Article:
Aljaafreh, A., Elzagzoug, E.Y., Abukhait, J. et al. (2023) A real-time olive fruit detection for harvesting robot based on YOLO algorithms. Acta Technologica Agriculturae, 26 (3). pp. 121-132. ISSN: 1338-5267

https://doi.org/10.2478/ata-2023-0017

sciendo

DOI: 10.2478/ata-2023-0017

Acta Technologica Agriculturae 3
Nitra, Slovaca Universitas Agriculturae Nitriae, 2023, pp. 121–132

A REAL-TIME OLIVE FRUIT DETECTION FOR HARVESTING ROBOT BASED ON YOLO ALGORITHMS

Ahmad ALJAAFREH[1], Ezzaldeen Y. ELZAGZOUG[2], Jafar ABUKHAIT[1]*, Abdel-Hamid SOLIMAN[3], Saqer S. ALJA'AFREH[4], Aparajithan SIVANATHAN[2], James HUGHES[2]

1Tafila Technical University, College of Engineering, Computer and Communication Engineering Department, Tafila, Jordan, a.aljaafreh@ttu.edu.jo
[2]The University of Sheffield, AMRC North West, Digital Department, Blackburn, United Kingdom, e.elzagzoug@amrc.co.uk (E. Y. E.), a.sivanathan@amrc.co.uk (A. S.), j.hughes@sheffield.ac.uk (J. H.)
[3]Staffordshire University, School of Digital, Technologies and Arts, Department of Engineering, Stoke-on-Trent, United Kingdom, a.soliman@staffs.ac.uk
[4]Mutah University, College of Engineering, Electrical Engineering Department, Karak, Jordan, eng.saqer_jaa@mutah.edu.jo
*correspondence: jafar@ttu.edu.jo

Deep neural network models have become powerful tools of machine learning and artificial intelligence. They can approximate functions and dynamics by learning from examples. This paper reviews the state-of-art of deep learning-based object detection frameworks that are used for fruit detection in general and for olive fruit in particular. A dataset of olive fruit on the tree is built to train and evaluate deep models. The ultimate goal of this work is the capability of on-edge real-time olive fruit detection on the tree from digital videos. Recent work in deep neural networks has led to the development of a state-of-the-art object detector termed You Only Look Once version five (YOLOv5). This paper builds a dataset of 1.2 K source images of olive fruit on the tree and evaluates the latest object detection algorithms focusing on variants of YOLOv5 and YOLOR. The results of the YOLOv5 models show that the YOLOv5 new network models are able to extract rich olive features from images and detect the olive fruit with a high precision of higher than 0.75 mAP_0.5. YOLOv5s performs better for real-time olive fruit detection on the tree over other YOLOv5 variants and YOLOR.

Keywords: object detection; olive harvesting; YOLO; deep learning; computer vision

Automation has contributed to different agriculture applications such as planting, harvesting, disease recognition, production estimation, quality control, water management, crop monitoring, control of insecticides, and soil quality and pesticides. Among these applications, harvesting is the process that has received the least amount of technological development for satisfactory automation, until now most fruit and vegetable harvesting is based on manual techniques (Jimenez et al., 2000).

In some European Mediterranean countries such as Spain, Italy, Greece, and Turkey, olive fruit has strong agricultural importance, being a big part of the economy. Currently, Spain is the leading producer of olives and produces 5,276,899 metric tons of olives on more than 2.4 million hectares of dedicated land. However, despite worldwide production tripling in the last 60 years, the worldwide olive oil consumption rate has kept pace with the production rate, as stated in the last blog of the International Olive Council (International Olive Council, 2021). This is unsurprising, as olive oil represents an important dietary source that has currently entered the production of other foods. Additionally, olive planting is a traditional part of the social, economic, and environmental importance to agriculture in many regions.

Accordingly, one of the important factors that could affect olive oil productivity is the harvesting method. Normally, olive harvesting is done by hand, which is a time-consuming, tedious, and costly process. It involves a large number of employees and, therefore, high labour costs. Mechanical olive harvesters can be used to minimize the time and cost of production. However, during processing, mechanical harvesters can cause damage such as local tissue degradation, combined with intracellular water output and the oxidation of phenolic compounds after impact. A vision system could be applied to help configure mechanical harvesters to support and sustain quality, time, and cost. To the best of authors' knowledge, an AI-based or smart and automated olive fruit harvester has not been investigated, although this technique has had success in harvesting other kinds of fruits.

Recently, agricultural autonomous robots, namely Agrobots, have emerged with several agricultural applications to increase productivity and operation safety (Mavridou et al., 2019). They mainly deploy computer vision and machine learning techniques in fruit detection, weed detection, plant disease detection, fruit quality prediction, and fruit maturity prediction (Koirala et al., 2019; Zhang et al., 2020; Bah et al., 2018). Robotic vision, based on feature algorithms or deep learning algorithms, is required in fruit detection to guide the robot arm to detach the fruit (Kang and Chen, 2020b).

Several studies have been conducted on image features' categorization for fruit/vegetable detection. The proposed scheme in Bolle et al. (1996) represents the first attempt to develop a system which can analyse the fruit's colour, texture, and density for supermarket applications. In Nguyen et al. (2016), colour and geometric features have been deployed in a clustering algorithm for red apple detection. In Wang and Xu (2018), multiple image features and the Latent Dirichlet Allocation (LDA) model were deployed for unsupervised instance segmentation of the fruit.

Deep learning algorithms have been used extensively in fruit detection because of their higher performance accuracy. In Kang and Chen (2020a), researchers developed a framework of a deep learning-based fruit detection algorithm and clustering-based classifier to assist fast labelling of training data. In Bresilla et al. (2019), an approach for fruit detection, based on state-of-the-art deep neural networks techniques using single-shot YOLO detectors to detect apples and pears in the tree canopy, has been presented. The results have shown that modifications to the input grid on the standard model of YOLO yield better detection.

A considerable number of studies have focused on evaluating the trees' health, disease, trees detection, trees counting and olive trees quality testing. In Beyaz and Ozturk (2016), the olive cultivars were identified using image processing techniques based on the genetic identification method. In Di Nisio et al. (2020), a hybrid approach (combination of multispectral information and spatial data) was presented to monitor the spread of olive quick decline syndrome (OQDS) in olive trees. In Martinelli et al. (2019), an image processing-based technique using iTRAQ method was introduced to detect and classify the spot disease based on using the analysis of olive tree leaf textures. A multi-step algorithm to automatically detect and count the olive trees in satellite images has been presented in Khan et al. (2018).

There is no existing research concerning olives detection for harvesting applications based on Deep Learning (DL) techniques. The main reason for the lack of research in this area is the existing challenges. This includes the olives' small fruit size that means the acquired images may contain fruit with any number, different sizes, colours, random position, and shape. In this study, working in a real-life environment created additional challenges that should be considered, such as olive fruits occlusion with themselves or with other parts of the tree (leaves or/and branches), shadows, and lighting conditions. These challenges show the importance of selecting a suitable machine learning model that has high detection accuracy and achieves real-time detection for on-edge applications.

