RIMC Volume 2, Issue 2, December 2025 | http://dx.doi.org/10.21622/RIMC.2025.02.2.1846
Journal of Robotics: Integration, Manufacturing & Control - eISSN 3009-7967

Real-time Object Detection and Diagnosis of Tomato Quality using YOLO

Youssry A. Mokhtar [1], and Essam H. Seddik [2]

[1] Arab Academy for Science, Technology, and Maritime Transport, Department of Mechanical Engineering, Alexandria, Egypt.

[2] Arab Academy for Science, Technology, and Maritime Transport, Department of Mechanical Engineering, Alamein, Egypt.

y.a.mokhtar@student.aast.edu, essam.seddik@aast.edu

Received: 04 November 2025 Accepted: 16 December 2025 Published: 23 December 2025

Abstract

Tomato quality plays a critical role in both customer satisfaction and the efficiency of post-harvest processing. Traditional sorting and grading methods are labor-intensive, subjective, and unsuitable for high-throughput operations. This study proposes a smart AI-based vision system capable of real-time classification of tomato quality into three classes: fresh, damaged, and unripe. The system employs advanced deep learning techniques, specifically a custom-trained YOLO object-detection model, to analyze key visual attributes such as color, texture, and surface defects. A diverse, labeled custom dataset of tomato images was collected to train and evaluate the model. This dataset included tomato images with the three health conditions according to the output classes of the neural network. The results show that the system has achieved high accuracy and strong robustness across varying lighting conditions and backgrounds, making it suitable for deployment in real agricultural and industrial environments. By enabling fast, automated, and objective quality assessment, the proposed system significantly enhances the reliability and efficiency of tomato grading and contributes to improved food supply chain management.

Key-words: Tomato Quality Classification, Computer Vision, YOLO, Deep Learning

I. Introduction

A. Problem statement

With an annual production of about 186 million metric tons, tomatoes are one of the most consumed vegetables in the world (FAO, 2022). Food safety, waste reduction, and consumer pleasure all depend on maintaining the quality after harvest. Sorting and grading tomatoes according to their ripeness, freshness, and physical flaws is still quite difficult, though, particularly in high-volume production settings like farms, warehouses, and supermarkets. With an annual production of about 186 million metric tons, tomatoes are one of the most consumed vegetables in the world (FAO, 2022). Food safety, waste reduction, and consumer pleasure all depend on maintaining the quality after harvest. Sorting and grading tomatoes according to their ripeness, freshness, and physical flaws is still quite difficult, though, particularly in high-volume production settings like farms, warehouses, and supermarkets [1].

B. Literature review

Since Deep Learning (DL) showed the best results in the literature review, a modified YOLOv5 object detection model was used to

CC BY NC
Copyright © 2025, authors

solve this problem. The DL model categorizes tomatoes according to their color, outer look, damage symptoms, and obvious defects after being trained on a dataset of more than 2,000 annotated tomato photos. The suggested system can be used in robotic arms or conveyor belts in agricultural environments and is built for real-time response [2]. Mukesh Dalal and Payal Mittal systematically reviewed recent advancements in using deep learning models (YOLO v9, v10, EfficientDet, Transformer-based models, and hybrid frameworks) for real-time object detection in agriculture (crops, fruits, diseases), noting enhanced precision but emphasizing challenges like data scarcity and the need for edge computing [3]. Campos Soto, Rojas Pino, and Aguilera Carrasco systematically review the use of deep learning models (CNNs, R-CNN, YOLO) for fruit classification and detection, emphasizing challenges like data scarcity and occlusion while proposing future research into multi-modal integration and computational efficiency [4]. Tan et al. review the application of deep learning in fruit and vegetable picking robots, summarizing key technologies in visual perception, path planning, and end effector control to highlight the shift towards intelligent, automated harvesting [5]. Dewi, Thiruvady, and Zaidi propose a novel Fruit Classification System that applies Neural Architecture Search (NAS) to automatically refine the deep learning network topology, achieving a superior 99.98% mAP for classifying 15 distinct fruit categories [6].

