LIGHTWEIGHT FRESH JUJUBE TARGET SEGMENTATION ALGORITHM BASED ON IMPROVED YOLOV8: YOLOSEG-JUJUBE
/
基于改进型 YOLOV8 的轻量级鲜枣目标分割算法：YOLOSEG-JUJUBE

Huamin ZHAO*1), Defang XU*2)
1). College of Agricultural Engineering, Shanxi Agricultural University, Jinzhong /China;
2). Department of Mathematics and Artificial Intelligence, Lvliang University, Lvliang /China
Corresponding authors: Huamin Zhao; Tel: +86-18001093228; E-mail: zhaohuamin@sxau.edu.cn
Defang Xu, Tel: +86-18935460012; E-mail: xudefang0012@163.com
DOI: https://doi.org/10.35633/inmateh-76-51

Keywords: *YOLOseg-Jujube；YOLOv8n；Fruit detection；Area estimation*

ABSTRACT
Fruit instance segmentation algorithms are critical for target localization in fruit-picking robots, enabling accurate area estimation of fruits within images. However, existing methods often face limitations such as high computational cost, poor adaptability to low-power devices, and reduced detection performance in complex environments. To address these challenges, YOLOSeg-Jujube was proposed - an improved instance segmentation algorithm based on YOLOv8 - validated on a jujube image dataset annotated with bounding polygons. The architecture of YOLOSeg-Jujube was optimized through ablation studies to identify the most effective structure. The final network design includes Focus, CBS, Conv4cat, SPD, and SPPFr modules as the backbone, and a YOLOv8 head incorporating the SIoU loss function. Compared to YOLOv4-tiny, YOLOv5n, YOLOv7-tiny, and YOLOv8n, YOLOSeg-Jujube achieves reductions in parameter size of 62.8%, 9.8%, 73.9%, and 23.6%, respectively. The model achieves 83.5% B_mAP and 83.2% S_mAP, outperforming mainstream YOLO variants in both segmentation accuracy and area estimation of target objects. YOLOSeg-Jujube is robust, fast, and computationally efficient, making it suitable for deployment on resource-constrained platforms. Furthermore, it demonstrates strong potential for recognizing ripeness stages of fresh jujubes, providing technical support for intelligent harvesting systems in the field.

摘要
水果实例分割算法是水果采摘机器人目标定位的重要方法。能够确保对图像中的目标进行正确的区域估算。与此同时，模型在复杂农业环境下计算成本高、不便于在低功耗计算设备上部署等都对水果采摘机器人用实例分割造成限制。因此，为了解决这些问题，本研究在 YOLOv8 的基础上设计了 YOLOseg-Jujube，并在自然光照环境下的红枣图像数据集上进行了验证。YOLOseg-Jujube 网络由 Focus、CBS、Conv4cat、SPD 和 SPPFr 组成骨干网络，YOLOv8 head 和 SIoU loss 组成头部网络。通过对比，YOLOseg-Jujube 的参数分别比 YOLOv4-tiny、YOLOv5n、YOLOv7-tiny 和 YOLOv8n 低 62.8%、9.8%、73.9% 和 23.6%。YOLOseg-Jujube 的 B_mAP 为 83.5%，S_mAP 为 83.2%，YOLOseg-Jujube 在分割目标的面积估计方面性能优于 YOLO 主流变体的算法。YOLOseg-Jujube 算法稳健、快速、准确，计算成本低，并且能够识别枣果的成熟阶段，可为田间鲜枣采摘机器人的研究提供技术支撑。

INTRODUCTION
Jujube (*Ziziphus jujuba*) is a fruit that is known around the world for its high in vitamins and minerals and low calories. The fruit comes with a host of several health benefits to humans. Nevertheless, it is a difficult task to sort and harvest, including hand-picking the fruit, since their maturity stages, for example, change from green to dark red. Meanwhile, jujube fruit can be divided into white, crisp, or fully mature depending on its color, flesh firmness, and composition. A fruit detection method using deep learning offers the best solution to recognize and locate the jujube fruit targets in an image or video according to *Yichen et al., (2022)* and *Defang et al., (2023).* This method allows for proper monitoring, quick sorting, and accurate harvesting of fruit. Notwithstanding, an extensive investigation on jujube fruit instance segmentation is needed to consolidate fruit detection.

Furthermore, it is essential to develop an algorithm with fewer parameters, greater robustness, and better compatibility with low-power computing devices such as the Nvidia Jetson Nano and Raspberry Pi, which will help extend the packing robot's battery life and enhance picking efficiency.

The ability to detect fruit targets using computer vision with deep learning has been widely investigated by researchers over the years, providing a groundbreaking outcome in fruit-quality detection and estimation *(Wang Z., Walsh, & Koirala, 2019)*, fruit counting *(Yanchao et al., 2022)*, yield prediction *(Koirala, Walsh, Wang, & McCarthy, 2019)*, harvesting/picking robot applications *(Lawal, 2021)*, and so on. However, the complex environment, such as occlusion, illumination, background, and nonstructural field, where fruits are grown, is still a difficult factor for fruit detection. Additionally, it is limited to rectangular bounding boxes, which in turn led to the introduction of fruit instance segmentation. Fruit instance segmentation is more granular with target shape deterministic through pixel characterization compared to fruit detection.

The application of the You Only Look Once object detector has demonstrated remarkable performance for fruit detection. Using YOLOv3, *Redmon & Farhadi, (2018*), *Wendong, Yakun, Jing, & Gang, (2021)* improved the method, and reported an average precision (AP) of 95.56% and speed of 35.5 frames per second (fps) for real-time object detection revealed AP of 90.05% and speed of 29.4 fps for kiwifruits detection in orchard. *Fu et al., (2021*), *Yang-Yang et al., (2019),* reported AP of 88.8% and speed of 40 fps for muskmelon fruit detection, *Peichao, Hao, Kunfeng, Jiachao, & Shanda, (2023*), designed MYOLO to achieved mean AP of 97.03% and speed of 50.6 fps for mushrooms detection, and *Liu G., Nouaze, Mbouembe, & Kim, (2020)* proposed YOLO-Tomato to reached 94.58% of AP and revealed that the factors of fruit detection are solvable. Next, *Yunchao, et al., (2023)*, reported 92.07% of AP and speed of 32.3 fps based on YOLOv4 *(Bochkovskiy, Wang, & Liao, 2020)* to detect each camellia oleifera fruit targets in orchard. The modification of YOLOv4-tiny by *Borja & Tofael, (2021),* revealed a speed of more than 50 fps and AP of 94.19% for real-time pear fruit detection and counting. *Huang & Wu, (2023),* designed GCS-YOLOv4-tiny and reported AP of 93.42% to detect different growth stages of fruits. Owning to YOLOv5 *Defang et al., (2023),* proposed YOLO-Jujube and achieved AP of 88.8% and a speed of 245 fps to detect jujube fruit, and the counting method of red jujube by *Yichen et al., (2022)* recorded AP of 94% and speed of 35.5 fps. *Bin Y et al.(2021)* reported AP of 86.75% and a speed of 66.7 fps to detect apple fruit targets, and *Bin Z. et al., (2022)* realized AP of 97.4% to detect a dragon fruit using modified YOLOv5. The proposed YOLOv5s-cherry by *Gai et al., (2021)* for cherry detection revealed F_1-score of 0.08 and 0.03 more than the YOLOv4 and YOLOv5, respectively. The YOLOv7 by *Wang, Bochkovskiy, & Liao, (2023),* was declared to have outperformed YOLOv4 and YOLOv5 in object detection performance. *Bin et al., (2022),* revealed AP of 95.6% and 96.0% respectively using YOLOv7 and YOLOv7-tiny for dragon fruit detection, and *Junyang et al., (2022),* modified the YOLOv7 to achieved AP of 97.29% and a speed of 14.4 fps for citrus detection. In quest for improvement, YOLOv8 by *Jocher et al, (2023),* was introduced, having a new feature with anchor free that can directly predict the center of the target, rather than offset of the known anchor box. Anchor free reduces the number of box predictions to speeds up Non-Maximum Suppression (NMS) post processing. YOLOv8 is designed to be flexible, easy to use, fast and accurate compared to previous versions of YOLO framework, but still in the development stage. Regardless, the YOLO variants algorithm for fruit detection has a complicated network topology that is hard to understand, cannot estimate the area of fruit targets in an image, cannot properly address occlusions by leaves, branches or other clusters, and are anchor-based detectors excluding YOLOv8. Furthermore, the YOLOv8 network is yet to be experimented for fruit detection, including for fruit instance segmentation.

To approach fruit instance segmentation, several investigations have been conducted using Mask R-CNN *(He, Gkioxari, Dollár, & Girshick, 2017)* as two-stage detector. *Santos, et al., (2020),* addresses occlusions and recorded F_1-score up to 91%, and the DualSeg proposed by *Jinhai et al., (2023),* reported 83.7% for grape instance segmentation. *Chan et al., (2021),* demonstrated robustness against various illuminations and complex backgrounds, and yielded AP of 94.7% for mangoes instance segmentation. The amodal segmentation algorithm proposed by *Jordi et al., (2023)*, was designed to predict the complete shape of apple different growth stages under visible and occluded regions and it achieved an AP between 44% and 51% with a mean absolute error (MAE) of 4.5 mm. *Dandan & Dongjian, (2022)*, improved Mask RCNN by fusing an attention module into the backbone network to enhance feature extraction ability and apples were accurately segmented under various conditions, having a recall, precision, F_1-score, and segmentation mAP of 97.1%, 95.8%, 96.4% and 0.917, respectively, with average run-time of 4 fps. *Pengyu et al, (2021),* developed a suppression branch and added it

Mask R-CNN for apple segmentation. The experimental results showed a significantly enhanced F_1-score of 90.5% and detection time of 4 fps. The MASU R–CNN proposed by *Tian et al, (2020),* for instance segmentation of apple flowers reached precision of 96.43%, recall of 95.37%, F_1-score of 95.90%, and mAP of 59.4%, outperforming other state-of-the-art models. *Liu et al., (2019),* reported AP and inference time of 89.47% and 2.9 fps, respectively, on improved Mask-RCNN for cucumber instance segmentation. Nevertheless, the research experimented on Mask-RCNN for fruit instance segmentation tends to have a very large weight-size, which slower the detection speed. The detection speed of a single-stage is faster than a two-stage detector, according to *Koirala et al., (2019)* and *(Olarewaju M. Lawal, 2021)*. For this reason, *Kang & Chen, (2020),* developed single-stage DaSNet-v2 for fruit detection and instance segmentation, and semantic segmentation of branches under complex orchard environment. The results demonstrated a weight-size of 8.1 MB with speed of 18 fps, but still require further performance improvement. Recently, *Mubashiru, (2023),* designed YOLOv5-LiNet based on YOLOv5 for fruits instance segmentation, which achieved box AP of 89.3%, instance segmentation AP of 88.5%, and real-time speed of 385 fps with weight-size of 3.0 MB. However, there are a few experiments using YOLO framework, including anchor-free YOLOv8 for fruit instance segmentation. Therefore, this article developed YOLOseg-Jujube based on improved YOLOv8n architecture to real-time detect and segment fruit targets in the complex environment.

