2024 11th International Conference on Signal Processing and Integrated Networks (SPIN) | 979-8-3503-0843-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/SPIN60856.2024.10511673

Comparative study of YOLOv8 and YOLO-NAS for agriculture application

Yogesh Kumar
computer science Engineering
UPES
Dehradun, India
yogeshchauhan114@gmail.com

Pankaj Kumar
computer science Engineering
Nirma University
Ahmedabad, India
pankaj.kumar@nirmauni.ac.in

***Abstract*—Agricultural tasks have significantly improved as a result of ongoing machine learning (ML) improvements. Deep learning (DL), which has a significant capacity for extracting high-dimensional features from fruit images, is widely applied to the automated detection and harvesting of fruits. In the fields of fruit recognition and automated harvesting, Convolutional Neural Networks (CNNs) have demonstrated the ability to attain speed and accuracy levels that rival human performance. This article compares the performance of YOLOv8m with YOLO-NASl for grapes detection. In this research, the YOLOv8 and YOLO-nas object detection models, including their different scales, were trained using a publicly available Embrapa WGISD dataset. The dataset consists of 300 digital images of grapes growing in vineyard settings, and it includes a total of 4,432 annotations. The performance of the YOLOv8m and YOLO-NASl model were evaluated using metrics such as recall, precision, and the mean average precision (mAP@50). In the subset of test data, YOLOv8m achieved the top overall performance, with a precision (0.855), mAP@50 (0.885), and recall (0.827), while best recall was obtained from YOLO-NASl (0.934).**

***Index Terms*—Object detection, Fruit detection, YOLOv8, YOLO-NAS, Deep learning, Agriculture application**

I. Introduction

The cultivation, observation, and harvest of crops have all been revolutionised by agricultural automation, which has become a crucial component of contemporary farming practises. The precise and effective detection of fruits in orchards is an important aspect component of agricultural automation. The accurate location and identification of ripe fruits has enormous potential for improving resource utilisation, lowering labour costs, and raising yield quality overall. Traditional fruit identification techniques frequently involved manual labour, which is time-consuming, laborious, and error-prone. The field of fruit identification has seen a significant transformation with the development of machine learning, computer vision, and technological advances in sensors. Today's automated technologies may offer in-the-moment, non-intrusive assessments of fruit maturity and output, empowering farmers to make wise choices and act quickly. Recently, there has been a lot of interest in using robotics as well as artificial intelligence (AI) approaches for automating agricultural activities (as seen in Figure 1). The study of artificial neural networks (ANN) led to the development of the deep learning (DL) idea [1]. Many academics have studied fruit identification and recognition using DL for autonomous harvesting since DL has a significant ability for extracting highly complex features from fruit photos. Convolutional neural networks (CNN) is adept in effectively identifying patterns in a multidimensional space, which is developed by [2], [3]. Numerous academics have studied in-depth and extensively fruit identification and recognition based on DL for autonomous harvesting due to DL's potent capacity to extract high-dimensional features from fruit photos. [4] To find apparent flaws in sour lemons, an upgraded CNN (15, 16, and 18 layers) has been proposed. The upgraded CNN was found to outperform conventional methods for extracting fruit features, such as the decision tree, local binary pattern, support vector machine, k-nearest neighbour and histogram of oriented gradient (HOG), by achieving a 100% accuracy. [5] A CNN-based fruit detecting system was suggested by the author. To assess the proposed approach, the Fruits-360 dataset was used. Training and testing accuracy are, respectively, 99.79% and 100%. One of the most well-known and powerful fruit detection algorithms is YOLO. It is able to detect and categorise target fruits in a single image. Different version of YOLO already introduced [6]–[16]. Due to the benefits of YOLO-based systems for fruit detection and classification are commonly used. [17] Developed a technique based on YOLO-v2 to find and count the number of mangoes in fruit images captured by a UAV. Model has been taken processing time 80ms and got on average detected accuracy 96.1%. A mechanical harvesting technique columncomb has been proposed for effective gathering of litchi fruits by [18]. To boost fruit detection accuracy, author combines YOLOv5 with classical image processing technique [19]. In this work we are comparing YOLOv8 medium and YOLO-NAS large models on Embrapa WGISD grapes dataset, mean average precision (mAP@50), precision and recall are the comparison metrices used for analysing the performance of the models.