The smart system of olive harvesting should have the ability to detect and localize olive fruit from digital images. The proposed design aims to guarantee the efficiency and consistency of the method of olive harvesting. This paper builds a dataset of 1200 source images of olive fruit on the tree and evaluates the latest object detection algorithms focusing on variants of YOLOv5 and YOLOR. The results of the YOLOv5 models show that the YOLOv5 new network models are able to extract rich olive features from images and detect the olive fruit with a high precision of higher than 0.7 mAP_0.5.

The rest of the paper is organized as follows: Section 2 gives a brief background on YOLO approaches and the olive detection model implementation. Section 3 demonstrates and discusses the results while section 4 concludes the paper along with future work.

Material and methods

YOLO

You Only Look Once (YOLO) is a new approach for object detection (Redmon and Farhadi, 2017). In YOLO, objects can be detected and located at one glance (Du, 2018). YOLO divides the input image into N grids, each with equal dimensions of S×S. Each grid is responsible for the detection and localization of the object it contains. YOLO predicts the coordinates of bounding boxes directly using fully connected layers on the top of the convolutional feature extractor. Several versions of YOLO have been developed to enhance its performance, YOLO, YOLOv2 and YOLO9000, YOLOv3, YOLOv4, and YOLOv5. YOLO models have a high performance and are appropriate for on-device deployment.

In 2020, Glenn Jocher introduced YOLOv5 (Jocher, 2020), whilst the model architecture remains close to YOLOv4, it derives most of its performance improvement from PyTorch training procedures. The major YOLOv5 improvements include mosaic data augmentation and auto-learning bounding box anchors. The release of YOLOv5 includes different models' sizes: YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. YOLOv5 is superior to YOLOv4 in terms of speed, accuracy, and size (Nelson and Solawetzet, 2020).

YOLOR was published in May 2021 (Wang et al., 2021) and stands for "You Only Learn One Representation". It is proposed as a unified network to encode implicit knowledge and explicit knowledge together. YOLOR aims to implement a technique that can serve many tasks for a given one input. YOLOR is designed to be specifically for object detection, rather than other machine learning use cases such as object identification or analysis.

Olive detection model implementation

To simplify the implementation of the olive detection model, Fig. 1 shows the main and detailed steps of the implementation process. The process includes three main steps:

1. Data preparation: in which olive trees' images are collected, pre-processed, annotated, and augmented to build the dataset for training, evaluating, and testing the YOLO model.
2. Model implementation: in which a deep learning model is selected and trained on both the training and validation datasets and then, evaluated on the test dataset.
3. Model inference: in which the detection model is deployed on real-life olive images.

Data preparation

The proposed detection model aimed to detect the olive fruit on twigs and branches. This model was implemented

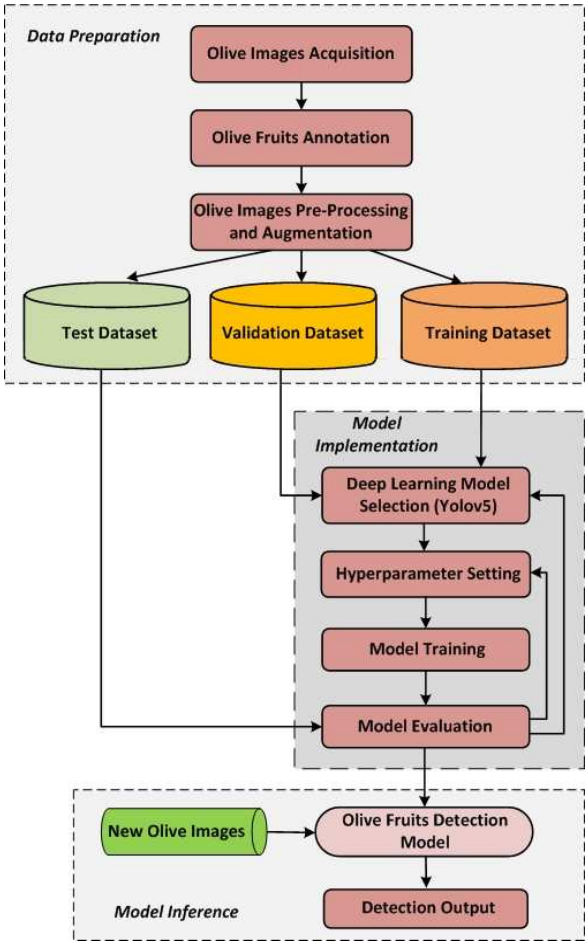


Fig. 1 Flowchart of the olive detection model implementation

and trained on real olive fruit images that are acquired under different circumstances. Thus, it can be deployed on-edge to detect olive fruits on the tree using a digital camera. This sub-section demonstrated the image acquisition process of olive trees and branches, in addition to steps followed to annotate olive fruit and implementing a state-of-art dataset.

Image acquisition

The images of olive twigs and branches were captured using an RGB camera with a resolution of 2736 × 3648 pixels from 10 olive farms in Jordan. A set of 1200 source images was collected of different olive trees (Nabali, Rasie Nassohi and Souri) in Irbid, Tafila, Madaba, and Karak cities. All images were captured under natural daylight from 10:00 am to 6:00 pm to obtain varying illumination conditions. The digital camera was mounted at a height between 0.5 and 1.5 m with a distance range to tree of between 0.5 and 1.5 m. These distances were selected to simulate the camera mounted on a robotic arm in the final phase of the system. This should assist a future automated harvesting robot in locating and harvesting the designated areas. Images were categorized based on capturing illumination and shadowing as 820 images under high illumination (collection time from 10:00 am to 3:00 pm) and 380 under low illumination (collection time from 3:00 pm to 6:00 pm). Additional categorization was also based on olive fruit colours, using 750 and 450 images of green and black olives, respectively. These variations of source images increased the detection efficiency and made the detection model invariant to scale, colour, and illumination. All images were resized to a resolution of 1094 × 1459 pixels. Figure 2 shows the samples of olive images.

Fig. 2 Sample examples of olive images

Annotation, preprocessing, augmentation, and dataset preparation

Olive fruit in the original images was labelled using LabelMe, an open access annotation tool (Russell et al., 2008). A number of 40,834 different olive fruits were annotated with class "olive" across the 1200 images. These annotations include partially occluded and clearly visible olive fruits. Figure 3 shows some annotations of olive fruits on the original images. Images were uploaded to Roboflow, a computer vision platform to construct the dataset, and divided into training, validation, and test datasets. All images were resized to 800×600 pixels, augmented to increase the size of image dataset to 4800 images. Three augmentation parameters were applied to the source images: Rotation: between -18° and +18°; Brightness: between -19% and +19%, and Blurring: up to 1.25 px.

Model implementation

The performance of object detection improved considerably after the advent of the YOLO and R-CNN families. Convolutional layers are employed in most image-related neural networks. Convolutional Neural Networks (CNNs) learn image representations by performing a sliding window approach. The selection of a detection model, training and evaluation are described in this section.

Model selection

YOLOv5s was selected as the olive object detection model after trialling and comparing it with YOLOv5x and YOLOR for accuracy and speed. For the real-time application discussed in the paper, speed is more critical than accuracy. Different YOLO variants models such as Yolo 5 X, S and YoloR were tested on the olive dataset which has 1200 source images. The source images were augmented to generate 4800 images using the Roboflow platform. The dataset was divided into three subsets: a set of 4000 training images, a set of 400 validation images, and a set of 400 testing images.

