Green Intelligent Systems and Applications

Research Article
Volume 5(2), 2025, 191−204
https://doi.org/10.53623/gisa.v5i2.934

Implementation of the You Look Only Once (YOLOv11) Algorithm to Detect the Ripeness of Golden Melons

Lucky Tandoballa, Ery Hartati*

Faculty of Computer Science and Engineering, Universitas Multi Data Palembang, South Sumatra, Indonesia

*Correspondence: ery_hartati@mdp.ac.id

SUBMITTED: 30 November 2025; REVISED: 20 December 2025; ACCEPTED: 23 December 2025

ABSTRACT: Melon is a horticultural commodity with high economic value, and characteristics such as sweetness, aroma, texture, and phytonutrient content significantly influenced consumer preference. Conventional methods for determining melon ripeness were time-consuming, required considerable expertise, and were often prone to subjective errors, ultimately slowing the production and distribution process. This study aimed to detect the ripeness level of golden melon fruit non-destructively using the YOLOv11 algorithm, focusing on external physical characteristics as the basis for classification. The objectives included applying transfer learning to categorize golden melon into ripe and unripe classes and evaluating model performance using precision, recall, mAP50, mAP50-95, and F1-score. The research methodology consisted of a literature review, dataset collection from previous studies, system design, implementation, and performance testing. The dataset was divided into 70% training, 20% validation, and 10% testing data, and the Adam optimizer was used during the training phase. Based on four experimental scenarios, scenario 3 produced the best and most consistent results, achieving a precision of 90.58%, a recall of 90.79%, an mAP50 of 97.31%, an mAP50-95 of 88.84%, and an F1-score of 92.97%. These findings demonstrated that scenario 3 offered optimal performance for detecting golden melon ripeness. Thus, the model was highly reliable overall.

KEYWORDS: Adam optimizer; melon ripeness detection; object detection; YOLOv11

1. Introduction

Melon is one of the agricultural commodities in the form of a horticultural crop with high economic value [1]. Melon (*Cucumis melo* L.), watermelon (*Citrullus lanatus*), cucumber (*Cucumis sativus* L.), and pumpkin (*Cucurbita*) all belong to the Cucurbitaceae plant family, which consists of approximately 90 genera and 750 species [2]. According to data from the Central Statistics Agency (BPS), melon production increased by 13.8% in the past three years compared to total production in 2020 [3]. This increase indicated that melon production in Indonesia continued to grow annually, making quality control particularly fruit ripeness sorting, an increasingly important process prior to market distribution.

Consumers generally preferred melons with high sweetness, affordable prices, crisp flesh, medium to large fruit sizes, a strong aroma, and long shelf life [4]. One common indicator of melon ripeness was the thickness and roughness of the skin mesh; a thicker and rougher mesh typically indicated a ripe fruit [5]. In practice, determining harvest time relied heavily on visual characteristics such as fruit shape, skin color, size, and sound, as well as physical parameters like fruit firmness [6]. However, conventional ripeness assessment was time-consuming, required expert judgment, was prone to subjectivity, and could hinder production efficiency. Consequently, the development of automated systems capable of accurately and rapidly detecting fruit ripeness became increasingly important, particularly in the context of agricultural monitoring using robotic systems [7].

To achieve accurate and consistent results, automation systems in agriculture commonly utilized machine learning methods to identify ripe and harvestable fruit [8]. Among various machine learning algorithms, Support Vector Machine (SVM) had been widely used and demonstrated good performance in solving digital image classification problems [9]. Several previous studies explored melon ripeness detection using machine learning approaches under relatively controlled or non-challenging dataset conditions, where images were captured directly from melon objects. For instance, melon ripeness detection enhanced with attention mechanisms using the YOLOv8 algorithm achieved a precision of 97.9%, recall of 96.2%, mAP50 of 98.1%, and mAP50–95 of 94.1% [10]. Other studies employing SVM combined with Gray Level Co-Occurrence Matrix (GLCM) feature extraction reported accuracies ranging from 76.0% to 82.0% [1, 4, 5].

More recently, research began to consider more challenging dataset conditions. A study in [11] investigated melon ripeness detection under various environmental conditions such as different viewing angles, target overlap, leaf shading, and variations in fruit size and scale using an optimized YOLOv8 model with a lightweight MobileNetV3 backbone and Coordinate Attention mechanism. This approach achieved a precision of 85.9% and an mAP50 of 97.4%. Although these results demonstrated the robustness of YOLOv8-based architectures under challenging conditions, the variability in image capture time, lighting conditions, shadows, and bright illumination remained a significant challenge for accurate object recognition in real agricultural environments.

Despite the promising performance of YOLOv8- and SVM-based methods, existing studies had not yet explored the potential of newer YOLO architectures on challenging melon ripeness datasets. In particular, the performance of YOLOv11, an advanced version of the YOLO family, had not been evaluated for golden melon ripeness detection under complex environmental conditions. This represented a clear research gap, as YOLOv11 introduced substantial architectural improvements that might enhance detection stability, accuracy, and efficiency compared to earlier versions.

YOLOv11, released by Ultralytics in 2025, incorporated a ConvNeXtV3 backbone and an optimized Dynamic Head, resulting in improved feature representation and more stable object detection. Compared to YOLOv8 and YOLOv10, YOLOv11 reportedly increased mean Average Precision (mAP) by approximately 3–5% and achieved processing speeds that were 20–25% faster [12]. Previous work applying YOLOv11 to fruit ripeness classification, such as mango ripeness detection, achieved an overall accuracy of 97.3%, with precision, recall, and F1-score all exceeding 97% [13]. These findings suggested that YOLOv11 had strong potential for real-time agricultural applications; however, its effectiveness had not yet been validated on challenging golden melon ripeness datasets.

Therefore, this study aimed to implement and evaluate the YOLOv11 algorithm with the Adam optimizer for detecting golden melon ripeness based on external physical characteristics of the fruit. This research specifically focused on challenging dataset conditions, including variations in camera angles, target overlap, leaf shading, differences in melon size and scale at various growth stages, and varying image capture times. The central research question guiding this study was: How did the YOLOv11 algorithm perform in detecting golden melon ripeness under diverse environmental and imaging conditions, as measured by precision, recall, and mean Average Precision (mAP)? The findings of this study were expected to contribute to the development of faster, more accurate, and more efficient automated melon ripeness detection systems, thereby supporting quality maintenance prior to distribution and sale to consumers.

