Sensors and Materials, Vol. 35, No. 9 (2023) 3441–3449
MYU Tokyo

S & M 3403

Marine Debris Detection Using Optimized You Only Look Once Version 5

Tae-Young Lee, Seung Bae Jeon, and Myeong-Hun Jeong*

Department of Civil Engineering, Chosun University,
309 Pilmun-daero, Dong-gu, Gwangju 61452, Republic of Korea

(Received April 27, 2023; accepted August 8, 2023)

Keywords: marine debris, object detection, marine observation, deep learning, YOLOv5

Marine debris is one of the most widespread pollution problems facing oceans and waterways. It threatens the marine ecosystem and navigation safety. Thus, the extensive and timely monitoring of marine debris is crucial. In this study, we aimed to detect marine debris from images. You Only Look Once Version 5 (YOLOv5) was used to train a model to detect marine debris. Our experimental results reveal that the model achieves a mean average precision (mAP) of 96.8% and an inference speed of 1.3 ms per image. The optimized YOLOv5 has a higher mAP than a faster region-based convolutional neural network and requires less inference time than a single-shot multibox detector.

1. Introduction

Marine debris is debris exposed on the ocean surface and in the oceans. The accumulation of marine debris in oceans, including man-made debris such as plastics, metal, glass, and nets, causes significant harm to marine life and human health.(1) Marine debris tends to become more polluted and less buoyant the longer it remains in the ocean; moreover, it sinks deeper into the ocean, making it more difficult to remove. Despite these risks, the amount of marine debris continues to grow each year, underscoring efforts to remove it.(2–4) However, as massive amounts of garbage are drifting in vast seas, directly removing them by humans is challenging in terms of cost and time.

To address the marine debris problem, methods using autonomous underwater vehicles (AUVs) are attracting attention, and research to develop them more effectively is underway.(5,6) As the development of these AUVs continues to progress rapidly, the use of AUVs in ocean cleanup is becoming recognized as a more reliable method.

The higher the accuracy of recognizing marine debris and the higher the detection speed, the more effective the detection. In addition, the smaller the machine, the easier it is to install. In this context, an artificial intelligence (AI) algorithm should result in a lighter system. Xue *et al.* attempted to use an AI algorithm to detect marine debris objects by using ResNet50-You Only Look Once Version 3 (YOLOv3).(7) Zocco *et al.* applied the improved EfficientDets network.(6)

*Corresponding author: e-mail: mhjeong@chosun.ac.kr
https://doi.org/10.18494/SAM4477

YOLOv5 small has also been applied to detect marine debris.(5) However, its performance was poor because it learned without sufficient training data.

In this study, we aimed to detect marine debris by utilizing a sufficient amount of data to train an algorithm and improving the object detection performance based on the optimized model. We also compared the performance of the proposed model with those of state-of-the-art methods.

The methods section below provides a review of the object detection algorithm, transfer learning, and genetic algorithm (GA). The following section presents the results, followed by the discussion and conclusions.

2. Data, Materials, and Methods

2.1 Dataset of marine debris

In this study, we used the marine debris images provided by the AI Hub site of the National Information Society Agency. These data comprised 43306 images of 11 types of marine debris objects obtained near the coast of the Republic of Korea. The details of these 11 objects are listed in Table 1. This dataset consists of 38494 training sets and 4812 test sets. We divided the dataset into 33269 training sets, 5225 validation sets, and 4812 test sets for deep learning; the ratio was intended to be approximately 8:1:1.

2.2 YOLOv5

Object detection is a technique based on surrounding visual objects in a bounding box to display the detection instances of objects. Traditional object detection algorithms in deep learning-based computer vision tasks are largely divided into one-stage and two-stage detectors. A one-stage detector divides an image into regions, predicts bounding boxes, and classifies class probabilities all at once. The object detection speed is "breakneck" and helpful for real-time detection applications. Representative models include YOLO and the single-shot multibox detector (SSD).(8,9)

Table 1
Overview of variables used in this study.