The network architecture of YOLOv5s, as shown in Fig. 4, consists of three parts: (1) Backbone: CSPDarknet, (2) Neck: PANet, and (3) Detect: YOLO Layer. The image was first inputted to CSPDarknet for feature extraction, and then fed to PANet for feature fusion. Finally, YOLO Layer outputs detection results (class, score, location, size).

Model hyperparameters optimization

The use of network architecture, such as YOLOv5s, and optimizing hyperparameters are both effective and robust in detecting and localizing olives as an object. Hyperparameters are variables that determine the network topology for example, how the network is trained (e.g., Learning Rate). Initially, hyper parameters were selected before training and optimizing the network parameters (weights and bias).

Tuning hyperparameters of the YOLO model to improve model performance and precision is a challenging job, because of the time required to train models based on different hyperparameters such as Anchor, Learning Rate, and Weight Decay with wide range of values. Table 1 illustrates four different hyperparameter combinations (A, B, C, and D) for olive detection model performance evaluation.

Anchor: is a predefined boundary defined boxes with a set of height and size. They are used to improve the accuracy and speed of the model.

Learning Rate: is a tuned parameter determining the step size of each iteration to minimize loss function.

Weight Decay: is a regularization technique used to avoid the overfitting of the model.

Fig. 3 Examples of annotated images including occluded and non-occluded olive fruits

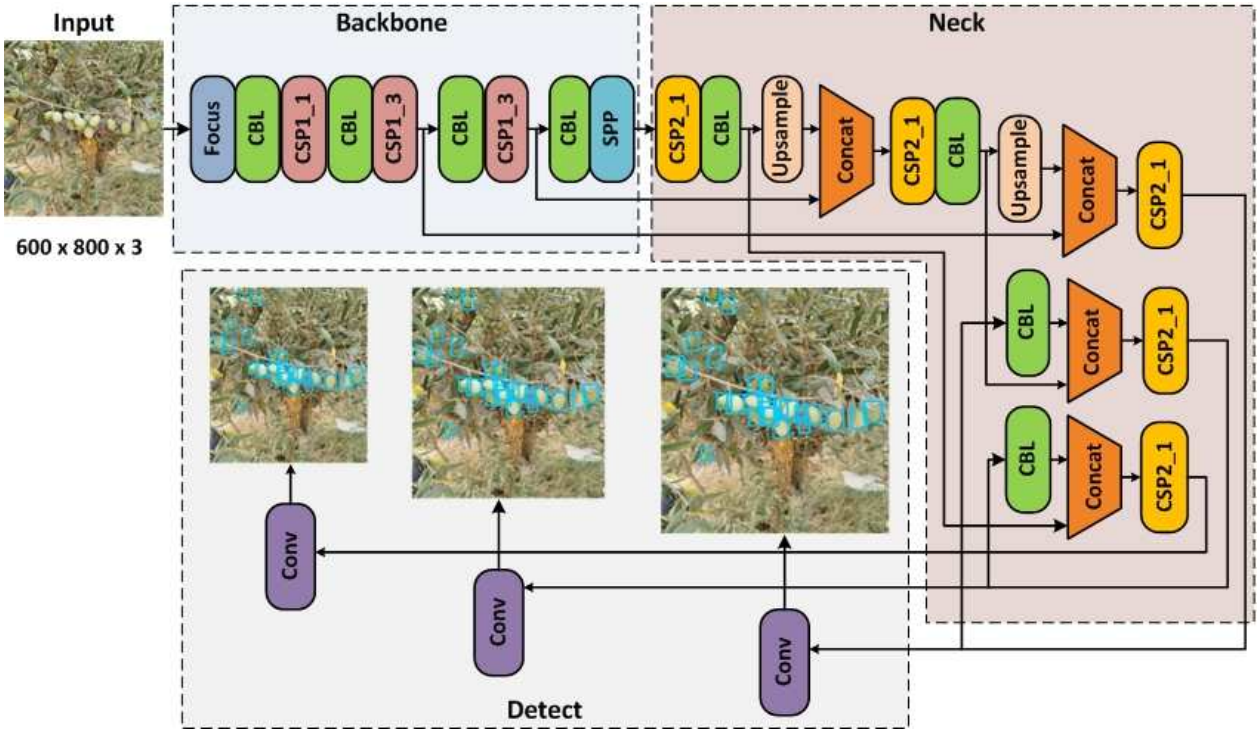


Fig. 4 YOLOv5s architecture
CSP – cross stage partial tetwork; CBL – convolution-batch normalization – leak ReLU; SPP – spatial pyramid pooling; Concat – concatenation function; Conv – convolutional layer

Table 1 Optimized hyperparameters

	Anchor	Learning rate	Weight decay
A	4	0.01	0.0005
B	6	0.01	0.0005
C	3	0.001	0.0005
D	4	0.01	0.05

Experimental results of object detection models are shown in Table 2. These results are based on augmented dataset which was trained using hyperparameter optimization and transfer learning. The mean average precision (mAP_0.5) was compared for the different models to evaluate accuracy. In addition, the precision parameter is an additional parameter calculating the accuracy of the models. The hyperparameter group A has the best precision compared to the other hyperparameter groups. Comparing the models in the group A, the YOLOv5x model seems to provide higher precision compared to the YOLOv5s model.

Different hyperparameter combinations were evaluated using the YOLOv5s model for the olive images dataset. This evaluation process took approximately 96 hours using a high-performance computer with 2X Nvidia A100 GPUs, 2 TB memory, and approximately 4800 images. The patch was set to 64 and epoch to 300 for evaluation.

The output results illustrated in Fig. 5 show the best hyperparameter found from the evaluation training on olive images using YOLOv5s.

The training box loss was compared for different models with different combination of hyperparameters (A, B, C, and D), as shown in Fig. 6. The YOLOv5 model S with the hyperparameter optimized categories D and C shows low loss compared to the categories A and B (Fig. 6a), which

Table 2 The difference in precision between the A, B, C, and D categories once models have been trained

Name	Hyperparameter	mAP_0.5	Precision	Box_loss	Obj_loss
YOLOv5x	D	0.7708	0.4279	0.0605	0.2696
YOLOv5s		0.7265	0.3871	0.0639	0.2818
YOLOv5x	C	0.7116	0.3991	0.0463	0.2104
YOLOv5s		0.6827	0.3794	0.0517	0.2269
YOLOv5x	B	0.7330	0.4873	0.0619	0.2574
YOLOv5s		0.7384	0.4045	0.0703	0.3066
YOLOv5x	A	0.7559	0.4675	0.0507	0.2255
YOLOv5s		0.7413	0.4366	0.0565	0.2538

