

YOLOv11 and Faster R-CNN for Palm Fruit Ripeness Detection: Consistent Class-Level Difficulty with Explainable AI Insights

Queenta Paradissa Ramadhani^{1*}, Favian Dewanta², Teddy Surya Gunawan³

^{1, 2}Department of Electrical Engineering, Telkom University, Indonesia

³Department of Electrical and Computer Engineering, International Islamic University Malaysia, Malaysia

ABSTRACT

Direct comparison between one-stage and two-stage object detectors under matched training conditions remains limited for palm oil Fresh Fruit Bunch (FFB) ripeness detection. This study compares five YOLOv11 variants and two Faster R-CNN variants using 16,842 annotated instances from Roboflow Universe across six ripeness categories under a controlled training configuration. YOLOv11m achieved the best overall performance with an mAP50-95 of 89.00%, outperforming Faster R-CNN R-101-FPN (85.91%) by 3.09 percentage points, while providing higher inference speed (64.42 versus 23.30 FPS) and a substantially smaller model size (38.65 versus 459.93 MB). Repeated evaluation across three random seeds showed stable performance, with mean mAP50-95 values of $89.01 \pm 0.19\%$ for YOLOv11m and $86.12 \pm 0.38\%$ for R-101-FPN. Both representative models exhibited the same class-level difficulty pattern, with Unripe achieving the highest AP and Abnormal the lowest. XGrad-CAM analysis further showed more dispersed activation patterns for the more difficult categories. These findings indicate that YOLOv11m provides the most favorable accuracy-speed-model-size trade-off among the evaluated models, while improving difficult categories may also require targeted data collection and annotation refinement.

Keywords: YOLOv11, Faster R-CNN, Deep Learning, Object Detection, Explainable AI, Precision Agriculture

Open Access

Type: Research Paper

Article:

Received: April 15, 2026

Revised: July 16, 2026

Accepted: September 13, 2026

Issued: October 30, 2026

© Ramadhani et al. (2026).



This is an open-access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license

*Correspondence Address:

queentaparadissa@student.telkomuniversity.ac.id

1. INTRODUCTION

Palm oil is one of the largest agricultural businesses in Southeast Asia, where Malaysia and Indonesia alone produce more than 85% of all palm oil produced worldwide [1]. Economic sustainability of the business is directly correlated with the Oil Extraction Rate (OER), a measure of the percent of the crude palm oil extraction from the Fresh Fruit Bunches (FFB) at harvesting time. Palm Oil Extraction Rate directly correlates with FFB ripening level at harvesting time [2]. Today, there is a tendency to estimate the ripening level using the method of visual evaluation. But visual evaluation has many limitations that can lead to economic loss due to improper harvesting time, namely, premature harvest resulting in low oil extraction rate, or delayed harvesting, which leads to high free fatty acid (FFA) level [3]. These challenges have motivated extensive research into automated computer vision and deep learning approaches for objective and scalable ripeness classification [4].

This paper compares five YOLOv11 variants (n, s, m, l, x) and two Faster R-CNN variants (R-50-FPN, R-101-FPN) for FFB ripeness detection across six ripeness categories, using a dataset of 16,842 annotated instances. The central finding is that per-class detection difficulty follows the same pattern regardless of detector paradigm. Unripe is consistently the easiest category to detect and Abnormal the hardest, for both one-stage and two-stage architectures, suggesting that the observed difficulty may be more closely related to the visual characteristics of the fruit than to the detector architecture examined. This pattern is further supported by XGrad-CAM visualization applied to both the best-performing YOLOv11 and Faster R-CNN models, which shows that model attention becomes progressively more dispersed as class difficulty increases. The primary contributions are: a systematic comparison of YOLOv11 and Faster R-CNN for FFB ripeness detection; evidence, supported by both quantitative and explainable-AI analysis, that detection difficulty remains consistent across the examined one-stage and two-stage architectures for the categories and models examined and a practical implication that improving detection of difficult categories such as Abnormal is likely to benefit more from targeted data

collection than from switching detector architecture.

1.1 Fresh Fruit Bunch Ripeness Assessment

FFB ripeness classification follows standardized grading criteria defining six distinct categories (Unripe, Underripe, Ripe, Overripe, Empty Bunch, Abnormal). Unripe characterized by dark and compact fruitlets with near-zero loose fruit, Underripe with partially orange fruitlets, Ripe exhibiting fully orange fruitlets with optimal loose fruit count, Overripe showing excessive loose fruitlets with elevated FFA content, Empty Bunch comprising bare rachis without fruitlets, and Abnormal encompassing deformed or disease-affected bunches [5]. These ripeness categories are associated with differences in oil yield and quality, making accurate classification economically critical. Misclassification of FFB ripeness can reduce extraction efficiency and affect the overall quality of a processing batch, which motivates the need for automated and objective detection systems [6].

The use of deep learning algorithms has proved to be effective in the classification of ripeness of FFBs. Gunawan et al. conducted a study on the use of five YOLOv8 models in detecting the ripeness of FFBs classified into six categories and found the peak mAP₅₀₋₉₅ of 92.7% using YOLOv8m, thus setting up a strong baseline for automated FFB detection [7]. YOLOv11 was used in recent studies for ripeness detection and grading of strawberries and fruits in different classes respectively [8], [9].

However, each of these studies evaluates a single detection paradigm in isolation. Gunawan et al. benchmark only YOLOv8 variants without a two-stage counterpart [7]. Ma et al. and Kamat et al. similarly restrict their evaluation to one-stage or hybrid one-stage architectures [8], [9]. This pattern is also reflected in the broader literature. Lai et al., in their systematic review of FFB ripeness detection methods, noted that most studies evaluate a single architecture family in isolation [3]. Josdaan et al. similarly report strong YOLOv8 results without a two-stage comparison [10]. Goh et al.'s recent review of FFB classification methods likewise identifies cross-architecture benchmarking as an underexplored direction [11]. Without a two-stage reference point evaluated under

comparable training conditions, it remains unclear whether performance differences between studies reflect architectural characteristics or training setup. This gap motivates a matched-configuration comparison between YOLOv11 and Faster R-CNN, which the present study addresses.

1.2 Object Detection

The architectures for convolutional object detection are typically split into two broad categories: one-stage and two-stage detectors. The two-stage detectors include the Faster R-CNN models where a pipeline of two stages is applied; in particular, an object detection process starts from generating a set of object proposals by a Region Proposal Network (RPN), and then these proposals are classified and refined by a Region of Interest (ROI) head [12], [13]. As a result, two-stage detectors can provide strong localization accuracy but require greater computational complexity and inference time. By introducing Feature Pyramid Networks (FPN) it is possible to further boost multi-scale detection by feature aggregation on different backbone levels [14]. The YOLO family of one-stage detectors performs both tasks (object localization and classification) in a single pass [15]. Further advancements in the field of YOLO family models have made these models increasingly accurate and efficient [16], [17].