MATERIALS AND METHODS

Dataset description

The jujube fruit dataset used in this paper was obtained by *Defang et al., (2023)*, set for ripeness inspection method. According to *Defang et al., (2023)*, the images were collected from Gaolang Red Date Picking Garden, Linxian County, Luliang, Shanxi, China using Huawei mate30pro and mate40pro of resolution 1904×4096 and 2736×3648 pixels, respectively. With reference to white and crisp mature stage under complex environment such as leaf occlusion, overlap occlusion, dense target fruit, branch occlusion, illumination, earth background, sky background, and other fruit natural scenes, these images were taken in the morning, midday and afternoon with constantly changing shooting angle and distance. Table 1 describes the image dataset details under different conditions. A total of, 1959 images in JPG format and recorded video in mp4 were collected. Furthermore, these images were randomly divided into 80% train-set, 15% valid-set, and 5% test-set, and the ground truth bounding boxes of each target in an image were hand labeled into 80R (Crisp mature ripeness) and 70R (White mature ripeness). Nevertheless, the image dataset was improved for jujube fruit instance segmentation in this paper using Labelme annotation tool. Similarly, the supposed polygon shape of the fruit targets in an image was drawn without special attention to the condition of the image, and the annotation files were saved in COCO format. Then, the COCO format was converted into the Poly-YOLO txt format. The Poly-YOLO txt format takes target category followed by xy to xn yn for instance segmentation, where xy is the coordinate for n polygon point. A total of, 10912 and 1921 bounding polygons were created respectively from 1569 train-set and 292 valid-set, similar to the box's information of *Defang et al., (2023)*. The 98 test-set images and the provided video were not annotated, but rather treated as unseen data, used to measure the level of algorithm robustness towards generalization.

Table 1

Image dataset details under different conditions

Time taken	Occlusions	Dense targets	Close targets	illumination	Others
Morning	280	95	131	105	39
Midday	268	99	150	110	33
Afternoon	270	85	166	92	36

Source: (Defang et al., 2023)

YOLOseg-Jujube

The architecture of YOLOseg-Jujube makes use of some key components to perform both fruit detection and instance segmentation tasks according to Fig. 1. As a build on YOLOv8n, it consists of backbone, neck and head network. The backbone is basically a series of convolutional layers that extract relevant features from the input image. It comprised of Focus, CBS, Conv4cat, SPD, and SPPFr network.

Similar to YOLOv5， the Focus network performs a slicing operation to split high-resolution feature maps into multiple low-resolution feature maps of four times, to obtain every pixel in an image without information lost. In this way, the width (W), height (H) information is concatenated in the channel space, then convolved to a downsampled feature map according to Fig. 1.

As the 1st downsampling network in layer 0, it increases the generalization capability. The CBS is a convolution layer activated with SiLU *(Elfwing, Uchibe, & Doya, 2017)* after batch normalization (BN) layer. The placed CBS_3,2 in 1st layer is for downsampling to provide a more scalable approach, CBS_1,1 in 8th layer is for dimensionality reduction and retaining the salient features, and CBS_3,1 placed in 9th layer has the ability to learn features common to different situations for better generalization. The knowledge of integrated Conv4cat into 2nd, 4th and 6th layer of backbone was derived from *Defang et al., (2023)* and *Du et al., (2020)*, where the complementary features of low-layer concatenate high-layer to enable sharing of information and the ability to learn more diverse features. For having similar W and H, where C is the number of channels, the $X \in R^{H\times W\times C1}$ of feature maps in CBS_1,1, $Y \in R^{H\times W\times C2}$ of feature maps in CBS_3,1 and $Z \in R^{H\times W\times C3}$ of feature maps in CBS_1,1 concatenated to produced $O \in R^{H\times W\times (C1 + C2 + C3)}$ of features as indicated in Fig. 1 and stated in Eq. 1.

$$O = [X, Y, Z] \tag{1}$$

The space-to-depth (SPD) proposed by *Sunkara & Luo, (2022)*, shown in Fig. 2 was placed at the 3rd, 5th and 7th layer of the backbone to permutes the spatial blocks of the input into the depth dimension. This network was used to combine feature maps of different size without discarding any feature data. The idea of SPD was to solve the performance degradation with low-resolution images and small objects, and have shown to outperforms the state-of-the-art deep learning models using YOLOv5 detector. Since the image dataset is associated with dense fruit targets, SPD was employed serving both as a feature extractor and as downsampling operation. According to Fig. 2, the intermediate feature map X of size ($S \times S \times C_1$) in Fig. 2(b) is sliced into a sequence of sub-feature maps, each with a shape of ($S/2$, $S/2$, C_1). This operation effectively downsamples X by a factor of 2 in Fig. 2(c). These sub-feature maps concatenate along the channel dimension to obtain a feature map X' in Fig. 2(d) with a reduced spatial dimension by *scale* and an increased channel dimension by *scale*^{2}. After that, a non-strided convolution added to retain all the discriminative feature information X'' as in Fig. 2(e). The SPD feature transformation can be defined by Eq. 2.

$$X(S,\ S,\ C_1) \rightarrow X'\left(\frac{S}{scale}, \frac{S}{scale}, scale^2 C_1\right) \rightarrow X''\left(\frac{S}{scale}, \frac{S}{scale}, C_2\right) \tag{2}$$

The SPPF layer was originally used to speed up the computation of the network by pooling features of various scales into a fixed-size feature map. However, it experienced loss of feature information during learning, including a slower detection speed for having three maxpooling layers with a convolution layer being concatenated. For this reason, SPPFr was designed to have a single maxpooling layer concatenated with a convolution layer. As in the operation described by Equation (1), the feature maps from the CBS module are concatenated with those from the MaxPool layer, as shown in Figure 1. The SPPFr module, serving as the final layer of the backbone, was added at the 10th position to enhance feature representation, reduce missed detections, and improve detection speed.

YOLOseg-Jujube used similar neck network as YOLOv8n. The neck combines path aggregation network (PAN) and feature pyramid network (FPN) for multiscale feature fusion. The Upsample layers of 11th and 14th increase the resolution of the feature maps collected from the backbone and concatenated with 6th layer of Conv4cat and 4th layer of Conv4cat, respectively, followed by C2f module. The C2f module as shown in Fig. 1 combines the high-level features with contextual information to improve detection accuracy. It consists of two CBS_1,1, split and three Bottlenecks, which is placed at the 13th, 16th, 19th and 22nd layer of the neck. Meanwhile, CBS_3,2 in 17th and 20th layer does downsampling operation before being concatenated with 13th and 10th layers, respectively. Having obtained feature maps from backbone, the C2f 16th, 19th and 22nd layer of neck passes the information to detection module for small, medium and large scale.

Furthermore, the framework of YOLOseg-Jujube makes used of an anchor free paradigm whose head network is designed to be decoupled, meaning that it processes classification, and regression tasks independently. This enables each branch to focus on its respective task and to improve the overall accuracy of the algorithm.

The detection module is part of the head, and uses a set of convolution and linear layers to map high-dimensional features and predict bounding boxes and polygons, and class probabilities. During training, the loss function plays a critical role in updating the network's weights by measuring the difference between the predicted bounding boxes and the ground truth coordinates. For classification (Cls) loss, binary cross-entropy (BCE) was used, as shown in Figure 1.