Class name	Number of objects	Ratio
Styrofoam_Piece	14506	5.25
Plastic_ETC	10656	3.86
Styrofoam_Bouy	4099	1.48
Styrofoam_Box	2733	0.99
Pet_Bottle	41287	14.96
Plastic_Buoy	107386	38.90
Metal	21918	7.94
Plastic_Buoy_China	6017	2.18
Rope	34565	12.52
Net	19149	6.94
Glass	13742	4.98

In contrast, a two-stage detector proceeds in two stages: extracting object proposals and then using classifiers to verify the class. The two-stage detector is relatively slow, but it has a higher accuracy than the one-stage detector. A faster region-based convolutional neural network (R-CNN) exemplifies this model and has been utilized in various research areas.(10,11)

Among the aforementioned models, YOLO has been continually updated. It has been used in many studies owing to its excellent performance and ease of use.(12–14) YOLOv5 has nano, small, medium, and large sizes, which vary according to the number of parameters in the layers. The mean average precision (mAP) is commonly used to evaluate the performance of an object detection algorithm. For YOLOv5 small, the "Common Objects in Context" (COCO) mAP@0.5 is 56.8 and COCO mAP@[0.5, 0.95] is 37.4. Here, "COCO mAP" indicates that the algorithm was evaluated using the MS-COCO dataset. mAP@0.5 is the value when the intersection over union (IOU) threshold is based on 0.5, and mAP@[0.5, 0.95] is the IOU threshold ranging from 0.5 to 0.95 with an interval of 0.05 each. The corresponding mAP is averaged. In general, mAP@0.5 is used.

mAP is the average of the APs in a class. The AP is an index used to account for both precision and recall; its value is derived from a precision–recall curve and the calculated area under the curve. In object detection, precision refers to the class matching accuracy of the detected object and recall indicates the number of matched objects in the image.(15) In this study, we utilized YOLOv5 to train on marine debris datasets to construct a real-time object detection algorithm with a small file size and good precision for an autonomous surface vessel.

2.3 Transfer learning

The YOLO model was created by learning on the Microsoft (MS)-COCO dataset with 80 classes, including humans, cars, and animals. MS-COCO is a challenging dataset used to train and evaluate commonly used object detection algorithms.(16) Therefore, new learning is required to detect marine debris. However, retraining the model configuration requires a long time. A method called transfer learning has been used to overcome this problem.

In this study, transfer learning denotes extracting the features of marine debris by taking a pretrained model and learning only a specific layer, instead of learning the entire network anew. If this learned weight is used, a new model can be reconstructed quickly and effectively using only a small amount of data. In addition, unbiased results can be obtained through transfer learning, as it uses a pretrained model built through a validation dataset rather than learning the entire layer using a personal dataset.(17)

It is essential to set the hyperparameters in transfer learning to successfully train the model. A hyperparameter is an exact condition that affects model learning by identifying personal data characteristics. The process of optimizing these settings is called fine-tuning and is essential for optimizing the model output. Therefore, efforts have been made to identify and configure the optimal scenarios among the parameter values significantly impacting learning through fine-tuning to complete the corresponding model. Among the numerous fine-tuning methods to construct an optimized YOLO model, in this study, we attempted to construct optimal hyperparameter scenarios using a GA, as GAs are known to have advantages over other optimization methods.(18)

2.4 GA

Theories of genetics and the evolution of organisms inspired the GA concept. This optimization method is based on the ability of organisms to adapt to their environments. This principle is based on the fact that when genetic information is transmitted from parents to the next generation, the genes of the next generation preferentially transmit genetic information with excellent environmental adaptability in each genome. By this mechanism, the algorithm first constructs a random set called a population of genomes, selects genes through gene encoding, and then continues to create new sets through crossover and mutation.

In this way, gene evolution proceeds, eliminating bad hyperparameter scenarios and automatically discovering optimal scenarios suitable for the objective function. In the context of this study, the hyperparameter set determined by the GA can be used to construct an optimized model by transferring marine debris data to the YOLOv5 model.

3. Results

3.1 Optimized hyperparameters

To optimize the hyperparameters using the GA, the number of generations must first be determined. More than 300 generations are recommended.(19) However, the number varies from study to study. In this study, we newly configured 29 hyperparameter conditions with optimal values through a total of 300 generations. The detected hyperparameter values are shown in Fig. 1. Each of the 29 hyperparameter values are visualized and the optimal values are indicated with a cross symbol. Figure 1 includes the hyperparameter values obtained from 300 generations and the GA fitness function values, which indicate the performance according to those hyperparameter values. The hyperparameter values are on the x-axis and the fitness scores are on the y-axis. When the y-axis value is considered maximum, the corresponding x-axis value is finally displayed as a cross symbol. For example, the momentum in the third position is a tuning parameter that can improve the threshold of the gradient descent algorithm. The figure shows that the value is adjusted to a value between 0.8 and 0.98, and eventually, 0.956 is selected as the optimal value.