Recent studies have compared YOLO and Faster R-CNN detectors under different conditions. Gholinavaz et al. compared YOLO variants and Faster R-CNN under adverse weather and noise conditions, reporting that Faster R-CNN demonstrated strong robustness and precise localization in several degraded scenarios, whereas the evaluated YOLO models provided favorable computational efficiency for real-time detection [18]. However, this evaluation was conducted on general-purpose benchmarks under simulated weather noise, and it remains unclear whether the same localization advantage holds for fine-grained, domain-specific categories such as ripeness stages, where inter-class visual differences are far subtler than in standard object categories. A comprehensive review of YOLO evolution from YOLOv1 to YOLOv8 highlights that recent innovations including improved feature extraction mechanisms and attention modules have substantially advanced the accuracy-efficiency frontier, though such reviews largely

synthesize benchmark results from general datasets (e.g., COCO, PASCAL VOC) rather than agricultural imagery, limiting their direct applicability to plantation-specific deployment conditions [17]. YOLOv11 uses C3k2 modules for feature representation and C2PSA for attention, which may help when detecting occluded or visually complex objects in agricultural settings [19]. Systematic reviews of deep learning for crop grading confirm that both Faster R-CNN and YOLO have been applied to agricultural detection, but direct comparative evaluations on the same dataset remain limited [20]. Research on fruit target recognition for robotic harvesting has consistently highlighted the trade-off between detection accuracy and real-time performance as the central design consideration for field deployment, motivating the need for systematic cross-paradigm comparisons [21].

Beyond conventional one-stage and two-stage convolutional detectors, transformer-based architectures such as DETR and RT-DETR have emerged as alternative object detection approaches, formulating detection as an end-to-end set prediction problem and reducing reliance on hand-crafted components such as anchor generation and non-maximum suppression [22], [23]. The present study deliberately focuses on the comparison between one-stage YOLOv11 and two-stage Faster R-CNN under controlled training conditions. Faster R-CNN was selected as the two-stage baseline because of its extensive use and well-documented behavior in the object detection literature, providing a stable reference for comparison with YOLOv11. Extending the benchmark to transformer-based detectors such as RT-DETR is therefore reserved for future work.

In response to this identified gap, the study compares YOLOv11 and Faster R-CNN for six-class FFB ripeness detection under controlled training conditions. The analysis goes beyond overall accuracy by examining class-level performance, the types of errors that occur, consistency across different runs, and XGrad-CAM visualizations. The study examines whether the difficulty of detecting certain classes remains consistent across different one-stage and two-stage architectures. This helps determine whether future improvements should mainly focus on changing the detector's design or on improving the data quality and representation of difficult FFB

categories. This manuscript presents an original development of the authors' research, based on the authors' own experimental design, analysis, and interpretation.

2. METHODS

This study follows a systematic process that includes four primary phases: preparing the dataset, training and evaluating the model, analyzing its performance, and conducting a comparative assessment. The dataset preparation stage involves sourcing, splitting, and augmentation. The model training stage trains both YOLOv11 and Faster R-CNN variants using comparable optimizer, batch size, and training exposure. The performance analysis stage evaluates all models using standardized metrics. Finally, the comparative assessment stage synthesizes findings to derive deployment recommendations.

2.1 Dataset

The dataset used in this study is derived from a publicly available FFB ripeness detection dataset on Roboflow Universe, originally comprising annotated images across six ripeness categories. Annotations were inherited directly from the source dataset and were not independently re-verified by the authors; inter-annotator agreement could therefore not be assessed, which is a relevant consideration given the morphological ambiguity of categories such as Abnormal.

The original dataset exhibited significant class imbalance, particularly in the Empty Bunch category. Before offline augmentation, the Empty Bunch class contained only 266 images, with 215 images in the training subset, making it substantially under-represented compared with the other classes. To address this imbalance, offline augmentation was applied to the training subset of the Empty Bunch class using the Albumentations library. Table 1 summarizes the augmentation operations and their parameters.

Table 1. Offline Augmentation

Technique	Parameter	Prob.
Horizontal Flip	Flips the image horizontally	0.5
Vertical Flip	Flips the image vertically	0.3
Rotate 90°	Randomly rotates the image by 90°	0.5
Brightness, Contrast	Brightness $\pm 30\%$, Contrast $\pm 30\%$	0.7
Hue, Saturation, Value	Hue ± 20 , Saturation ± 30 , Brightness ± 20	0.5

Technique	Parameter	Prob.
Gaussian Blur	Blur with kernel size 3×3 to 7×7 pixels	0.3
Gaussian Noise	Random noise on pixel values	0.3
Random Shadow	Adds random shadow to the image	0.3
CLAHE	Contrast Limited Adaptive Histogram Equalization	0.3

Augmentation was applied exclusively to the training subset; the validation and test subsets consist solely of original images from Roboflow. Restricting augmentation to the Empty Bunch class directly addresses its pronounced under-representation, but means that other classes did not receive the same additional visual variability. This strategy may alter the effective training distribution of the Empty Bunch class, and its potential influence on class-specific performance is therefore considered when interpreting the results.

The original dataset was partitioned into training, validation, and test subsets before offline augmentation. Offline augmentation was subsequently applied only to Empty Bunch samples in the training subset, ensuring that augmented versions of training images were not introduced into the validation or test subsets. After the preprocessing and augmentation steps, the final dataset included 4,885 images, each containing a total of 16,842 annotated instances. The final distribution included 3,905 training images containing 13,396 instances, 495 validation images with 1,734 instances, and 485 test images featuring 1,712 instances. The preprocessed and augmented dataset was subsequently published on Roboflow Universe to ensure reproducibility [24]. Table 2 presents the final class distribution across the training, validation, and test subsets.

Table 2. Dataset Class Distribution

Class	Instances	(%)	Train/Val/Test
Abnormal	3,152	18.7	2,568/273/311
Empty Bunch	3,280	19.5	2,531/378/371
Overripe	2,911	17.3	2,291/329/291
Ripe	2,711	16.1	2,223/225/263
Underripe	2,646	15.7	2,086/297/263
Unripe	2,142	12.7	1,697/232/213
Total	16,842	100.0	13,396/1,734/1,712

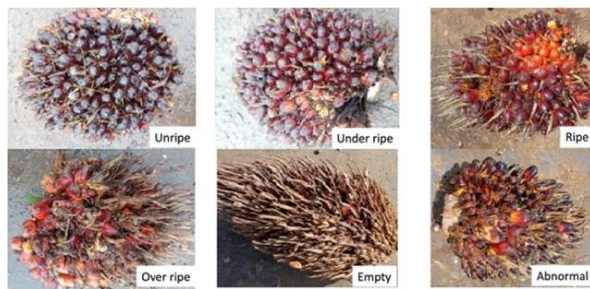


Figure 1. Palm Fruit Ripeness Category

2.2 YOLOv11 Model Architecture

YOLOv11 follows the standard YOLO single-stage detection paradigm comprising three primary components, backbone, neck, and detection head. The backbone extracts multi-scale feature representations using C3k2 modules, which replace the C2f blocks used in YOLOv8, and incorporates a C2PSA (Cross-Stage Partial with Spatial Attention) module after the SPPF block to enhance feature representation. The neck performs multi-scale feature aggregation through an FPN/PAN structure using C3k2 blocks. The decoupled detection head produces class predictions and bounding-box regression outputs using Distribution Focal Loss (DFL) [25].

