



Mask R-CNN and YOLOv8 Comparison to Perform Tomato Maturity Recognition Task

Jean Carlo Camacho¹ and Manuel Eugenio Morocho-Cayamcela²

Yachay Tech University, Urcuquí, Ecuador
{jean.camacho,mmorocho}@yachaytech.edu.ec

Abstract. This work explores the segmentation and detection of tomatoes in different maturity states for harvesting prediction by using the laboro tomato dataset to train a mask R-CNN and a YOLOv8 architecture. This work aims to test the mask R-CNN architecture and the proposed methodology efficiency on the benchmark paper [12]. The evaluation metric intersection over union (IoU) 0.5 showed an average precision of 67.2% with a recall of 78.9% over the laboro tomato dataset and an IoU average precision of 92.1% with a recall of 91.4% over the same dataset. The benchmark paper authors perform segmentation and classification in a separate process using color analysis algorithms and use the determination coefficient (R^2) for how accurately the tomato was set into the three maturity classes.

The results show that the state-of-the-art YOLOv8 has a R^2 of 0.809, 0.897, and 0.968 in the ripe, half-ripe, and green categories, respectively. However, the Mask R-CNN results are acceptable, with 0.819, 0.809, and 0.893 in the ripe, half-ripe, and green categories, respectively. The YOLOv8 model performed better than the one used in the benchmark paper by detecting, segmenting, and classifying tomatoes. Moreover, the color-analysis technique used in the benchmark paper results inefficiently because the classification results showed no linear relation between the predictions and the real values.

Keywords: Object detection · deep learning · precision agriculture · maturity recognition · Mask R-CNN · YOLO

1 Introduction

This text discusses the industrialization processes in precision agriculture and their potential applications in recognizing crops, estimating maturity time, detecting diseases, and nutritional deficiencies using computer vision systems [4, 14, 19–21]. These applications can reduce manual labor, improve harvesting accuracy, and even be deployed on autonomous robots for picking mature tomatoes [5]. However, a significant challenge lies in obtaining a suitable dataset for accurate recognition and instance segmentation. Generating a custom dataset

may be necessary for crops with limited available information, requiring environmental control, multiple angles, and expert labeling for supervised or semi-supervised tasks.

The laboro dataset, containing multiple classes and diverse images, is chosen for this study [12]. While the original dataset had 160 images, the current version has grown to 804 images.

One of the main challenges while performing this comparison was recreate the original authors methodology, to recreate the results and perform an accurate comparison.

This paper aims to replicate the instance segmentation results using the Mask R-CNN model on the laboro tomato dataset and compare them with YOLOv8, a state-of-the-art model for object recognition and instance segmentation. The proposed approach involves training the models with data augmentation, transfer learning, to improve performance in terms of mean average precision (mAP) and R^2 coefficients for predicting tomato instances.

2 Related Works

In the work of Sandro Magalh *et al.* [10], the authors performed object detection with the single-shot multi-box detector (SSD) and YOLO architectures to detect tomatoes. They compared some traditional machine learning (ML) techniques that used to work only for ripe tomatoes. However, with the implementation of object detection networks, it was possible to detect green, half-ripe, and ripe tomatoes. The models were trained with the author's own generated dataset on a greenhouse. The images contain tomatoes on the tomato plant collected by the mobile robot AgRob v16, the same robot that performed recognition with the trained model. The authors collected videos to generate the dataset and used a frame every three seconds. They used only the "tomato" class, which included tomatoes in all the ripeness states, and performed data augmentation techniques to give robustness to the model. The results showed that SSD MobileNet v2 was the best generalized and performing model. However, YOLOv4 Tiny also had achievable results with the best prediction times. For future works, the authors mentioned the importance of a regularization mechanism on the models and created sub-classes to detect ripeness states for harvesting procedures.