The study by Ioannis D. Apostolopoulos, Mpesi Tzani, and Sokratis I. Aznaouridis proposes a novel, generalizable machine learning (ML) model—specifically leveraging Vision Transformers (ViT)—to objectively assess fruit quality (distinguishing between good and rotten) across various fruit types [7]. The review by Ignacio Rojas Santelices, Cano, Moreira, and Peña Fritz thoroughly analyzes Artificial Vision Systems for fruit inspection and classification, categorizing methodologies by algorithms (e.g., CNN, ANN) and features (Color, Shape, and Texture) to detail the field's current state and common practices for automated quality control [8]. Sarron et al. utilized a YOLOv5 network for mango yield estimation across diverse orchards, finding that correcting for occlusion and detection errors required incorporating categorical covariates (region, cropping system) into the linear model to significantly improve generalization (R^2 from 0.34 to 0.66) [9]. Wang, Fang, Mo, Gan, and Sun reviewed deep learning models for tomato ripeness detection, noting that while existing YOLO-based methods offer high accuracy in controlled settings, they often fail in complex outdoor environments due to reliance on substantial memory and computational resources. This led them to propose the lightweight YOLOv11-MHS model to achieve high precision with reduced overhead in challenging agricultural scenes [10]. Wu, Huang, Song, and Zhou (2025) reviewed deep learning models for tomato ripeness detection, noting that while existing YOLO-based methods offer high accuracy in controlled settings, they often fail in complex outdoor environments due to reliance on substantial memory and computational resources. This led them to propose the lightweight YOLO-PGC model to achieve high precision with reduced overhead in challenging agricultural scenes [11]. The literature review by Wang, Xu, Hu, Zhang, Li, Zhu, and Liu identifies that accurate tomato yield estimation is challenging in field environments due to fruit occlusion and overlap, which limit the performance of traditional and older machine vision techniques. Consequently, the study emphasizes the need for lightweight, deep learning-based networks (like their improved YOLO11n) that can maintain high detection accuracy while being efficient enough for real-time deployment on edge computing devices [12].

II. Dataset

A. Data collection

In this study, a custom image dataset of tomatoes was collected and annotated to add to the YOLO's COCO dataset to include different tomato health conditions. The modified dataset was used to train and evaluate the proposed AI vision system. The dataset is divided into three quality-based classes.

Figure 1 displays an example of each of the three conditions.

Figure 1: Dataset classes.

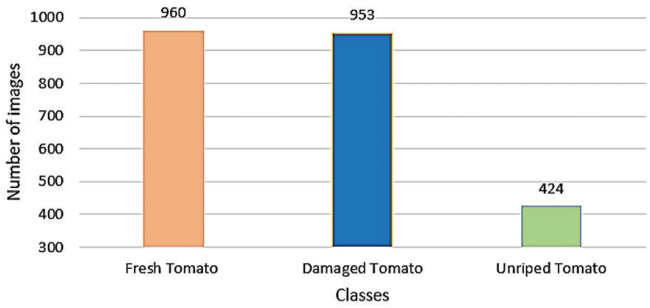

Figure 2: Dataset distribution.

The dataset was designed to reflect real-world conditions, including variations in lighting, background, and angle. As shown in the distribution, the number of unripe tomato samples is relatively smaller than the other two categories. This imbalance is due to the distinct visual features of unripe tomatoes, particularly their color and surface characteristics, which differ significantly from both fresh and damaged tomatoes. These unique features result in distinct feature extraction patterns, allowing the model to effectively learn their representations with fewer samples. Nonetheless, data augmentation techniques such as flipping, rotation, and contrast adjustment were applied to improve model robustness and mitigate class imbalance.

Figure 2 displays a bar chart titled "Dataset Classes", which illustrates the distribution of images across three tomato categories: *f*resh, *d*amaged, and *u*nripe *t*omato*es*. The dataset comprises 960 images of fresh tomatoes and 953 images of damaged tomatoes, indicating that these two classes are nearly equivalent in size. Conversely, the unripe tomato class contains only 424 images, representing a substantially smaller proportion of the dataset. This disparity reveals a noticeable class imbalance, with the unripe tomato category being significantly underrepresented.

The use of color-coded bars, clearly labeled axes, and explicit numerical values enhances the clarity of this distribution.

Emphasizing the importance of applying data augmentation or other balancing techniques prior to model development.