Five variants spanning a wide complexity spectrum were evaluated, YOLOv11n (nano, 5.23 MB), YOLOv11s (small, 18.30 MB), YOLOv11m (medium, 38.65 MB), YOLOv11l (large, 48.84 MB), and YOLOv11x (extra-large, 109.11 MB). All variants were initialized with COCO-pretrained weights and fine-tuned for 100 epochs using the SGD optimizer (Ultralytics v8.4.142). Training hyperparameters: initial learning rate (lr0) 0.01 with cosine learning rate decay (lrf=0.01), momentum 0.937, weight decay 0.0005, input resolution 640×640 pixels, batch size 8, and a 3-epoch warmup.

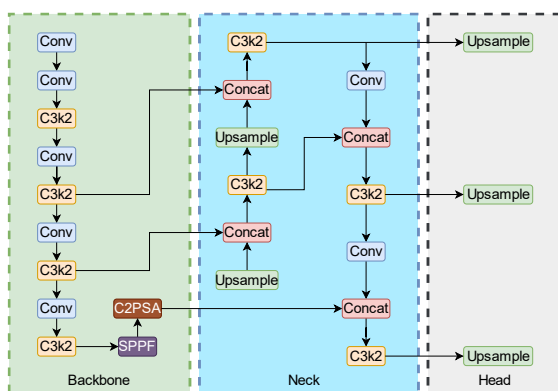


Figure 2. YOLOv11 architecture

2.3 Faster R-CNN Model Architecture

Faster R-CNN is a two-stage object detector that introduces a Region Proposal Network (RPN) to generate candidate object regions, sharing convolutional features with the detection network to enable efficient region proposals. In the first stage, the RPN generates a set of rectangular object proposals with associated objectness scores [13]. In the second stage, each proposal is classified and its bounding box is refined by a dedicated Region of Interest (ROI) head using features pooled via ROI Align [26]. The Feature Pyramid Network (FPN) integration enables multi-scale detection by aggregating feature maps from multiple backbone stages, substantially improving detection performance on objects of varying sizes [27].

Two backbone variants were evaluated: R-50-FPN (41.32M parameters, 314.93 MB) and R-101-FPN (60.26M parameters, 459.93 MB), implemented using Detectron2 v0.6 and initialized with COCO-pretrained weights. Both models were trained using SGD with a base learning rate of 0.01, momentum of 0.937, weight decay of 0.0005, a batch size of 8, an ROI batch size of 128 per image, and a warmup period of 1,954 iterations. Training was conducted for 48,812 iterations. The learning rate was reduced by a factor of 0.1 at iterations 34,168 and 43,931.

2.4 Experimental Setup

Both YOLOv11 and Faster R-CNN experiments were conducted on the same NVIDIA A100-SXM4-40GB GPU using Python 3.13, PyTorch 2.11.0+cu128, and CUDA 12.8. The input resolution was standardized to 640×640 pixels across all models. Table 3 summarizes the training configurations for both architectures.

Table 3. Training Configuration

Parameter	YOLOv11	Faster R-CNN
Framework	Ultralytics (v8.4.142)	Detectron2 (v0.6)
Input Resolution	640×640	640×640
Pretrained Weights	COCO	COCO
Optimizer	SGD	SGD
Batch Size	8	8
Initial Learning Rate	0.01	0.01
Momentum	0.937	0.937
Weight Decay	0.0005	0.0005
Random Seed	42	42
Warmup	3 epochs	1,954 iterations
Total Training Exposure	100 epochs (~390,500 images seen)	48,812 iterations (~390,500 images seen)

The optimizer, initial learning rate, momentum, weight decay, batch size, input resolution, and total training exposure were matched across the two architectures to reduce major training-related confounding factors. Although the training schedules are expressed differently in epochs for YOLOv11 and iterations for Faster R-CNN, both configurations correspond to approximately 390,500 image exposures.

No extensive hyperparameter search was performed for either architecture. The objective was not to identify the individually optimal configuration for each detector, but to compare the architectures under a controlled and reproducible training protocol. Therefore, the reported results should be interpreted as comparative performance under the specified configuration rather than as the absolute optimal performance of each architecture.

A fixed random seed of 42 was used for the full model benchmark to ensure reproducibility and consistent comparison across all variants. To assess run-to-run stability, the best-performing representative model from each architecture was additionally trained using seeds 123 and 2026. The results from seeds 42, 123, and 2026 were then summarized using the mean and standard deviation.

Training time ranged from 0.81 hours (YOLOv11n) to 1.78 hours (YOLOv11x) over 100 epochs, and from 0.19 hours (Faster R-CNN R-50-FPN) to 0.25 hours (R-101-FPN) over 48,812 iterations. Peak GPU memory usage during YOLOv11 training ranged from 2.35 GB (YOLOv11n) to 16.0 GB (YOLOv11x); corresponding figures for Faster R-CNN were not captured in the training logs, as Detectron2 does not report this metric by default.

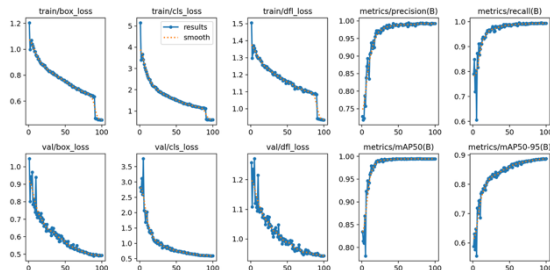


Figure 3. Training and validation curves for YOLOv11m

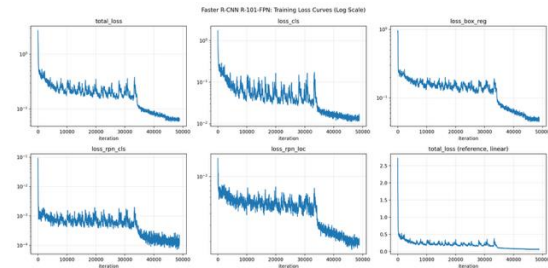


Figure 4. Training loss curves for R-101-FPN

2.5 Evaluation Metrics

Detection performance was evaluated using the following metrics. Precision, Recall, and F1-score were used to evaluate detection performance and the balance between false-positive and false-negative predictions, defined as:

$$Precision = \frac{TP}{(TP + FP)} \quad (1)$$

$$Recall = \frac{TP}{(TP + FN)} \quad (2)$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (3)$$

where TP, FP, and FN denote true positives, false positives, and false negatives respectively. mAP50 measures mean Average Precision at an Intersection over Union (IoU) threshold of 0.50, while mAP50-95 averages AP across ten IoU thresholds from 0.50 to 0.95 at intervals of 0.05. The mAP50-95 metric provides a more stringent measure of localization quality and is used as the primary ranking criterion in this study [28], [29].