3.2 Optimized YOLOv5

In this study, an optimized model was constructed by training YOLOv5 through the 29 optimal hyperparameter sets identified by the GA. The YOLOv5 model was divided into four categories on the basis of the number of parameters in the layers: nano, small, medium, and large. Table 2 shows the comparison of the general YOLOv5 and optimized YOLOv5. The performance of each model changes owing to the difference in the number of parameters. In mAP@0.5, the figures of YOLOv5 are excellent, and there is no clear performance improvement between the general YOLOv5 and the optimized YOLOv5. However, there is an evident improvement in the mAP@[0.5, 0.95] values.

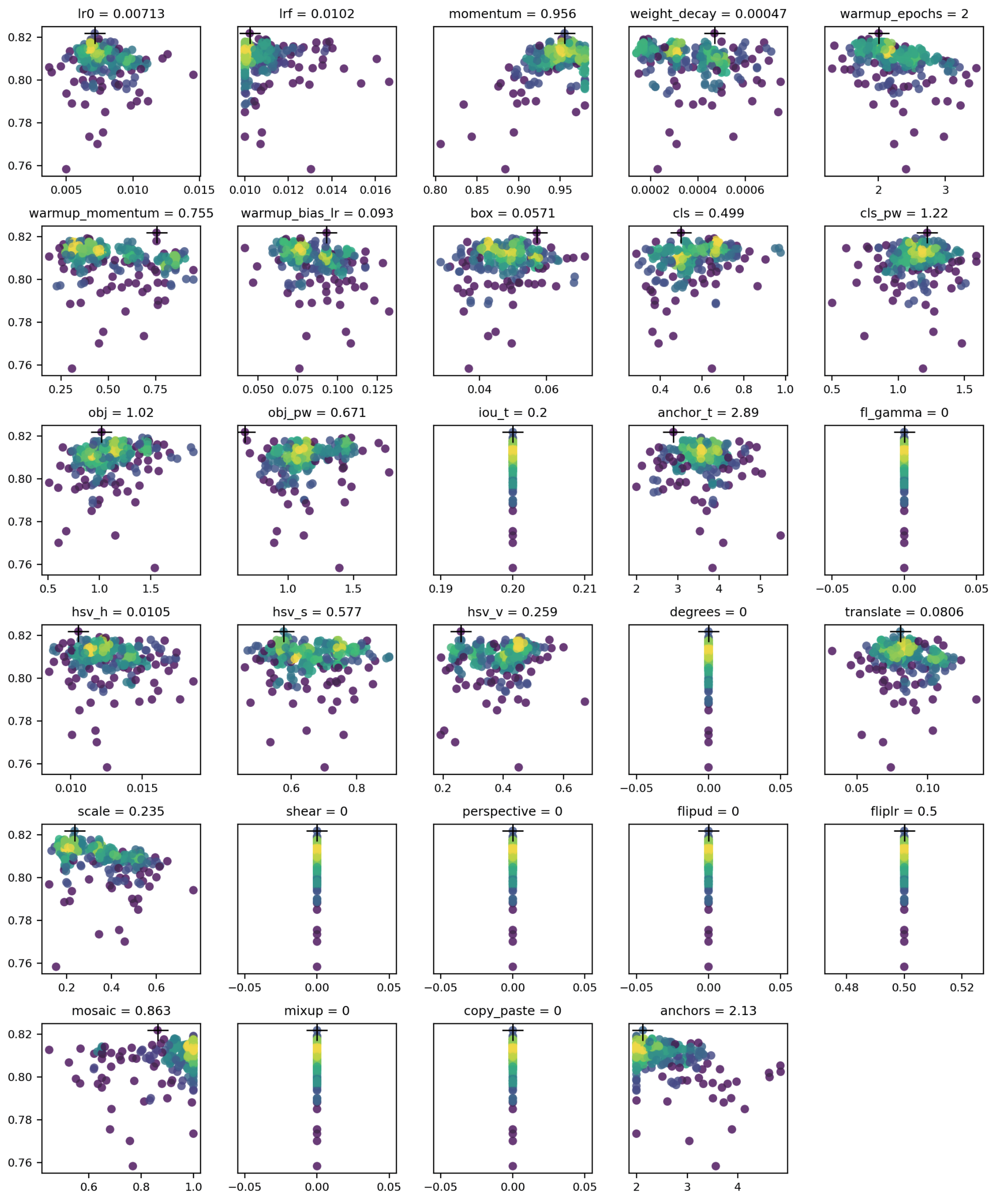

Fig. 1. (Color online) Best combinations from hyperparameter generation with GA.

Table 2
Comparison of general YOLOv5 and optimized YOLOv5.

Model	mAP@0.5	mAP@[0.5, 0.95]
YOLOv5 nano	0.934	0.771
Optimized YOLOv5 nano	0.940	0.788
YOLOv5 small	0.966	0.832
Optimized YOLOv5 small	0.968	0.847
YOLOv5 medium	0.979	0.872
Optimized YOLOv5 medium	0.977	0.881
YOLOv5 large	0.983	0.888
Optimized YOLOv5 large	0.981	0.895

Figure 2 shows the detection of marine debris in a test image. Although complicated and entangled, the proposed model can effectively detect marine debris. Furthermore, Fig. 3 shows the AP@0.5 values of YOLOv5 small for 11 classes of marine litter. The average mAP@0.5 of these values is 0.968.

3.3 Performance comparisons with other models

Table 3 shows the comparison of the performance characteristics of the four YOLOv5 models according to their sizes. The performance of YOLOv5 is compared with that of the SSD as a representative one-stage detector and that of the faster R-CNN as a representative two-stage detector.