B. Annotation

All photos were labelled using the Roboflow annotation platform, which provides a robust toolset and user-friendly interface for labelling computer vision datasets, to prepare the dataset for training and evaluation. A bounding box and one of three class labels: fresh, damaged, or unripe, were manually applied to each tomato instance in the pictures. Trained annotators carried out the labelling, visually examining each tomato to determine its class based on color, surface texture, and obvious flaws. While built-in features like auto-zoom, keyboard shortcuts, and error checking expedited the annotation workflow, Roboflow's integrated label management system helped guarantee consistent labeling throughout the dataset. To reduce human bias and mislabeling, annotation quality was confirmed by manual inspection and cross-validation by a second reviewer. A total of 2,337 labelled images were produced. Roboflow's dataset management capability was then used to export and version-control these annotations, ensuring repeatability and traceability during the model training procedure. Before exporting the finished dataset, Roboflow was also utilized to carry out data augmentation tasks, including flipping, rotation, and brightness/contrast correction, directly within the platform.

III. Training

Building dependable and broadly applicable machine learning models requires dividing the dataset into subsets for testing, validation, and training. To measure the model's performance in the actual world, this procedure makes sure that it learns from one piece of data, is adjusted using another, and is then tested on entirely unseen data.

The model was trained to identify characteristics and patterns linked to each class (Fresh, Damaged, and Unripe) using the training data. During training, the validation set was used to track the model's performance and adjust hyperparameters to assist in avoiding overfitting. In order to ensure an objective assessment of the model's performance in the real world on unknown data, the testing set was finally set aside for the final review.

A. Checking integrity and fairness of the evaluation process

The dataset was divided stratified by class in order to preserve the fairness and integrity of the assessment procedure. This indicates that, as opposed to having any subset dominated by a single tomato quality class, each subset (training, validation, and test) includes a representative portion of all three tomato quality classes. For instance, the training set is not skewed towards fresh tomatoes, and the test set is not limited to unripe tomatoes. The model's robustness and practical application are enhanced by this balanced distribution, which guarantees that it learns equally from each category and is assessed on its capacity to generalise across all tomato quality kinds.

IV. Computer vision model

A. Model selection

When developing an image-based AI system, choosing the right computer vision model is essential since it has a direct impact on the solution's accuracy, speed, and overall performance. The model must be able to reliably detect small items, identify subtle surface flaws, and function consistently in a variety of lighting and background settings for this project, which entails the real-time classification of tomatoes into fresh, damaged, and unripe categories.

A variety of model families is frequently employed in computer vision tasks, such as object identification models that integrate localization and classification, semantic segmentation networks for pixel-level comprehension, and convolutional neural networks (CNNs) for picture classification. Object detection models are most suited for this task because they involve recognizing and categorizing several tomatoes in a single image.

Among these, state-of-the-art models such as YOLO (You Only Look Once), Faster R-CNN, and SSD (Single Shot Multibox Detector) are widely adopted.

B. YOLO

YOLO (You Only Look Once) is a state-of-the-art, real-time object detection algorithm that reframes object detection as a single regression problem, rather than the traditional two-step approach of region proposal followed by classification. Unlike older methods like R-CNN or Faster R-CNN that generate multiple candidate regions and then classify them separately, YOLO processes the entire image in one forward pass through a neural network, making it extremely fast and efficient.

The workflow of this research began by splitting the input image into a 13 x 13 grid, depending on the model version. A set number of bounding boxes within each grid cell was then predicted. The model produced a few parameters for each box, including the width and height of the box in relation to the image, the (x, y) coordinates of the box's centre in relation to the cell, and a confidence score that indicates the likelihood of an object being present as well as the accuracy of the predicted box. Furthermore, class probabilities for the detected object—such as whether it is an unripe, damaged, or fresh tomato—are output by each grid cell. The final detection confidence for each object is calculated by multiplying these class probabilities by the bounding box confidence scores.

Once all grid cells and bounding boxes have been predicted, YOLO uses a method known as Non-Maximum Suppression (NMS) to remove overlapping or redundant boxes, leaving only the ones with the highest confidence scores. This procedure minimizes false positives and guarantees that each object is detected once. The model learns by minimizing a mixed loss function that has elements for objectless confidence (determining whether an object exists), classification accuracy (making the right class prediction), and localization accuracy (bounding box regression).