During test-set evaluation, confidence thresholds were applied to filter low-confidence detections before fixed-threshold performance and qualitative analyses. The threshold settings used for YOLOv11 and Faster R-CNN followed the respective evaluation configurations and are reported explicitly to ensure reproducibility. The confidence threshold affects fixed-threshold outputs such as precision, recall, F1-score, confusion matrices, and qualitative detections, whereas mAP50 and mAP50-95 were obtained from the corresponding precision-recall evaluation procedures.

Computational efficiency was assessed using Frames Per Second (FPS), model size in megabytes (MB), and total parameter count in millions (M). FPS for Faster R-CNN was measured following a 10-iteration warm-up, then averaged over 100 inference passes on

individual test images (batch size 1) using Detectron2's DefaultPredictor; reported timing includes image loading (I/O) in addition to model inference. YOLOv11 FPS was measured using the Ultralytics validation pipeline under comparable settings (batch size 1). Neither TensorRT nor mixed-precision acceleration was used for either architecture.

The reported FPS values were obtained using the native evaluation pipelines of the respective frameworks. Because the exact timing scope of image loading, preprocessing, inference, and post-processing may differ between Ultralytics and Detectron2, small FPS differences should be interpreted with caution. A threshold of 24 FPS is used in this study as an operational indicator of real-time-capable inference under the present GPU-based benchmark [30]. Exceeding this threshold does not by itself establish suitability for field deployment, which would require validation on the target edge hardware under actual plantation conditions.

2.6 Explainability Analysis

To provide qualitative insight into class-level detection difficulty, XGrad-CAM was applied to the best-performing variant of each architecture. For each of the six ripeness categories, two test-set samples were selected, cropped to their bounding-box regions with 50% padding, and passed through the respective model to generate class activation maps. For YOLOv11, the selected target layer was the final block of the backbone; for Faster R-CNN, the target layer was the final stage of the ResNet backbone [31], [32], [33]. Activation maps were qualitatively compared across classes to assess whether attention dispersion correlates with per-class detection difficulty (Section 3).

3. RESULTS AND DISCUSSION

This section presents the detection performance of YOLOv11 and Faster R-CNN under the matched training configuration described in Section 2, followed by a class-level analysis examining whether detection difficulty is consistent across the two architectures.

3.1 Overall Detection Performance

Table 4 presents the overall detection performance of all seven models evaluated on the test set under the matched training

configuration described in Section 2.4. All YOLOv11 variants achieve mAP50 values above 99.2%, slightly exceeding both Faster R-CNN variants (98.39%–98.52%). The performance gap becomes more pronounced at mAP50-95, where YOLOv11m achieves the highest score of 89.00%, compared with 85.91% for Faster R-CNN R-101-FPN, corresponding to a 3.09 percentage-point advantage. YOLOv11 also achieves higher precision, recall, and F1-score than the Faster R-CNN variants, with the largest differences observed in recall and F1-score. In terms of inference speed, all YOLOv11 variants exceed the 24 FPS operational real-time threshold, ranging from 45.16 to 76.81 FPS. Faster R-CNN R-50-FPN achieves 29.18 FPS and remains above the threshold, whereas R-101-FPN reaches 23.30 FPS and falls marginally below it [28], [30]. Faster R-CNN also has substantially larger model sizes (314.93–459.93 MB) than YOLOv11 (5.23–109.11 MB), indicating a greater computational and storage burden for resource-constrained deployment.

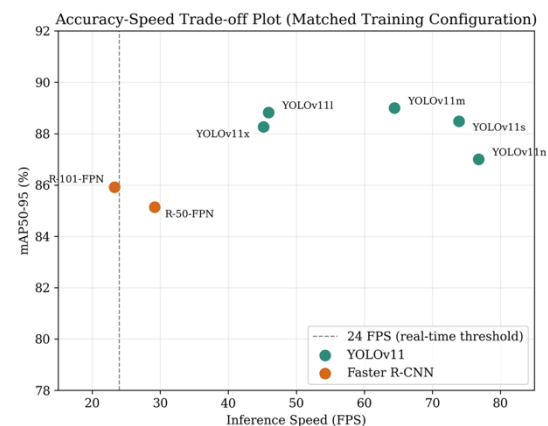


Figure 5. Accuracy-speed tradeoff

The consistently high mAP50 values (>98% for all seven models) warrant discussion, as near-saturated accuracy can sometimes indicate an overly homogeneous dataset rather than genuine model capability. Several factors support the plausibility of these results in the present case. First, the six ripeness categories exhibit distinguishable overall bunch characteristics, including color, fruitlet density, and surface texture, although adjacent ripeness categories may still share similar visual characteristics. This helps make mAP50, with its less stringent IoU measure, possible while still keeping open the possibility of variation in

terms of localization precision. Secondly, the growing difference seen under the mAP50-95 criteria, where the score ranges between 85.14% to 89.00%, not saturating towards its maximum value, shows that the localization of bounding boxes is a distinguishing factor among the models. Thirdly, offline data augmentation was done on the training set of the Empty Bunch class but not on the validation or test sets that had original images only.

Nonetheless, because the source dataset's image-acquisition conditions, including the number of distinct plantation sites or capture sessions, are not fully documented, the possibility that images across splits share background, lighting, or acquisition characteristics cannot be entirely excluded. Evaluation on an independently collected, more heterogeneous dataset is therefore identified as a direction for future work.

Table 5. Repeated Training Across Random Seeds
mAP 50-95 (%)

Model	YOLOv11m	R-101-FPN
Seed 42	89.00	85.91
Seed 123	89.21	85.88
Seed 2026	88.83	86.56
Mean $\pm$ SD	89.01 $\pm$ 0.19	86.12 $\pm$ 0.38

To assess the stability of the observed performance across different random initializations, YOLOv11m and Faster R-CNN R-101-FPN were additionally evaluated using three random seeds (42, 123, and 2026). As shown in Table 5, YOLOv11m achieved a mean mAP50-95 of 89.01 $\pm$ 0.19%, while Faster R-CNN R-101-FPN achieved 86.12 $\pm$ 0.38%. YOLOv11m therefore maintained an average advantage of 2.89 percentage points across the evaluated seeds. The relatively small standard deviations indicate that the performance of both representative models remained stable across repeated runs.