The overall detection speed of the YOLOv5 model ranges from 0.8 to 5.0 ms, confirming that real-time object detection is possible. In addition, mAP@0.5 ranges from 0.940 to 0.981, and it is evident that the model has excellent object detection performance. YOLOv5 nano is a small model with only 3 MB; thus, its performance is lower than those of the other models. However, this model can be used depending on the application, because it still performs relatively well.

In comparison with other models, we utilized SSD 512, which has the highest accuracy among the SSD models. YOLOv5 is 40 times faster than SSD 512. Although the mAP@0.5 of YOLOv5 is similar to that of SSD 512, the mAP@[0.5, 0.95] of YOLOv5 is superior to that of SSD 512. Moreover, the YOLOv5 model is much faster than the Faster R-CNN, but not inferior in terms of accuracy.

4. Discussion

In this study, 300 generations were used to construct an optimized YOLO model. The number of generations indicates how many generations are required to identify the optimal hyperparameters in the process of evolution using the GA. As the numbers of generations and combinations increase, the result may or may not be good because the hyperparameter set, which is the result obtained through the process, evolves into a random sample. If this method is considered a way to obtain an answer close to the optimal solution even if the actual optimal solution cannot be obtained, we aimed to see if we could obtain good results. However, there is a resource limit, as this approach increases the GPU time. Accordingly, this study was limited to