A. Related Work

Fruit harvesting is still primarily done manually, despite the fact that agricultural machinery is frequently employed in orchards for ditching, fertilisation, pruning fruit trees, sprinkling irrigation, and other chores. However, due to the progress in computer technology and the swift evolution of autonomous

979-8-3503-0843-3/24/$31.00 ©2024 IEEE

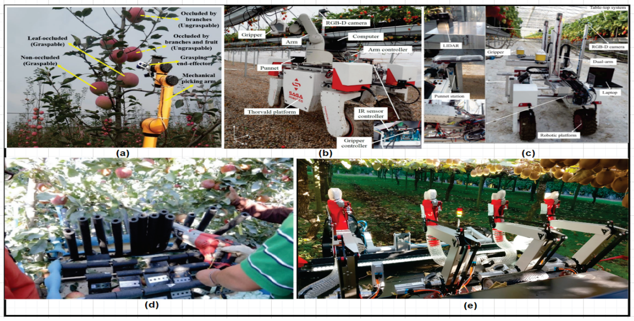

Fig. 1. Robots used for harvesting **(a and d)** Robot that is used to harvest apple [20], [21]; **(b and c)** Robot that is used to harvest strawberry [22], [23]; **(e)** Robot that is used to harvest kiwifruit [24]

control methods, particularly in the domains of robotics and computer vision technology, research and innovations in the field of fruit harvesting mechanisms have experienced rapid growth and advancement. [25] Using Intel RealSense cameras, MaskRCNN was utilised to identify the tomatoes from images captured in a production greenhouse. [26] To distinguish the apples into different classes, a multi-class fruit identification method based on deep learning is developed. VGG16 was used for this implementation. [27] YOLOv3-based approach for detecting green mangoes. [28] A novel method for fruit detection in environments that are natural is proposed. This method can be used by robots that automatically harvest fruit as well as systems for estimating yields and quality control. [27] YOLOv3-based proposed light-weight detecting technique for green mangoes. This Light-YOLOv3 model has an F1 score that is 4.5% higher than YOLOv3 and an execution speed that is five times faster, which is more than enough to suit the real-time operation needs of choosing robots. [29] Developed a YOLOv3-based technique for locating the litchi fruits and it stems in a night environment. This technique identified the fruit stems' regions of interest (RoIs) according to the bounding boxes of the litchi fruits, and then segments the fruit stems one by one using U-Net to get an impressive recognition result. [30] To create the enhanced YOLOv3-litchi model, the original YOLOv3 network's output scale was modified, and its depth was decreased. Compared to YOLOv3, the enhanced YOLOv3-litchi model can identify litchi at various growth stages more quickly and precisely. [31] Provides a thorough analysis of fruit detection and recognition using deep learning for autonomous harvesting.

B. *YOLOv8's Network Architecture*

The C3 module is swapped out with the C2f module based on the CSP principle, and the backbone of YOLOv8 is identical with YOLOv5. The C2f module merged C3 and ELAN, building on the ELAN concept from YOLOv7, such that YOLOv8 could receive greater amounts of gradient flow knowledge while still maintaining its lightweight nature. The more popular SPPF module was kept utilised at the ending of the backbone, and three Maxpools of capacity 5×5 were passed serially before each layer was concatenated to ensure accuracy of objects of varying scales while also maintaining a low weight. The feature fusion approach still employed by YOLOv8 in the neck section is PAN-FPN, which improves the fusion and usage of feature layer data at various scales. The neck module was created by the authors of YOLOv8 by combining the final decoupled head architecture, numerous C2f modules, and two up-sampling. The final component of the neck in YOLOv8 was constructed using the same concept as the head in YOLOx. It increased accuracy by combining confidence and regression boxes. All YOLO versions are supported by YOLOv8, which may also switch between them at will. Its broad hardware compatibility (CPU-GPU) further increases its adaptability. Figure 2 depicts the YOLOv8 network architectural diagram.