The consistency observed across repeated runs further supports the reliability of the comparative findings and indicates that the

overall ranking between the two representative models is not highly sensitive to random initialization. To complement these quantitative results, representative detection outputs from YOLOv11m and Faster R-CNN R-101-FPN are presented in Figures 6 and 7, respectively. These examples provide a qualitative view of the models' prediction behavior across different ripeness categories and illustrate their ability to localize multiple bunches under varied visual conditions.

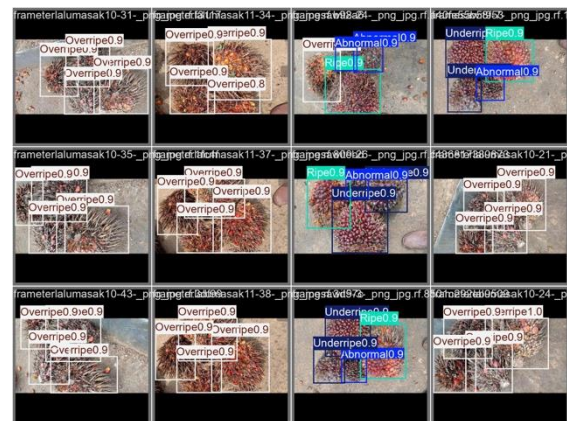


Figure 6. Detection prediction samples for YOLOv11m

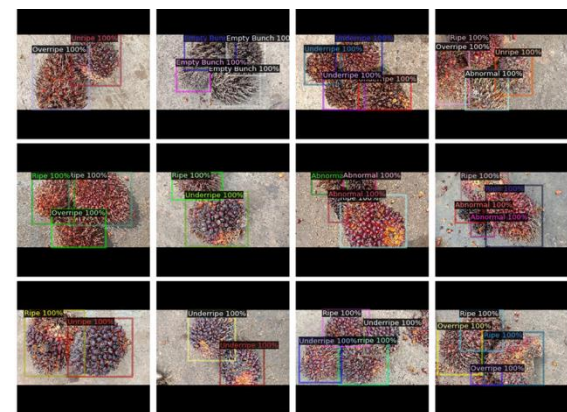


Figure 7. Detection prediction samples for R-101-FPN

3.2 Per-Class Detection Analysis and Error Characteristics

Table 4. Overall Detection Performance

Model	mAP50 (%)	mAP50-95 (%)	Prec. (%)	Recall (%)	F1 (%)	FPS	Size (MB)
YOLOv11n	99.29	87.00	98.95	99.05	99.00	76.81	5.23
YOLOv11s	99.37	88.48	99.27	99.51	99.39	73.91	18.30
YOLOv11m	99.32	89.00	99.01	99.36	99.18	64.42	38.65
YOLOv11l	99.31	88.82	99.07	99.53	99.30	45.92	48.84
YOLOv11x	99.33	88.27	99.29	99.50	99.39	45.16	109.11
R-50-FPN	98.39	85.14	97.36	93.06	95.16	29.18	314.93
R-101-FPN	98.52	85.91	97.64	95.24	96.43	23.30	459.93

Table 6 presents the per-class AP50-95 results for the best-performing variant of each architecture.

Table 6. Per-Class AP50-95 (%)

Model	YOLOv11m	R-101-FPN
Abnormal	84.64	81.07
Empty Bunch	87.61	82.51
Overripe	88.30	84.37
Ripe	90.55	88.09
Underripe	90.87	89.35
Unripe	92.04	90.08

The central finding of this analysis is that both representative models exhibit the same per-class difficulty ranking. Unripe is consistently the easiest category to detect, followed by Underripe, Ripe, Overripe, and Empty Bunch, with Abnormal being the most difficult category for both the one-stage YOLOv11m and the two-stage Faster R-CNN R-101-FPN. This ranking is consistent across both representative models. Despite substantial differences in their detection mechanisms, both architectures show a similar pattern of class-level difficulty, suggesting that the observed differences may be strongly influenced by the visual characteristics of the FFB categories themselves. Unripe fruitlets tend to exhibit relatively consistent visual characteristics, whereas the Abnormal category may contain more heterogeneous appearances, making it more challenging to characterize consistently.

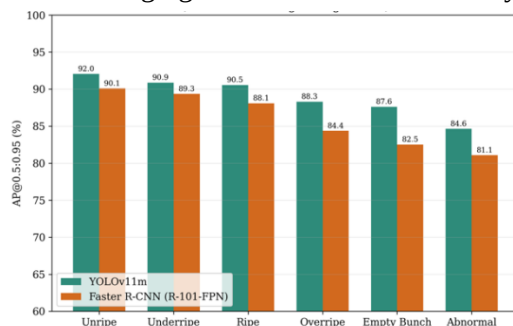


Figure 8. Per-Class AP Comparison

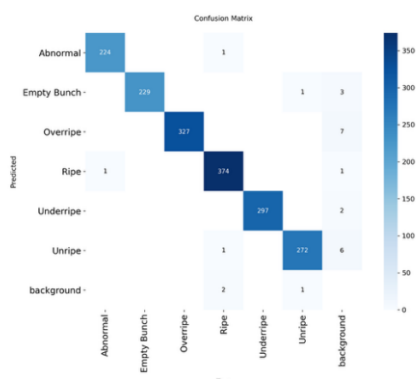


Figure 9. Confusion Matrix YOLOv11m

Inspection of the YOLOv11m confusion matrix shows that the residual errors are dominated by background-associated detections, particularly for the Overripe and Unripe classes (6–7 cases each), together with a small number of missed detections (2 cases for Ripe and 1 for Underripe). In contrast, direct confusion between ripeness categories occurred in only three isolated instances. For Faster R-CNN R-101-FPN, the observed errors were more frequently associated with confusion between visually adjacent ripeness stages, including Ripe misclassified as Unripe and Unripe misclassified as Overripe.

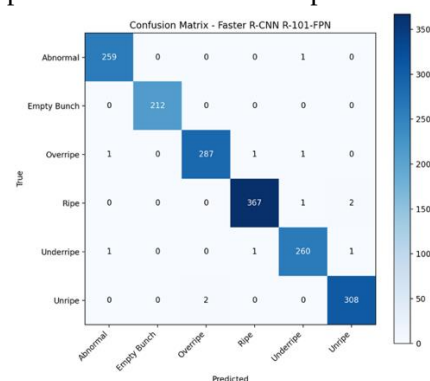


Figure 10. Confusion Matrix R-101-FPN

These results suggest that the two architectures may exhibit different error patterns despite sharing a similar overall per-class difficulty ranking. YOLOv11m shows more background-associated detection errors, whereas Faster R-CNN R-101-FPN exhibits more confusion among visually similar ripeness categories. These differences may be related to their distinct detection mechanisms; however, because no component-level ablation study was conducted, they cannot be causally attributed to specific architectural components such as C2PSA, anchor-free prediction, or the region-proposal stage.