In the work of Wenli Zhang *et al.* [21] proposed an object recognition architecture for fruit detection in edge computing applications. The problem the authors intend to solve is the speed of the state-of-the-art model's implementation. On the methodology, the authors propose an architecture based on Light-CSPNet as the backbone network, an improved feature extraction model, a down-sampling method, and a feature fusion model to perform real-time fruit detection. The authors trained the model over three different fruits (oranges, tomatoes, and apples) and compared the results against YOLOv3, YOLOv4, YOLOv3-tiny, YOLOv4-tiny, and their own proposed model. The YOLOv3 and YOLOv4 models performed better over precision, recall, and average detection precision (AP) but had a low FPS rate (8.1 and 4.6, respectively) for edge device applications.

On the other hand, the proposed model had a slightly better result than the YOLOv3-tiny and YOLOv4-tiny, with an FPS rate of 24.8 and an AP of 0.93 for oranges, 0.847 for tomatoes and 0.85 for apples.

While some authors present multiple variants of the same architecture, the work of Mobashiru Lawal [9] presented a comparison of the original YOLOv3 and a modified version of it optimized for tomato detection by introducing spatial pyramid pooling and Mish activation function. Three variants of the model were proposed, the A, B, and C versions that used different activation functions. The model YOLO-Tomato-A got an AP of 98.3% with a detection time of 48 ms, YOLO-Tomato-B got an AP of 99.3% with a detection time of 44 ms, and YOLO-Tomato-C got an AP of 99.5% with a detection time of 52 ms. The mish activation function and the SPP combination were the most optimal in this work. The dataset was obtained with a camera in a controlled environment and labeled by an expert.

These previous works show that the YOLO architecture has been a popular option for object detection or instance segmentation [5, 16, 17]. This is because YOLO architecture performs object detection in a single shot, which results in a very efficient methodology [13, 18]. In addition, most state-of-the-art solutions include convolutional neural network use, which performs far better than ML or traditional image processing techniques separately.

However, in the work of Taehyeong Kim *et al.* [8] and Daichi Minagawa *et al.* [11] used a custom deep neural network and Mask R-CNN as the architectures, respectively, to solve the harvest time of the tomatoes. With the Mask R-CNN, they performed object detection to separate the tomato region from the background, removing noise. With the pre-processed images, the authors proposed an image processing methodology of color analysis to calculate the red rate and classify the tomatoes using the image and the red color ratio. The dataset was collected using a robot that uses cameras and takes control of how many days there were until the harvesting day. The classifier was tested with the background and noise images, and against the pre-processed ones, the pre-processed ones showed higher accuracy when predicting the harvest time and days left until harvest time.

3 System Model and Methodology

3.1 Dataset

The Laboro Tomato: Instance segmentation dataset [1]. It is a dataset containing growing tomatoes in a greenhouse. The dataset separates the tomatoes into three ripening stages, ripe, half-ripe, and green. Furthermore, it contains two different types of tomatoes: cherry and regular. The dataset is designed for object detection and instance segmentation. It contains bounding box annotations and vertices representing the tomato masks to do segmentation tasks for each tomato, along with the class to which it belongs.

Moreover, the images were taken with two cameras, giving different image quality and resolutions. The dataset contains 643 images to train and 161 to

test, giving 804. To classify the tomatoes into different categories, the regular (big) and cherry (little) were different species that the experts already knew how to differentiate from the planting phase. Additionally, cherry tomatoes are considerably smaller than regular tomatoes. To classify into the different ripeness states, the authors used the percentage of red the tomato has as one of the criteria. Fully ripened presented a 90% or more, half-ripened was between 30–89% and green was between 0–30% of red color. However, there were other criteria, and experts decided on the final classification. Examples of each category are represented on Fig. 1.

In the benchmark paper [12], the authors mention that they picked up a subset of 60 big and 40 small tomatoes that matches the color criteria initially established to generate an artificial dataset. This carefully selected dataset will be mentioned as the dataset I, and the full laboro dataset will be represented as dataset II.