Compared to region-based methods, YOLO better captures global context and spatial linkages since it examines the entire image at once. Because of its architecture, YOLO is incredibly quick, able to operate in real-time (30+ FPS), and appropriate for uses such as security monitoring, autonomous driving, and, in this instance, real-time tomato quality detection. YOLO is the perfect option for smart agricultural systems that need real-time automated decision-making because of its speed-accuracy balance and capacity to identify several items in a single frame.

C. YOLOv5

YOLOv5, created by Ultralytics, is a highly favoured and extensively used object identification model because of its performance, speed, and adaptability. In order to balance accuracy with inference speed, YOLOv5 is implemented in PyTorch and comes in five primary pre-configured model sizes: YOLOv5n (Nano), YOLOv5s (Small), YOLOv5m (Medium), YOLOv5l (Large), and YOLOv5x (Extra Large). Although the depth and width of each version vary, they are all based on the same architecture, which ultimately influences the number of layers and channels as well as the size, accuracy, and speed of the model.

D. YOLOv5n

The YOLOv5n (Nano) model was chosen for this project because it strikes a great mix between speed, efficiency, and tolerable accuracy. With roughly 1.9 million parameters, YOLOv5n is the lightest version of the YOLOv5 family. This makes it ideal for real-time applications on devices with limited resources, like the Raspberry Pi, NVIDIA Jetson, or other edge computing platforms frequently found in agricultural settings. The key architectural advantages of the YOLO family, such as quick inference, end-to-end object recognition, and high spatial awareness, are still present in YOLOv5n despite its diminutive size. Even with different lighting and backdrop conditions, it can identify and categorize several tomato instances in a single image. Even while it is not as accurate as larger models like YOLOv5m or YOLOv5x, in well-structured, high-quality datasets, the performance loss is negligible. YOLOv5n is an ideal solution for on-site, automated quality assessment where speed and deployability are more important than slight precision gains. In this application, it successfully detected and classified tomatoes into fresh, damaged, and unripe categories with dependable accuracy and low latency.

The provided ***Figure 3*** illustrates the process of object detection using a system like YOLO (You Only Look Once). It shows an input image divided into an $ S\times S$ grid, where each cell predicts bounding boxes with confidence scores. A class probability map determines the object class for each grid cell, ultimately leading to the final tomato detection results with precise bounding boxes.

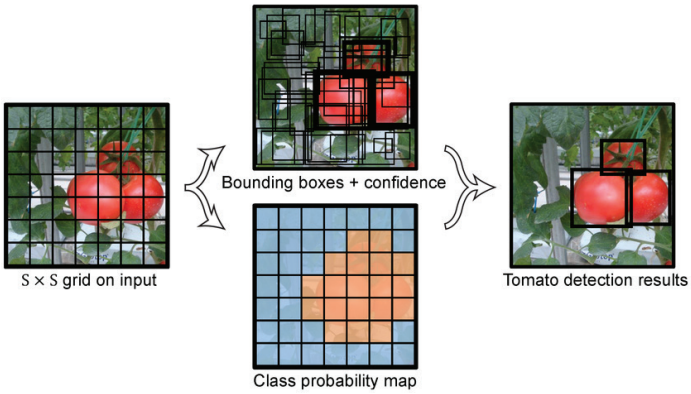

Figure 3: YOLO feature extraction.

E. YOLOv5n Layer architecture

The YOLOv5n (Nano) architecture is suitable for edge devices and embedded systems used in agricultural applications, as shown in ***Figure 3,*** because it offers real-time object detection with low computational overhead. The three main parts of it are the head, neck, and Backbone. To improve early-stage efficiency, the Backbone starts with a special Focus layer that slices the image, boosts the channel depth, and reduces spatial dimensions. The Backbone oversees extracting visual information from the input image.