3.3 Explainable AI Analysis

To further examine the source of class-level difficulty, XGrad-CAM was applied to YOLOv11m and Faster R-CNN R-101-FPN, the best-performing variant of each architecture. Figure 11 and Figure 12 present representative activation maps for all six ripeness categories for YOLOv11m and Faster R-CNN R-101-FPN, respectively.

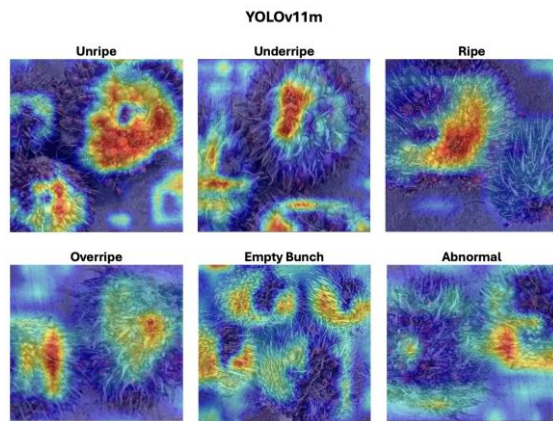


Figure 11. XGrad-CAM activation maps for YOLOv11m

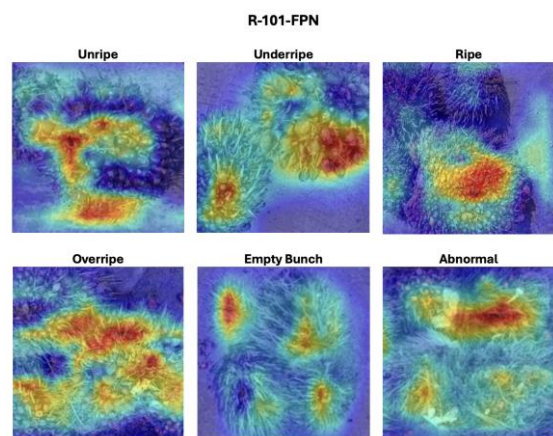


Figure 12. XGrad-CAM activation maps for R-101-FPN

For both models, activation maps for Unripe, Underripe, and Ripe, which achieved relatively high per-class AP, generally show more compact and localized activation around the fruit bunch. In contrast, Empty Bunch and Abnormal, the two lowest-performing categories, exhibit more dispersed activation patterns. For YOLOv11m, activation is distributed across multiple regions rather than being concentrated in a single area, while Faster R-CNN R-101-FPN shows broader and less sharply defined activation compared with the more localized patterns observed in the higher-performing classes. Overripe exhibits an intermediate pattern, consistent with its intermediate per-class AP. Because this analysis is based on a limited number of representative test samples rather than a quantitative assessment of the full test set, the activation maps should be interpreted as qualitative evidence only. Nevertheless, the observed patterns are broadly consistent with the per-class AP ranking in Table 6 and suggest that visually ambiguous categories may require less

concentrated feature attention across both architectures.

3.4 Cross-Architecture Comparison and Practical Implications

Table 7 summarizes the comparison between the optimal variant of each architecture.

Table 7. Comparison of YOLOv11m and R-101-FPN

Metric	YOLOv11m	R-101-FPN
mAP50	99.32	98.52
mAP50-95	89.00	85.91
F1-Score	99.18	96.43
FPS	64.42	23.30
Model Size	38.65	459.93

YOLOv11m is the more favorable choice when inference speed and model footprint are primary considerations, providing approximately 2.8× higher inference speed at about one-twelfth the model size, while also achieving a 3.09 percentage-point advantage in mAP50-95. The repeated-run analysis in Table 5 further shows that this performance advantage remains consistent across the evaluated random seeds.

Both representative models nevertheless exhibit the same overall per-class difficulty ranking, suggesting that the difficulty of categories such as Abnormal and Empty Bunch is not unique to a single detector architecture. Together with the qualitative XGrad-CAM observations in Section 3.3, this finding suggests that targeted data-centric improvements may be useful for these difficult categories. Such improvements may include collecting additional Abnormal samples covering a broader range of visual appearances and improving annotation consistency for morphologically ambiguous bunches. These observations do not exclude potential benefits from further architectural optimization, but indicate that dataset characteristics should also be considered when seeking additional performance gains [34].

The error analysis in Section 3.2 further shows that the two models exhibit different residual error patterns. YOLOv11m shows more background-associated detection errors, whereas Faster R-CNN R-101-FPN exhibits more confusion between visually similar ripeness categories. These patterns provide additional considerations for model selection and future improvement, although they cannot

be causally attributed to specific architectural components without dedicated ablation experiments.

3.5 Limitations

Several limitations should be considered when interpreting these results. First, repeated-run analysis was conducted only for the best-performing representative model of each architecture, YOLOv11m and Faster R-CNN R-101-FPN, rather than for all seven evaluated model variants. Although the relatively small standard deviations across three random seeds indicate stable performance for these representative models, variability across the remaining variants was not assessed. Second, the XGrad-CAM analysis in Section 3.3 was based on two representative test samples per class per model rather than on a systematic quantitative analysis of activation patterns across the full test set. The resulting visualizations should therefore be interpreted as qualitative and illustrative rather than conclusive evidence. Third, annotations were inherited from the source dataset without independent re-verification, and inter-annotator agreement could not be assessed. This limitation may be particularly relevant for morphologically ambiguous categories such as Abnormal. Finally, the source dataset does not fully document the diversity of image acquisition conditions, including plantation sites, capture sessions, illumination, occlusion, and viewpoint variation. Consequently, validation on an independently collected and more heterogeneous dataset is necessary to further assess the generalizability of the high detection performance observed in this study.

CONCLUSION

This study compared five YOLOv11 variants (n, s, m, l, and x) and two Faster R-CNN variants (R-50-FPN and R-101-FPN) for palm oil Fresh Fruit Bunch (FFB) ripeness detection across six ripeness categories. The models were evaluated on a dataset containing 16,842 annotated instances under a controlled training protocol with matched optimizer, initial learning rate, momentum, weight decay, batch size, input resolution, and approximately equivalent total training exposure. Among the evaluated models, YOLOv11m achieved the best overall performance, reaching an mAP50-

95 of 89.00%, compared with 85.91% for Faster R-CNN R-101-FPN, corresponding to a 3.09 percentage-point advantage. YOLOv11m also achieved substantially higher inference speed (64.42 versus 23.30 FPS) while requiring approximately one-twelfth the model size. Repeated-run analysis of the two representative models across three random seeds further supported the stability of this comparison. YOLOv11m achieved a mean mAP50-95 of $89.01 \pm 0.19\%$, whereas Faster R-CNN R-101-FPN achieved $86.12 \pm 0.38\%$. The relatively small variation across runs indicates that the observed performance pattern was consistent across the evaluated random initializations.