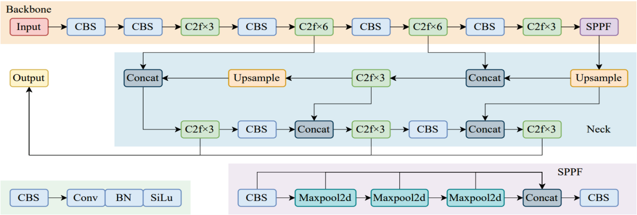

Fig. 2. Architecture for YOLO-v8 [10]

.

C. *YOLO-NAS Network Architecture*

YOLO-NAS [16] models' architecture found utilising Deci's exclusive NAS technology, AutoNAC engine. This engine was used to determine the ideal block types, block counts, and channel counts for each stage as well as their sizes and structures. The total number of architecture variants in the NAS search space is 10^4. The AutoNAC engine, which is hardware and data conscious, analyses every element of the inference stack, incorporating compilers and quantization, and then focuses in on an area known as the ”efficiency frontier” to discover the most effective models. To ensure that the model is compatible with Post-Training Quantization (PTQ), Quantization-Aware RepVGG (QA-RepVGG) blocks are introduced into the structure of the model during the NAS process. Benefits from 8-bit quantization and reparameterization are achieved by using quantization-aware ”QSP” and ”QCI” modules made up of QA-RepVGG blocks. This enables PTQ with the least amount of accuracy loss. The researchers additionally utilise a hybrid quantization technique that uses selective quantization of particular layers to enhance accuracy and latency tradeoffs while retaining overall performance. To enhance their ability to recognise objects, YOLO-NAS models additionally utilise methods of attention and time-based inference reparametrization.

II. Experimental Setup

A. *Dataset used*

In this experiment, YOLOv8m and YOLO-NASl models were trained using a publicly available dataset known as the

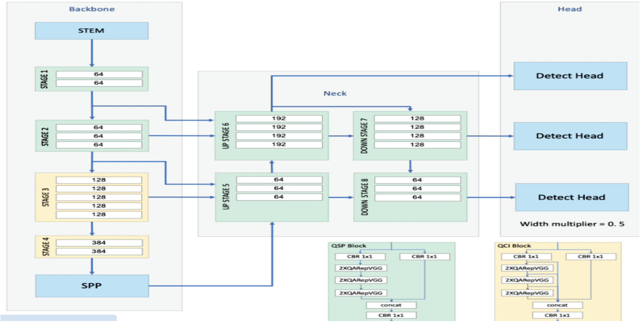

Fig. 3. Architecture for YOLO-NAS [16]

.

Embrapa WGISD [32]. This dataset was developed with the aim of offering images and annotations for the exploration of object detection and instance segmentation in the context of image-based monitoring and field robotics applied to viticulture. This dataset encompasses instances from five distinct grape varieties captured in real-field conditions, showcasing a range of variations in grape orientation, lighting conditions, and focus. Furthermore, it also encompasses genetic and phenological diversity, encompassing differences in attributes such as shape, color, and compactness among the grape instances. This dataset comprises 300 images, and within these images, there are a total of 4,432 grape clusters that have been marked and delineated using bounding boxes. In this experiment, 70%, 15% and 15% images used for training, validation and testing respectively.

B. Working principle of YOLOv8

A deep neural network architecture is used by YOLOv8, a variation of the YOLO object detection framework, to carry out effective and precise real-time object detection. By using various-sized anchors during prediction, it can recognise objects of varying sizes and aspect ratios thanks to the multi-scale technique it uses. A backbone network that extracts features from the given input image serves as the foundation of YOLOv8's design, which is followed by numerous detection heads to predict bounding boxes, object categories, and confidentiality. Predictions of various scales are produced by these detection heads. The depth and width of the network are balanced between computational effectiveness and detection efficiency. In addition, YOLOv8 makes use of anchor-based identification, focal loss, and feature pyramid fusion. These techniques work together to let the system manage a variety of object scales and retain high detection accuracy, making it appropriate for real-time object identification applications.