(a) big green tomato



(b) big half ripen tomato



(c) big full ripen tomato



(d) little green tomato



(e) little half ripen tomato



(f) little full ripen tomato

Fig. 1. Illustration that shows an example of each tomato category, where (a), (b), and (c) are big/normal tomatoes, and (d), (e), and (f) represent the little/cherry tomatoes, the images are organized from top to bottom as green, half-ripen, and full-ripen. (Color figure online)

3.2 YOLO Architecture

YOLO (you only look once) is a real-time object detection architecture proposed by Joseph Redmon *et al.* [13]. YOLO is a popular option for object detection because it can detect objects from any image in a single shot using anchor boxes 2 using a single neural network.

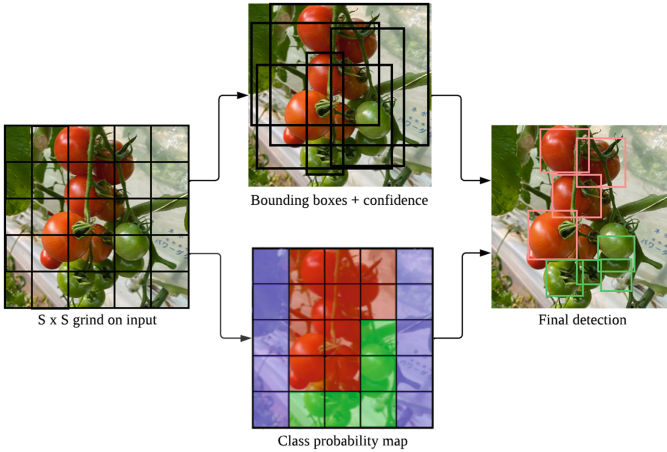


Fig. 2. Illustration that shows the basic of how YOLO architecture works [13].

YOLOv8 is the latest version of the YOLO architecture. However, there are many differences between these versions. On the open access GitHub repository of the roboflow team [2] is publicly available the YOLOv8 implementation, from which scheme is represented in Fig. 3. One of the main differences is that YOLOv8 is the first YOLO version that is an anchor-free model.

The anchor-free model considerably reduces the number of box predictions, which speeds up the non-maximum suppression (NMS) process, in which less likely candidates are sifted until a successful inference. Also, the C2f is implemented as a new convolution layer. In C2f, all the outputs from the bottleneck are concatenated, while in previous versions, only the last bottleneck was used. This details can be found on a bigger depth in [2].

3.3 Mask-R CNN Architecture

Mask-R CNN is an object detection and instance segmentation network proposed by Kaiming He *et al.* [7] with the Facebook AI Research group.

The fast R-CNN [6] was extended to process the feature maps into the RoI sections, using the RoI pooling layers, leading to a fast speed and great accuracy. The faster R-CNN has two outputs for each possible object: The class label and a bounding box offset. So the Mask R-CNN included the object mask attached to the object detection as shown in Fig. 4b.

Furthermore, the faster R-CNN contains a extra branch in contrast to the R-CNN, where the information from the pretrained VGG [15] CNN features maps are passed through a Conv layer, then the anchor boxes that are composed by the bounding box and the category, then this is passed through a non-maximum suppression layer (NMS) and sent to the RoI pooling layer as in Fig. 4b.