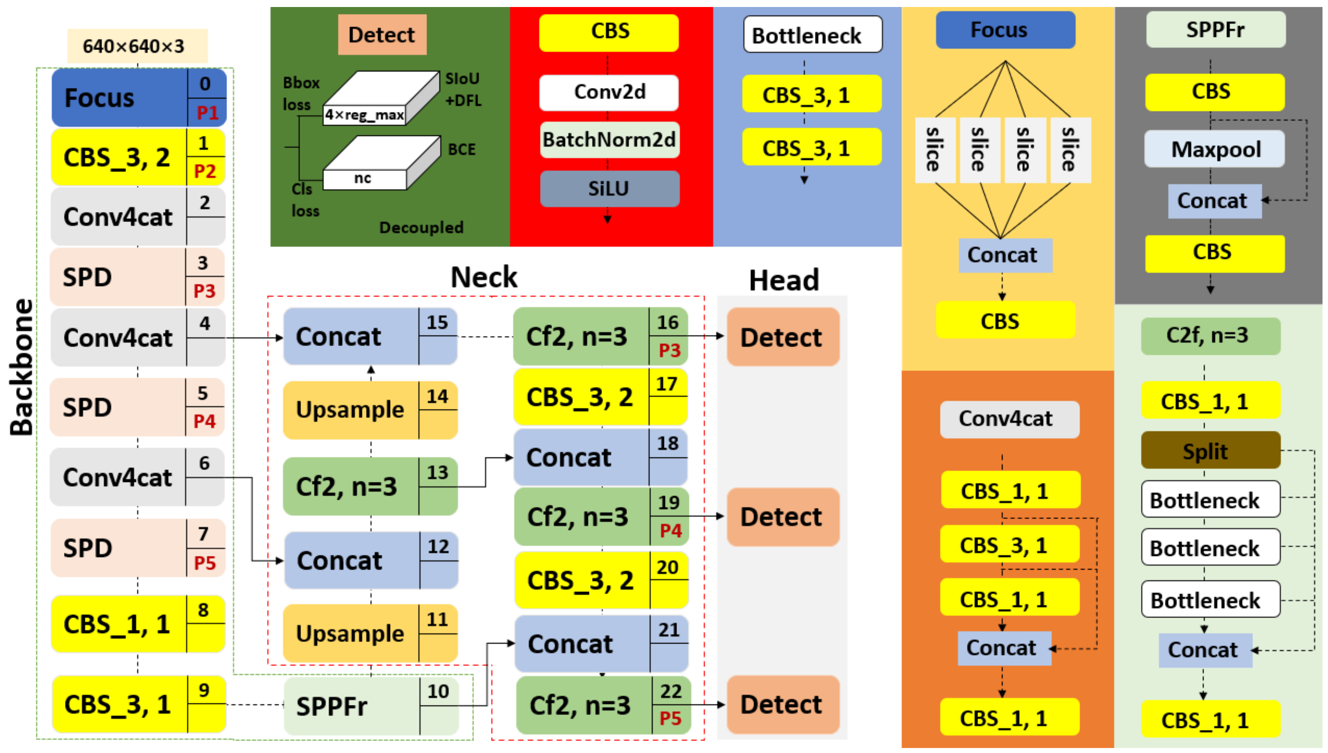


Fig. 1 - The network architecture of YOLOseg-Jujube

For regression loss, the proposed model adopts the SCYLLA-IoU (SIoU) loss function *(Gevorgyan, 2022)* combined with Distribution Focal Loss (DFL) *(Xiang et al., 2022)*, in contrast to the Complete IoU (CIoU) loss with DFL used in YOLOv8 *(Zheng et al., 2020)*. This substitution enhances bounding box regression by considering angle, distance, and shape alignment more effectively. SIoU loss improves the speed of training and the accuracy of inference depending on the cost of angular, distance, shape, and IoU. The loss function L_{box} embodies all the costs by Eq. 3, where the distance cost (Δ) takes into account the angle cost, shape cost (Ω) is unique for each dataset, and IoU is the difference between the predicted box (B) and real box (B^{gt}) with reference to union and intersection.

For DFL as shown in Eq. 4, it focuses on the distribution of the close to target location and enables probability density near the location. Here s_i is the sigmoid output, x_i and x_{i+1} is the interval, x is label.

$$L_{box} = 1 - \mathrm{IoU} + \frac{\Delta + \Omega}{2} \tag{3}$$

$$DFL_{(s_i, s_{i+1})} = -((x_{i+1} - x)\log(s_i) + (x - x_i)\log(s_{i+1})) \tag{4}$$

Finally, the overall architecture of YOLOseg-Jujube was designed to be low computation cost, fast, flexible, and efficient, while still achieving high fruit detection and instance segmentation accuracy.

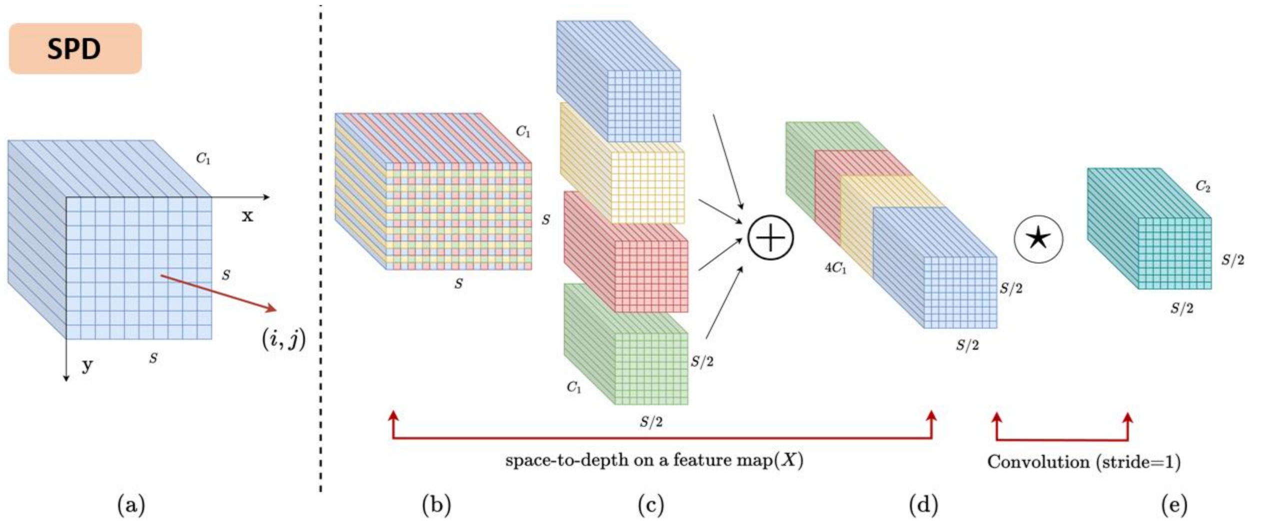


Fig. 2 - SPD-Conv for low resolution images and small objects (scale = 2, stride =1)
(Sunkara & Luo, 2022)

Experiment and evaluation

The training and testing of YOLOseg-Jujube, including compared YOLO-mainstream variants were implemented using YOLOv8.0.40 platform on hardware and environment information stated in Table 2. The algorithms received an input image of 640×640×3 pixels, 16 batch, 0.937 momentum, 0.0005 weight decay, 0.7 IoU, 0.015 hue, 0.7 saturation, 0.4 lightness, 1.0 mosaic, 0.5 scale, 0.1 translate, 7.5 Bbloss, 0.5 Clsloss, 1.5 DFloss, and 100 epochs for training from scratch. At the time, the weight for training the algorithms was randomly initialized from the scratch. The developed algorithms were exported from PyTorch (pt) to open neutral network exchange (ONNX), and to NCNN, developed by Tencent. The NCNN format was deployed using ncnn-android-yolov8 (Kang & Chen, 2020) on mobile phone. The NCNN was optimized to run faster than known frameworks on mobile phone CPUs.

Table 2

Hardware and environment description

Hardware	Configure	Environment	Version
System	Ubuntu20.04	Python	3.9.12
CPU	Core i7-12700F	Conda	23.1.0
GPU	RTX3060 (16G)	PyTorch	1.12.1
RAM	32G	CUDA	11.3.1
Hard-disk	1.0T	CUDNN	8.8.0

The algorithms were evaluated using the metrics stated in Eq. (5)-(11) respectively for precision (P), recall (R), average precision (AP), mean average precision (mAP), speed, number of parameters (params), and giga floating point operations per second (GFLOPs). TP is the true positive for correct detections, FN is the false negative for missed detections, FP is the false positive for incorrect detections, P(R) means that P is a function of R, AP is the area under curve (AUC) for single-class, mAP is mean AP values over multiclass, C is the total number of classes, j is the serial number, i is the input-size, k is the convolution kernel-size, o is the output-size and H×W is the size of outputted feature map. Speed is used to measure the real-time in frames per second (fps), params (10^6) is the number of trainable parameters, layer is the network topology, and GFLOPs estimate the number of floating-point arithmetic operations.

$$P = \frac{TP}{TP+FP} \tag{5}$$

$$R = \frac{TP}{TP+FN} \tag{6}$$

$$AP=\int_0^1 P_{(R)} dR \tag{7}$$

$$mAP = \frac{\sum_{j=1}^{C} AP_j}{C} \tag{8}$$

$$speed = frames/time \tag{9}$$

$$params = \left[i \times (k \times k) \times o \right] + o \tag{10}$$

$$GFLOPs = H \times W \times params \tag{11}$$

RESULTS AND DISCUSSION

Ablation studies on YOLOseg-Jujube

The ablation studies conducted on YOLOseg-Jujube are needed to find out what features of the algorithm are responsible for achieving a good performance. The process involves removing and adding some features of algorithm and seeing the effects on performance. Computation cost, detection accuracy, and speed were taken into consideration. Computation cost is simply based on the size of the algorithm in GFLOPs and params. While the layer 1–9 was held constant throughout the backbone network, Table 3 shows that changes in layer 0 and 10, including the neck network and loss function, caused a variation in computation cost. The added CBS_3,2/Focus and SPPF/SPPFr respectively in layer 0 and 10 of Method 1–12 show differences in params and GFLOPs. Moreover, the contribution to computation cost of methods with SPPFr is less than methods having SPPF. The shown number of layers in Table 3 as network topology is a support to this fact. Without no significant difference with the used loss function, the incorporated PAN-FPN constituted an increased in params and GFLOPs compared to the FPN. Using only params, the level of computation cost is measure as the equality of Method1, Method5 and Method6 is greater than the similarity of Method2–4, Method7 and Method10–12, and also higher than the equals of Method8 and Method9.