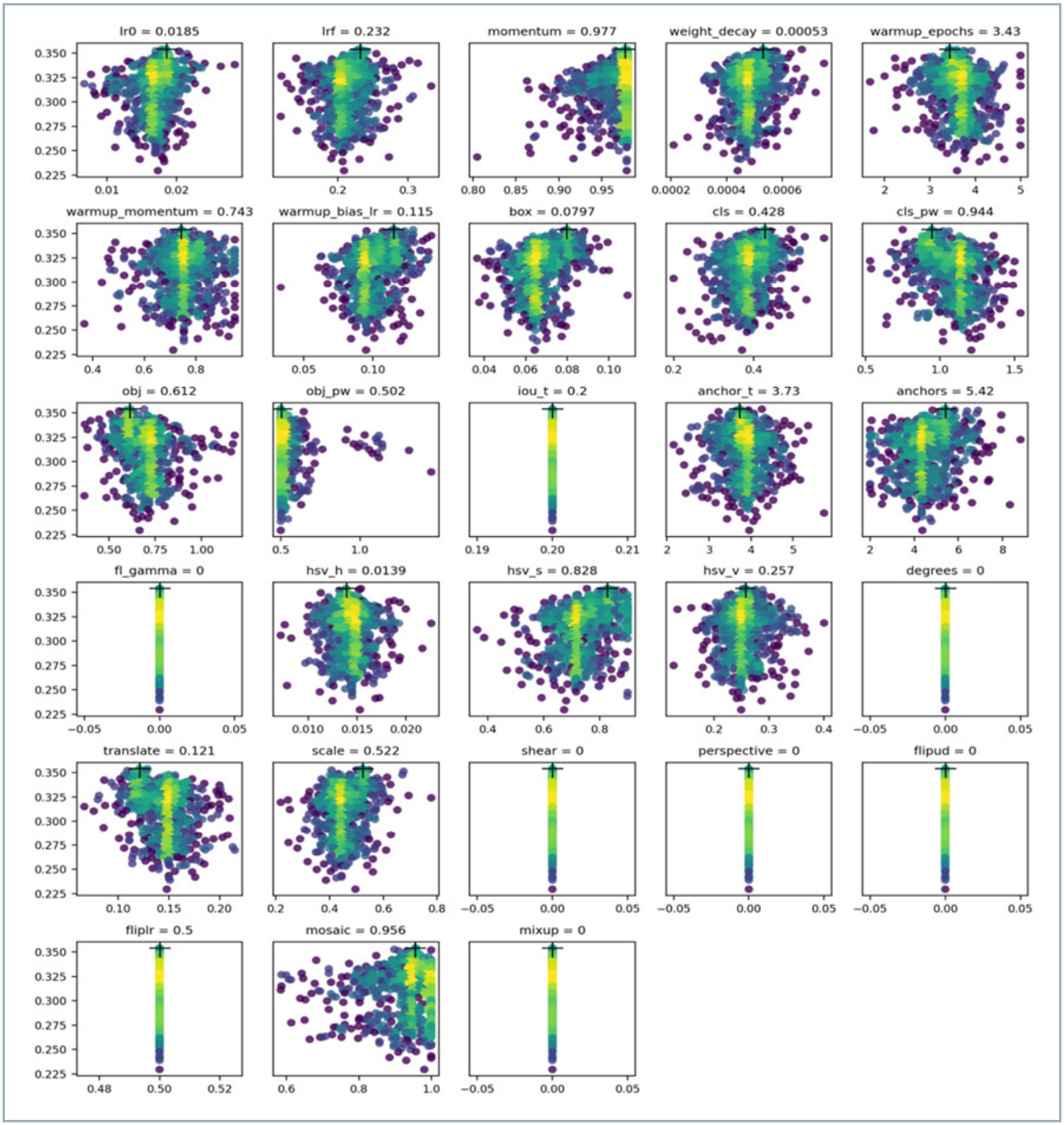
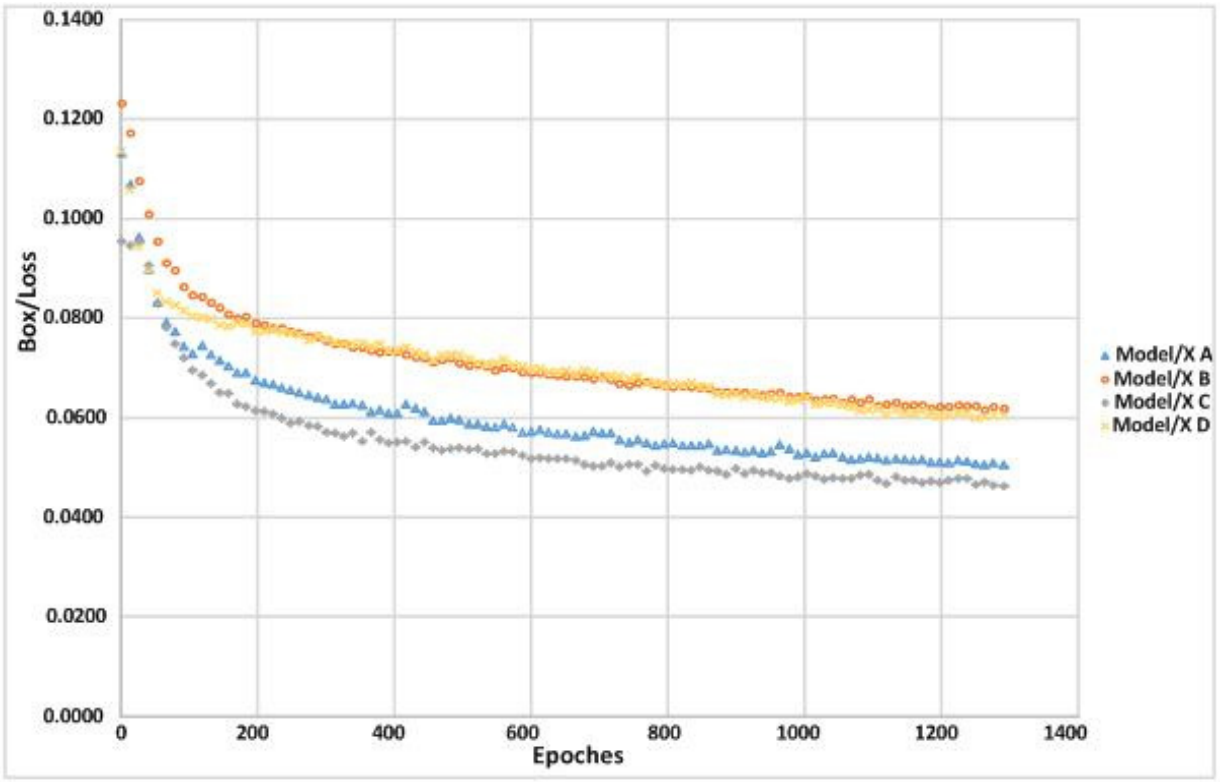
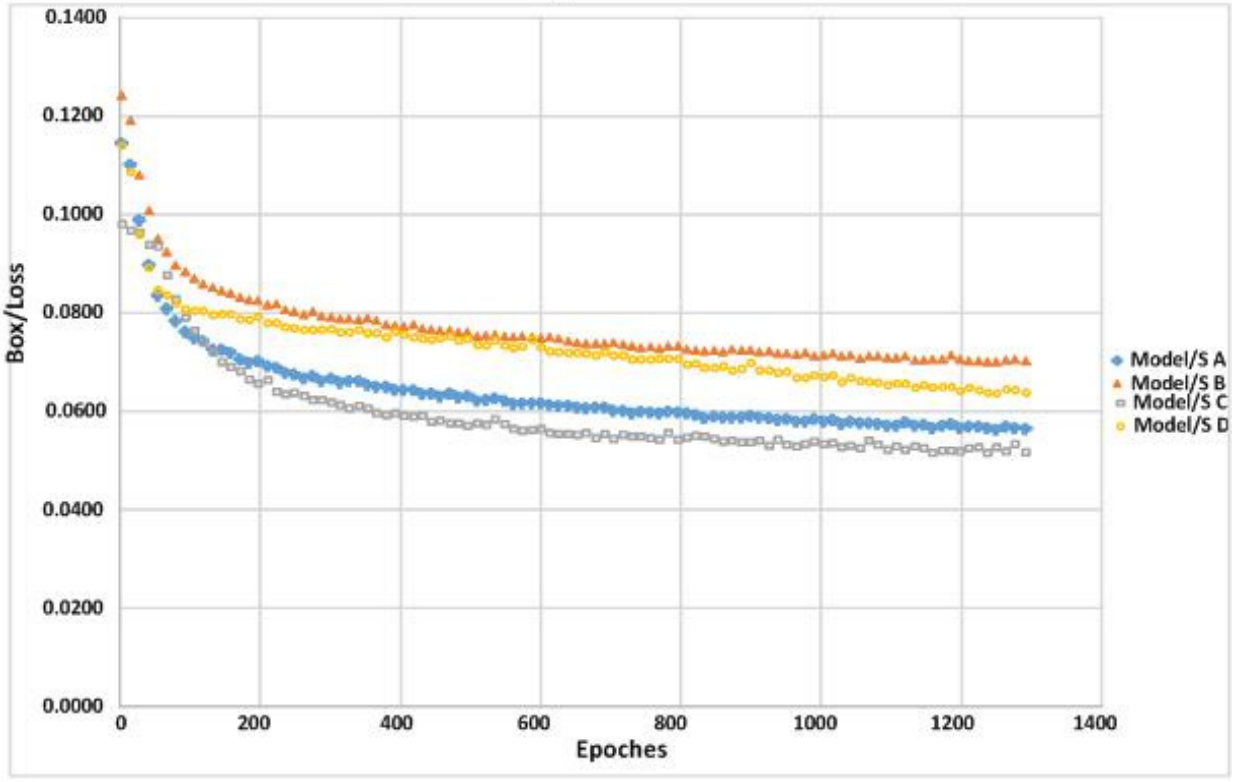