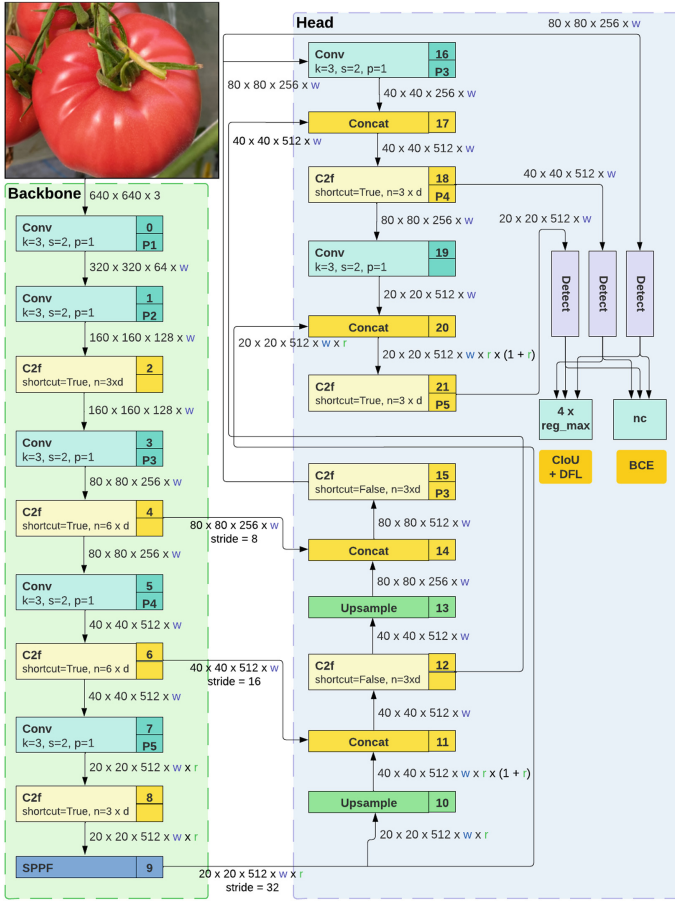


Fig. 3. A graphical representation of the YOLOv8 architecture, where the backbone region is composed of successive Conv and C2f layers and an SPPF layer at the end. From the backbone, the P3, P4, and P5 outputs are used as inputs to run inside the head layer, composed of the Unsample, Concat, C2f, and Conv layers. Finally, three outputs are obtained to pass through the Detect layer and separate the bounding box and class loss calculation.

However, Mask R-CNN architecture principal difference from faster R-CNN is that the pooling layer is replaced by a RoI align pooling layers, it predicts in parallel the RoI in a binary mask, which makes the classification independent of the mask prediction.

4 Methodology

4.1 Benchmark Paper Methodology

The methodology that the benchmark paper authors proposed consists of two fundamental steps. First, because they only used the class tomato to perform the classification, the Mask R-CNN model could only perform instance segmentation of the class “tomato.” Then with the generated mask, the authors extracted the tomato pixels to perform the color analysis in the hue-saturation-value (HSV) format. This analysis consist of computing the average color in the area of the mask for each individual prediction, then this average color is passed through multiple thresholds. For the fully ripened category, multiple threshold sets were proposed. For the first set, the minimum threshold values for each HSV value were 0, 140, and 145; the maximum were 5, 255, and 255. For the second set, the minimum threshold is 174, 120, and 135, respectively, and the maximum values are 179, 255, and 255. To define the fully ripened category, one of both sets representing the fully ripened is selected with a logic operator OR. For the half-ripened category, the minimum threshold values are 3, 144, and 155, and the maximum threshold values are 18, 255, and 255. For green tomatoes, any tomatoes that do not fall into this threshold is considered a green tomato.

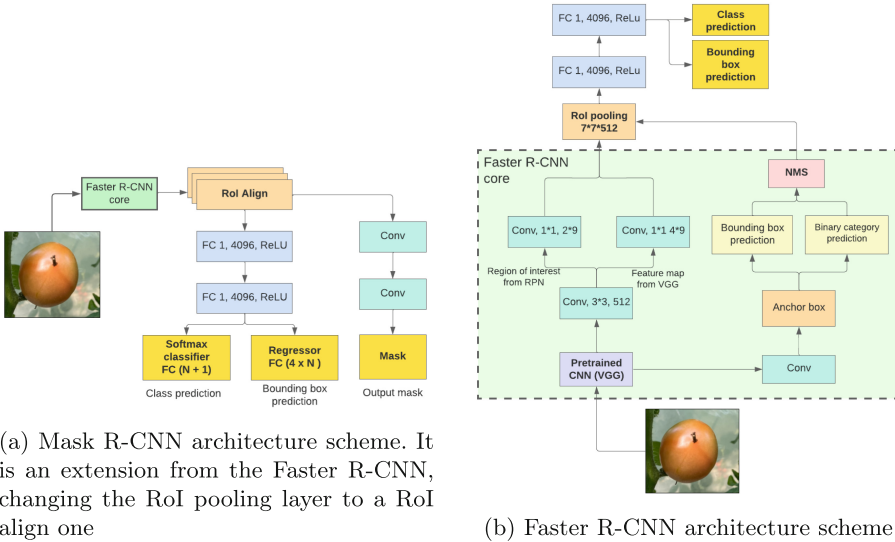


Fig. 4. Details of the Mask-RCNN architecture in (4a) and its implementation in the Faster R-CNN architecture in (4b).