2. Materials and Methods

2.1.Melon.

Melon (*Cucumis melo* L.) is a horticultural crop from the Cucurbitaceae family widely cultivated in tropical regions. It is known for its distinctive aroma, sweet taste, and crunchy texture, making it a popular fruit among consumers [11]. Melon is characterized by its high water content (90–95%) and high economic value in the agribusiness sector [14]. Consuming ripe melon provides higher sugar and nutrient content than unripe melon. Therefore, determining the ripeness of melon was crucial to maintain its quality and nutritional benefits [15].

2.2.Computer vision.

Computer vision is a branch of artificial intelligence that enables computers to "see," recognize, and interpret images or videos in a manner similar to humans. This technology combines digital image processing with machine learning methods to automatically detect, classify, segment, and track objects. The development of deep learning has driven significant progress in computer vision, particularly through the use of Convolutional Neural Network (CNN) architectures, which help identify and extract complex visual features [16].

2.3.Convolutional Neural Network (CNN).

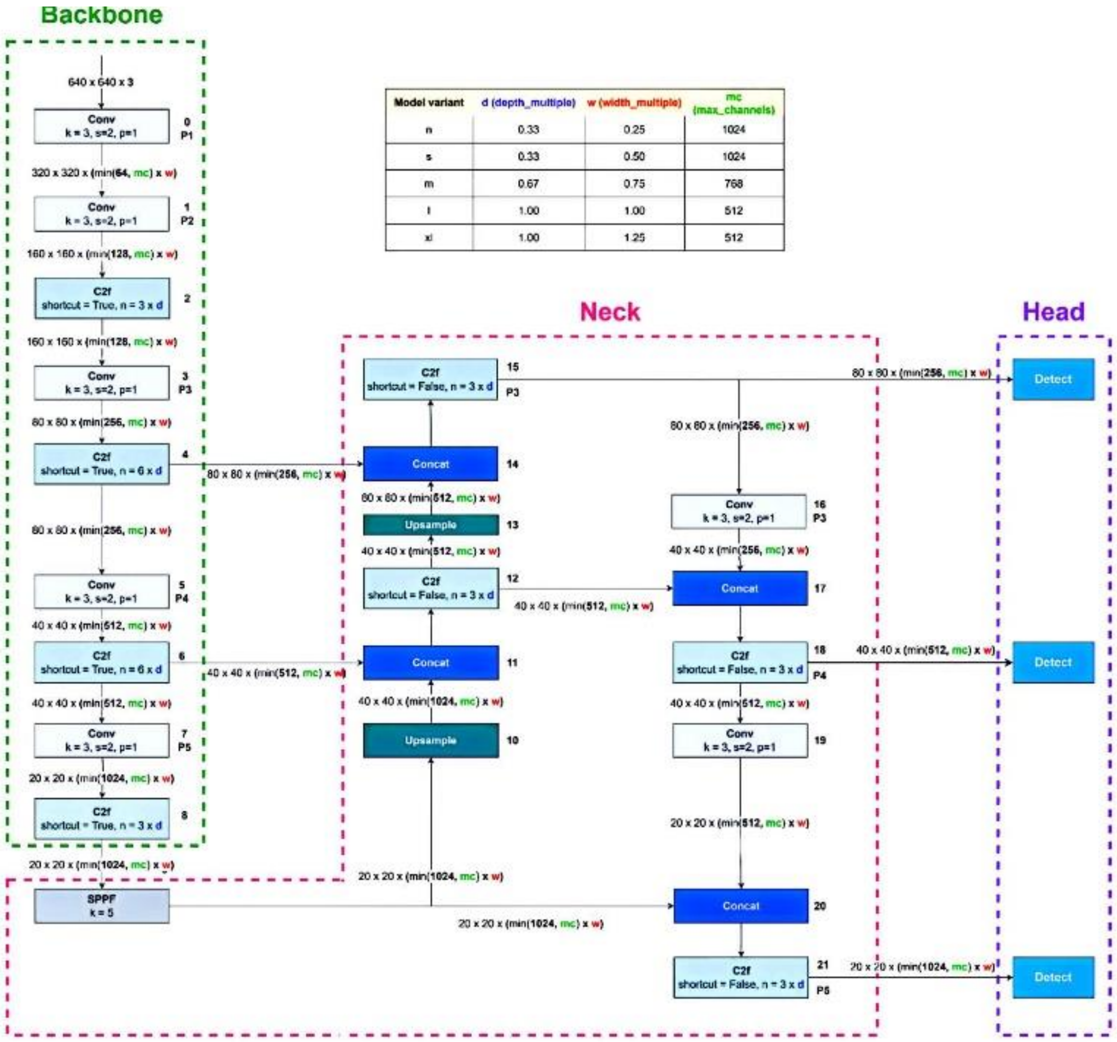
Convolutional Neural Network (CNN) is a deep learning architecture designed primarily to solve complex image processing problems [17]. In general, CNNs consist of several main layers, including convolution layers, non-linear activation functions (ReLU), pooling layers, and fully connected layers. The convolution layer extracts features from input data using kernels or filters, while the pooling layer reduces the spatial size of the data, speeding up computation and preventing overfitting to the training data [19, 20].

2.4.Transfer learning.

Transfer learning is a machine learning approach that leverages the knowledge of a previously trained model to solve problems on new datasets. Instead of training a model from scratch, this method adapts and fine-tunes existing model parameters to fit the characteristics of a new dataset. Transfer learning improves computational efficiency and reduces training time while maintaining accuracy and generalizability to new contexts. Knowledge gained from a source domain or dataset can thus be "transferred" to a target domain with similar characteristics [20].

2.5.YOLOv11.

