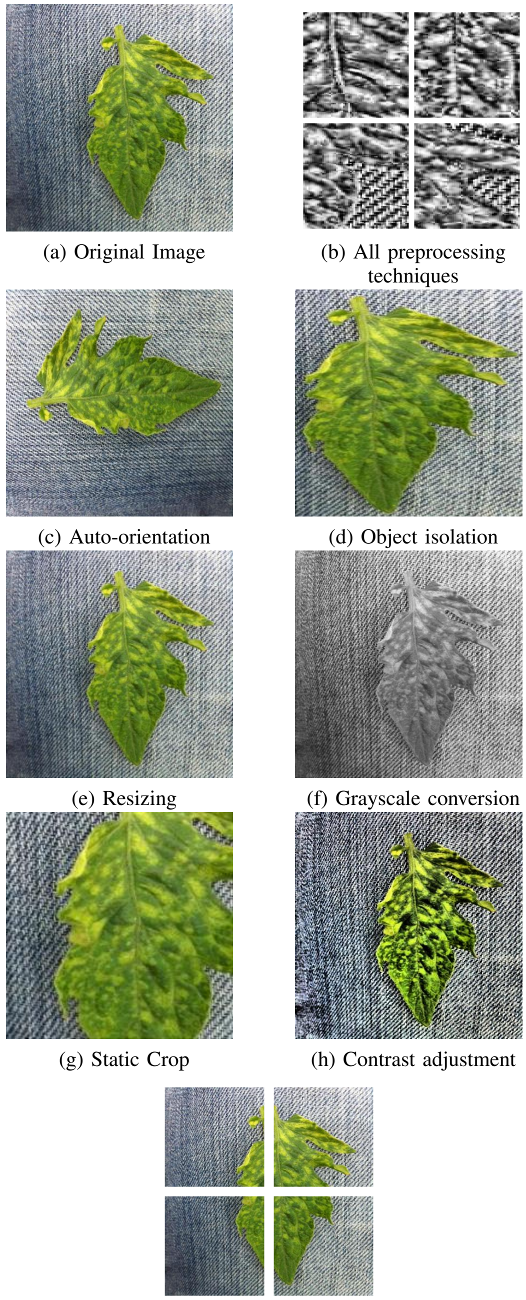
Vasavi et. al. [14] presented an overview of the existing studies in the area of forecasting crop leaf disease using image processing, machine learning, and deep learning algorithms. They studies a number of papers and concluded that the most frequently used image preprocessing techniques are: (i) cropping the leaves from the acquired images, (ii) color transformations, (ii) rescaling, (iv) background removal, (v) image enhancement, (vi) flipping, (vii) rotating, (viii) shearing, and (ix) image smoothing. Aslan et. al. [15] presented a comprehensive survey on leaf disease identification and classification, and the noted that image preprocessing techniques consist of four different tasks: (i) image resizing, (ii) noise removal, (iii) image enhancement, and (iv) image segmentation.

In this paper, we aim to compare seven different preprocessing techniques prior to the application of an object detector for plant diseases and pests prediction. Different metrics, such the accuracy, the training time, and the inference time of the object detection models will be studied with and without the application of the preprocessing techniques.

III. Image preprocessing techniques

In this study, we use the following seven image preprocessing techniques to improve the mean average precision of CNN models for plant diseases and pests detection:

- i Auto-orientation: a technique that strips an image from its Exchangeable Image File Format (EXIF) data by determining the orientation of a given image in order to display it the same way that they are stored on disk. EXIF files hold crucial information about images. These data files are produced by almost all digital cameras each time a picture is captured. The whole image's metadata, including the exposure level, picture's location, orientation, and any camera settings, are included in an EXIF file.
- ii Object isolation: generates an image (with one full-frame annotation) for each bounding box in the original dataset.
- iii Resizing: changes the image's size and, optionally, scales to a desired set of dimensions. Annotations are adjusted proportionally.
- iv Grayscale conversion: reduces the number of the image size by changing an RGB-channeled image into one with only a single grayscale channel. The weighted total of the associated red, green, and blue pixels is used to calculate the value of each grayscale pixel: G = 0.2125 RED + 0.7154 GREEN + 0.0721 BLUE.
- v Static crop: an image is reduced to a certain horizontal or vertical portion.
- vi Contrast adjustment: enhances an image with low contrast. For local contrast improvement, it uses the adaptive equalization technique, which makes use of histograms calculated across several tiling sections of the picture. Therefore, even in areas that are darker or brighter than the majority of the areas in an image, local details can be improved.
- vii Tiling: a technique for detecting small images by tiling the images. Tiling effectively focuses the detector on small objects while preserving the low input resolution required for quick inference.