To evaluate the methodology’s performance, the instance segmentation model and the color threshold classification methodology had different metrics to evaluate. Therefore, the mean average precision (mAP) and the recall metrics were

used to evaluate the model. The mAP is obtained as the mean of the ratio of correct detections (true positive) over the number of object detections. However, detection is achieved as correct if the intersection over union (IoU) is more than 0.5. The definition of the IoU is in Eq. 1. The mAP equation is defined in Eq. 2, where N is the number of classes, and AP is the average precision. Also, the precision is calculated as the number of true positives over the total detections as in Eq. 3.

$$IoU = \frac{Area(Mask_{groundtruth} \cap Mask_{predicted})}{Area(Mask_{groundtruth} \cup Mask_{predicted})} \quad (1)$$

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (2)$$

$$P = \frac{true_positives}{true_positives + false_positives} \quad (3)$$

The recall is defined as the number of positive matches over the number of ground truth objects, which is detailed in Eq. 4.

$$recall = \frac{true_positives}{true_positives + false_negatives} \quad (4)$$

Furthermore, to evaluate the classification performed by the color separation, the R^2 metric was used and is defined in Eq. 5, where SS_{res} is the sum of the squares of the residual error and SS_{tot} is the total sum of the errors. It works by measuring the amount of variance in the predictions explained by the dataset. The prediction is compared with the expert's labels classifying the tomatoes into different categories. Also is considered that the classes, the half-ripened are in the middle, so if a half-ripened tomato is a mismatch to any of the other categories, the variance is less than it would be a mismatch between a mature tomato and a green tomato.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \quad (5)$$

4.2 Proposed Methodology

In contrast to the author's work, the tomato classification, as well as the instance segmentation will be performed directly by the Mask R-CNN and YOLOv8 architectures. The color-based classification is removed, which will simplify the pipeline process, the details are represented in the Fig. 5b

Furthermore, as the benchmark paper only uses three labels (ripe, half-ripe, and green) this work will limit to use only three although, specifically the big tomatoes subset.

5 Results and Discussions

After training the Mask R-CNN [3] and YOLOv8 [2] models with 309 images for training and 67 for validating until the loss function was stable and with the same dataset distribution, both models performed predictions over 66 images. With the mask obtained from the predicted images, we used the color analysis technique to classify into the three existing classes. Furthermore, compare this classification with the obtained by the models-only technique as shown in Table 1. From this table, we can see that the classification methodology that uses color analysis to predict the classes performs poorly in contrast to the classification of the models, even the negative values represent that there is not relationship between the expected values and the real values. The replicated results on the R^2 metric represent a completely different result from what the benchmark paper authors obtained.

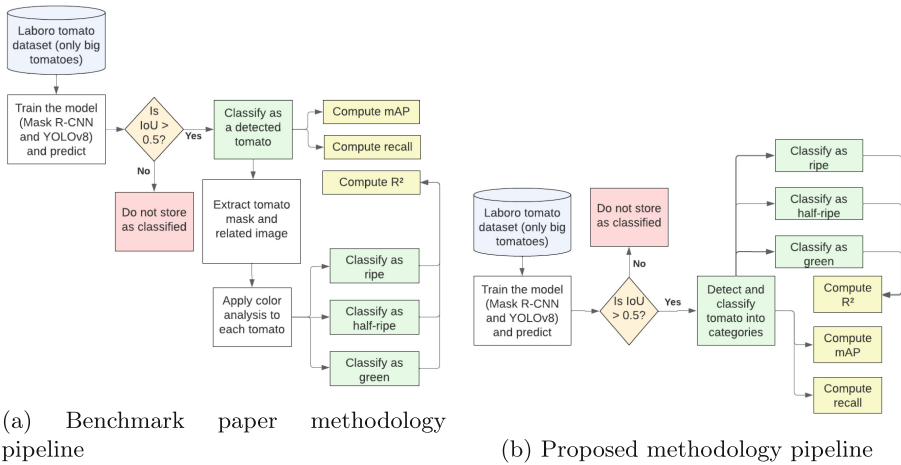


Fig. 5. Authors methodology pipeline in (5a) that will get contrasted with the proposed methodology that is illustrated in (5b).