You Only Look Once (YOLO) is a deep learning architecture designed for real-time object detection with high computational efficiency, making it suitable for dynamic video processing. The key feature of YOLO is its one-stage detection method, which allows it to detect and classify multiple objects simultaneously in a single pass [21]. The YOLOv11 algorithm employs a Dynamic Head mechanism to predict bounding box coordinates and class probabilities more adaptively. This approach improves detection accuracy and speeds up processing, especially for small objects or objects with varying positions. In YOLOv11, the input image is divided into an S × S grid. Each grid cell detects an object if the object's center falls within that cell and predicts several bounding boxes. The confidence score represents the model's certainty about an object's existence and accuracy, measured by the highest Intersection Over Union (IoU) value. After detection, results are processed using the Non-Maximum Suppression (NMS) technique to eliminate redundant predictions, retaining only the most accurate results. The YOLOv11 architecture is shown in Figure 1 [22].


Backbone
640 x 640 x 3
Conv k = 3, s=2, p=1
0 P1
320 x 320 x (min(64, mc) x w)
Conv k = 3, s=2, p=1
1 P2
160 x 160 x (min(128, mc) x w)
C2f shortcut = True, n = 3 x d
2
160 x 160 x (min(128, mc) x w)
Conv k = 3, s=2, p=1
3 P3
80 x 80 x (min(256, mc) x w)
C2f shortcut = True, n = 6 x d
4
80 x 80 x (min(256, mc) x w)
80 x 80 x (min(256, mc) x w)
Conv k = 3, s=2, p=1
5 P4
40 x 40 x (min(512, mc) x w)
C2f shortcut = True, n = 6 x d
6
40 x 40 x (min(512, mc) x w)
40 x 40 x (min(512, mc) x w)
Conv k = 3, s=2, p=1
7 P5
20 x 20 x (min(1024, mc) x w)
C2f shortcut = True, n = 3 x d
8
20 x 20 x (min(1024, mc) x w)
SPPF k = 5
Neck
C2f shortcut = False, n = 3 x d
15 P3
Concat
14
80 x 80 x (min(512, mc) x w)
Upsample
13
40 x 40 x (min(512, mc) x w)
C2f shortcut = False, n = 3 x d
12
40 x 40 x (min(512, mc) x w)
Concat
11
40 x 40 x (min(1024, mc) x w)
Upsample
10
20 x 20 x (min(1024, mc) x w)
20 x 20 x (min(1024, mc) x w)
80 x 80 x (min(256, mc) x w)
Conv k = 3, s=2, p=1
16 P3
40 x 40 x (min(256, mc) x w)
Concat
17
C2f shortcut = False, n = 3 x d
18 P4
40 x 40 x (min(512, mc) x w)
Conv k = 3, s=2, p=1
19
20 x 20 x (min(512, mc) x w)
Concat
20
C2f shortcut = False, n = 3 x d
21 P5
Head
80 x 80 x (min(256, mc) x w)
Detect
40 x 40 x (min(512, mc) x w)
Detect
20 x 20 x (min(1024, mc) x w)
Detect

Model variant	d (depth_multiple)	w (width_multiple)	mc (max_channels)
n	0.33	0.25	1024
s	0.33	0.50	1024
m	0.67	0.75	768
l	1.00	1.00	512
xl	1.00	1.25	512

Figure 1. YOLOv11 architecture.

YOLOv11 was introduced as the latest version, combining high speed with precise object detection accuracy due to improvements in its architectural design. This version employed ConvNeXtV3 as its main backbone, which captured clearer image details, and incorporated the Dynamic Head feature, allowing it to adapt to various object sizes and conditions. YOLOv11 increased the mean Average Precision (mAP) by 3–5% compared to YOLOv8 and YOLOv10, while processing data 20–25% faster [12].

2.6.Adam optimizer.

The Adam optimizer is an optimization method that plays a significant role in machine learning, particularly in the training of neural networks for deep learning applications. The term Adam stands for Adaptive Moment Estimation, which combines the advantages of two predecessor algorithms, namely RMSProp and Momentum, to achieve faster and more stable convergence. Its mechanism relies on calculating two types of estimates: the average and the uncentered variance of the gradient, which are then used to adjust the learning rate for each network parameter [23, 24].

2.7.Dataset collection.

The melon dataset used in this study was obtained from a greenhouse at the Shenzhen Experimental Base of the Chinese Academy of Agricultural Sciences, located on Pengfei Road, Dapeng New District, Shenzhen, Guangdong Province, China. Images were collected in October and November 2023. The field data collection process is shown in Figure 2 [11].

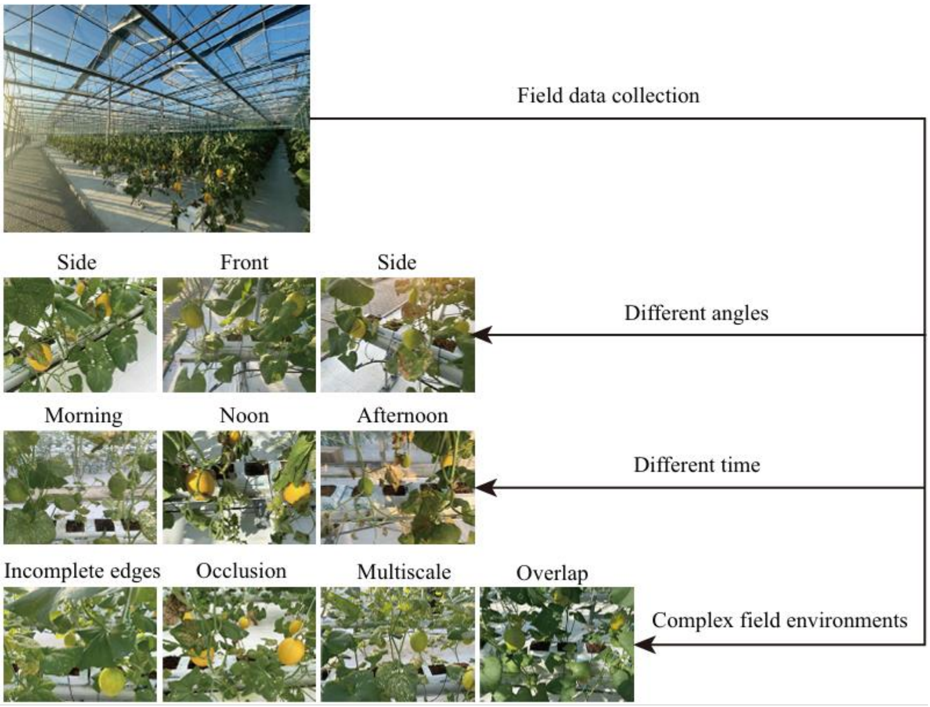


Figure 2. Field data collection process.