Table 3

Ablation studies on YOLOseg-Jujube based on computation cost

Algorithm	Layer 0	Layer 10	Neck	Loss	Layers	Params	GFLOPs
Method1	CBS_3,2	SPPF	PAN-FPN	CIoU	168	2.53	11.3
Method2	CBS_3,2	SPPFr	PAN-FPN	CIoU	167	2.49	11.2
Method3	CBS_3,2	SPPFr	PAN-FPN	SIoU	167	2.49	11.2
Method4	Focus	SPPFr	PAN-FPN	CIoU	169	2.49	11.3
Method5	Focus	SPPF	PAN-FPN	CIoU	170	2.53	11.4
Method6	Focus	SPPF	PAN-FPN	SIoU	170	2.53	11.4
Method7	Focus	SPPFr	PAN-FPN	SIoU	169	2.49	11.3
Method8	Focus	SPPFr	FPN	CIoU	153	2.16	11.0
Method9	Focus	SPPFr	FPN	SIoU	153	2.16	11.0
Method10	Focus	SPPFr	PAN-FPN	EIoU	169	2.49	11.3
Method11	Focus	SPPFr	PAN-FPN	DIoU	169	2.49	11.3
Method12	Focus	SPPFr	PAN-FPN	GIoU	169	2.49	11.3

The depicted validation losses in Fig. 3 are used to compliment computation cost and to predict the performance of Method1–12 by showing the consistence decreasing pattern. They measure the real position of fruit targets in an image during network training, where Fig. 3(a) is the bounding box, Fig. 3(a) is for instance segmentation via bounding polygon, Fig. 3(c) is for classification, and Fig. 3(d) is for DFL to enable general distribution of target position.

Having shown a closely related level of neural network depth among Method1–12, the validation of class loss is deeper than box loss, segment loss and DFL according. This is as a result of more correctness found in class loss compared to box loss, segment loss and DFL. Excluding Method10, which shows a less deep network in Fig. 3(a) because it has EIoU compared to others, it is difficult to examine the deepest and best performing network. But it can be addressed using Fig. 4 of mAP. This means that a decreasing loss of validation as algorithm learns is an increase in the mAP. Fig. 4(a) show that the aggregated mAP of 80R at 74.6% and 70R at 92.2% seen in Method7 is higher than other methods. Similarly, Figure 4(b) shows that Method 7 also achieves superior aggregated mAP values of 74.6% at 80R and 91.9% at 70R, further confirming its performance advantage over alternative approaches. Next to the mAP performance of Method7 is Method4, having scored 0.1% less under 80R with the same value under 70R of Fig. 4(a), and the same value under 80R with 0.1% less under 70R of Fig. 4(b). Despite having a similar backbone network, as indicated in Table 3, the improved mAP of Method7 was triggered with the added SIoU loss against the CIoU loss incorporated into Method4. The obtained results verify that the mAP level of Method7 is higher than the rest methods. For justification, the test-set images were tested on Method1–12 to investigate robustness towards generalization. And the sample of the tested image associated with sky background and occlusions is presented in Fig. 5. The methods were able to detect and segment a number of fruit targets perfectly, but were noted with difference in confidence score, including missed or false detection. For example, the confidence score of detected and segmented 70R target at the base of Fig. 5(g) recorded 62%, which is more than 59% of Fig. 5(a), 51% of Fig. 5(b), 34% of Fig. (c), 38% of Fig. 5(d), 29% of Fig. 5(e), 44% of Fig. 5(f), 47% of Fig. 5(h), 55% of Fig. 5(i), 33% of Fig. 5(j), 52% of Fig. 5(k), and 40% of Fig. 5(l). This demonstrated the superiority of Method7 over other methods.

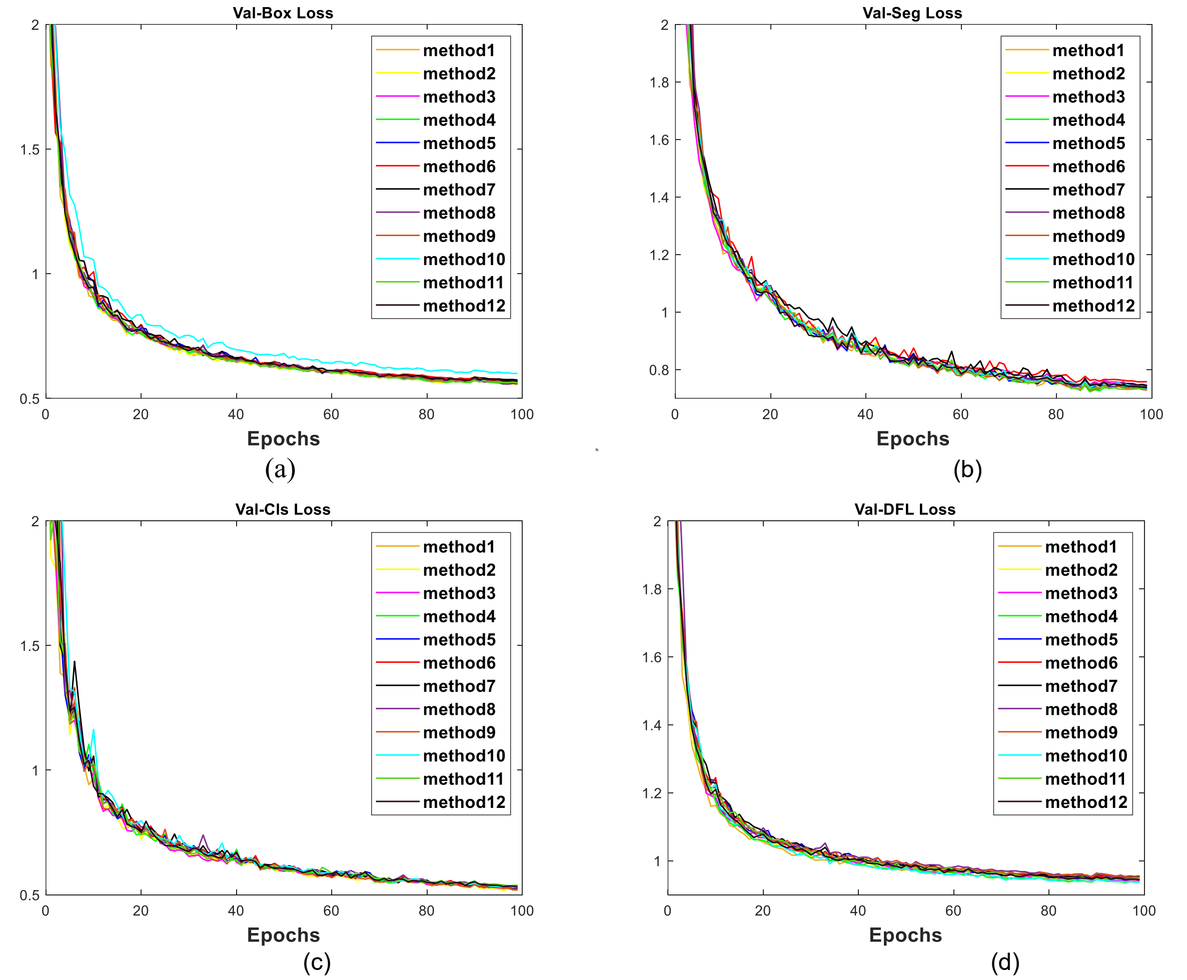
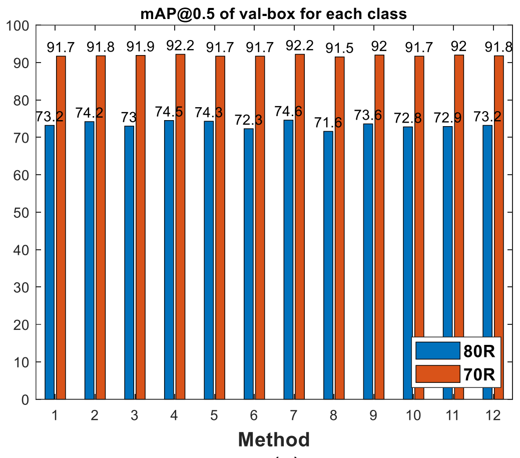
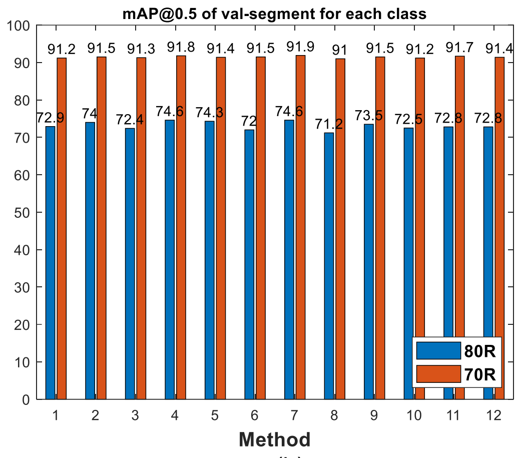


Fig. 3 - The validation of (a) box, (b) segmentation, (c) class, and (d) DFL loss during ablation on YOLOseg-Jujube