Fig. 5 Best olive training hyperparameters evaluated by the YOLOv5s model. The hyperparameters evolution has one subplot per hyperparameter. The *X*-axis shows the hyperparameter value vs *Y*-axis shows the fitness value. The higher concentration value is shown in yellow; the vertical distribution illustrates the deactivated hyperparameters and does not mutate



(a)



(b)

Fig. 6 YOLOv5 Box\Loss curves of the two models YOLOv5s and YOLOv5x and the combinations of hyperparameters settings A, B, C, and D vs epochs. Figure (a) is for YOLOv5s, and figure (b) is for YOLOv5x

implies the categories D and C demonstrated a better performance in classifying input data and output targets. However, Fig. 6b represents the model X, which shows slightly lower loss for the hyperparameter categories A and C compared to the model S categories A and C.

Experimental platform and model training

The model was trained using the YOLOv5 variants (YOLOv5s, YOLOv5x) and YOLOR. These models are evaluated in this work for olive fruit detection for only one class named "olive".

Model evaluation indicators

The model detection performance was evaluated using mean average precision (mAP), recall, precision, train and validation bounding box loss, and object classification loss metrics. The evaluation metrics that were used to evaluate the model are explained as follows:

Precision is a measure of a network's ability to accurately identify targets at a single threshold, calculated by:

$$precision = \frac{true\ positive}{actual\ results} \text{ or } \frac{Tp}{Tp + Fp} \tag{1}$$

Recall is a measure of the network's ability to detect its target, calculated by:

$$recall = \frac{olive\ positive}{predicted\ results} \text{ or } \frac{Tp}{Tp + Fn} \tag{2}$$

Intersection over Union (*IoU*) is a method used to compare two arbitrary shapes, i.e., object widths, heights, and location of two boxes into the original region. This will evaluate the precision of the object detector on particular data set (Rezatofighi et al., 2019), as in Eq. (3). Figure 7 shows how IoU is calculated diagrammatically.

$$IoU = \frac{area\ of\ overlap}{area\ of\ union} \tag{3}$$

Average precision is a method combining recall and precision for the entire ranking. It is the average of precision in a single ranking (Everingham et al., 2010):

$$AP = \frac{1}{|class|} \sum_{c \in class} \frac{TP(c)}{TP(c) + FP(c)} \tag{4}$$

Mean average precision (*mAP*) is the average of precision values at the rank where there is a relevant document. It is calculated from precision, recall, and interception over union *IOU*.

$$mAP = \frac{AP}{total\ number\ of\ class} \tag{5}$$

where: *Tp* – are the Bounding Boxes (BB) that have the intersection over union (*IoU*) with the ground truth (GT) above 0.5; *Fp* – two cases – (a) BB that have IoU with GT below 0.5, (b) BB that have *IoU* with GT that has already been detected; *Tn* – there are no true negatives, the image is expected to contain at least one object; *Fn* – images containing an object where the method failed to produce a BB

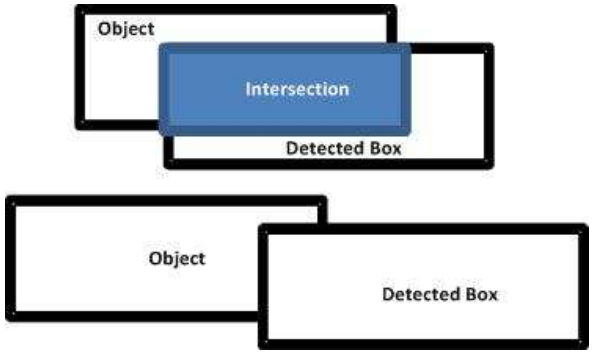


Fig. 7 Diagrammatic example intersection over union calculation

Training loss is the error cost of bad prediction based on the difference between the predicted value and the true value.

$$loss = \sum_{i=0}^{S^2} (coord\ Err + iou\ Err + cls\ Err) \tag{6}$$

Validation loss is the same metric as training loss, but it is not used to update the weights. However, it is used to find the best combination of hyperparameters in order to ensure better model generalization and avoid overfitting.

Validation loss is calculated by comparing the output *Y* of the validation set with ground truth Y_t using a loss function:

$$J = \frac{1}{N} \sum_{i=1}^{N} L(Y, Y_t) \tag{7}$$

where: *L* – the individual loss function based on the difference between predicted value and target

Results and discussion

This paper builds a dataset of 1.2 K source images of olive fruit on the tree and evaluates the latest object detection algorithms focusing on the variants of YOLOv5 and YOLOR. The results of the YOLOv5 models show that the YOLOv5 new network models can extract rich olive features from images and detect the olive fruit with a high precision above 0.75 mAP_0.5. YOLOv5s performs the best for real-time olive fruit detection on the tree over other YOLOv5 variants and YOLOR. Furthermore, the latency will be compared for different object detection models that have adequate precision.