This dataset consisted of images taken from various angles to ensure the model could extract melon features comprehensively. Temporal variability in the images posed a challenge for the model to recognize target objects amidst shadows and bright light. Although the greenhouse environment was controlled, providing good growth conditions and adequate plant spacing, several challenges remained, such as target overlap, leaf shading, and differences in size and scale of melons at various growth stages. A total of 3,806 high-quality melon images with a resolution of 4032×3024 were obtained. This dataset has been used in a research paper entitled *Melon Ripeness Detection by an Improved Object Detection Algorithm for Resource-Constrained Environments* [11], which is accessible at https://github.com/XuebinJing/Melon-Ripeness-Detection/tree/main?tab=readme-ov-file. From the total of 3,806 images, 3,756 melon images in .jpg format were selected for this study and resized to 640×640 pixels. The dataset was split into 70% training, 20% validation, and 10% testing, a commonly adopted ratio in deep learning studies to ensure sufficient training data while maintaining reliable validation and unbiased testing. Figure 3 shows the ripeness levels of the golden melon.

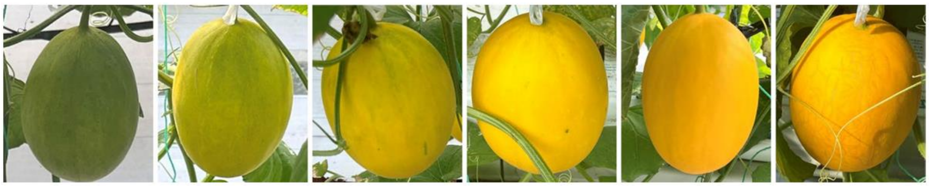


Figure 3. Level of ripeness of the golden melon.

2.8.Design.

Figure 4 shows the model design schematic, which began with the collection of a golden melon dataset. The images were then processed through a labeling stage to assign each image to the appropriate ripeness class. After labeling, the dataset was divided into three subsets: training, validation, and testing. During the training phase, the training dataset underwent image augmentation techniques, including random rotation, horizontal flipping, brightness adjustment, and scaling, to increase dataset diversity and improve model generalization. The augmented data were then used to train the YOLOv11 model. Meanwhile, the validation dataset was used to monitor the model’s performance throughout training and ensure that overfitting did not occur. The testing dataset also underwent image augmentation before being used in the final testing phase, which evaluated the trained model’s overall performance and its ability to detect melon ripeness accurately. Through this series of processes, the final output was a fully trained and validated YOLOv11 model ready for automated detection of golden melon ripeness.

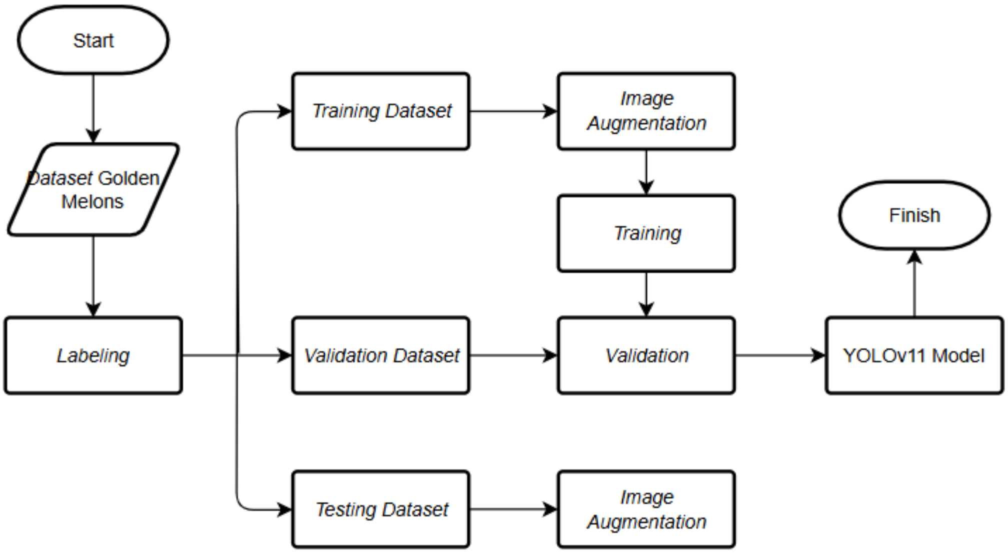


Figure 4. Model design schematic.

2.9.Implementation and testing.

In this stage, the previously designed system was implemented using Google Colab as the working platform. Google Colab was chosen due to its free access to GPU acceleration, ease of reproducibility, and widespread use in deep learning research. This setup enabled the system to process and classify the prepared dataset according to the planned learning model. The implemented system was then tested using the testing dataset to evaluate its performance. The testing involved various configurations, including 100 and 150 training cycles, a batch size of 16, and learning rates of 0.001 and 0.002. Table 1 presents the different testing configurations for the combined model.

Table 1. Model testing combination scenario.

Scenario	Epoch	Batch Size	Learning Rate
1	100	16	0.001
2	100	16	0.002
3	150	16	0.001
4	150	16	0.002

After completing the testing process, the results were analyzed to determine the model's effectiveness using commonly applied object detection metrics, namely precision, recall, F1-score, and mean Average Precision (mAP). These metrics were calculated based on the True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) values obtained from the fruit ripeness detection results. The mean Average Precision (mAP) was used to measure the average precision across all detected object classes, with its calculation formulated mathematically in Equations (1), (2), (3), and (4). Model performance was considered successful if it achieved a minimum mAP of over 80%, consistent with previous studies that reported YOLOv8 achieving an average mAP above 80% [10, 11].