C. Working principle of YOLO-NAS

YOLO framework efficiency and neural architecture search (NAS) methods are combined in YOLO-NAS, a novel development in object identification. To automatically find the best network designs for object detection, YOLO-NAS incorporates a search algorithm into the YOLO architecture. The objective of YOLO-NAS is to improve both precision and efficacy in real-time detection of objects by iteratively optimising design of architecture's, combinations of layer, and extraction of feature patterns. This method takes advantage of YOLO's single-pass object detection capabilities and NAS's capacity to independently modify neural networks to produce an effective and efficient solution for precise and quick detection of objects across a variety of application domains.

D. Methodology

In this section, we outline the precise methods for grapes detection utilising the cutting-edge object detection mechanism YOLOv8m and YOLO-NASl. Our strategy seeks to take advantage of YOLOv8 and YOLO-NAS architecture's advantages for precise and effective grapes detection across various agricultural settings. A number of crucial steps make up the methodology:

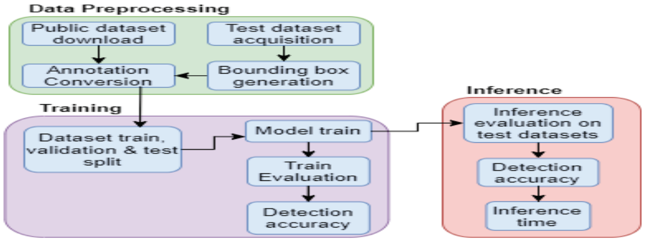

Fig. 4. Framework for conducting training and assessing the performance of models in experiment.

Figure 4 shows the framework used to detect grapes using YOLOv8 and YOLO-NAS technology. In **Data Preprocessing**, we start by accumulating a broad and representative dataset of pictures of several grapes varieties in various growth phases and lighting scenarios from the github repository of Embrapa WGISD. In **Data Annotation** phase The bounding boxes around the grapes are expertly annotated with careful consideration. For **Training** we used YOLOv8m and YOLO-NASl, which is well-known for its effectiveness and precision in real-time object recognition. The architecture's single-pass detection technique and capacity to manage an extensive amount of object classes fit the needs of object detection effectively. The Embrapa WGISD dataset is used to train the model. Optimised convergence is achieved by fine-tuning training parameters like learning rate, batch size, and optimisation technique. In **Model Evaluation** On a different validation dataset, the effectiveness of the trained model is assessed. The model's accuracy and capacity to identify fruits in various settings are measured using metrics including mAP, precision, and recall. In **Inference**, we analyse the performance of both the detection models on the test dataset in order to assess the efficacy of the YOLOv8m and YOLO-NASl based on the grapes detection. This sheds light on the relative advantages and disadvantages of these two techniques.

E. Training

In order to train YOLOv8m we installed the ultralytics package. The Yolo Command Line Interface (CLI) is made available via this. We don't be required to clone the repository separately or install the prerequisites, which is a major benefit. In order to train YOLO-NASl we installed the super-gradients package. Batch size is 8, image size 640×640 and 100 epochs used to train the model in both the case. On a computer with a 32GB Nvidia Tesla V100-PCIE GPU and a intel Xeon 256GB RAM processor, all the training experiments were conducted. Training time taken by the YOLOv8m and YOLO-NASl is 18.7 minutes and 43 minutes respectively.

F. Performance Assessment

mAP at 0.50 confidence levels (map50), Precision and Recall metrices used to evaluate the performance of the YOLOv8m and YOLO-NASl models for grapes detection. These metrics offer an unbiased way to quantify how accurately and effectively the models can identify grapes.