Convolutional layers used in conjunction with Batch Normalization and the SiLU activation function—which provides a smoother gradient flow than ReLU—come next. The usage of Cross Stage Partial (CSP) Bottlenecks, which divide the feature map into two paths—one undergoing transformation and the other preserved—and then concatenate them to improve gradient

flow and minimize computation, is a crucial component of the Backbone. A layer called Spatial Pyramid Pooling (SPP) completes the Backbone. It uses max-pooling at various scales to gather local and global information for improved object representation. The next step involves the use of convolutional layers in combination with Batch Normalisation and the SiLU activation function, which offers a smoother gradient flow than ReLU.

A key part of the Backbone is the use of Cross Stage Partial (CSP) Bottlenecks, which split the feature map into two paths, one of which is undergoing transformation and the other of which is maintained. These paths are then concatenated to enhance gradient flow and reduce computation. The Backbone is completed by a layer known as Spatial Pyramid Pooling (SPP). For better object representation, it collects local and global data using max-pooling at different sizes.

Figure 4 illustrates two key architectural modules of YOLOv5: the C3 (Cross Stage Partial Bottleneck) module and the SPPF (Spatial Pyramid Pooling - Fast) module. The C3 module is the primary processing unit, designed to improve gradient flow and reduce computation by splitting and concatenating feature paths. The SPPF module, used in the Backbone, efficiently gathers multi-scale global and local contextual information via a sequential process. pooling operations.

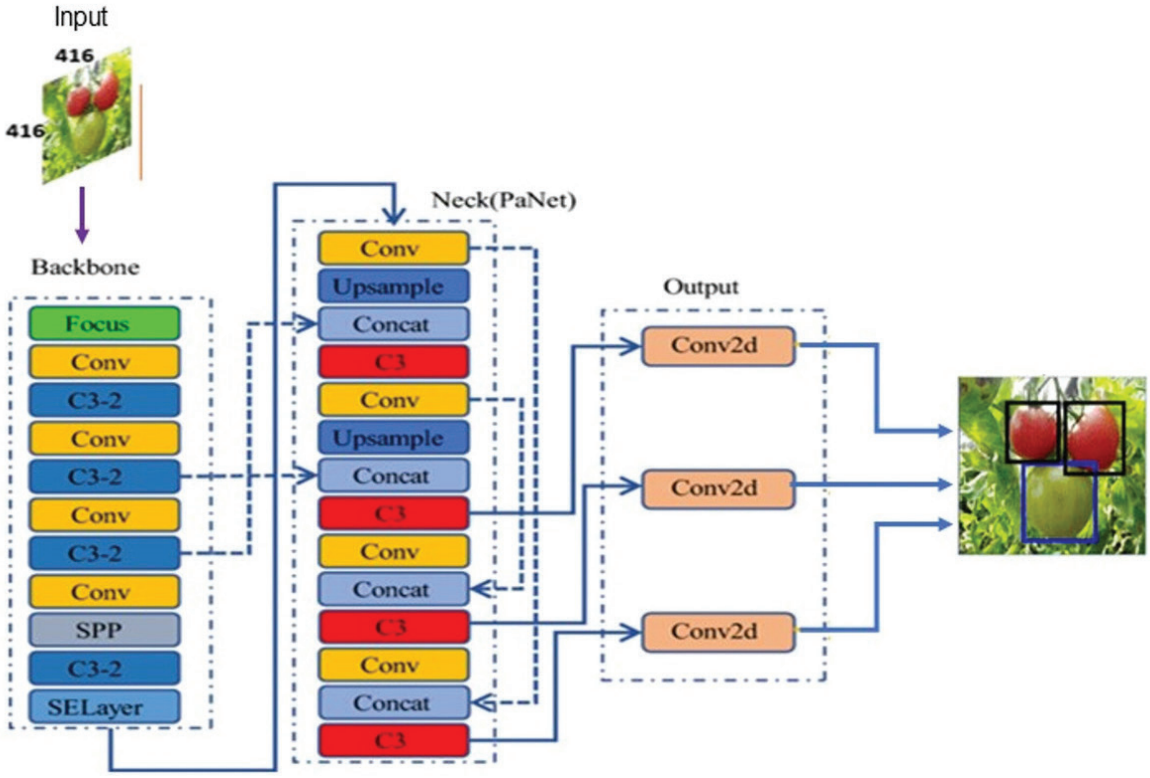

Figure 4: YOLOv5 layers.