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 - Score = 2 \text{ x} \frac{Precision \text{ x } Recall}{Precision + \ Recall} \tag{3}$$

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{4}$$

3. Results and Discussion

3.1.Comparison experiments trainning YOLOv11 model.

This study conducted four testing scenarios for the YOLOv11 model using the Adam optimizer. The results of each scenario, which utilized various combinations of training parameters, are presented below.

3.1.1 Results for scenario 1 and scenario 2.

In the first scenario, the model was trained using a relatively limited amount of training data with a moderate learning rate. Based on the performance graph in Figure 5(a), precision and recall increased rapidly at the beginning of the epochs, indicating that the model began to recognize objects effectively from the early stages of training. Both mAP50 and mAP50–95 increased gradually until reaching a stable state, demonstrating a fairly good convergence process. The training and validation loss curves showed a consistent downward trend, although slight fluctuations occurred due to the limited training data. Overall, scenario 1 showed that a learning rate of 0.001 provided effective learning, but the model's performance was still constrained by the relatively small amount of training data.

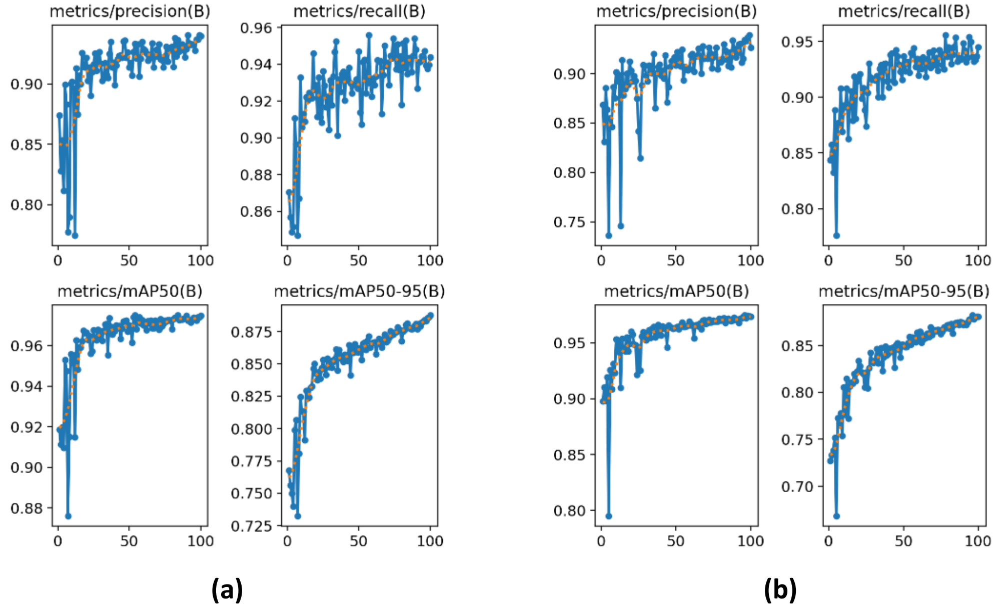


Figure 5. Performance curves of The YOLOv11 model in scenario 1 (a); Performance curves of the YOLOv11 model in scenario 2 (b).

The second scenario used the same amount of training data as scenario 1 but employed a higher learning rate. The graph in Figure 5(b) shows that precision and recall increased faster than in scenario 1, indicating a more aggressive learning process. Both mAP50 and mAP50–95 reached high values in fewer epochs. However, the graphs also displayed more pronounced fluctuations, particularly in the validation metrics, suggesting that a high learning rate on limited data can compromise training stability. This scenario demonstrates that a learning rate of 0.002 can accelerate convergence but may reduce stability and recall when the training dataset is still small.

3.1.2. Results for scenario 3 and scenario 4.

In the third scenario, the training dataset was increased to 150 images while maintaining a learning rate of 0.001. As shown in Figure 6(a), precision and recall increased consistently and were more stable compared to scenario 1. Both mAP50 and mAP50–95 reached high values with smoother curves, indicating an improvement in the model's generalization ability. The training and validation loss curves decreased steadily, suggesting that the additional training data helped the model learn more effectively. These results indicate that increasing the amount of training data significantly improved the stability and overall performance of the model, even without changing the learning rate.

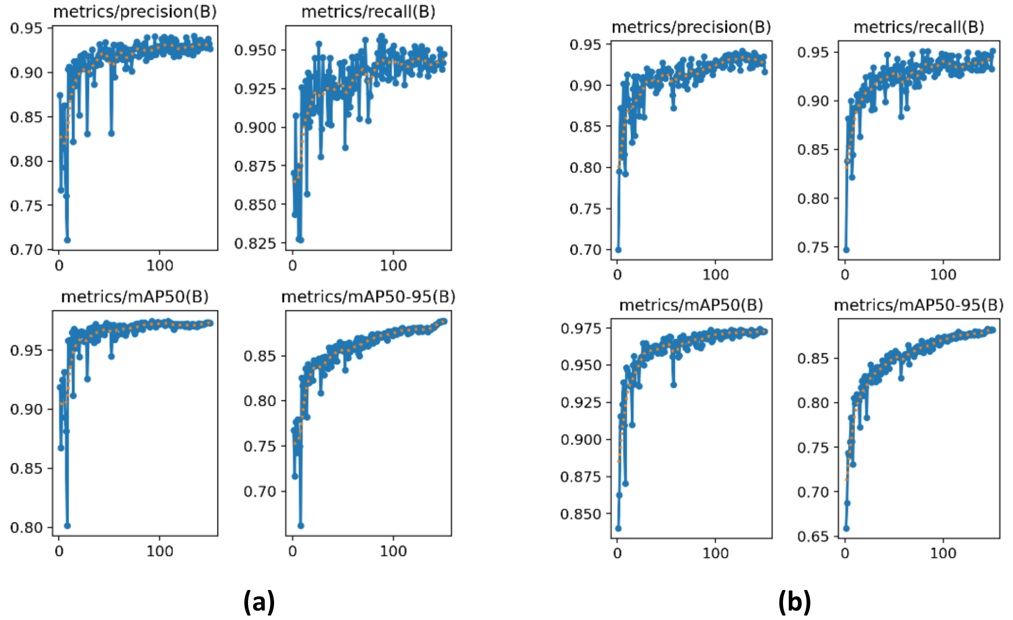


Figure 6. Performance curves of the YOLOv11 model in scenario 3 (a); Performance curves of the YOLOv11 model in scenario 4 (b).