(a) Original Image
(b) All preprocessing techniques
(c) Auto-orientation
(d) Object isolation
(e) Resizing
(f) Grayscale conversion
(g) Static Crop
(h) Contrast adjustment
(i) Tiling

Fig. 1: Visualization of the original image and and the preprocessed images

In Figure 1, we present an example of an original image from the PlantDoc dataset [16] and the corresponding images after the application of each preprocessing technique separately and combined.

IV. Computational study

In this section, we present the experimental study with the object detection algorithms either by using the image preprocessing techniques or without them. The PlantDoc [16] dataset is used, which contains a total of 2,345 images with 13 different plant species and 18 classes of diseases. For the training and detection of the plant diseases, we used two state-of-the-art object detection algorithms: one single-stage detector, namely YOLOV5 [7] and a two-stage detector, namely Faster-RCNN [8]. The "You Only Look Once" (YOLO) object detection technique divides images into a grid layout. In the grid, each cell is in charge to find objects inside of it. Due to its accuracy and speed, YOLO is one of most widely-used algorithms for object detection. Faster R-CNN is a single-stage model that is trained end-to-end and is used for object detection. Faster R-CNN is able to accurately predict the positions of various items efficiently and correctly. Both algorithms' models are pre-trained on the COCO dataset [17].

The experiments were carried out on a Dell PowerEdge R710 server with an Intel Intel Xeon Silver 4214 CPU processor with 48 CPU cores and 128 GB of RAM running under Ubuntu 20.10 64-bit, and on an NVIDIA V100 Tensor Core GPU with 32 GB of memory. A custom deep learning workflow orchestrator was implemented with PyTorch [18], which supports custom models and pretrained models from OpenMMLab [19] and TorchVision [20]. For the experiments we trained and evaluated each preprocessing technique once and all preprocessing techniques in the dataset.

We combine the image resize technique to all other preprocessing techniques to decrease the runtime of the object detectors. In addition, we used several of the COCO challenge metrics to assess the overall accuracy of the trained models. The metrics are the following:

- AP: the median average precision (mAP) averaged over ten intersection over union (IoU) thresholds (i.e., 50%, 55%, 60%, ..., 95%). This is the main metric of our experimental process
- $AP^{\text{IoU=.50}}$: AP at IoU=50%
- $AP^{\text{IoU=.75}}$: AP at IoU=75%
- $AR^{\text{maxdets=1}}$: the maximum recall given one detection per image
- $AR^{\text{maxdets=10}}$: the maximum recall given ten detection per image
- $AP^{\text{maxdets=100}}$: the maximum recall given 100 detection per image

The AP results of YOLOv5 are presented in Table I, while the AR results are presented in Table II. The best performance for each metric is in bold. Combining all techniques increased the AP by 13.1%. In addition, the techniques auto-orientation, object isolation, and static crop provided better results in terms of accuracy in contrast to the original unprocessed images. The grayscale conversion had the worst performance of 19% since color provides a meaningful signal to the model and colors are fairly similar in their appearance [21]. Regarding the $AP^{\text{IoU=.50}}$ metric, the best accuracy was obtained by the object isolation with an accuracy of 38.7%. Similarly, for the $AP^{\text{IoU=.75}}$ metric, which is a strict metric, the best accuracy was obtained for the auto-orientation technique with an accuracy of 32.1%. The combinations of all techniques also succeed in the best AR for max detections 1, 10, and 100. The recall remains stable at 41% for all three cases. Since we found that the grayscale conversion and tiling had negative results, we reran the experiment using only the other five methods. The results were not as expected, as the combination of all other techniques seems to decrease the AP metrics, except for the $AP^{\text{IoU=.50}}$ metric.

The AP results of Faster-RCNN are shown in Table III, while the AR results are shown in Table IV. Similarly to YOLOv5, there was also an increase of 13.4% in the accuracy by combining all preprocessing techniques even though none of the techniques individually caused a significant increase of the accuracy. Regarding the $AP^{\text{IoU=.50}}$ metric, the best accuracy was obtained by the original unprocessed images with an accuracy of 42.8%. Similarly, for the $AP^{\text{IoU=.75}}$ metric, the best accuracy was achieved by the contrast adjustment technique with an accuracy of 28.6%. The best technique for the AR metric for max detections 1, 10, and 100 is to combine all the preprocessing techniques together. The recall is 29.4%, 48.1%, and 50.2% for the three cases, respectively.

Finaly, Table V includes the average execution times per epoch in seconds for the YOLOv5 and Faster-RCNN training. The training times have decreased, meaning that the preprocessing techniques had a positive effect on the model training time. We observe that the YOLOv5 algorithm decreases its average time per epoch by about 81%–86% for the preprocessing techniques auto-orientation, object isolation, and static crop. However, the average precision is increased. The grayscale conversion technique, although it significantly reduces the execution time of each epoch, does not achieve an improvement in the results. This is due to the loss of a large part of the information from the image. This results in the training of the neural network being more difficult, and consequently not being able to find better results. Tiling is the most time-consuming preprocessing technique. The tiling technique takes as input one image and produces four images, thus, increasing the dataset three times.