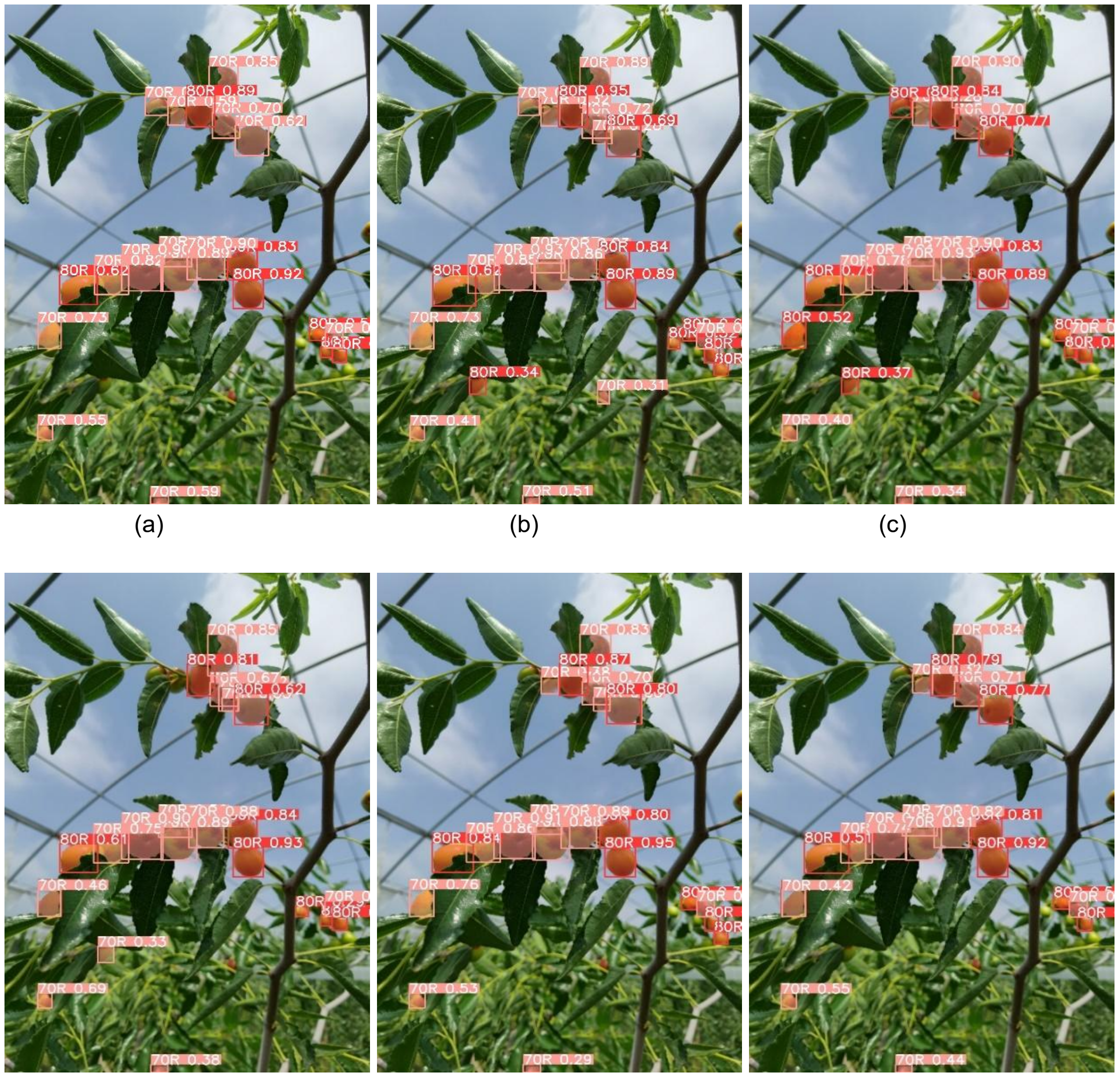


(a)



(b)

Fig. 4 - The mAP@50% of validation (a) box and (b) segmentation for each class during ablation on YOLOseg-Jujube



(a) (b) (c)

(d) (e) (f)

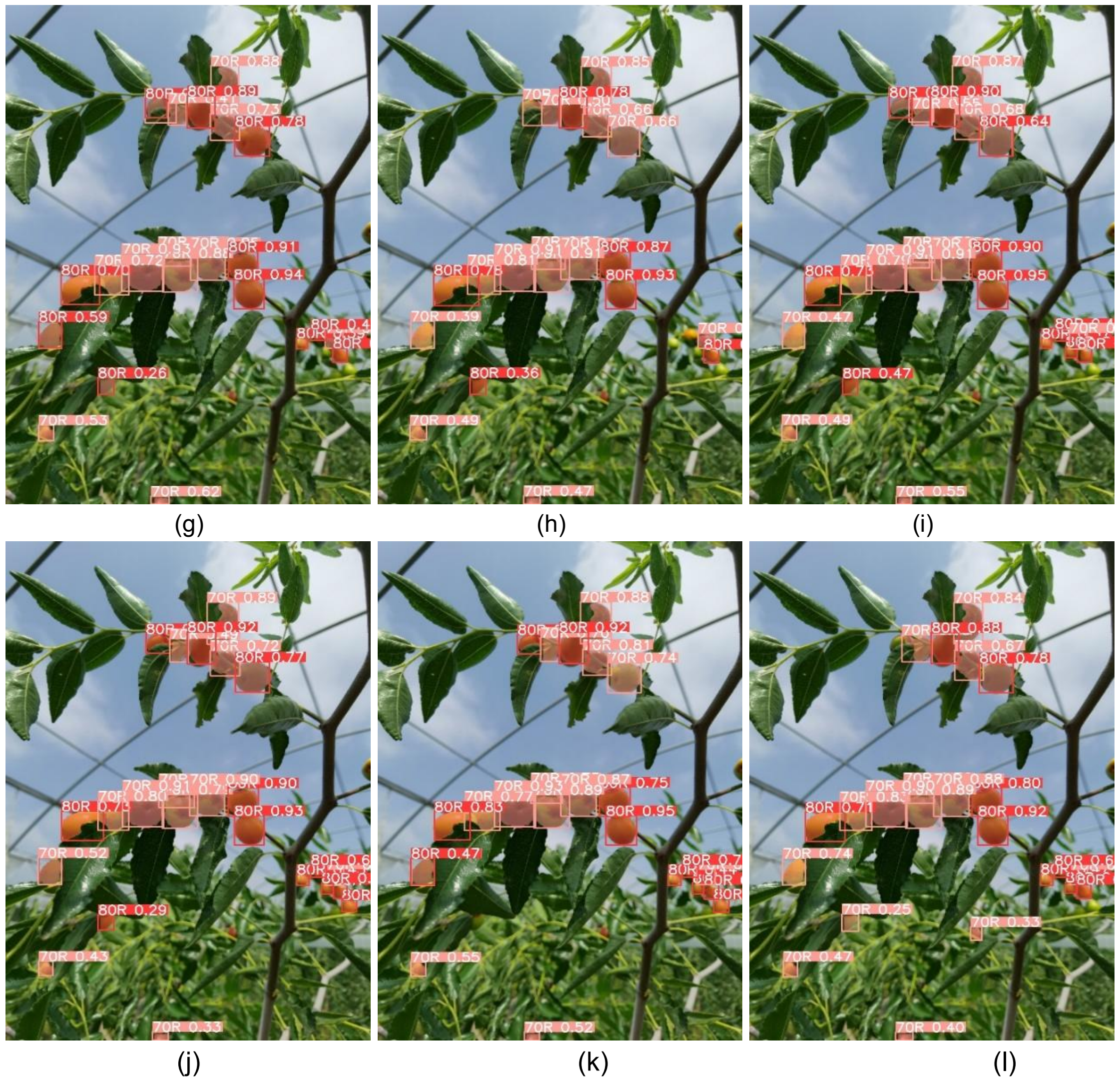


Fig. 5 - The detected and segmented fruit targets of class 80R and 70R obtained from an image with shy background and occlusions using (a)–(l) for Method1–12

Table 4 provides a comprehensive analysis of mAP for both box and segmentation, including speed, to show the outstanding performance of Method7 compared to Method1–6 and Method8–12. The mAP is more accurate because it measures the overall global relationship between P and R. Using Method7, the 83.5% of B_mAP is 1.1%, 0.5%, 1.0%, 0.2%, 0.5%, 1.5%, 1.9%, 0.7%, 1.2%, 1.0% and 1.1% more accurate respectively than Method1–6, Method8–12 under fruit targets detection, and 83.2% of S_mAP is 1.1%, 0.5%, 1.3%, 0.1%, 0.3%, 1.5%, 2.1%, 0.7%, 1.3%, 1.0% and 1.1% more accurate respectively than Method1–6, Method8–12 under fruit targets instance segmentation. Considering computation cost as a criterion for a faster detection speed, Method7 at 323 fps is faster than Method1 at 309 fps, Method2 at 315 fps, Method3 at 321 fps, Method4 at 320 fps, Method5 at 306 fps, Method6 at 317 fps, Method10 at 313 fps, Method11 at 319 fps, and Method12 at 318 fps, but not than Method8 at 342 fps and Method9 at 344 fps. The faster speed of Method8 and Method9 is attributed to applied FPN, unlike Method7 having PAN-FPN as shown in Table 4. Considering the overall performance of all methods, Method7 is the best and was chosen for our YOLOseg-Jujube after the ablation studies.

Table 4

Summary of Ablation study on YOLOseg-Jujube using valid-set and videos

Algorithm	B_P (%)	B_R (%)	B_ mAP (%)	S_P (%)	S_R (%)	S_ mAP (%)	Speed (fps)
Method1	78.8	76.4	82.4	78.3	76.8	82.1	309

Algorithm	B_P (%)	B_R (%)	B_ mAP (%)	S_P (%)	S_R (%)	S_ mAP (%)	Speed (fps)
Method2	77.1	79.6	83.0	77.2	79.3	82.7	315
Method3	82.4	74.7	82.5	82.2	74.6	81.9	321
Method4	83.8	74.3	83.3	83.9	74.3	83.1	320
Method5	80.8	75.7	83.0	80.8	75.4	82.9	306
Method6	81.3	76.3	82.0	81.2	76.1	81.7	317
Method7	80.1	77.7	83.5	80.4	77.2	83.2	323
Method8	79.8	74.5	81.6	79.7	74.1	81.1	342
Method9	76.4	80.3	82.8	81.6	75.4	82.5	344
Method10	77.1	77.9	82.3	77.6	77.3	81.9	313
Method11	78.8	76.6	82.5	78.8	76.7	82.2	319
Method12	78.4	78.4	82.5	78.0	78.2	82.1	318

Comparison with YOLO-mainstream variants

Having considered Method7 as YOLOseg-Jujube, its performance was compared to YOLO-mainstream variants as shown in Table 4. For computation cost, the obtained params from YOLOseg-Jujube are 62.8%, 9.8%, 73.9%, and 23.6% respectively less compared to YOLOv4-tiny, YOLOv5n, YOLOv7-tiny and YOLOv8n. In the case of GFLOPs, YOLOseg-Jujube is 82.7%, 83.8% and 5.8% lower than YOLOv4-tiny, YOLOv7-tiny and YOLOv8n, respectively, but not with YOLOv5n at -2.7%. With reference to B_mAP and S_mAP, the displayed results in Table 4 demonstrated the outstanding performance of YOLOseg-Jujube against YOLO-mainstream variants. For having 83.5% of B_mAP and 83.2% of S_mAP, YOLOseg-Jujube is 0.2% and 0.3%, 2.7% and 2.4%, 0.6% and 0.4%, and 2.1% and 1.9% more accurate than YOLOv4-tiny, YOLOv5n, YOLOv7-tiny and YOLOv8n, respectively. Furthermore, the algorithms were evaluated for their real-time detection speed using video as an important aspect of performance. The findings demonstrated that YOLOseg-Jujube is faster than YOLOv4-tiny YOLOv5n, YOLOv7-tiny and YOLOv8n as indicated in Table 4. Therefore, the aggregated performance in terms of reduced computation cost, accuracy and speed indicated that the algorithm of YOLOseg-Jujube was better than YOLOv4-tiny YOLOv5n, YOLOv7-tiny and YOLOv8n. YOLOseg-Jujube is robust against changeable environment and capable of understanding the ripeness stages of jujube fruit targets in an image.