The fourth scenario combined the largest amount of training data with the highest learning rate. As shown in Figure 6(b), precision, recall, mAP50, and mAP50–95 increased very rapidly and reached the highest values compared to all other scenarios. The performance graph appeared to be the most stable, despite the relatively high learning rate, due to the sufficient amount of training data. The loss curve decreased sharply at the beginning of the epoch and quickly reached convergence, indicating an efficient training process. This scenario was concluded to be the optimal configuration in the study, as it achieved the highest accuracy, fastest convergence, and best stability. Table 2 presents the test results for all four scenarios.

Table 2. Scenario testing results.

Scenario	Precision	Recall	mAP50	mAP50-95	F1-score
1	0.9513	0.8950	0.9748	0.8875	0.9222
2	0.9547	0.8878	0.9758	0.8833	0.9199
3	0.9058	0.9079	0.9731	0.8884	0.9297
4	0.9445	0.8877	0.9743	0.8826	0.9150

Although Scenario 4 produced slightly higher metric values and converged faster due to a higher learning rate, it was more sensitive to data variations and exhibited less stable training behavior. This increased the risk of overfitting, especially when the dataset contained complex visual variations. Scenario 3, on the other hand, demonstrated more stable and balanced performance, particularly in recall and mAP50–95. The higher recall indicated that the model detected true objects more effectively, including those affected by occlusion, leaf shading, and varying lighting conditions. Furthermore, the improved mAP50–95 suggested that Scenario 3 provided more accurate localization across different IoU thresholds, reflecting better generalization to diverse object sizes and positions.

Across all scenarios, a trade-off between precision and recall was observed. Scenarios with higher precision tended to miss more objects, whereas Scenario 3 maintained a better balance between these metrics. This balance was reflected in Scenario 3 achieving the highest

F1-score, indicating the most optimal compromise between precision and recall and confirming the overall effectiveness of the model in handling detection accuracy and consistency simultaneously. Therefore, Scenario 3 was selected as the optimal configuration because it offered stable training, strong generalization, and robustness to dataset variability, making it more suitable for real-world agricultural applications.

The YOLOv11-based model achieved a competitive mAP50 of 97.31% with higher consistency across scenarios, in contrast to other studies utilizing YOLOv8 for melon ripeness assessment, which reported mAP50 values ranging from 97.4% to 98.1% [10, 11]. The increased recall and mAP50–95 in Scenario 3 suggested stronger generalization under challenging environmental conditions, despite a slightly reduced precision. Differences in performance between experiments could be attributed to variations in dataset size, image resolution, and environmental factors.

Compared to other CNN-based architectures commonly used for object detection and classification, such as Faster R-CNN or SSD, YOLOv11 offered an end-to-end detection framework that balanced accuracy and computational efficiency. While two-stage detectors may achieve high precision, they generally required longer training and inference times, making them less suitable for real-time agricultural applications. In contrast, YOLOv11 was designed for faster inference with fewer computational resources, enabling efficient deployment in field conditions. The relatively stable training behavior and consistent performance observed in this study further supported the suitability of YOLOv11 for datasets with environmental variability. Additionally, the efficient inference speed and moderate GPU usage of YOLOv11 made it practical for real-time monitoring systems, where both detection accuracy and computational efficiency were critical.

3.2. Confusion matrix.

Figure 7 presents the confusion matrix of the best-performing model obtained from Scenario 3. The confusion matrix provides detailed insight into the classification behavior of the model, particularly in distinguishing between ripe and unripe fruit classes. Based on the matrix, the model correctly classified 977 unripe samples and 776 ripe samples, demonstrating strong capability in recognizing both maturity stages. However, some misclassification patterns were observed. A total of 24 unripe samples were misclassified as ripe, while 16 ripe samples were misclassified as unripe. These errors primarily occurred due to visual similarities in color, texture, and shape between ripe and unripe fruits, especially during transitional maturity stages where visual boundaries were less distinct. Such overlaps are common in fruit ripeness detection tasks and highlight the inherent difficulty of differentiating adjacent ripeness levels based solely on visual cues.

Additionally, several samples from both ripe and unripe classes were predicted as background, with 114 unripe and 34 ripe samples falling into this category. This suggests that some fruits were not detected confidently, likely due to partial occlusion, small object size, or low contrast with the background. Despite this, the number of background-related misclassifications remained relatively limited compared to the total correct detections, indicating that the model maintained strong object localization capability. The background class itself exhibited minimal confusion with fruit classes, as reflected by the low number of background samples predicted as ripe or unripe. This indicates that the model effectively distinguished fruit objects from non-fruit regions, reducing false positives and improving overall detection reliability.