In Figures 2–3, we present the performance of the YOLOv5 and Faster-RCNN algorithms, respectively. The x-axis is the mAP, while the y-axis is the average execution time per epoch. Techniques appearing on the lower right part are the best since they combine a large mAP value with a low average execution time per epoch. In both cases, the grayscale technique is the fastest technique but it also produces lowest mAP. For the YOLOv5 algorithm, the average time per epoch of the original dataset and the average time per epoch of all the preprocessing techniques together are relatively close, but the latter produces a larger mAP value. Static crop method produces a very good accuracy by also reducing the execution time per epoch compared to the original images. Regarding the Faster-RCNN algorithm, the combination of all preprocessing techniques dramatically reduces the execution time per epoch and on the same time increases the mAP value.

V. CONCLUSION

Image preprocessing techniques are utilized prior to the application of object detection models without any justification.

Techniques	**AP**	**AP50**	**AP75**
Original images	0.266	0.376	0.301
All techniques	**0.301**	0.302	0.302
Auto-orientation	0.271	0.381	**0.321**
Object isolation	0.279	0.387	0.311
Grayscale conversion	0.190	0.314	0.194
Static crop	0.292	0.406	0.270
Contrast adjustment	0.245	0.245	0.245
Tiling	0.201	0.338	0.202
Combination of 5 techniques	0.251	**0.399**	0.292

TABLE I: Average precision of YOLOv5 on the PlantDoc dataset

Techniques	**maxDets=1**	**maxDets=10**	**maxDets=100**
Original images	0.194	0.370	0.396
All techniques	**0.410**	**0.410**	**0.410**
Auto-orientation	0.201	0.381	0.405
Object isolation	0.198	0.374	0.402
Grayscale conversion	0.190	0.370	0.385
Static crop	0.272	0.379	0.387
Contrast adjustment	0.316	0.316	0.316
Tiling	0.200	0.339	0.379
Combination of 5 techniques	0.216	0.380	0.406

TABLE II: Average recall of YOLOv5 on the PlantDoc dataset

Techniques	**AP**	**AP50**	**AP75**
Original images	0.260	**0.428**	0.028
All techniques	**0.295**	0.441	0.040
Auto-orientation	0.225	0.359	0.237
Object isolation	0.269	0.435	0.039
Grayscale conversion	0.210	0.341	0.221
Static crop	0.263	0.395	0.268
Contrast adjustment	0.260	0.415	**0.286**
Tiling	0.235	0.422	0.219
Combination of 5 techniques	0.236	0.382	0.271

TABLE III: Average precision of Faster R-CNN on the PlantDoc dataset

Techniques	**maxDets=1**	**maxDets=10**	**maxDets=100**
Original images	0.200	0.426	0.459
All techniques	**0.294**	**0.481**	**0.502**
Auto-orientation	0.197	0.394	0.407
Object isolation	0.215	0.418	0.467
Grayscale conversion	0.212	0.389	0.406
Static Crop	0.291	0.442	0.445
Contrast adjustment	0.221	0.412	0.425
Tiling	0.283	0.432	0.440
Combination of 5 techniques	0.212	0.379	0.399

TABLE IV: Average recall of Faster R-CNN on the PlantDoc dataset

Techniques	**YOLOv5**	**Faster-RCNN**
Original images	115	344
All techniques	80	75
Auto-orientation	18	75
Object isolation	16	75
Grayscale conversion	**15**	70
Static crop	21	**68**
Contrast adjustment	16	72
Tiling	108	255

TABLE V: Average training time per epoch (in seconds) with and without preprocessing