Table 4

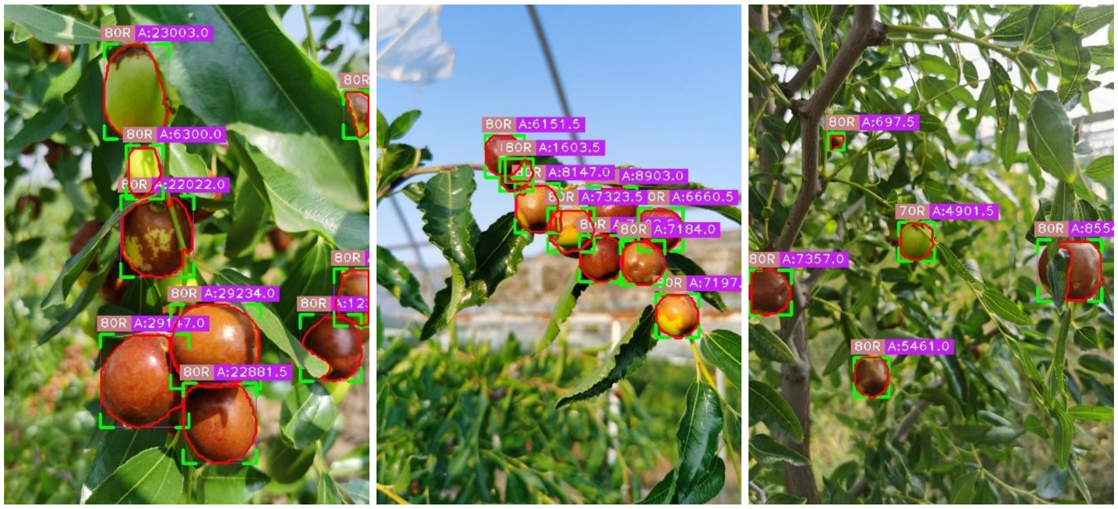
Performance comparison of algorithms using valid-set and videos

Algorithm	Params	GFLOPs	B_P %	B_R %	B_ mAP%	S_P %	S_R %	S_ mAP%	Speed (fps)
YOLOv4-tiny	6.70	65.5	76.5	79.3	83.2	76.4	78.8	82.9	297
YOLOv5n	2.76	11.0	79.3	75.5	80.8	79.2	75.7	80.8	273
YOLOv7-tiny	9.55	69.9	81.0	76.0	82.9	81.3	76.0	82.8	230
YOLOv8n	3.26	12.0	79.5	75.7	81.4	79.3	75.6	81.3	272
YOLOseg-Jujube	2.49	11.3	80.1	77.7	83.5	80.4	77.2	83.2	323

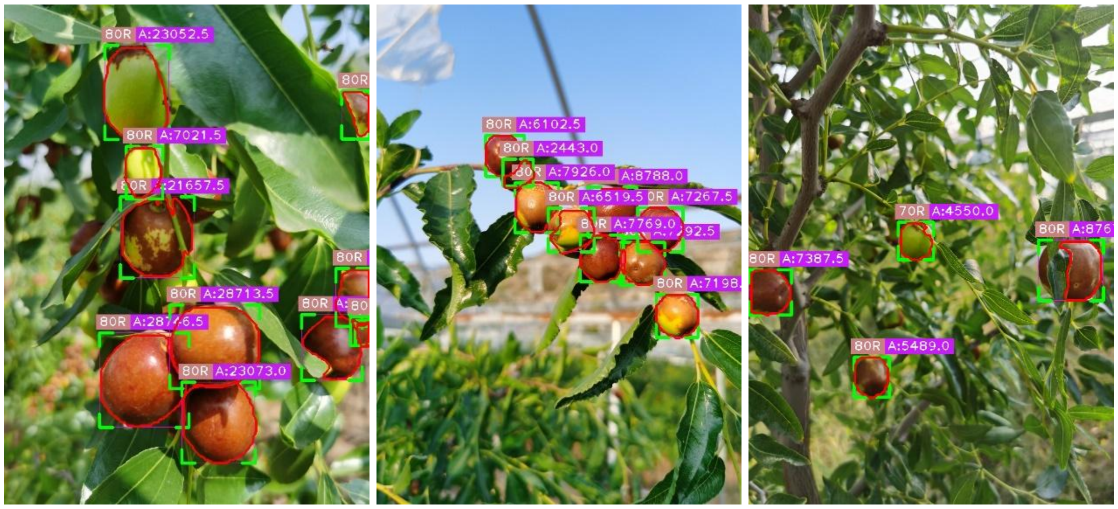
To establish the superiority of YOLOseg-Jujube to YOLO-mainstream variants, the algorithms were evaluated on test-set images with (1) fruit occlusions, (2) fruit clusters, and (3) dense targets, and were subjected to an estimation of the area of targets. Fig. 6 shows that the algorithms were able to segment and estimate the area of targets according to their classes very well, but there were some differences in the area value, including missed and false detections. Using the (1) fruit occlusion image, Fig. 6(a–c) of YOLOv4-tiny, YOLOv5n and YOLOv7-tiny were noted with two false detections unlike Fig. 6(d–e) of YOLOv8n and YOLOseg-Jujube with one false detection. Each segmented target is not completely drawn in the occlusion situation, which results in only an estimation of its area. Future investigation is required.

For (2) fruit clusters, all the algorithms segmented and estimated area perfectly, except for the difference in area value. Nevertheless, the obtained aggregated area values for YOLOseg-Jujube of Fig. 6(e) are higher than YOLOv4-tiny of Fig. 6(a), YOLOv5n of Fig. 6(b), YOLOv7-tiny of Fig. 6(c) and YOLOv8n of Fig. 6(d). This indicated that YOLOseg-Jujube instance segmentation is more robust than the YOLO-mainstream variants. In the case of

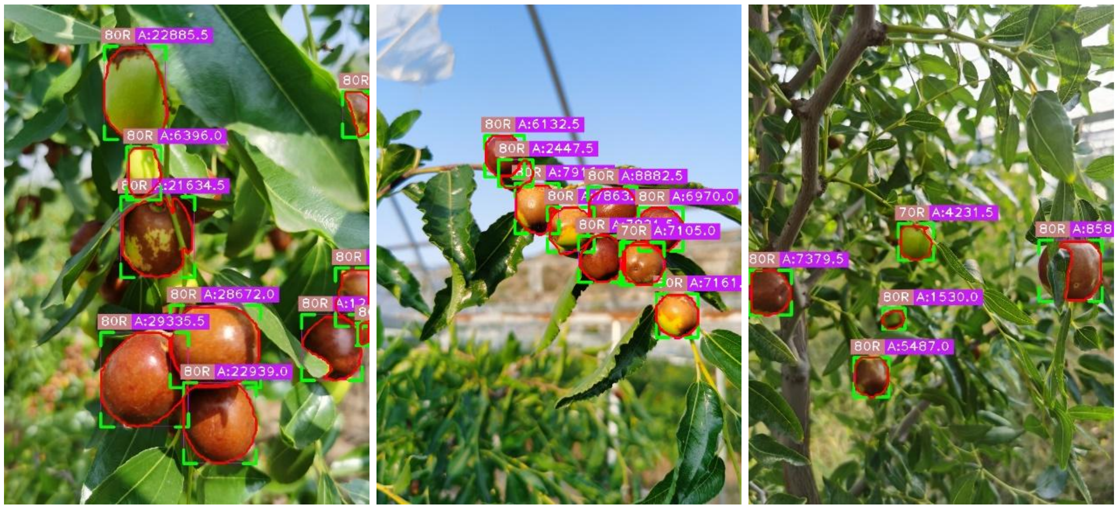
(3) dense targets, missed detections was observed in Fig. 6(a–c), but no missed detection was found in Fig. 6(d) of YOLOv8n and Fig. 6(e) of YOLOseg-Jujube. With this area calculation performance, YOLOseg-Jujube is the best candidate for the analysis of fruit-related phenotypic traits, such as number, size, shape and color.