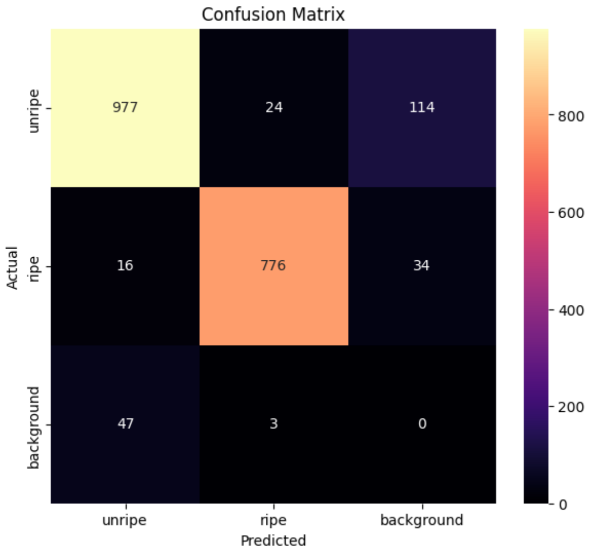
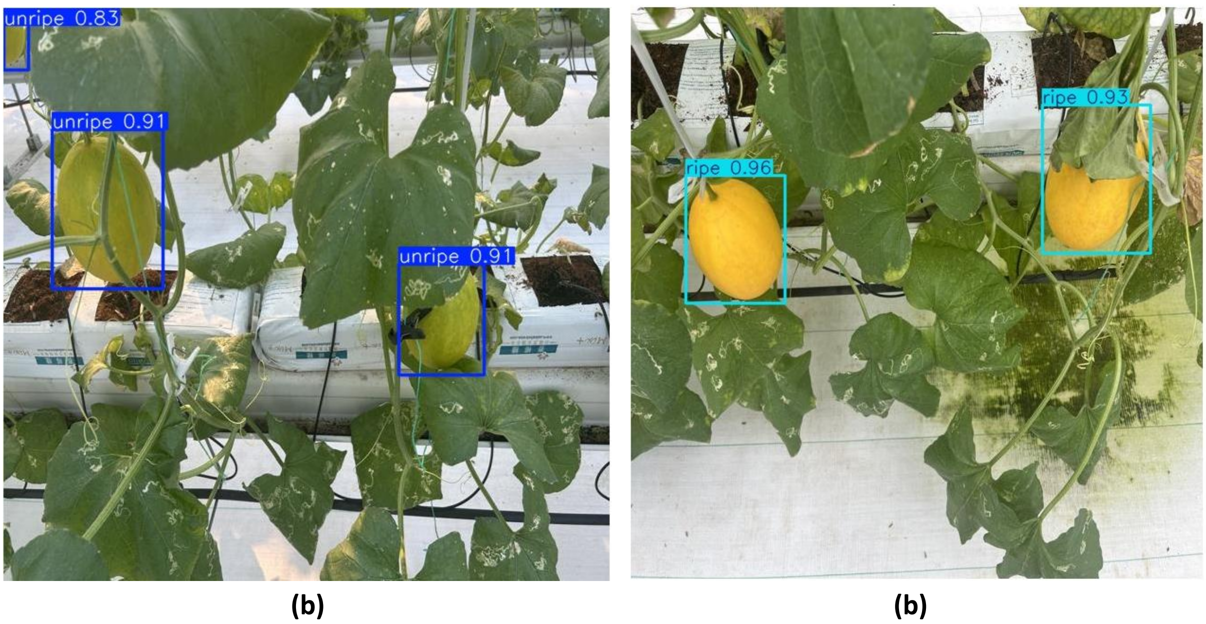


Figure 7. Best Model Confusion Matrix Results.

3.3. Detection.

Figure 8 further illustrated qualitative detection results under various real-world conditions. The detection visualizations demonstrated that the model was robust under different lighting conditions, including uneven illumination and moderate shadows. The model was also able to correctly detect fruits under partial occlusion, such as overlapping fruits or leaves covering parts of the object, although minor reductions in confidence were observed in heavily occluded cases. Additionally, the model performed well in complex background scenarios, where color similarity between fruit and background elements could have potentially caused misdetections.



(b) **(b)**

Figure 8. Detection of YOLOv11 in experiment unripe (**a**); Detection of YOLOv11 in experiment ripe (**b**).

Overall, the combined analysis of the confusion matrix and detection visualizations confirmed that Scenario 3 provided the most consistent and balanced performance. The model achieved high precision and recall with minimal misclassification between ripe and unripe classes while maintaining robustness across varying environmental conditions. This balance between quantitative accuracy and qualitative robustness supported the selection of Scenario 3 as the optimal configuration for fruit maturity detection in this study.

4. Conclusions

This study demonstrated the effectiveness of the YOLOv11 algorithm for non-destructive detection of golden melon ripeness under challenging visual conditions, including variations in lighting, occlusion, and fruit scale. The main contribution of this research was the evaluation of a newer YOLO architecture on a complex agricultural dataset, addressing the limited prior exploration of YOLOv11 for melon ripeness detection. Among the evaluated configurations, Scenario 3 achieved the most balanced and stable performance, attaining a precision of 90.58%, a recall of 90.79%, an mAP50 of 97.31%, an mAP50–95 of 88.84%, and the highest F1-score of 92.97%. These results indicated an optimal trade-off between precision and recall, as well as strong object localization and generalization capability across varying environmental conditions. Practically, the proposed YOLOv11-based model was well suited for real-world agricultural applications, such as automated fruit sorting, harvest monitoring, and smart farming systems, where reliable and real-time detection is essential. The high recall reduced the risk of missed detections, while the strong mAP performance confirmed robustness in handling variations in object size and spatial positioning. However, this study was limited by the use of a greenhouse-based dataset and binary ripeness classes, as well as the absence of a detailed computational efficiency analysis. Future work should focus on expanding the dataset to open-field environments, implementing multi-stage ripeness classification, and enabling real-time deployment on edge devices to further enhance robustness and practical applicability.

Acknowledgments

Not applicable.

Author Contribution

All authors contributed equally to the conception and design of the study, material preparation, data collection, and data analysis.

Data Availability

The data supporting the findings of this study are available from the corresponding author upon reasonable request.

Competing Interest

All authors declare no competing financial, personal, or professional interests that could have influenced the work reported in this paper.

References

[1] Saputra, R. A.; Puspitasari, D.; Baidawi, T. (2022). Deteksi Kematangan Buah Melon dengan Algoritma Support Vector Machine Berbasis Ekstraksi Fitur GLCM. *Jurnal Infortech*, *4*(2).

[2] Shahwar, D.; Khan, Z.; Park, Y. (2023). Molecular Marker-Assisted Mapping, Candidate Gene Identification, and Breeding in Melon (*Cucumis melo* L.): A Review. *International Journal of Molecular Sciences*, *24*(20). https://doi.org/10.3390/ijms242015490.