At the class level, both representative models exhibited the same overall difficulty ranking, with Unripe achieving the highest per-class AP and Abnormal the lowest. XGrad-CAM visualizations also showed more localized activation patterns for higher-performing categories and more dispersed activation for difficult categories such as Abnormal and Empty Bunch. Because the explainability analysis was qualitative and based on a limited number of samples, these observations should not be interpreted as evidence of a causal relationship. Nevertheless, the consistency between the per-class AP results and activation patterns suggests that visual ambiguity within the FFB categories contributes to the observed detection difficulty across both architectures.

The two architectures also exhibited different residual error patterns. YOLOv11m showed more background-associated detection errors, whereas Faster R-CNN R-101-FPN exhibited more confusion between visually similar ripeness categories. These findings suggest that future improvements should consider both model design and dataset characteristics. In particular, difficult categories such as Abnormal and Empty Bunch may benefit from additional targeted samples covering greater visual diversity and from improved annotation consistency.

Under the present NVIDIA A100 GPU-based benchmark, YOLOv11m provides the most favorable accuracy-speed-model-size trade-off among the evaluated models and demonstrates potential for real-time FFB ripeness detection. However, the reported GPU inference performance should not be interpreted as direct evidence of field-deployment

readiness. Future work should therefore evaluate the models on resource-constrained edge hardware and under more challenging field conditions, including variations in illumination, occlusion, motion blur, viewing angle, and plantation environment. Repeated-run evaluation should also be extended to the remaining model variants, while comparisons with additional contemporary detectors, including transformer-based architectures such as RT-DETR, would provide a broader assessment of architectural performance.

DECLARATIONS

Author Contributions: The first author conceived the study, designed the methodology, developed the software, curated the data, performed the formal analysis and investigation, prepared the visualizations, and wrote the original draft of the manuscript. The second author supervised the research, validated the results, and contributed to reviewing and editing the manuscript. The third author also supervised the research, contributed to the methodology, and reviewed and edited the manuscript. All authors have read and approved the final version of the manuscript.

Funding: This research was supported by the Ministry of Higher Education, Science, and Technology of the Republic of Indonesia (Kemendikristek) under grant numbers 1609/LL4/PG/2026 and 069/LIT07/PPM-LIT/2026. The funder had no role in the study design, data collection, analysis, interpretation, manuscript preparation, or decision to publish.

Conflict of Interest: The authors declare no conflict of interest.

Generative Artificial Intelligence Use Statement: During the preparation of this manuscript, the authors used Claude (Anthropic) solely for language editing and grammar checking. The tool was not used to generate the scientific content, research data, analysis, interpretation, or conclusions. The authors reviewed and edited all output and take full responsibility for the content of this publication.

Ethics Approval and Informed Consent: Not applicable. This study used images of palm oil

fresh fruit bunches and did not involve human participants, animals, personal or confidential data, or vulnerable groups; therefore, ethics approval and informed consent were not required.

Data Availability: The palm fruit ripeness dataset used in this study is publicly available on Roboflow Universe at <https://universe.roboflow.com/queenta-paradissa/dataset-6-4260-citra>. The source code and trained model weights are available from the corresponding author upon reasonable request.

REFERENCES

- [1] H. Varkkey, A. Tyson, and S. A. B. Choiruzzad, "Palm oil intensification and expansion in Indonesia and Malaysia: Environmental and socio-political factors influencing policy," *For. Policy Econ.*, vol. 92, pp. 148–159, Jul. 2018, doi: 10.1016/j.forpol.2018.05.002.
- [2] N. D. Hadi, B. L. Handoko, A. S. L. Lindawati, and H. Sarjono, "Palm Oil Industry Dynamics: Assessing P/B Ratios of Indonesian Palm Oil Companies through Palm Tree Profile, Average FFB Yield, and Palm Oil Extraction Rate," *Journal of Applied Data Sciences*, vol. 6, no. 1, pp. 236–246, Jan. 2025, doi: 10.47738/jads.v6i1.483.
- [3] J. W. Lai, H. R. Ramli, L. I. Ismail, and W. Z. Wan Hasan, "Oil Palm Fresh Fruit Bunch Ripeness Detection Methods: A Systematic Review," *Agriculture (Switzerland)*, vol. 13, no. 1, pp. 156–172, Jan. 2023, doi: 10.3390/agriculture13010156.
- [4] N. Paul, G. C. Sunil, D. Horvath, and X. Sun, "Deep learning for plant stress detection: A comprehensive review of technologies, challenges, and future directions," *Comput. Electron. Agric.*, vol. 229, no. 1, pp. 1–23, Feb. 2025, doi: 10.1016/j.compag.2024.109734.
- [5] N. Mohamad Hamidi, A. R. Saili, and S. Syahlan, "A Systematic Literature Review of Factors Affecting the Yield of Fresh Fruit Bunch and Oil Extraction Rate among Smallholders in Malaysia," *International Journal of Academic Research in Business and Social*