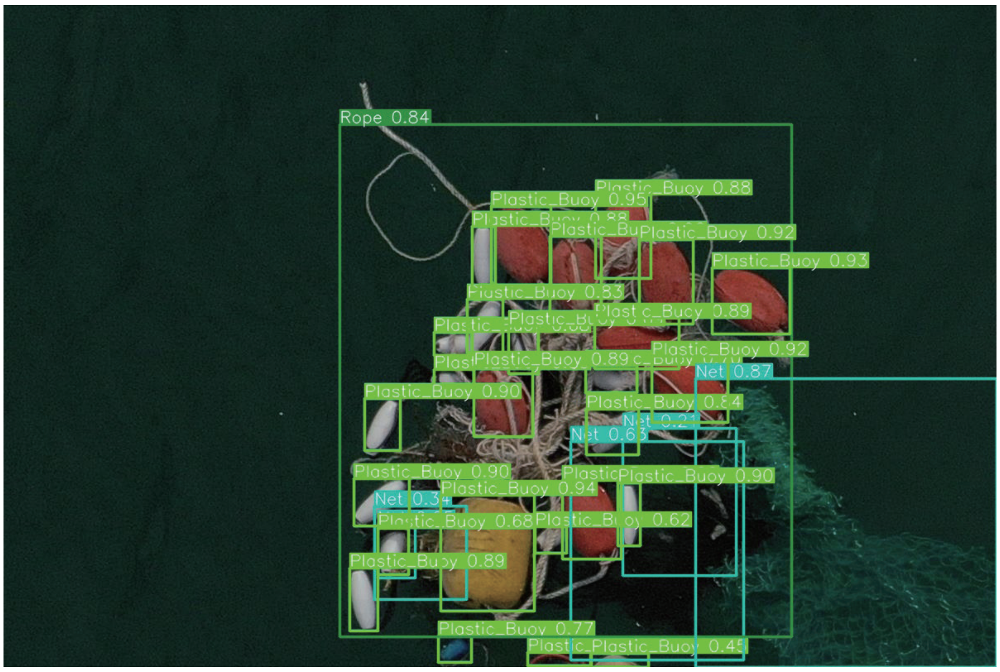

Fig. 2. (Color online) Marine debris detection.

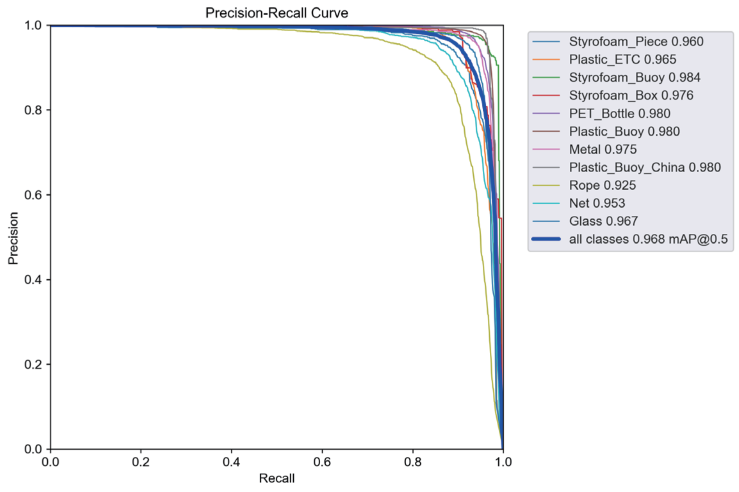

Fig. 3. (Color online) Precision–recall curve for mAP@0.5 using YOLOv5 small.

Table 3
Performance comparison.

	Model	mAP@0.5	mAP@[0.5, 0.95]	Inference	File size (MB)
One-stage detector	YOLOv5 nano	0.940	0.788	0.8 ms	3
	YOLOv5 small	0.968	0.847	1.3 ms	13
	YOLOv5 medium	0.977	0.811	3.0 ms	40
	YOLOv5 large	0.981	0.895	5.0 ms	90
	SSD 512	0.949	0.799	0.20 s	137
Two-stage detector	Faster R-CNN	0.968	0.851	0.24 s	315

300 trials. In addition, the GA algorithm with 300 generations was applied to YOLOv5 small to obtain the optimal hyperparameters and relearned the model on the basis of the hyperparameter set. However, examining the results obtained by applying the GA algorithm to YOLOv5 nano, medium, and large is time-consuming. In the future, research should focus on improving the model by determining how significant the difference is as the number of generations increases and how improved the results are when additional efforts are made to identify the optimal values for each model.

According to the experimental results, the larger the size of the YOLO model, the better the performance. When the optimized hyperparameters were applied to improve each model's performance, the performance improvement was evident in the smaller models with lower performance. However, at the same time, considering the size of the model, YOLOv5 large weighs 90 MB, and YOLOv5 small weighs only 13 MB. In terms of the difference in accuracy, that in mAP@0.5 is 0.013 and that in mAP@[0.5, 0.95] is as small as 0.048. Thus, it is judged that YOLOv5 small is effective for mounting in AUVs.

In this study, we attempted to maximize the use of YOLOv5 small through an optimization process. However, this model has some structural limitations. The near-maximal performance did not change significantly. For this reason, it will be necessary to create structurally improved models in future research by adding model layers.