Models' precision comparison

To evaluate the final detections, mAP_0.5 was measured with averaging over IoU thresholds at [0.5: 0.05: 0.95] for the hyperparameter category A. The mAP_0.5 results in this system were compared for YOLOv5s with the original YOLOv5s results. The mAP_0.5 results were also compared

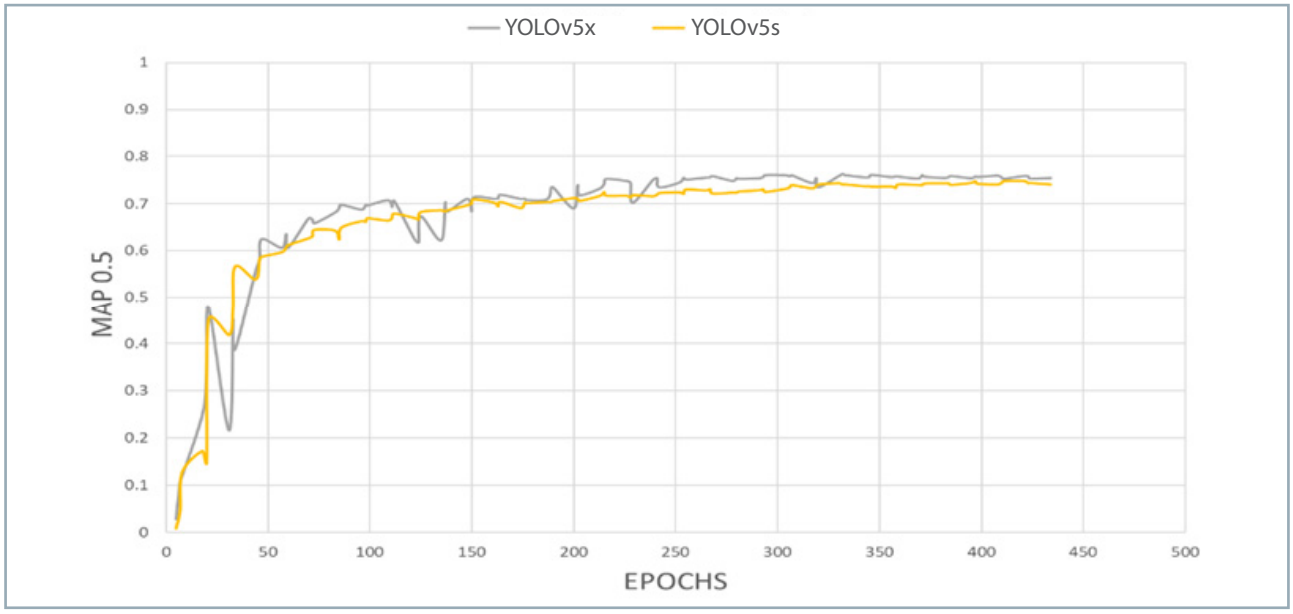


Fig. 8 Mean average precision (mAP_0.5) curve for the model YOLOv5x and YOLOv5s

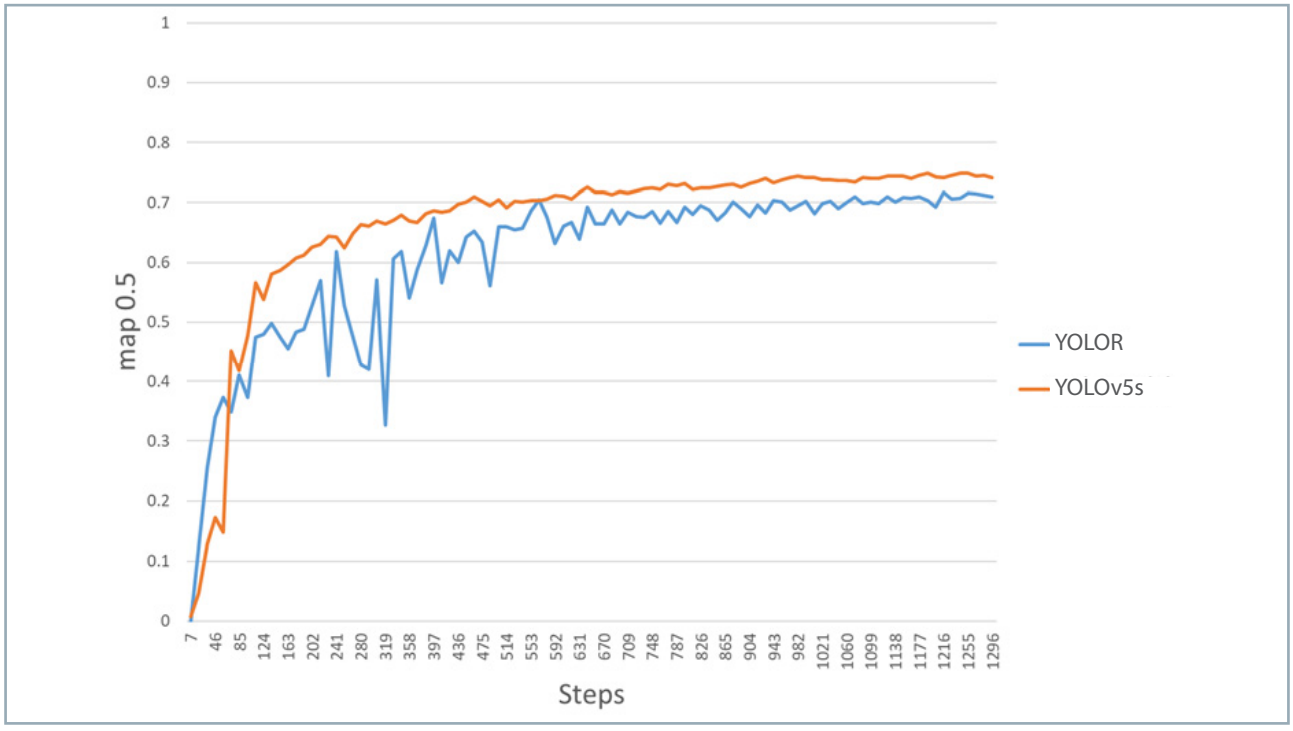


Fig. 9 mAP_0.5 comparison between YOLOv5s and YOLOR

for YOLOv5s, YOLOv5x, and YOLOR when trained on the olive dataset.

In this section, the mAP_0.5 value was compared for the two variants of YOLOv5, named YOLOv5s and YOLOv5x. These two variants were specifically chosen because YOLOv5x is the largest and YOLOv5s is the smallest. Hence, there was no need to compare mAP_0.5 with other YOLOv5 variants. Figure 8 shows mAP_0.5 for the model YOLOv5x and YOLOv5s. As shown in Fig. 8, the steady-state mAP_0.5 value is approximately 0.75 for both models. The mAP_0.5 value is considered high compared with the mAP_0.5 value of the original YOLOv5s, as shown in Table 3. This mAP_0.5 value indicates high detection precision for the olive harvesting application.

The original YOLOv5 is a family of object detection architectures and models pre-trained on the COCO dataset. YOLOv5s was trained using a V100 GPU.