Mean Average Precision (mAP) serves as a widely employed evaluation metric in the field of object detection. It assesses the balance between precision and recall by computing the average precision (AP) for individual classes and subsequently determining the average of these values across all classes [33]. The Average Precision (AP) assesses precision at various recall levels by calculating the area under the precision-recall curve. It is described mathematically as shown in Equation 1, where precision(r) is the precision at a particular recall level (r). Better object detection performance is indicated by a greater mAP when recall and precision are taken into account.

$$AP = \int_0^1 precision(r)\, dr \tag{1}$$

Multiple versions of mean average precision (mAP) are customized to correspond with distinct IoU thresholds. To illustrate, mAP@0.5 computes the average precision specifically at an IoU threshold of 0.5.

$$mAP(50) = \frac{1}{n}\sum_{k=1}^{n} AP_k \tag{2}$$

Precision stands as a crucial metric utilized in object detection tasks for assessing the model's precision in generating positive predictions. It quantifies the proportion of accurately identified positive instances within the total instances that the model predicted as positive. [33].

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

Recall, which is also known as the true positive rate or sensitivity, evaluates the proportion of actual positive instances that the model accurately identifies. [33], [34].

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

1) Loss: Loss is a numerical representation of the difference between a model's predicted outputs and the real or anticipated values. A decrease in the loss value indicates a higher level of agreement between predictions and actual ground truth data, and the goal during training is to minimize this loss in order to improve the model's performance.

The loss functions that are used vary among various YOLO versions. In YOLOv8m, cls loss, box loss and dfl are used. YOLO-NASl utilizes cls loss, dfl loss and IoU loss. Below, we give a brief explanation of each of these losses.

- **Classification loss (cls):** Classification loss [35] uses categorical cross-entropy loss to quantify the discrepancy between the expected probability for various classes and the actual class labels.
- **Box loss:** The box loss [36] is used to measure how much the predicted bounding box coordinates differ from the actual coordinates of the ground truth boxes. This function typically utilizes metrics such as mean squared error (MSE) or smooth L1 loss to calculate this disparity.
- **Distributional Focal Loss (dfl):** The use of Distributional Focal Loss [37] is intended to tackle class imbalance problems in object detection by assigning higher significance to intricate instances that present a more significant difficulty in achieving precise classification.
- **Intersection over Union loss (IoU):** The IoU loss [38] assesses the alignment between predicted bounding boxes and the real ground truth boxes by utilizing the IoU metric, which quantifies the extent of overlap between the predicted and ground truth bounding boxes.

III. Results

The performance metrics displayed in Table I provides insightful information about the differences in performance and characteristics between YOLOv8m and YOLO-NASl, where table indicated that the map@50, precision and recall of the YOLOv8m are 88%, 85% and 82% respectively. The same metrices for the YOLO-NAS were 81%, 22% and 93%. These metrices illustrating its general efficiency in detecting and localizing objects. It's important to note that YOLO-NASl achieves impressive Recall scores, exceeding 0.93. This indicates that the model excels in identifying the vast majority of true positive grape instances. This attribute could be of utmost importance in situations where the primary objective is to maximize the detection of positive cases. However, when it comes to Precision scores, a distinct trend emerges. YOLO-NASl exhibits significantly lower Precision scores, particularly when contrasted with its high Recall scores. Poor precision scores imply that although the models are effective at identifying positive cases, they also produce a large number of false positives, means identifying non-grapes events as grapes. These false warnings may result in resource misallocation and an ineffective system. In conclusion, the YOLO-NAS architecture's strong recall may be advantageous in circumstances when skipping a good instance can have negative consequences, such as opponent ship identification. Turning attention to the YOLOv8m model, there is a sub-

TABLE I
EVALUATION METRICS ON TESTING PHASE TO ASSESS THE PERFORMANCE OF BOTH YOLOV8M AND YOLO-NASL.

Model	Precision	Recall	mAP@50
YOLOv8m	0.855	0.827	0.885
YOLO-NASl	0.220	0.934	0.812

stantial enhancement in precision and mAP@50 compared to YOLO-NASl. Specifically, the precision reaches 0.855, while the mAP@50 reaches 0.885. Figure 5 and 6 shows the progress metrices of YOLOv8m and YOLO-NASl models for 100 epochs. During the early epochs, there is a significant amount of variation observed in YOLOv8m model. mAP@50 in YOLOv8m model demonstrate good convergence rate. In YOLO-NAS model a significant amount of variation observed in precision. Metrices such as mAP@50 and recall demonstrate a good convergence rate. The patterns we have observed underscore the intricate dynamics of how the evaluated models reach convergence and maintain stability. The YOLO-NASl model exhibited slower convergence and a delay in achieving stability in its precision metrics. To attain their best performance, these models need hyperparameter tuning to ensure strong convergence and increased stability.