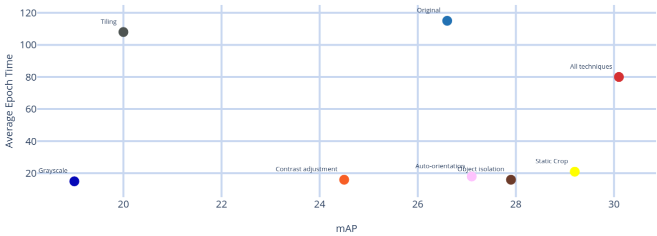


Fig. 2: YOLOv5 performance with and without preprocessing

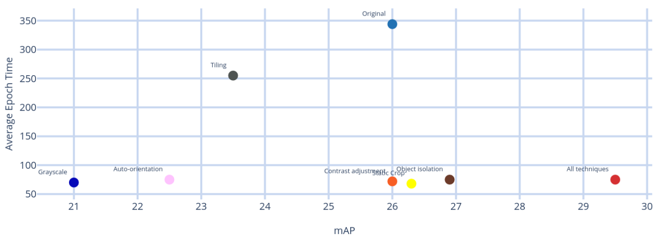


Fig. 3: Faster-RCNN performance with and without preprocessing

The main goal of this work was to provide computational evidence of several preprocessing techniques, which can help object detection models achieve better accuracy rates. In this study, we applied seven different image preprocessing techniques to the PlantDoc dataset and trained two state-of-the-art object detection algorithms (YOLOv5 and Faster-RCNN) in order to assess the effect of the preprocessing techniques on the detection accuracy and training time. We conclude that the detection accuracy increases when preprocessing techniques are applied prior to the object detection algorithms. The accuracy was increased by about 13% for both algorithms when all preprocessing techniques used at once. Furthermore, there was also a slight decrease in the average execution time per epoch in the training process when all preprocessing techniques were used.

Image preprocessing is not a panacea for achieving higher accuracy or less training time but helps. In future work, we plan to experiment with other image preprocessing techniques, such as image denoising, image filtering, and image normalization, and other object detection algorithms, such as YOLOv7 [22], EfficientDet [23], SSD [24] and RetinaNet [25].

Acknowledgements

This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation (project code MIS 5047196).

References

[1] Food and A. O. of the United Nations, "New standards to curb the global spread of plant pests and diseases," 2022. [Online]. Available: https://www.fao.org/news/story/ en/item/1187738/icode/

[2] A. Abbas, S. Jain, M. Gour, and S. Vankudothu, "Tomato plant disease detection using transfer learning with c-gan synthetic images," *Computers & Electronics in Agriculture*, vol. 187, p. 106279, 2021.

[3] Z. Chen, R. Wu, Y. Lin, C. Li, S. Chen, Z. Yuan, S. Chen, and X. Zou, "Plant disease recognition model based on improved yolov5," *Agronomy*, vol. 12, no. 2, p. 365, 2022.

[4] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers & Electronics in Agriculture*, vol. 145, pp. 311–318, 2018.

[5] B. Liu, Y. Zhang, D. He, and Y. Li, "Identification of apple leaf diseases based on deep convolutional neural networks," *Symmetry*, vol. 10, no. 1, p. 11, 2017.

[6] T. Sieberth, R. Wackrow, and J. Chandler, "UAV image blur-its influence and ways to correct it." *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 40, 2015.

[7] G. Jocher, "ultralytics/yolov5: v3.1 - Bug fixes and performance improvements," https://github.com/ultralytics/yolov5, Oct. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4154370

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[9] K. Dharavath, F. A. Talukdar, and R. H. Laskar, "Improving face recognition rate with image preprocessing," *Indian Journal of Science and Technology*, vol. 7, no. 8, pp. 1170–1175, 2014.

[10] P. Vasuki, J. Kanimozhi, and M. B. Devi, "A survey on image preprocessing techniques for diverse fields of medical imagery," in *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)*. IEEE, 2017, pp. 1–6.

[11] R. W. Schmidt, S. Woutersen, and F. Ariese, "Ramanlight—a graphical user-friendly tool for pre-processing and unmixing hyperspectral raman spectroscopy images," *Journal of Optics*, vol. 24, no. 6, p. 064011, 2022.

[12] A. J. Belay, A. O. Salau, M. Ashagrie, and M. B. Haile, "Development of a chickpea disease detection and classification model using deep learning," *Informatics in Medicine Unlocked*, p. 100970, 2022.

[13] H. Huang, J. Deng, Y. Lan, A. Yang, L. Zhang, S. Wen, H. Zhang, Y. Zhang, and Y. Deng, "Detection of helminthosporium leaf blotch disease based on UAV imagery," *Applied Sciences*, no. 3, p. 558, 2019.

[14] P. Vasavi, A. Punitha, and T. V. N. Rao, "Crop leaf disease detection and classification using machine learning and deep learning algorithms by visual symptoms: A review," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 2, p. 2079, 2022.

[15] M. F. Aslan, A. Durdu, K. Sabanci, E. Ropelewska, and S. S. Gültekin, "A comprehensive survey of the recent studies with UAV for precision agriculture in open fields and greenhouses," *Applied Sciences*, vol. 12, no. 3, p. 1047, 2022.

[16] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, "PlantDoc: a dataset for visual plant disease detection," in *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, 2020, pp. 249–253.

[17] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015.

[18] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[19] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu *et al.*, "Mmdetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[20] S. Marcel and Y. Rodriguez, "Torchvision the machine-vision package of torch," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 1485–1488.

[21] S. Ganguly, P. Bhowal, D. Oliva, and R. Sarkar, "BLeafNet: a bonferroni mean operator based fusion of cnn models for plant identification using leaf image classification," *Ecological Informatics*, vol. 69, p. 101585, 2022.

[22] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.

[23] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

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