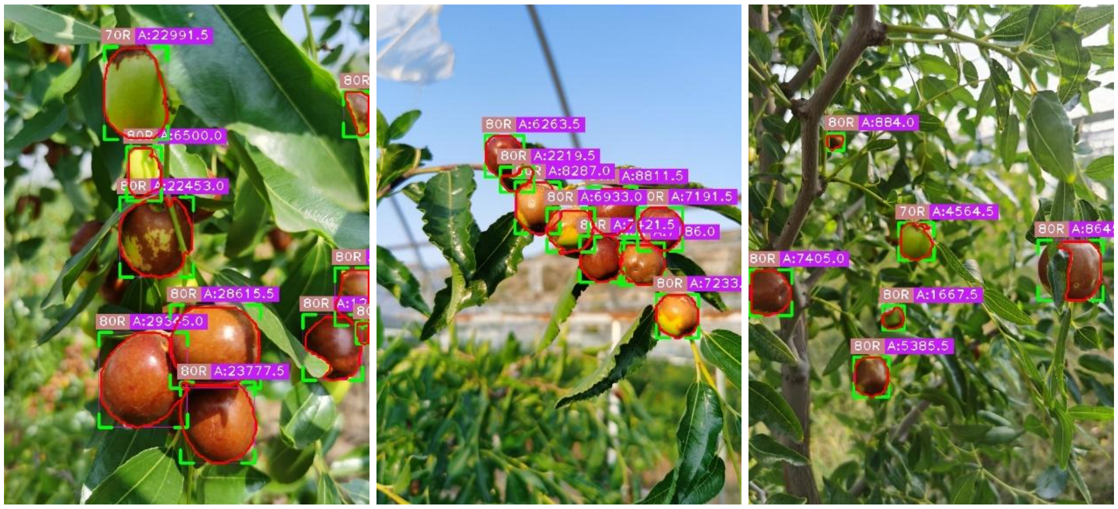
(a)



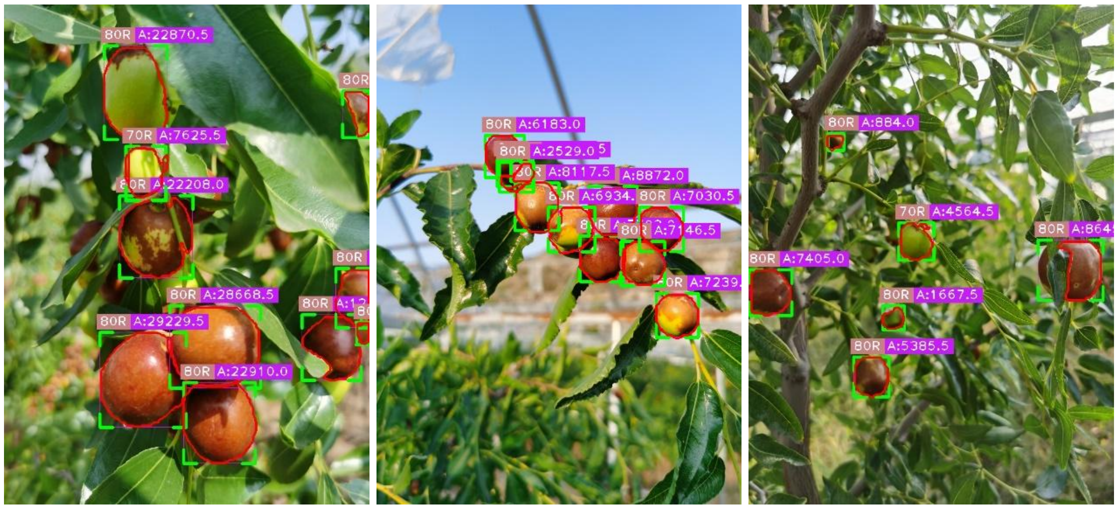
(b)



(c)



(d)



(e)

Fig. 6 - The segmented targets with area obtained from images of (1) fruit occlusions, (2) fruit clusters, and (3) dense fruit evaluated on (a) YOLOv4-tiny, (b) YOLOv5n, (c) YOLOv7-tiny, (d) YOLOv8n and (e) YOLOseg-Jujube

Furthermore, the developed YOLOseg-Jujube algorithm was evaluated and compared with YOLO-mainstream variants on low-power computing device.

Table 5 provides details of the exported ONNX and NCNN model deployed on mobile phone. Table 5 shows that ONNX's size and exported time of YOLOseg-Jujube is smaller than YOLOv4-tiny YOLOv5n, YOLOv7-tiny and YOLOv8n. This is in support of the claims in Table 4 that the complexity of YOLOseg-Jujube is reduced compared to YOLO-mainstream variants.

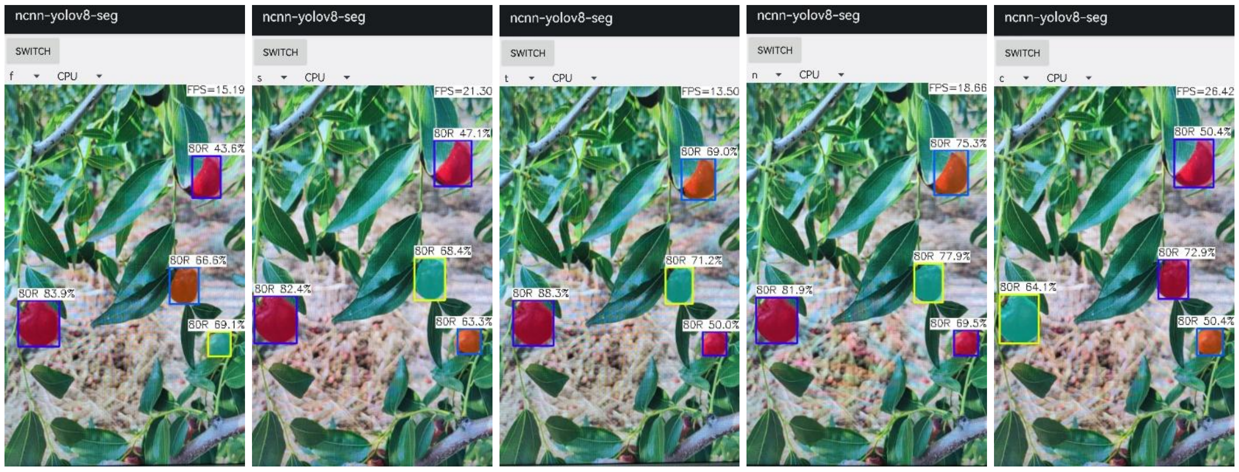
The obtained param and bin of the NCNN model also showed similar attributes, where the bin of YOLOseg-Jujube is 62.5%, 10.2%, 73.8% and 23.2% less than YOLOv4-tiny, YOLOv5n, YOLOv7-tiny and YOLOv8n, respectively. With reference to the displayed Fig. 7, the detection speed of algorithms was obtained. Subjecting the algorithms to real-time instance segmentation, a number of fruit targets were detected and segmented perfectly as shown in Fig. 6, but with a floating confidence score. Nevertheless, the real-time speed is very crucial to mobile phone.

Table 5 and Fig. 7(e) show that the real-time segmentation speed of YOLOseg-Jujube noted to be 26.24 fps is 73.9%, 24.0%, 95.7% and 41.6% faster than YOLOv4-tiny of Fig. 7(a), YOLOv5n of 7(b), YOLOv7-tiny of 7(c) and YOLOv8n of 7(d), respectively. This demonstrated that a decrease in bin results to an increase in speed. The obtained results also indicated that YOLOseg-Jujube is compatible and deployment friendly with low-power computing devices.

Table 5

Comparison of exported algorithms using simplified mode

ONNX	**YOLOv4-tiny**	**YOLOv5n**	**YOLOv7-tiny**	**YOLOv8n**	**YOLOseg-Jujube**
Add	3	7	–	6	–
BatchNormalization	3	–	–	–	–
Concat	13	19	20	19	22
Conv	59	87	85	75	66
Maxpool	3	3	6	3	2
Mul	47	78	76	66	57
Reshape	7	7	7	7	7
Resize	2	2	2	2	18
Sigmoid	47	78	76	66	57
Slice	–	–	–	8	4
Transpose	1	1	1	1	1
Size (MB)	25.6	10.5	36.5	12.5	9.5
Export (s)	10.7	8.9	8.3	3.9	3.4
NCNN					
Param (KB)	11.7	17.7	18.4	16.7	15.0
Bin (MB)	12.7	5.3	18.2	6.2	4.76
Speed (fps)	15.19	21.30	13.50	18.66	26.42



(a) (b) (c) (d) (e)

Fig. 7 - The output of fruit targets segmentation tested on a mobile phone using algorithm of (a) YOLOv4-tiny, (b) YOLOv5n, (c) YOLOv7-tiny, (d) YOLOv8n and (e) YOLOseg-Jujube

CONCLUSIONS AND FUTURE WORK

This paper explored the YOLOv8n based architecture to design YOLOseg-Jujube for fruit instance segmentation to consolidate fruit detection, including to solve the factors of complex environment faced by fruit detection. The ablation studies conducted on YOLOseg-Jujube resulted in the algorithm consisting of Focus, CBS, Conv4cat, SPD, and SPPFr as backbone network, and YOLOv8' head with SIoU loss as head network. The computation cost in terms of params show that YOLOseg-Jujube is 62.8%, 9.8%, 73.9%, and 23.6% less than YOLOv4-tiny, YOLOv5n, YOLOv7-tiny and YOLOv8n, respectively. Likewise, the 83.5% of B_mAP and 83.2% of S_mAP noted in YOLOseg-Jujube is 0.2% and 0.3%, 2.7% and 2.4%, 0.6% and 0.4%, and 2.1% and 1.9% more accurate than YOLOv4-tiny, YOLOv5n, YOLOv7-tiny and YOLOv8n, respectively. The real-time speed of YOLOseg-Jujube demonstrated to be faster than YOLOv4-tiny YOLOv5n, YOLOv7-tiny and YOLOv8n on computer with GPU-RTX3060 and mobile phone, including having to be the best candidate for the analysis of fruit-related phenotypic traits based on the conducted area calculation performance. The aggregated performance demonstrated that YOLOseg-Jujube is robust for generalization, low in computation cost, accurate, fast, flexible for deployment on low-power computing devices, efficient and able to understand the ripeness stages of jujube fruit targets in an image and video. However, future investigations will require detection accuracy improvements, to solve each segmented target not completely drawn in the occlusion situation for proper area estimation, a simpler network topology, and the performance of other converted model format on mobile phones.

ACKNOWLEDGEMENTS

This research was funded by the Lvliang Key R&D Projects, grant number 2024NY03; The Basic Research Project of Shanxi Province, grant number 202103021223145 and College Student Innovation and Entrepreneurship Training Program, grant number 20-034.