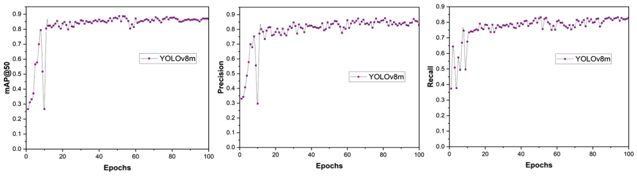

Fig. 5. The YOLOv8m model, trained on a publicly available grape dataset for 100 epochs, achieved the following performance metrics: mean average precision at 0.50 confidence levels (mAP50), precision and recall.

In terms of precision, YOLOv8m performs substantially better than YOLO-NASl, obtaining an impressive 85% precision. As a result, it can be concluded that YOLOv8m performs better at reducing false positive detections and offering more precise grape identifications.

With an impressive recall rate of 93%, YOLO-NASl surpasses YOLOv8m in terms of recall performance. Consequently, YOLO-NASl excels at capturing a greater proportion of actual grapes within the dataset, thereby enhancing its ability for grape detection.

The YOLOv8m model exhibits superior performance, particularly when it comes to mean average precision, achieving an impressive 88%. This demonstrates that YOLOv8m excels in tasks related to grape detection by effectively striking a balance between precision and recall.

Regarding the analysis of losses, Figure 7 displays the mean losses observed throughout the training and validation phases for YOLOv8m and YOLO-NASl. In all three categories—box, dfl, and cls—the YOLOv8m model shows greater validation-phase losses than training-phase losses. This disparity indicates a certain level of overfitting, wherein the models become proficient at understanding the training data but encounter difficulties when applying this knowledge to new, unfamiliar data. The YOLO-NASl model demonstrates nearly identical training and validation losses, which indicates less overfitting. Figure 8 illustrates that both models accurately identify the relevant instances of interest.

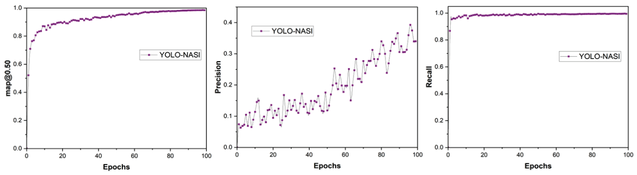

Fig. 6. The YOLO-NASl model, trained on a publicly available grape dataset for 100 epochs, achieved the following performance metrics: mean average precision at 0.50 confidence levels (mAP@0.50), precision and recall.

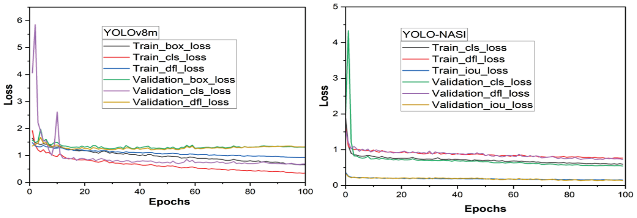

Fig. 7. Training and validation losses of YOLOv8m and YOLO-NASl

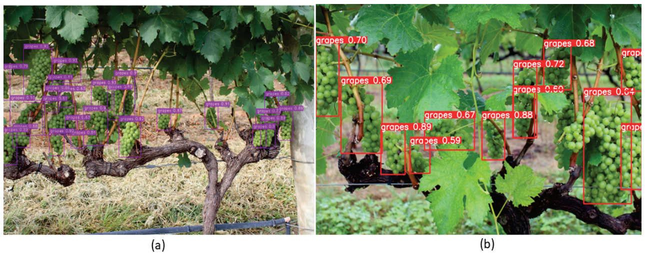

Fig. 8. Figure **(a)** and **(b)** depicts the inference test by YOLO-NASl and YOLOv8 respectively.