When the work was done, the authors used dataset I, which contained specially selected images. Those images fitted well on the thresholds proposed by the authors and contained only 100 images with tomato samples. Figure 6 shows a more realistic color intraclass variation between the tomatoes. For example, on the left image is a half-ripened tomato next to a fully-ripened one that looks similar to the only one in the right image. Even all half-ripened tomatoes in the right image could be considered in a similar color range as the fully-ripened one.

Moreover, some tomato masks include a great portion of the tomato plant and leaves as in the left image from Fig. 6 where a fully-ripened tomato could be classified as green because it is occluded by the tomato plant, which can represent a bias in the color analysis, increasing the portion of green in the image

composition. By adding more green values, ripe tomatoes can be misclassified as half-ripened and half-ripened tomatoes as green. The mask accuracy also impacts the quality of the color analysis because the background in most of the samples contains leaves or parts of the tomato plant that also puts additional green color. Also the Mask R-CNN tends to generate duplicated masks for the same element, which is detected two or more times and could represent as an issue to perform the color evaluation technique.

Comparing the YOLOv8 and the Mask R-CNN models as classifiers, both perform well enough, achieving an R^2 of 0.809 in the ripe class with the YOLOv8 model as the lowest value. However, YOLOv8 was the best model in the half-ripe and green labels.



Fig. 6. Tomatoes images samples from the laboro dataset [1], where the yellow line surrounds fully-ripened tomatoes, the pink line surrounds half-ripened tomatoes, and the purple line surrounds green tomatoes.



(a) Mask R-CNN segmentation and classification sample with a close-up to the boundaries of the generated mask

(b) YOLOv8 segmentation and classification sample with a close-up to the boundaries of the generated mask

Fig. 7. YOLOv8 (b) and Mask R-CNN (a) image segmentation and classification with close-up images that highlight the mask quality.

Table 1. R^2 metric over the three different classes, comparing the Mask R-CNN, YOLOv8 models, and the color analysis in the classification task.

Dataset	Classification methodology	R^2 (ripe)	R^2 (half-ripe)	R^2 (green)
I	Mask R-CNN + color analysis ^a	0.92	0.75	0.94
II	Mask R-CNN + color analysis	-0.335	-0.214	-0.702
II	YOLOv8 + color analysis	-0.325	-0.124	-0.518
II	Mask R-CNN	0.819	0.809	0.803
II	YOLOv8	0.809	0.897	0.968

^aimages/Original results from the benchmark paper authors [12]

From the R^2 metric of the Mask R-CNN and YOLOv8 models presented in the Table 1, it is eventually that both models perform well on the segmentation task. However, the R^2 metric is insufficient to choose which model is better for the segmentation and classification task over tomatoes. In the Table 2, it is shown the recall and mAP from both models. Generally, both metrics are inverse as better the precision, worse the recall, and bit-wise. For the YOLOv8 model, both metrics are superior to the Mask R-CNN. The better performance of YOLOv8 could be because of its architecture which could be superior to the Mask R-CNN.

Table 2. Recall and mAP from the segmentation task, where the Mask R-CNN and YOLOv8 were trained with the same dataset distribution.

Dataset	Model	Recall	mAP (IoU 0.5)
I	Mask R-CNN + color analysis ^a	78.9%	67.2%
II	Mask R-CNN	77%	75.26%
II	YOLOv8	83.4%	83.7%

^aOriginal results from the benchmark paper authors [12]

Finally, some examples from the results are shown in Fig. 8 that looks similar, the YOLOv8 and Mask R-CNN models perform equally; however, in the sub-images 8a, 8d, and 8g we can appreciate more mismatches from the original images in the sub-images 8c, 8f, and 8i that the YOLOv8 model in the 8b, 8e, and 8h. Also, the masks from the YOLOv8 model are neater than those from the Mask R-CNN. However, the YOLOv8 masks also include more leaves and sections from the plants than the Mask R-CNN masks, just like in the sub-image 8h and 8g in the inferior right corner.