5. Conclusions

The increasing amount of marine debris threatens marine ecosystems. In this study, we utilized a YOLOv5 model based on transfer learning from marine debris data to build an AI model for marine debris detection. The poor performance of this model observed in previous studies resulted from a learning process based on insufficient data and a limited optimization process. In contrast, we constructed 29 optimized hyperparameter combinations using the GA algorithm and more training data to improve the model's performance. The experimental results in this study demonstrated that the detection performance of the proposed model achieved mAP@0.5 of 0.968 and that the model size was suitable for loading into AUVs.

Acknowledgments

This study was supported by research funds from Chosun University (2022).

References

1 S. C. Gall and R. C. Thompson: Mar. Pollut. Bull. **92** (2015) 170. https://doi.org/10.1016/j.marpolbul.2014.12.041
2 M. C. Goldstein, M. Rosenberg, and L. Cheng: Biol. Lett. **8** (2012) 817. https://doi.org/10.1098/rsbl.2012.0298
3 K. Bucci, M. Tulio, and C. M. Rochman: Ecol. Appl. **30** (2020) 2. https://doi.org/10.1002/eap.2044
4 A. Cózar, S. Aliani, O. C. Basurko, M. Arias, A. Isobe, K. Topouzelis, and C. Morales-Caselles: Front. Mar. Sci. **8** (2021) 571796. https://doi.org/10.3389/fmars.2021.571796
5 C. S. Chin, A. B. H. Neo, and S. See: Proc. 2022 IEEE/ACIS 22nd International Conference on Computer and Information Science (ICIS, 2022) 20–24. https://doi.org/10.1109/ICIS54925.2022.9882484
6 F. Zocco, C. I. Huang, H. C. Wang, M. O. Khyam, and M. Van: eprint arXiv: 2203.07155 (2022). https://arxiv.org/abs/2203.07155

7 B. Xue, B. Huang, W. Wei, G. Chen, H. Li, N. Zhao, and H. Zhang: IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **14** (2021) 12348. https://ieeexplore.ieee.org/document/9626595
8 J. Redmon, S. Divvala, R. Girshick, and A. Farhadi: Proc. IEEE Conf. Computer Vision and Pattern Recognition (2016) 779–788. https://arxiv.org/abs/1506.02640
9 W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg: Proc. European Conf. Computer Vision (EECV, 2016) 21–37. https://doi.org/10.1007/978-3-319-46448-0_2
10 S. Ren, K. He, R. Girshick, and J. Sun: Proc. IEEE Conf. Neural Information Processing Systems—Natural and Synthetic (2015) 28. https://doi.org/10.48550/arXiv.1506.01497
11 H. Y. Oh, M. S. Khan, S. B. Jeon, and M. H. Jeong: Appl. Sci. **12** (2022) 5553. https://doi.org/10.3390/app12115553
12 F. Zhou, H. Zhao, and Z. Nie: Proc. 2021 IEEE Int. Conf. Power Electronics, Computer Applications (ICPECA, 2021) 6–11. https://doi.org/10.1109/ICPECA51329.2021.9362711
13 M. Sozzi, S. Cantalamessa, A. Cogato, A. Kayad, and F. Marinello: Agronnomy **12** (2022) 319. https://doi.org/10.3390/agronomy12020319
14 Q. H. Phan, V. T. Nguyen, C. H. Lien, M. T. K. Hou, and N. B. Le: Plants **12** (2023) 790. https://doi.org/10.3390/plants12040790
15 T. Y. Lee, M. H. Jeong, and A. Peter: Sens. Mater **34** (2022) 251. https://doi.org/10.18494/SAM3732
16 T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, and C. L. Zitnick: Proc. European Conf. Computer Vision (ECCV, 2014) 740–755. https://doi.org/10.1007/978-3-319-10602-1_48
17 J. Yosinski, J. Clune, Y. Bengio, and H. Lipson: eprint arXiv: 1411.1792 (2014). https://arxiv.org/abs/1411.1792
18 C. L. Chang, S. L. Lo, and S. L. Yu: Environ. Monit. Assess. **117** (2006) 145. https://doi.org/10.1007/s10661-006-8498-0
19 Ultralytics: https://github.com/ultralytics/yolov**5** (2020).