IV. CONCLUSION

In this paper, Comparison of the effectiveness of YOLOv8m and YOLO-NASl in the task of grape detection has revealed useful information. Due to its significant advantage in precision and mAP@0.50, YOLOv8m is a favorable option for tasks that require a reduction in false positives and the maintenance of a high overall level of accuracy. On the other hand, YOLO-NASl performs exceptionally well in recall, showing that it can accurately identify the majority of the grapes in the dataset. The decision between these models should be based on the particular needs of the application, with YOLOv8m being preferred for activities requiring high precision and YOLO-NASl for applications prioritising comprehensive detection.

Future investigations should prioritize the development of hybrid models or fine-tuning methodologies that integrate the most advantageous aspects of both YOLOv8 and YOLO-NAS architectures, aiming to strike a harmonious equilibrium between recall and precision. This endeavor holds the potential to enhance the overall efficacy of grape detection systems across a wide array of practical scenarios, such as precision agriculture and vineyard management. Additionally, to push the boundaries of grape recognition technology, further research can emphasize the optimization of these models tailored to specific grape varieties and varying environmental conditions.

References

[1] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[2] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[4] A. Jahanbakhshi, M. Momeny, M. Mahmoudi, and Y.-D. Zhang, “Classification of sour lemons based on apparent defects using stochastic pooling mechanism in deep convolutional neural networks,” *Scientia Horticulturae*, vol. 263, p. 109133, 2020.

[5] S. Sakib, Z. Ashrafi, and M. A. B. Siddique, “Implementation of fruits recognition classifier using convolutional neural network algorithm for observation of accuracies for various hidden layers,” *arXiv preprint arXiv:1904.00783*, 2019.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[7] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[8] ——, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[9] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “Yolov4: Optimal speed and accuracy of object detection,” *arXiv preprint arXiv:2004.10934*, 2020.

[10] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Yifu), C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, and M. Jain, “ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation,” Nov. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7347926

[11] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, “You only learn one representation: Unified network for multiple tasks,” *arXiv preprint arXiv:2105.04206*, 2021.

[12] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, “Yolox: Exceeding yolo series in 2021,” *arXiv preprint arXiv:2107.08430*, 2021.

[13] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie *et al.*, “Yolov6: A single-stage object detection framework for industrial applications,” *arXiv preprint arXiv:2209.02976*, 2022.

[14] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7464–7475.

[15] G. Jocher, A. Chaurasia, and J. Qiu, “Ultralytics yolov8,” 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[16] S. Aharon, Louis-Dupont, Ofri Masad, K. Yurkova, Lotem Fridman, Lkdci, E. Khvedchenya, R. Rubin, N. Bagrov, B. Tymchenko, T. Keren, A. Zhilko, and Eran-Deci, “Super-gradients,” 2021. [Online]. Available: https://zenodo.org/record/7789328

[17] J. Xiong, Z. Liu, S. Chen, B. Liu, Z. Zheng, Z. Zhong, Z. Yang, and H. Peng, “Visual detection of green mangoes by an unmanned aerial vehicle in orchards based on a deep learning method,” *Biosystems Engineering*, vol. 194, pp. 261–272, 2020.

[18] C. Li, J. Lin, B. Li, S. Zhang, and J. Li, “Partition harvesting of a column-comb litchi harvester based on 3d clustering,” *Computers and Electronics in Agriculture*, vol. 197, p. 106975, 2022.

[19] Z. Miao, X. Yu, N. Li, Z. Zhang, C. He, Z. Li, C. Deng, and T. Sun, “Efficient tomato harvesting robot based on image processing and deep learning,” *Precision Agriculture*, vol. 24, no. 1, pp. 254–287, 2023.

[20] B. Yan, P. Fan, X. Lei, Z. Liu, and F. Yang, “A real-time apple targets detection method for picking robot based on improved yolov5,” *Remote Sensing*, vol. 13, no. 9, p. 1619, 2021.

[21] L. He, H. Fu, M. Karkee, and Q. Zhang, “Effect of fruit location on apple detachment with mechanical shaking,” *Biosystems Engineering*, vol. 157, pp. 63–71, 2017.