From Fig. 7, we can appreciate that most of the background is green, and despite this, the green tomato was the most accurate label, but also is very mismatched by the Mask R-CNN that detects tomatoes by similar mismatch on the shape and color of leaves as shown on the Fig. 7a, that can explain the lower value on R^2 by the replicated Mask R-CNN model. On the other hand, YOLOv8 has high accuracy in predicting green tomatoes. This accuracy can also be due to the considerably superior amount of green tomatoes in the dataset compared



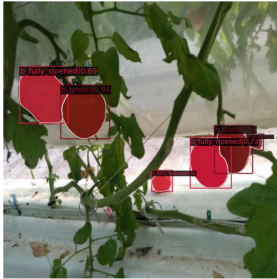
(a) Mask R-CNN full-ripen prediction



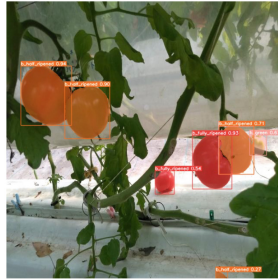
(b) YOLOv8 full-ripen prediction



(c) Original picture with full-ripen samples



(d) Mask R-CNN half-ripen prediction



(e) YOLOv8 half-ripen prediction



(f) Original picture with half-ripen samples



(g) Mask R-CNN green prediction



(h) YOLOv8 green prediction



(i) Original dataset with green samples

Fig. 8. Illustration that contains in (a), (d), and (g) the Mask R-CNN segmentation and classification, in (c), (f), and (i), the instance segmentation and classification prediction by YOLOv8 model, and in (b), (e), and (h), and the original images from laboro dataset.

to the ripe and half-ripe tomatoes. Furthermore, in Fig. 7b on the right picture, the mask area can be appreciated that covers the tomato area in a sharper way than the Mask R-CNN model on Fig. 7a in the right picture though.

6 Conclusions

In this work, we compared the Mask R-CNN and YOLOv8, for instance segmentation and classification task. In addition, we compared the models for classification against a color-based classification technique in the laboro dataset, where the models performed much better in the classification task.

The classification performed by the model is better than the color-based classification due to the number of features the models can extract from the whole image. Not only the color but the contrast, the full illumination of the picture, and the surrounding shapes can be an essential part of the image to proportion an accurate classification. The color thresholds are an inefficient measure that is not scalable not only for the specific dataset of tomatoes but also for detecting ripeness in other crops.

On the other hand, the original benchmark paper results are not consistent when testing on images with different real-world conditions. This inconsistency can be due to the few samples the laboro dataset had when this work was developed and the few images that could be destined for testing. Moreover, the authors apply this methodology to a synthetic dataset that can add biases on the classification.

Furthermore, the YOLOv8 seems to be a better classification and instance segmentation model than the Mask R-CNN. Nevertheless, training both models by changing the hyper-parameters and analyzing the resulting metrics is necessary.

7 Future Works

The color analysis methodology to perform classification is inefficient and very inaccurate, and it is better not to consider that to solve maturity recognition task problems. It could be a handful solution as a support to label new data to train a model, and there is not any model that could already do this by classifying the data.

On the other hand, to get an approach to how other architectures work with the maturity recognition task, the results of this work could be replicated, emphasizing those well-known architectures that perform well working with images.

Also, the resultant model of this work could be embedded in a system to detect tomatoes. Then, using a dataset that contains tomato disease labels as well, it could be used for spraying on focused areas, reducing the amount of pesticide.

This work is centered on tomato maturity recognition with images, but a model using real-time video that performs the same object recognition and instance segmentation task is feasible.