Recently developed YOLOR was also tested and compared with YOLOv5s using the olive dataset. Figure 9 shows that the mAP_0.5 values of both YOLOv5s and YOLOR after sufficient training are 0.75 and 0.7, respectively. There is approximately 0.05 mAP_0.5 difference between YOLOR

Table 3 Comparison between the original YOLOv5s trained on COCO dataset (Jocher, 2020) and YOLOv5s model trained on olive dataset

Model	Dataset	Dataset size	No. of classes	Size (pixels)	mAP_0.5	GPU	Speed (ms)	Params (M)
Original YOLOv5s	COCO	5000	80	640 × 640	55.4	V100	2.0	7.3
YOLOv5s in this paper	Olive	4800	1	600 × 800	75.0	A100	16.0	7.3

Table 4 Detection speed comparison between YOLOR, YOLOv5x, and YOLOv5s

	YOLOv5s	YOLOv5x	YOLOR
Latency	0.016s	0.031s	0.039s
Speed	62 FPS	32 FPS	25 FPS

and YOLOv5s. Therefore, YOLOv5s outperforms YOLOR in terms of precision.

Models' latency comparison

The main purpose of this research is to process the real-time streaming video for the olive harvesting process. The detection of olives on the tree should be real-time and on edge as well; therefore, low latency is crucial for olive harvesting application. In this subsection, latency is compared for the YOLOv5s, YOLOv5x, and YOLOR models that achieved an adequate precision as shown in the previous section.

Object detection models were tested on 50 images to compare the detection speed, which is the opposite of the model latency. Table 4 shows a comparison of detection latency among the YOLOv5s, YOLOv5x, and YOLOR models.

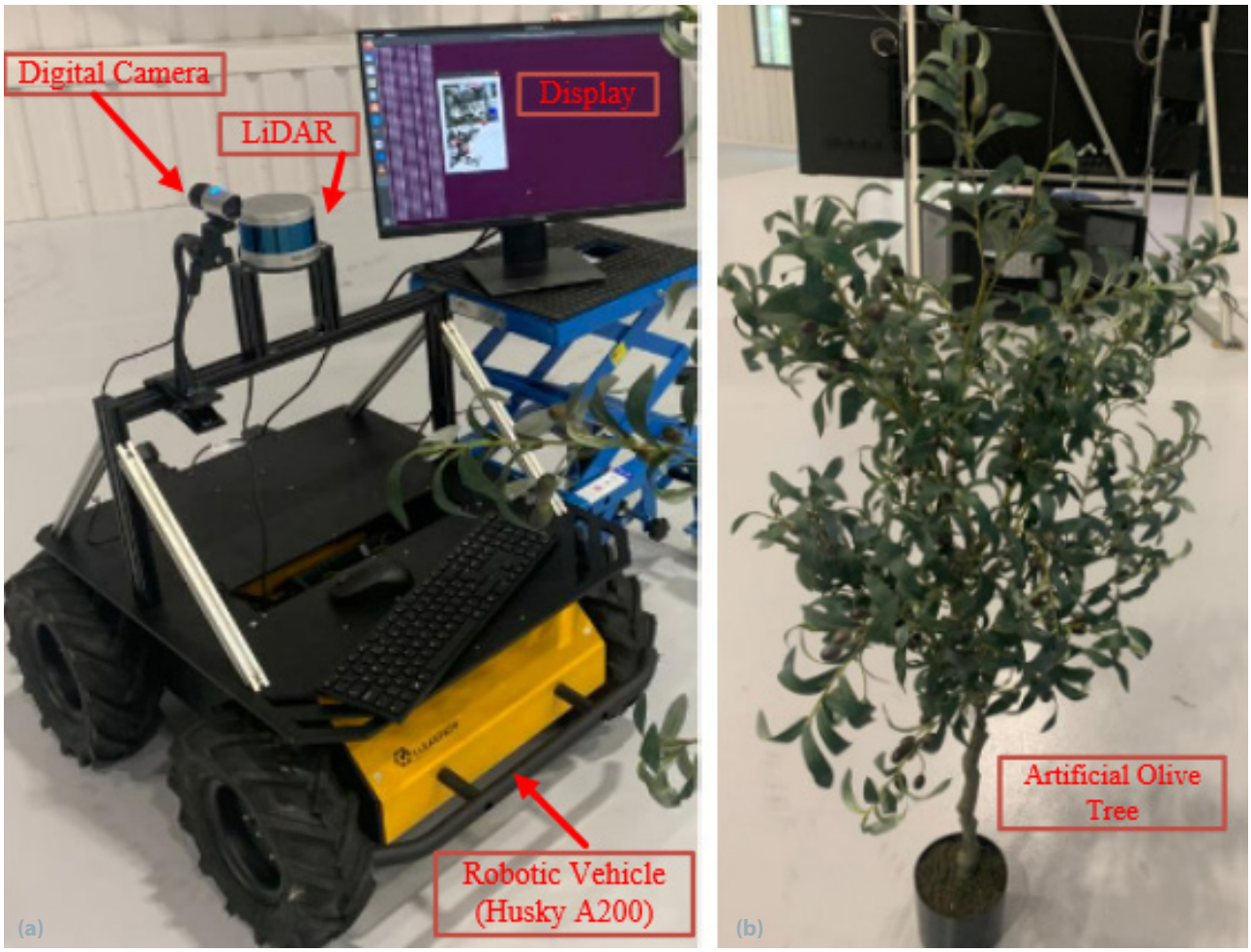


Fig. 10 a) Hardware setup of the real-time olive detection model implementation; b) Artificial olive tree

As shown in Table 4, YOLOv5s outperforms the other two models in olive detection speed. An image took an average of 0.016 seconds. Therefore, YOLOv5s can achieve 62 frames per second (FPS).

Real-time implementation

The detection model was also deployed on edge to validate its real-time performance. The hardware (as shown in Fig. 10a) has been setup using the following components:

1. Lenovo Thinkcentre M710Q PC is equipped with an Intel Core i3-6100T processor and 8 GB Memory on Ubuntu 20.0;
2. Microsoft LifeCam Studio Webcam;
3. Husky A200 UGV;
4. Velodyne LiDAR 3d;
5. digital screen.

The experiment was performed at AMRC North West labs on an artificial olive tree (shown in Fig. 10b).

Object detection inference time depends on the hardware specifications where the model is implemented. Moreover, inference time also depends on images size.

The Husky robot moves around the tree capturing a real-time video using the mounted webcam, as shown in Fig. 10. The size of each captured image is 415 × 289 pixels. Real-time olive fruit detection and recognition is performed on the captured images and recognition results are displayed on the digital screen.

Inference time is proportionally dependent on the number of detected olive fruits. The actual range of inference time per image ranges between 56 ms and 3.395 s.

Conclusion

The need for agricultural revolution is well recognized and the application of real-time systems in various aspects of the farming industry is a necessity. Artificial intelligence and data analysis play a key role in real-time systems and many opportunities for its application exist within the agriculture harvesting chain. Yet there are areas of the farming industry considered to be challenging for the current digitalization revolution such as olive harvesting. This project, novel olive harvesting is focused on researching the best practices to improve olive harvesting and farm productivity.

The use of a real-time compatible system with high speed (YOLOv5) capabilities to localize olives on the tree as an object has been discussed in this paper. This application helps improve the quality and productivity of olive farms and could provide the required data for forecasting futuristic yielding.