REFERENCES

[1] Bin, Y., Pan, F., Xiaoyan, L., Zhijie, L., & Fuzeng, Y. (2021). A Real-Time Apple Targets Detection Method for Picking Robot Based on Improved YOLOv5. *Remote Sensing, 13*(9). https://doi.org/10.3390/rs13091619

[2] Bin, Z., Rongrong, W., Huiming, Z., Chenghai, Y., Yuyang, X., Meng, F., & Wei, F. (2022). Dragon fruit detection in natural orchard environment by integrating lightweight network and attention mechanism. *Frontiers in Plant Science, 13*. https://doi.org/10.3389/fpls.2022.1040923

[3] Bochkovskiy, A., Wang, C. Y., & Liao, H. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv:2004.10934.* doi:https://doi.org/10.48550/arXiv.2004.10934:

[4] Borja, P. A. I., & Tofael, A. (2021). Real Time Pear Fruit Detection and Counting Using YOLOv4 Models and Deep SORT. *Sensors, 21*(14), 4803. https://doi.org/10.3390/s21144803

[5] Chan, Z., Pengfei, C., Jing, P., Xiaofan, Y., Changxin, C., Shuqin, T., & Yueju, X. (2021). A mango picking vision algorithm on instance segmentation and key point detection from RGB images in an open orchard. *Biosystems Engineering, 206*, 32-54. https://doi.org/10.1016/j.biosystemseng.2021.03.012

[6] Dandan, W., & Dongjian, H. (2022). Fusion of Mask RCNN and attention mechanism for instance segmentation of apples under complex background. *Computers and Electronics in Agriculture, 196*, 106864. https://doi.org/10.1016/j.compag.2022.106864

[7] Defang, X., Huamin, Z., Mubashiru, L. O., Xinyuan, L., Rui, R., & Shujuan, Z. (2023). An Automatic Jujube Fruit Detection and Ripeness Inspection Method in the Natural Environment. *Agronomy, 13*(2), 451. https://doi.org/10.3390/agronomy13020451

[8] Du, C., Wang, Y., Wang, C., Shi, C., & Xiao, B. (2020). Selective feature connection mechanism: Concatenating multi-layer CNN features with a feature selector. *Pattern Recognition Letters, 129*, 108-114. https://doi.org/10.1016/j.patrec.2019.11.015

[9] Elfwing, S., Uchibe, E., & Doya, K. (2017). Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks, 107*, 3-11. https://doi.org/10.1016/j.neunet.2017.12.012

[10] Fu, L., Feng, Y., Wu, J., Liu, Z., Gao, F., Majeed, Y., . . . Cui, Y. (2021). Fast and accurate detection of kiwifruit in orchard using improved YOLOv3-tiny model. *Precision Agriculture, 22*(3), 754-776. https://doi.org/10.1007/s11119-020-09754-y

[11] Gai, R., Li, M., & Chen, N. (2021). *Cherry detection algorithm based on improved YOLOv5s network*. Paper presented at the 2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys). https://ieeexplore.ieee.org/abstract/document/9780909

[12] Gevorgyan, Z. (2022). SIoU Loss: More Powerful Learning for Bounding Box Regression. *arXiv e-prints*, arXiv:2205.12740. https://ui.adsabs.harvard.edu/abs/2022arXiv220512740G

[13] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. *arXiv e-prints*, arXiv:1703.06870. doi:10.48550/arXiv.1703.06870: https://ui.adsabs.harvard.edu/abs/2017arXiv170306870H

[14] Huang, M.-L., & Wu, Y.-s. (2023). GCS-YOLOV4-Tiny: A lightweight group convolution network for multi-stage fruit detection. *Mathematical biosciences and engineering : MBE, 20 1*, 241-268.

[15] Jinhai, W., Zongyin, Z., Lufeng, L., Huiling, W., Wei, W., Mingyou, C., & Shaoming, L. (2023). DualSeg: Fusing transformer and CNN structure for image segmentation in complex vineyard environment. *Computers and Electronics in Agriculture, 206*, 107682. https://doi.org/10.1016/j.compag.2023.107682

[16] Jocher, G., Chaurasia, A., & Qiu, J. (2023). ultralytics. https://github.com/ultralytics/ultralytics

[17] Jordi, G.-M., Mar, F.-F., Eduard, G., M., B. P., Jochen, H., Josep-Ramon, M., . . . Javier, R.-H. (2023). Looking behind occlusions: A study on amodal segmentation for robust on-tree apple fruit size estimation. *Computers and Electronics in Agriculture, 209*, 107854. https://doi.org/10.1016/j.compag.2023.107854

[18] Junyang, C., Hui, L., Yating, Z., Daike, Z., Hongkun, O., & Xiaoyan, C. (2022). A Multiscale Lightweight and Efficient Model Based on YOLOv7: Applied to Citrus Orchard. *Plants, 11*(23), 3260. https://doi.org/10.3390/plants11233260

[19] Kang, H., & Chen, C. (2020). Fruit detection, segmentation and 3D visualisation of environments in apple orchards. *Computers and Electronics in Agriculture, 171*, 105302. https://doi.org/10.1016/j.compag.2020.105302

[20] 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'. *Precision Agriculture, 20*(6), 1107-1135. https://doi.org/10.1007/s11119-019-09642-0

[21] Lawal, O. M. (2021). Development of tomato detection model for robotic platform using deep learning. *Multimedia Tools and Applications, 80*(17), 26751-26772. https://doi.org/10.1007/s11042-021-10933-w

[22] Liu, G., Nouaze, J. C., Mbouembe, P. L. T., & Kim, J. H. (2020). YOLO-Tomato: A Robust Algorithm for Tomato Detection Based on YOLOv3. *Sensors, 20*(7), 2145. https://doi.org/10.3390/s20072145

[23] Liu, X., Zhao, D., Jia, W., Ji, W., Ruan, C., & Sun, Y. (2019). Cucumber Fruits Detection in Greenhouses Based on Instance Segmentation. *IEEE ACCESS, 7*, 139635-139642. https://ieeexplore.ieee.org/abstract/document/8843929

[24] Mubashiru, L. O. (2023). YOLOv5-LiNet: A lightweight network for fruits instance segmentation. *PloS one, 18*(3), e0282297. https://doi.org/10.1371/journal.pone.0282297

[25] Olarewaju M. Lawal. (2021). YOLOMuskmelon: Quest for Fruit Detection Speed and Accuracy Using Deep Learning. *IEEE ACCESS, 9*, 15221-15227. https://ieeexplore.ieee.org/abstract/document/9328755

[26] Peichao, C., Hao, F., Kunfeng, L., Jiachao, Z., & Shanda, L. (2023). MYOLO: A Lightweight Fresh Shiitake Mushroom Detection Model Based on YOLOv3. *Agriculture, 13*(2), 392. https://doi.org/10.3390/agriculture13020392

[27] Pengyu, C., Zhaojian, L., Kyle, L., Renfu, L., & Xiaoming, L. (2021). Deep Learning-based Apple Detection using a Suppression Mask R-CNN. *Pattern Recognition Letters*(147), 206-211. https://doi.org/10.1016/j.patrec.2021.04.022

[28] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv e-prints*, 1804.02767.

[29] Santos, T. T., Souza, L. L. d., Santos, A. A. d., & Avila, S. (2020). Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association. *Computers and Electronics in Agriculture, 170*, 105247. https://doi.org/10.1016/j.compag.2020.105247

[30] Sunkara, R., & Luo, T. (2022). *No more strided convolutions or pooling: A new CNN building block for low-resolution images and small objects.* Paper presented at the Joint European conference on machine learning and knowledge discovery in databases.

[31] Tian, Y., Yang, G., Wang, Z., Li, E., & Liang, Z. (2020). Instance segmentation of apple flowers using the improved mask R–CNN model. *Biosystems Engineering, 193*, 264-278. https://doi.org/10.1016/j.biosystemseng.2020.03.008

[32] Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.* Paper presented at the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.

[33] Wang, Z., Walsh, K., & Koirala, A. (2019). Mango Fruit Load Estimation Using a Video Based MangoYOLO—Kalman Filter—Hungarian Algorithm Method. *Sensors, 19*(12), 2742. https://doi.org/10.3390/s19122742

[34] Wendong, G., Yakun, L., Jing, Z., & Gang, J. (2021). An improved Tiny YOLOv3 for real-time object detection. *Systems Science & Control Engineering, 9*(1), 314-321. https://doi.org/10.1080/21642583.2021.1901156

[35] Xiang, L., Chengqi, L., Wenhai, W., Gang, L., Lingfeng, Y., & Jian, Y. (2022). Generalized Focal Loss: Towards Efficient Representation Learning for Dense Object Detection. *IEEE transactions on pattern analysis and machine intelligence, 45*(3), 3139-3153. https://ieeexplore.ieee.org/abstract/document/9792391

[36] Yanchao, Z., Wenbo, Z., Jiya, Y., Leiying, H., Jianneng, C., & Yong, H. (2022). Complete and accurate holly fruits counting using YOLOX object detection. *Computers and Electronics in Agriculture, 198*, 107062. https://doi.org/10.1016/j.compag.2022.107062

[37] Yang-Yang, Zheng, Jian-Lei, Kong, Xue-Bo, Jin, . . . Zuo. (2019). CropDeep: The Crop Vision Dataset for Deep-Learning-Based Classification and Detection in Precision Agriculture. *Sensors, 19*(5), 1058. https://doi.org/10.3390/s19051058

[38] Yichen, Q., Yaohua, H., Zhouzhou, Z., Huanbo, Y., Kaili, Z., Juncai, H., & Jiapan, G. (2022). A Counting Method of Red Jujube Based on Improved YOLOv5s. *Agriculture, 12*(12), 2071. https://doi.org/10.3390/agriculture12122071

[39] Yunchao, T., Hao, Z., Hongjun, W., & Yunqi, Z. (2023). Fruit detection and positioning technology for a Camellia oleifera C. Abel orchard based on improved YOLOv4-tiny model and binocular stereo vision. *Expert Systems With Applications, 211*, 118573. https://doi.org/10.1016/j.eswa.2022.118573

[40] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020). *Distance-IoU loss: Faster and better learning for bounding box regression.* Paper presented at the Proceedings of the AAAI Conference on Artificial Intelligence.