References

1. Laboro tomato: Instance segmentation dataset. <https://github.com/laboroai/LaboroTomato>. Accessed 5 Jan 2023
2. Ultralytics yolov8. <https://github.com/ultralytics/ultralytics>. Accessed 20 Jan 2023
3. Chen, K., et al.: MMDetection: Open MMLab detection toolbox and benchmark. arXiv preprint [arXiv:1906.07155](https://arxiv.org/abs/1906.07155) (2019)
4. Chiu, M.T., et al.: Agriculture-vision: a large aerial image database for agricultural pattern analysis. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2825–2835 (2020). <https://doi.org/10.1109/CVPR42600.2020.00290>
5. Ge, Y., et al.: Tracking and counting of tomato at different growth period using an improving YOLO-deepsort network for inspection robot. *Machines* **10**(6), 489 (2022). <https://doi.org/10.3390/machines10060489>
6. Girshick, R.: Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision (ICCV). IEEE (2015). <https://doi.org/10.1109/iccv.2015.169>
7. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN (2017). <https://doi.org/10.48550/ARXIV.1703.06870>. <https://arxiv.org/abs/1703.06870>
8. Kim, T., Lee, D.H., Kim, K.C., Choi, T., Yu, J.M.: Tomato maturity estimation using deep neural network. *Appl. Sci.* **13**(1), 412 (2022). <https://doi.org/10.3390/app13010412>
9. Lawal, M.O.: Tomato detection based on modified YOLOv3 framework. *Sci. Rep.* **11**(1), 1447 (2021). <https://doi.org/10.1038/s41598-021-81216-5>
10. Magalhães, S.A., et al.: Evaluating the single-shot MultiBox detector and YOLO deep learning models for the detection of tomatoes in a greenhouse. *Sensors* **21**(10), 3569 (2021). <https://doi.org/10.3390/s21103569>
11. Minagawa, D., Kim, J.: Prediction of harvest time of tomato using mask R-CNN. *AgriEngineering* **4**(2), 356–366 (2022). <https://doi.org/10.3390/agriengineering4020024>
12. Rahim, U.F., Mineno, H.: Highly accurate tomato maturity recognition: combining deep instance segmentation, data synthesis and color analysis. In: 2021 4th Artificial Intelligence and Cloud Computing Conference. ACM (2021). <https://doi.org/10.1145/3508259.3508262>
13. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection (2015). DOI: <https://doi.org/10.48550/ARXIV.1506.02640>
14. Shafi, U., Mumtaz, R., García-Nieto, J., Hassan, S.A., Zaidi, S.A.R., Iqbal, N.: Precision agriculture techniques and practices: from considerations to applications. *Sensors* **19**(17), 3796 (2019). <https://doi.org/10.3390/s19173796>
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). <https://doi.org/10.48550/ARXIV.1409.1556>
16. Su, F., Zhao, Y., Wang, G., Liu, P., Yan, Y., Zu, L.: Tomato maturity classification based on SE-YOLOv3-MobileNetV1 network under nature greenhouse environment. *Agronomy* **12**(7), 1638 (2022). <https://doi.org/10.3390/agronomy12071638>
17. Sun, L., et al.: Lightweight apple detection in complex orchards using YOLOV5-PRE. *Horticulturae* **8**(12), 1169 (2022). <https://doi.org/10.3390/horticulturae8121169>
18. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOV7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors (2022). <https://doi.org/10.48550/ARXIV.2207.02696>. <https://arxiv.org/abs/2207.02696>

19. Wang, C., et al.: A review of deep learning used in the hyperspectral image analysis for agriculture. *Artif. Intell. Rev.* **54**(7), 5205–5253 (2021). <https://doi.org/10.1007/s10462-021-10018-y>
20. Zhang, N., Wang, M., Wang, N.: Precision agriculture—a worldwide overview. *Comput. Electron. Agric.* **36**(2–3), 113–132 (2002). [https://doi.org/10.1016/S0168-1699\(02\)00096-0](https://doi.org/10.1016/S0168-1699(02)00096-0)
21. Zhang, W., et al.: Lightweight fruit-detection algorithm for edge computing applications. *Front. Plant Sci.* **12**, 740936 (2021). <https://doi.org/10.3389/fpls.2021.740936>