[22] Y. Xiong, Y. Ge, L. Grimstad, and P. J. From, “An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation,” *Journal of Field Robotics*, vol. 37, no. 2, pp. 202–224, 2020.

[23] Y. Xiong, C. Peng, L. Grimstad, P. J. From, and V. Isler, “Development and field evaluation of a strawberry harvesting robot with a cable-driven gripper,” *Computers and electronics in agriculture*, vol. 157, pp. 392–402, 2019.

[24] H. A. Williams, M. H. Jones, M. Nejati, M. J. Seabright, J. Bell, N. D. Penhall, J. J. Barnett, M. D. Duke, A. J. Scarfe, H. S. Ahn *et al.*, “Robotic kiwifruit harvesting using machine vision, convolutional neural networks, and robotic arms,” *biosystems engineering*, vol. 181, pp. 140–156, 2019.

[25] M. Afonso, H. Fonteijn, F. S. Fiorentin, D. Lensink, M. Mooij, N. Faber, G. Polder, and R. Wehrens, “Tomato fruit detection and counting in greenhouses using deep learning,” *Frontiers in plant science*, vol. 11, p. 571299, 2020.

[26] F. Gao, L. Fu, X. Zhang, Y. Majeed, R. Li, M. Karkee, and Q. Zhang, “Multi-class fruit-on-plant detection for apple in snap system using faster r-cnn,” *Computers and Electronics in Agriculture*, vol. 176, p. 105634, 2020.

[27] Z.-F. Xu, R.-S. Jia, H.-M. Sun, Q.-M. Liu, and Z. Cui, “Light-yolov3: fast method for detecting green mangoes in complex scenes using picking robots,” *Applied Intelligence*, vol. 50, pp. 4670–4687, 2020.

[28] G. Lin, Y. Tang, X. Zou, J. Cheng, and J. Xiong, “Fruit detection in natural environment using partial shape matching and probabilistic hough transform,” *Precision Agriculture*, vol. 21, pp. 160–177, 2020.

[29] C. Liang, J. Xiong, Z. Zheng, Z. Zhong, Z. Li, S. Chen, and Z. Yang, “A visual detection method for nighttime litchi fruits and fruiting stems,” *Computers and Electronics in Agriculture*, vol. 169, p. 105192, 2020.

[30] H. Wang, L. Dong, H. Zhou, L. Luo, G. Lin, J. Wu, and Y. Tang, “Yolov3-litchi detection method of densely distributed litchi in large vision scenes,” *Mathematical Problems in Engineering*, vol. 2021, pp. 1–11, 2021.

[31] F. Xiao, H. Wang, Y. Xu, and R. Zhang, “Fruit detection and recognition based on deep learning for automatic harvesting: An overview and review,” *Agronomy*, vol. 13, no. 6, p. 1625, 2023.

[32] T. Santos, L. De Souza, A. Dos Santos, and A. Sandra, “Embrapa wine grape instance segmentation dataset–embrapa wgisd,” *Zenodo*, 2019.

[33] R. Padilla, S. L. Netto, and E. A. Da Silva, “A survey on performance metrics for object-detection algorithms,” in *2020 international conference on systems, signals and image processing (IWSSIP)*. IEEE, 2020, pp. 237–242.

[34] R. Padilla, W. L. Passos, T. L. Dias, S. L. Netto, and E. A. Da Silva, “A comparative analysis of object detection metrics with a companion open-source toolkit,” *Electronics*, vol. 10, no. 3, p. 279, 2021.

[35] S. Wu, J. Yang, X. Wang, and X. Li, “Iou-balanced loss functions for single-stage object detection,” *Pattern Recognition Letters*, vol. 156, pp. 96–103, 2022.

[36] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, “A comprehensive survey of loss functions in machine learning,” *Annals of Data Science*, pp. 1–26, 2020.

[37] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, and J. Yang, “Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 002–21 012, 2020.

[38] S. Du, B. Zhang, and P. Zhang, “Scale-sensitive iou loss: An improved regression loss function in remote sensing object detection,” *IEEE Access*, vol. 9, pp. 141 258–141 272, 2021.