This Precision–Confidence Curve shown in ***Figure 5*** shows how the precision of your tomato classification model changes as you adjust the confidence threshold. The x-axis represents the model's confidence level, and the y-axis represents precision, which measures how many predictions made above that confidence level is correct. The curves for Fresh and Unripe tomatoes (orange and green) stay high across almost all confidence levels, meaning the model identifies these classes very accurately even when its confidence is low. In contrast, the Damaged Tomato curve (light blue) remains low at lower confidence levels and increases only gradually, showing that this class is more difficult for the model and produces more false positives. The thick blue curve represents overall precision across all classes, and it reaches perfect precision (1.00) at a confidence of about 0.924. Overall, the plot

highlights strong performance for Fresh and Unripe tomatoes and weaker consistency for Damaged tomatoes, helping you choose an appropriate confidence threshold depending on whether you prefer higher accuracy or more detections.

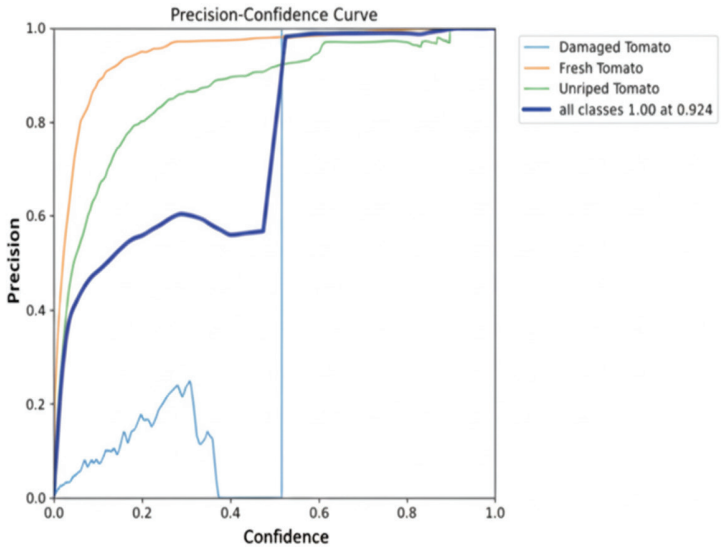

Figure 5: Precision-confidence curve.

The figure below (***Figure 6***) presents a normalized confusion matrix that evaluates the performance of a classification model across four categories: Damaged Tomato, Fresh Tomato, Unripe Tomato, and Background. The diagonal values represent correct predictions, showing that the model performs exceptionally well for Fresh Tomato, with a precision of 0.98, and achieves strong accuracy for Unripe Tomato and Damaged Tomato, with scores of 0.86 and 0.70, respectively. The matrix also indicates that a small proportion of Damaged Tomato images (0.03) and Unripe Tomato images (0.11) were misclassified as background, while minor confusion is observed between the tomato classes, such as 0.01 of unripe tomatoes predicted as damaged and 0.02 of fresh tomatoes predicted as background. Overall, the matrix highlights that while the model demonstrates high accuracy for most categories, some misclassification occurs, particularly with background and visually similar tomato classes—indicating areas where further model refinement or dataset enhancement may be beneficial.

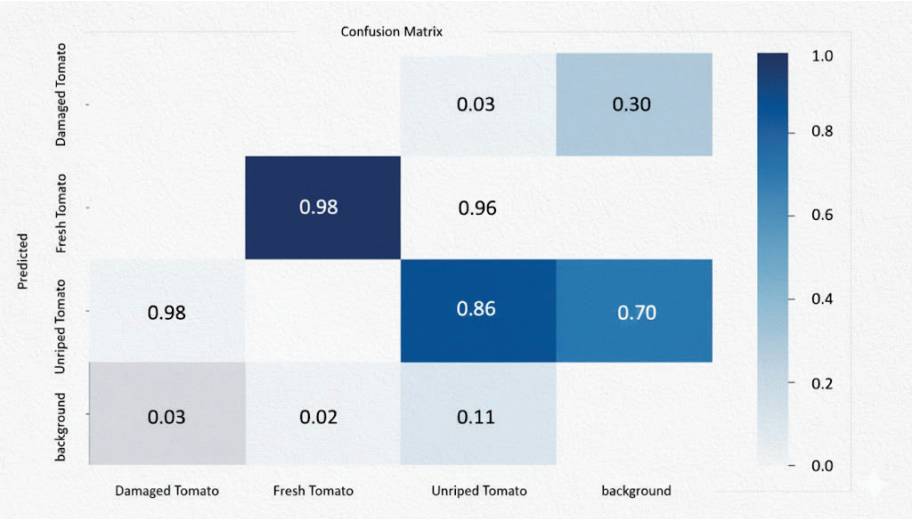

Figure 6: Confusion matrix.