- Sciences, vol. 14, no. 9, pp. 214–229, Sep. 2024, doi: 10.6007/ijarbss/v14-i9/22274.
- [6] A. P. G. Kadir et al., “Oil Palm Economic Performance in Malaysia and R&D Progress in 2022,” *J. Oil Palm Res.*, vol. 35, no. 2, pp. 193–216, Jun. 2023, doi: 10.21894/jopr.2023.0028.
- [7] T. S. Gunawan, M. Kartiwi, H. Mansor, and N. M. Yusoff, “Palm Fruit Ripeness Detection and Classification Using Various YOLOv8 Models,” in *ICSIMA 2023 - 9th IEEE International Conference on Smart Instrumentation, Measurement and Applications*, Kuala Lumpur, Malaysia: Institute of Electrical and Electronics Engineers Inc., Nov. 2023, pp. 193–198. doi: 10.1109/ICSIMA59853.2023.10373435.
- [8] H. Ma et al., “YOLOv11-GSF: an optimized deep learning model for strawberry ripeness detection in agriculture,” *Front. Plant Sci.*, vol. 16, 2025, doi: 10.3389/fpls.2025.1584669.
- [9] P. Kamat, S. Gite, H. Chandekar, L. Dlima, and B. Pradhan, “Multi-class fruit ripeness detection using YOLO and SSD object detection models,” *Discover Applied Sciences*, vol. 7, no. 9, Sep. 2025, doi: 10.1007/s42452-025-07617-7.
- [10] J. Josdaan, V. C. Tamsil, J. Harefa, and K. Jingga, “Revolutionizing palm oil ripeness classification: Utilizing YOLOv8 for ultra-precise ripeness detection,” in *Procedia Computer Science*, Elsevier B.V., 2024, pp. 700–709. doi: 10.1016/j.procs.2024.10.296.
- [11] J. Y. Goh, Y. Md Yunos, and M. S. Mohamed Ali, “Fresh Fruit Bunch Ripeness Classification Methods: A Review,” *Food and Bioprocess Technology*, vol. 18, no. 1, pp. 183–206, Jun. 2024, doi: 10.1007/s11947-024-03483-0.
- [12] E. Edozie, A. N. Shuaibu, U. K. John, and B. O. Sadiq, “Comprehensive review of recent developments in visual object detection based on deep learning,” *Artif. Intell. Rev.*, vol. 58, no. 9, pp. 1–97, Sep. 2025, doi: 10.1007/s10462-025-11284-w.
- [13] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.
- [14] A. A. J. Pazhani and C. Vasanthanayaki, “Object detection in satellite images by faster R-CNN incorporated with enhanced ROI pooling (FrRNet-ERoI) framework,” *Earth Sci. Inform.*, vol. 15, no. 1, pp. 553–561, Mar. 2022, doi: 10.1007/s12145-021-00746-8.
- [15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.
- [16] M. L. Ali and Z. Zhang, “The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection,” *Computers*, vol. 13, no. 12, p. 336, Dec. 2024, doi: 10.3390/computers13120336.
- [17] J. Terven, D. M. Córdova-Esparza, and J. A. Romero-González, “A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS,” *Mach. Learn. Knowl. Extr.*, vol. 5, no. 4, pp. 1680–1716, Dec. 2023, doi: 10.3390/make5040083.
- [18] S. Gholinavaz, N. saeedi, and S. S. Gharehveran, “Robustness analysis of YOLO and Faster R-CNN for Object detection in Realistic Weather Scenarios with Noise Augmentation,” *Sci. Rep.*, vol. 15, no. 1, pp. 1–27, Dec. 2025, doi: 10.1038/s41598-025-28737-5.
- [19] G. Jocher and R. Munawar, “Ultralytics YOLO11,” Ultralytics Inc. Accessed: Jun. 27, 2026. [Online]. Available: <https://docs.ultralytics.com/models/yolo11#overview>
- [20] M. Dalal and P. Mittal, “A Systematic Review of Deep Learning-Based Object Detection in Agriculture: Methods, Challenges, and Future Directions,” *Computers, Materials and Continua*, vol. 84, no. 1, pp. 57–91, Jun. 2025, doi: 10.32604/cmc.2025.066056.
- [21] S. A. Ghazalli, H. Selamat, N. Khamis, and M. F. Haniff, “Short Review on Palm

- Oil Fresh Fruit Bunches Ripeness and Classification Technique,” *Journal of Advanced Research in Applied Mechanics*, vol. 106, no. 1, pp. 37–47, 2023, doi: 10.37934/aram.106.1.3747.
- [22] T. Shehzadi, K. A. Hashmi, M. Liwicki, D. Stricker, and M. Z. Afzal, “Object Detection with Transformers: A Review,” *Sensors*, vol. 25, no. 19, p. 6025, Oct. 2025, doi: 10.3390/s25196025.
- [23] G. Liu, D. Yang, J. Ye, H. Lu, Z. Wang, and Y. Zhao, “A real-time welding defect detection framework based on RT-DETR deep neural network,” *Advanced Engineering Informatics*, vol. 65, p. 103318, May 2025, doi: 10.1016/j.aei.2025.103318.
- [24] Q. Paradissa, “Palm Fruit Ripeness Dataset,” Roboflow Universe. Accessed: Mar. 28, 2026. [Online]. Available: <https://universe.roboflow.com/queenta-paradissa/dataset-6-4260-citra>
- [25] J. Bento, T. Paixão, and A. B. Alvarez, “Performance Evaluation of YOLOv8, YOLOv9, YOLOv10, and YOLOv11 for Stamp Detection in Scanned Documents,” *Applied Sciences*, vol. 15, no. 6, p. 3154, Mar. 2025, doi: 10.3390/app15063154.
- [26] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” in 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy: IEEE, Oct. 2017, pp. 2980–2988. doi: 10.1109/ICCV.2017.322.
- [27] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature Pyramid Networks for Object Detection,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA: IEEE, Jul. 2017, pp. 936–944. doi: 10.1109/CVPR.2017.106.
- [28] R. Padilla, S. L. Netto, and E. A. B. da Silva, “A Survey on Performance Metrics for Object-Detection Algorithms,” in 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), Brazil: IEEE, Jul. 2020, pp. 237–242. doi: 10.1109/IWSSIP48289.2020.9145130.
- [29] G. Lavanya and S. D. Pande, “Enhancing Real-time Object Detection with YOLO Algorithm,” *Journal of EAI Endorsed Transactions on Internet of Things*, vol. 10, no. 1, pp. 1–9, Dec. 2023, doi: 10.4108/eetiot.4541.
- [30] H. Zhang and X. Yang, “A Survey on Algorithms and Performance Metrics for Object Detection,” *International Core Journal of Engineering*, vol. 9, no. 9, pp. 113–145, Sep. 2023, doi: 10.6919/ICJE.202309_9(9).0017.
- [31] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization,” in 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy: IEEE, Oct. 2017, pp. 618–626. doi: 10.1109/ICCV.2017.74.
- [32] Z. Feng, M. Zhu, L. Stanković, and H. Ji, “Self-Matching CAM: A Novel Accurate Visual Explanation of CNNs for SAR Image Interpretation,” *Remote Sensing*, vol. 13, no. 9, p. 1772, May 2021, doi: 10.3390/rs13091772.
- [33] R. Fu, Q. Hu, X. Dong, Y. Guo, Y. Gao, and B. Li, “Axiom-based Grad-CAM: Towards Accurate Visualization and Explanation of CNNs,” in Proceedings of the British Machine Vision Conference 2020, Manchester, England: British Machine Vision Association, Jan. 2020. doi: 10.5244/C.34.146.
- [34] X. A. Inbaraj, C. Villavicencio, J. J. Macrohon, J.-H. Jeng, and J.-G. Hsieh, “Object Identification and Localization Using Grad-CAM++ with Mask Regional Convolution Neural Network,” *Electronics*, vol. 10, no. 13, p. 1541, Jun. 2021, doi: 10.3390/electronics10131541.