[3] Produksi Tanaman Buah-Buahan 2021. (accessed on 1 October 2025) Avaialable online: https://www.bps.go.id/id/statistics-table/2/NjIjMg==/produksi-tanaman-buah-buahan.html.

[4] Zakiah, R. A.; Wahjuni, S.; Suwarno, W. B. (2023). Pemilihan Algoritma Machine Learning untuk Perangkat dengan Komputasi Terbatas pada Deteksi Kematangan Buah Melon Berjala. *Jurnal Ilmu Komputer dan Agri-Informatika*, *10*(2), 189–199. https://doi.org/10.29244/jika.10.2.189-199.

[5] Abiyyu, I. F.; Tawakal, H. A. (2021). Pengembangan Aplikasi Pendeteksi Kematangan Buah Melon: Studi Kasus Aplikasi Melonku. *Jurnal Informasi Terpadu*, *7*(1), 27–32. https://doi.org/10.54914/jit.v7i1.331.

[6] Pamungkas, W. A.; Bintoro, N. (2021). Karakteristik Kematangan Buah Melon 'Premier' (*Cucumis melo* L.) Berdasarkan Sifat Akustik. *Jurnal Agrointek*, *15*(3), 715–727. https://doi.org/10.21107/agrointek.v15i3.9621.

[7] Budiarti, N. A. E.; Wahjuni, S.; Suwarno, W. B.; Wulandari. (2021). Research on Melon Fruit Selection Based on Rank with YOLOv4 Algorithm. *Journal of Physics: Conference Series*, *2123*(1). https://doi.org/10.1088/1742-6596/2123/1/012036.

[8] Suradiradja, K. H. (2022). Algoritme Machine Learning Multi-Layer Perceptron dan Recurrent Neural Network untuk Prediksi Harga Cabai Merah Besar di Kota Tangerang. *Faktum Exacta*, *14*(4), 194. https://doi.org/10.30998/faktorexacta.v14i4.10376.

[9] Amrozi, Y.; Yuliati, D.; Susilo, A.; Novianto, N.; Ramadhan, R. (2022). Klasifikasi Jenis Buah Pisang Berdasarkan Citra Warna dengan Metode SVM. *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, *11*(3), 394–399. https://doi.org/10.32736/sisfokom.v11i3.1502.

[10] Umar, U.; Sardjono, T. A.; Kusuma, H.; Yani, M.; Widyantara, H. (2025). Optimizing YOLOv8 for Enhanced Melon Maturity Detection with Attention Mechanisms: A Case Study from Puspalebo Orchard. *International Journal of Informatics and Visualization*, *9*(3), 947–956. https://doi.org/10.62527/joiv.9.3.2942.

[11] Jing, X.; Wang, Y.; Li, D.; Pan, W. (2024). Melon ripeness detection by an improved object detection algorithm for resource constrained environments. *Plant Methods*, *20*(1). https://doi.org/10.1186/s13007-024-01259-3.

[12] Khanam, R.; Hussain, M. (2024). YOLOv11: An Overview of the Key Architectural Enhancements. *Arxiv*, 1–9. http://arxiv.org/abs/2410.17725.

[13] Lipiński, S.; Sadkowski, S.; Chwietczuk, P. (2025). Application of AI in Date Fruit Detection—Performance Analysis of YOLO and Faster R-CNN Models. *Computation*, *13*, 149. https://doi.org/10.3390/computation13060149.

[14] Alqoria, N. T.; Utaminingrum, F. (2021). Rancang Bangun Sistem Deteksi Kemanisan Buah Melon dengan Metode Gray Level Co-occurrence Matrix (GLCM) dan Support Vector Machine (SVM). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, *5*(6), 2472–2477.

[15] Wijaya, V. C.; Utaminingrum, F. (2022). Deteksi Tingkat Kemanisan Buah Melon melalui Ekstraksi Fitur Local Binary Pattern dengan Klasifikasi K-NN berbasis Raspberry Pi 4. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, *6*(1), 52–57. http://j-ptiik.ub.ac.id.

[16] Mozaffari, M.H. (2025). Deep Learning for Computer Vision Application. *Electronics*, *14*, 2874. https://doi.org/10.3390/electronics14142874.

[17] Krichen, M. (2023). Convolutional Neural Networks: A Survey. *Computers*, *12*, 151. https://doi.org/10.3390/computers12080151.

[18] Zewen, L.; Fan, L.; Wenjie, Y.; Shouheng, P.; Jun, Z. (2021). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(12), 6999–7019.

[19] Hadi, M. U.; Qureshi, R.; Ahmed, A.; Iftikhar, N. (2023). A lightweight CORONA-NET for COVID-19 detection in X-ray images. *Expert Systems with Applications*, *225*. https://doi.org/10.1016/j.eswa.2023.120023.

[20] Firmansah, R. A.; Santoso, H.; Anwar, A. (2023). Transfer Learning Implementation on Image Recognition of Indonesian Traditional Houses. *Jurnal Teknik Informatika*, *4*(6), 1469–1478. https://doi.org/10.52436/1.jutif.2023.4.6.767.

[21] Wening, M.; Hendry, J.; Santoso, W. A. (2025). Traffic Signs Detection System Based on You Only Look Once (YOLOv8) using Raspberry Pi. *Jurnal PEKOMMAS*, *10*(1), 61–70. https://doi.org/10.56873/jpkm.v9i1.5832.

[22] Hidayatullah, P.; Syakrani, N.; Sholahuddin, M. R.; Gelar, T.; Tubagus, R. (2025). YOLOv8 to YOLO11: A Comprehensive Architecture In-depth Comparative Review. *Arxiv*. http://arxiv.org/abs/2501.13400.

[23] Cheng, W.; Pu, R.; Wang, B. (2025). AMC: Adaptive Learning Rate Adjustment Based on Model Complexity. *Mathematics*, *13*, 650. https://doi.org/10.3390/math13040650.

[24] Mienye, I.D.; Swart, T.G. (2024). A Comprehensive Review of Deep Learning: Architectures, Recent Advances, and Applications. *Information*, *15*, 755. https://doi.org/10.3390/info15120755.