V. Model performance evaluation

A. YOLOv5n model performance evaluation

Precision – 97.3% The percentage of anticipated positive cases (detections) that turn out to be accurate is known as precision. In this case, a precision of 97.3% indicates that the model is **97.3% accurate** in predicting whether a tomato is fresh, damaged, or unripe. Because of its great precision, the model seldom mislabels or mistakenly classifies non-tomato objects as tomatoes (false positives). This high precision is vital, as it ensures that when the model signals a detection, the **robotic arm** can trust the classification and act with high confidence, minimizing false picks and maximizing process reliability.

Recall – 63.1% Recall measures the model's ability to detect all actual objects. A Recall of 63.1% means that the model successfully detects about 63 out of every 100 actual tomatoes present in the images. This suggests that some objects are being missed. This low Recall reveals a significant limitation: the **robot** is missing nearly 4 out of every 10 available tomatoes due to challenges like occlusion or lighting. Therefore, while the harvested crop will be accurately sorted, the overall yield collection will be substantially incomplete, necessitating further optimization before full **robot** deployment.

B. Model performance test

Figure 7 shows a single tomato with visible signs of damage. The tomato surface has dark spots, bruising, and areas of decay, indicating biological or physical deterioration. This crucial detection is necessary to prevent the robotic arm from selecting and harvesting a compromised fruit. These features—such as discoloration, wrinkling, and surface lesions—are commonly associated with damaged or spoiled tomatoes.

Figure 8 shows the output of the tomato detection model, where two tomatoes are identified and classified using bounding boxes. The tomato on the left is labeled "Fresh Tomato" with 79% confidence, indicating it is ready for the robot to pick, while the tomato on the right is labeled "Unripe Tomato" with 94% confidence, signaling it should be ignored. This demonstrates the model's ability to detect multiple tomatoes in one image and accurately distinguish between different ripeness stages, effectively acting as the **robot's** 'eyes' for selective harvesting.

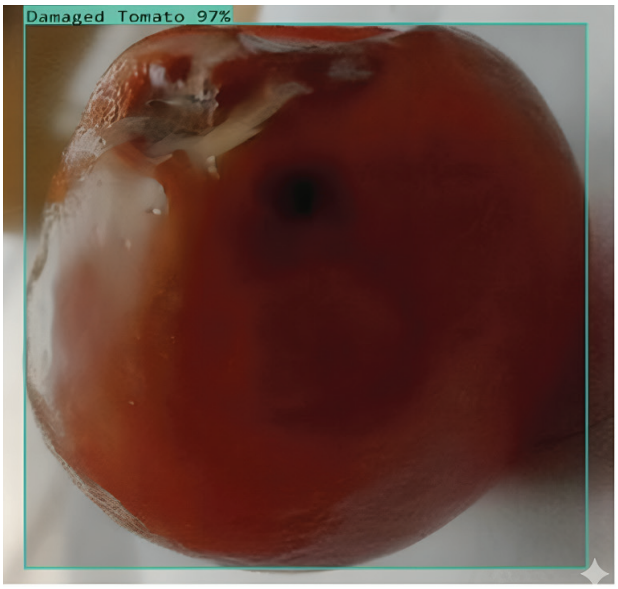

Figure 7: Damaged tomato detection.

Figure 7: Multi-label tomato detection.