Considering multiple versions of AI algorithms taking into account real-time speed, accuracy, and model size, the YOLOv5 model satisfied all olive project requirements. The YOLOv5 model is small in size (i.e., requires less processing) with high speed (i.e., best suited for real-time applications) compared to the other models such as the YOLOv5 model x. Notwithstanding that, the model is able to localize more than 95% of the olives on the tree.

Acknowledgement

This work was supported by the Royal Academy of Engineering/Transforming Systems through Partnership Programme (TSP) under Grant (Project no: TSP 1372), the funded project entitles "A Novel Design for Smart Olive Harvester".

References

BAH, M. D. – HAFIANE, A. – CANALS, R. 2018. Deep learning with unsupervised data labeling for weed detection in line crops in UAV images. In Remote Sensing, vol. 10, no. 11, article no. 1690. DOI: https://doi.org/10.3390/rs10111690

BEYAZ, A. – OZTURK, R. 2016. Identification of olive cultivars using image processing techniques. In Turkish Journal of Agriculture and Forestry, vol. 40, no. 5, pp. 671–683. DOI: https://doi.org/10.3906/tar-1504-95

BOLLE, R. M. – CONNELL, J. H. – HAAS, N. – MOHAN, R. – TAUBIN, G. 1996. VeggieVision: A produce recognition system. In Proceedings of the Third IEEE Workshop on Applications of Computer Vision, pp. 244–251. DOI: https://doi.org/10.1109/acv.1996.572062

BRESILLA, K. – PERULLI, G. D. – BOINI, A. – MORANDI, B. – CORELLI GRAPPADELLI, L. – MANFRINI, L. 2019. Single-shot convolution neural networks for real-time fruit detection within the tree. In Frontiers in Plant Science, vol. 10, article no. 611. DOI: https://doi.org/10.3389/fpls.2019.00611

DI NISIO, A. – ADAMO, F. – ACCIANI, G. – ATTIVISSIMO, F. 2020. Fast detection of olive trees affected by *Xylella fastidiosa* from UAVs using multispectral imaging. In Sensors, vol. 20, no. 17, article no. 4915. DOI: https://doi.org/10.3390/s20174915

DU, J. 2018. Understanding of object detection based on CNN family and YOLO. In Journal of Physics: Conference Series, vol. 1004, article no. 012029. DOI: https://doi.org/10.1088/1742-6596/1004/1/012029

EVERINGHAM, M. – VAN GOOL, L. – WILLIAMS, C. K. I. – WINN, J. – ZISSERMAN, A. 2010. The PASCAL visual object classes (VOC) challenge. In International Journal of Computer Vision, vol. 88, no. 2, pp. 303–338. DOI: https://doi.org/10.1007/s11263-009-0275-4

INTERNATIONAL OLIVE COUNCIL. 2021. Worlds olive oil production has tripled. Available at: https://www.internationaloliveoil.org/worlds-olive-oil-production-has-tripled/

JIMENEZ, A. R. – CERES, R. – PONS, J. L. 2000. A survey of computer vision methods for locating fruit on trees. In Transactions of the ASAE, vol. 43, no. 6, pp. 1911–1920. DOI: https://doi.org/10.13031/2013.3096

JOCHER, G. 2020. YOLOv5. Available at: https://github.com/ultralytics/yolov5

KANG, H. – CHEN, C. 2020a. Fast implementation of real-time fruit detection in apple orchards using deep learning. In Computers and Electronics in Agriculture, vol. 168, article no. 105108. DOI: https://doi.org/10.1016/j.compag.2019.105108

KANG, H. – CHEN, C. 2020b. Fruit detection, segmentation and 3D visualisation of environments in apple orchards. In Computers and Electronics in Agriculture, vol. 171, article no. 105302. DOI: https://doi.org/10.1016/j.compag.2020.105302

KHAN, A. – KHAN, U. – WALEED, M. – KHAN, A. – KAMAL, T. – MARWAT, S. N. K. – MAQSOOD, M. – AADIL, F. 2018. Remote sensing: An automated methodology for olive tree detection and counting in satellite images. In IEEE Access, vol. 6, pp. 77816–77828. DOI: https://doi.org/10.1109/access.2018.2884199

KOIRALA, A. – WALSH, K. B. – WANG, Z. – MCCARTHY, C. 2019. Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO'. In Precision Agriculture, vol. 20, no. 6, pp. 1107–1135. DOI: https://doi.org/10.1007/s11119-019-09642-0

MARTINELLI, F. – MARCHESE, A. – GIOVINO, A. – MARRA, F. P. – DELLA NOCE, I. – CARUSO, T. – DANDEKAR, A. M. 2019. In-field and early detection of *Xylella fastidiosa* infections in olive using a portable instrument. In Frontiers in Plant Science, vol. 9, pp. 2007. DOI: https://doi.org/10.3389/fpls.2018.02007

MAVRIDOU, E. – VROCHIDOU, E. – PAPAKOSTAS, G. A. – PACHIDIS, T. – KABURLASOS, V. G. 2019. Machine vision systems in precision agriculture for crop farming. In Journal of Imaging, vol. 5, no. 12, article no. 89. DOI: https://doi.org/10.3390/jimaging5120089

NELSON, J. – SOLAWETZ, J. 2020. YOLOv5 is here: State-of-the-art object detection at 140 FPS. Available at: https://blog.roboflow.com/yolov5-is-here/

NGUYEN, T. T. – VANDEVOORDE, K. – WOUTERS, N. – KAYACAN, E. – DE BAERDEMAEKER, J. G. – SAEYS, W. 2016. Detection of red and bicoloured apples on tree with an RGB-D camera. In Biosystems Engineering, vol. 146, pp. 33–44. DOI: https://doi.org/10.1016/j.biosystemseng.2016.01.007

REDMON, J. – FARHADI, A. 2017. YOLO9000: Better, faster, stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp. 6517–6525. DOI: https://doi.org/10.1109/cvpr.2017.690

REZATOFIGHI, H. – TSOI, N. – GWAK, J. – SADEGHIAN, A. – REID, I. – SAVARESE, S. 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 658–666. DOI: https://doi.org/10.48550/arXiv.1902.09630

RUSSELL, B. C. – TORRALBA, A. – MURPHY, K. P. – FREEMAN, W. T. 2008. LabelMe: A database and web-based tool for image annotation. In International Journal of Computer Vision, vol. 77, no. 1, pp. 157–173. DOI: https://doi.org/10.1007/s11263-007-0090-8

WANG, C. Y. – YEH, I. H. – LIAO, H. Y. M. 2021. You only learn one representation: Unified network for multiple tasks. arXiv preprint arXiv:2105.04206. DOI: https://doi.org/10.48550/arXiv.2105.04206

WANG, Y. – XU, L. 2018. Unsupervised segmentation of greenhouse plant images based on modified Latent Dirichlet Allocation. In PeerJ, vol. 6, article no. e5036. DOI: https://doi.org/10.7717/peerj.5036

ZHANG, Y. – SONG, C. – ZHANG, D. 2020. Deep learning-based object detection improvement for tomato disease. In IEEE Access, vol. 8, pp. 56607–56614. DOI: https://doi.org/10.1109/access.2020.2982456