VI. Conclusion

This paper presented a real-time AI-based computer vision system for automated tomato quality detection using a lightweight YOLOv5n object detection model. The proposed approach addressed the limitations of traditional manual grading by enabling fast, objective, and scalable classification of tomatoes into fresh, damaged, and

unripe categories under diverse lighting and background conditions. A custom annotated dataset reflecting real-world variability was developed to train and evaluate the model. Experimental results demonstrated high precision (97.3%), indicating strong reliability in correct classification and low false-positive rates, which are critical for automated harvesting and sorting applications. While the recall rate (63.1%) revealed challenges in detecting all tomato instances—particularly under occlusion and complex scenes—the system proved effective as a real-time decision-support tool for quality assessment in smart agricultural environments.

Future improvements will focus on enhancing Recall and overall robustness through dataset expansion, improved class balancing, and additional data augmentation using images collected from real harvesting fields. Further research may explore advanced or hybrid deep learning architectures, including newer YOLO variants and attention-based models, to better capture subtle surface defects while maintaining real-time performance on edge devices. Additionally, model optimization techniques such as pruning and quantization can be employed to reduce computational overhead. Integrating the proposed vision system with robotic manipulators or conveyor-based sorting platforms, along with depth sensing and grasp planning, represents a key step toward fully autonomous tomato harvesting and grading systems, contributing to reduced post-harvest losses and improved efficiency in smart agriculture.

REFERENCES

[1] J. A. Clark, "Automation in horticulture: The future of crop grading and handling," 2022. [Online]. Available: https://www.cabidigitallibrary.org/doi/pdf/10.5555/20220163943

[2] M. D. Forecast, "Tomato Market Size, Share, Trends & Growth Report, 2033," 2025. [Online]. Available: https://www.marketdataforecast.com/market-reports/tomato-market

[3] M. Dalal and P. Mittal, "A Systematic Review of Deep Learning-Based Object Detection in Agriculture: Methods, Challenges, and Future Directions," *Computers, Materials & Continua*, vol. 84, no. 1, pp. 57–91, 2025, doi: 10.32604/cmc.2025.066056.

[4] S. Espinoza, C. Aguilera, L. Rojas, and P. G. Campos, "Analysis of Fruit Images With Deep Learning: A Systematic Literature Review and Future Directions," *IEEE Access*, vol. 12, pp. 3837–3859, 2024, doi: 10.1109/ACCESS.2023.3345789.

[5] Y. Tan, X. Liu, J. Zhang, Y. Wang, and Y. Hu, "A Review of Research on Fruit and Vegetable Picking Robots Based on Deep Learning," *Sensors*, vol. 25, no. 12, p. 3677, Jun. 2025, doi: 10.3390/s25123677.

[6] C. Dewi, D. Thiruvady, and N. Zaidi, "Fruit Classification System with Deep Learning and Neural Architecture Search," 2024.

[7] I. D. Apostolopoulos, M. Tzani, and S. I. Aznaouridis, "A General Machine Learning Model for Assessing Fruit Quality Using Deep Image Features," *AI*, vol. 4, no. 4, pp. 812–830, Sep. 2023, doi: 10.3390/ai4040041.

[8] I. Rojas Santelices, S. Cano, F. Moreira, and Á. Peña Fritz, "Artificial Vision Systems for Fruit Inspection and Classification: Systematic Literature Review," *Sensors (Basel)*, vol. 25, p. 1524, Dec. 2025, doi: 10.3390/s25051524.

[9] J. Sarro *et al.*, "Fruit yield estimation using image analysis is also about correcting the number of detections | International Society for Horticultural Science," 2023. [Online]. Available: https://www.ishs.org/ishs-article/1360_42

[10] D. WANG, Z. FANG, M. MO, J. GAN, and Z. SUN, "Tomato ripeness detection method based on improved YOLOv11 lightweight model," *Front Agric Sci Eng*, vol. 13, 2025, doi: 10.15302/j-fase-2025657.

[11] Q. Wu, H. Huang, D. Song, and J. Zhou, "YOLO-PGC: A Tomato Maturity Detection Algorithm Based on Improved YOLOv11," *Applied Sciences*, vol. 15, no. 9, p. 5000, Apr. 2025, doi: 10.3390/app15095000.

[12] A. Wang *et al.*, "Tomato Yield Estimation Using an Improved Lightweight YOLO11n Network and an Optimized Region Tracking-Counting Method," *Agriculture (Switzerland)*, vol. 15, no. 13, 2025, doi: 10.3390/agriculture15131353.