INSTANCE SEGMENTATION OF INDVIDUAL CAVENDISH BANANA (*Musa acuminata*) AND RIPENESS CLASSIFICATION USING YOLOv11

Adrian Lloyd L. Dalion

INTRODUCTION

Bananas (*Musa spp.*) are among the most important tropical fruits globally, contributing significantly to both consumption and agricultural trade. Of these, the Cavendish variety (*Musa acuminata*) dominates international banana markets due to its long shelf life, consistent appearance, and suitability for large-scale export operations (Maseko et al., 2024). In the Philippines, bananas are a leading fruit crop, contributing substantially to the livelihoods of millions of Filipinos and accounting for a major share of the country's agricultural exports. In 2022, the Philippines remained one of the world's top banana producers, with total output exceeding 9 million metric tons (Philippine Statistics Authority, 2023). Cavendish bananas, the primary variety for export, made up more than half of this volume. The Davao Region serves as the country's leading production hub, supported by extensive plantations, grower cooperatives, and postharvest infrastructure (PCAARRD, n.d.).

Ensuring the uniform ripeness of Cavendish bananas is important for meeting export quality standards. Variability in ripeness among banana fingers within a single bunch can significantly affect shelf life and marketability. Hailu et al. (2013) showed that fruit position in the bunch influences ripening, often resulting in uneven maturity. This poses a challenge for export markets, which require consistent quality. As outlined in the Codex Standard for Bananas, international shipments must contain bananas of uniform variety, origin, and ripeness (Food and Agriculture Organization & World Health

Organization, 1997/2022). Traditional manual grading remains the standard in most post-harvest facilities, but this approach is subjective, labor-intensive, and often limited to bunch-level assessments. As a result, suboptimal sorting contributes to postharvest losses and quality rejections during export inspection (Manalo et al., 2023). This underscores the need for an automated, fine-grained approach to assess ripeness per finger.

In this context, computer vision and deep learning models have gained traction in agricultural quality assessment, offering non-destructive, repeatable, and scalable solutions. Among these, the YOLO (You Only Look Once) object detection framework has become a widely used model for real-time object detection and image analysis. YOLOv11, one of the latest versions in the YOLO series, extends its capabilities to include instance segmentation, allowing not only the detection but also the pixel-level delineation of individual objects within an input image. In addition to segmentation, it also supports object classification, enabling the assignment of different class labels to each detected instance. Architectural improvements such as the C3k2 block for enhanced multi-scale feature representation and the C2PSA (Channel and Spatial Attention) module for refined attention to object details significantly boost both segmentation accuracy and classification performance, making YOLOv11 well-suited for applications requiring precise object localization and category prediction. (Ultralytics, 2024). This is particularly valuable for fruit classification tasks involving dense or clustered objects, such as banana fingers in a bunch.

Recent peer-reviewed studies have demonstrated the applicability of YOLO-based models in banana ripeness detection. Cayha et al. (2023) used YOLOv5 to classify the ripeness levels of Cavendish bananas which achieved a mean Average Precision (mAP) of

98.8%, while Wang et al. (2025) developed an enhanced YOLOv9-based model for banana ripeness detection under complex lighting conditions, achieving high performance with a mAP50 of 98.9% and mAP50-95 of 95.9%. These works highlight the feasibility of combining RGB imagery and YOLO segmentation for ripeness classification in real-world agricultural settings.

This study aims to apply YOLOv11 to segment individual fingers of Cavendish banana bunches and classify their ripeness stages using RGB image data. Unlike previous approaches that rely on traditional classification or unsupervised clustering, this method integrates segmentation and classification in a single end-to-end model. The project is designed with the goal of establishing a baseline system that could enhance ripeness detection accuracy, reduce labor dependency, and inform future development of intelligent grading tools for the banana industry.

By focusing on per-finger ripeness detection, this study aligns with current agricultural innovation priorities in the Philippines. It supports national goals for postharvest loss reduction, digital agriculture, and export competitiveness. It contributes to SDG 2 (Zero Hunger) by promoting improved food quality and reducing spoilage, SDG 9 (Industry, Innovation, and Infrastructure) by applying advanced AI technologies in agriculture, and SDG 12 (Responsible Consumption and Production) by enabling more precise sorting and minimizing waste across the supply chain (United Nations, 2015).

Objective of the Study

The general objective of this study is to design and evaluate a YOLOv11-based model for detecting and classifying the ripeness of individual Cavendish banana fingers

using RGB image data within the context of postharvest operations in Davao City, Philippines. Specifically, the study aims to:

1. develop a comprehensive database of bananas with annotated finger-level segmentation to enable the classification of individual fingers;
2. train and optimize a YOLOv11-based model to perform precise instance segmentation of individual banana fingers from bunch-level RGB images;
3. classify the ripeness stage of each segmented finger using predefined features derived from the model output;
4. evaluate the segmentation and classification performance of the YOLOv11 model using metrics such as Intersection over Union (IoU), mean Average Precision at multiple IoU thresholds, confusion matrix analysis, precision, recall, and F1-score.

The scope of this study is limited to the design and evaluation of a YOLOv11-based model for segmentation and ripeness classification of individual Cavendish banana fingers using RGB images collected under postharvest conditions in Davao City, Philippines. The model is trained and tested on a manually annotated dataset, with performance evaluated using metrics such as Intersection over Union (IoU), mean Average Precision (mAP), precision, recall, F1-score, and confusion matrix analysis. While YOLOv11 supports real-time detection, this study does not cover hardware integration, real-time deployment, or field implementation. Additionally, factors such as varying lighting conditions, other banana cultivars, and environmental variations are beyond the scope of this research.

The succeeding chapters will cover the review of related literature on YOLO-based fruit segmentation and ripeness detection in Chapter 2, and the methodology in Chapter 3, including dataset preparation, YOLOv11 training, and performance evaluation.

REVIEW OF LITERATURE

Cavendish Banana

The Cavendish banana (*Musa acuminata*, AAA group) plant is a large herbaceous species characterized by a pseudostem formed from tightly packed leaf sheaths that provide structural support. As a triploid hybrid, it is naturally seedless, making it ideal for consumption. It belongs to the genus *Musa* and the family Musaceae, which also includes other banana and plantain species (Mathew & Negi, 2016). At the apex of the pseudostem, a terminal inflorescence emerges, which eventually develops into a fruit bunch. This bunch is borne on a central stalk known as the peduncle, segmented into nodes or crowns and internodes. Each node bears a hand of bananas, arranged in circular tiers along the stalk (Guo et al., 2021). A hand consists of multiple individual fruits called fingers (Morphology of Banana Plant | Improving the Understanding of Banana, n.d.). In commercial cultivation, growers commonly thin the developing bunch, retaining about six to nine hands to optimize fruit size and quality. This structural arrangement, from pseudostem to fruit fingers, is typical of *Musa acuminata* hybrids, including the Cavendish variety (Guo et al., 2021).

Cavendish Banana Production in the Philippines

Bananas are among the Philippines' most important fruit crops. National banana output has hovered around 9 million metric tons per year in recent years, making the country one of the world's largest producers (DAFF, 2024). Cavendish bananas, the main export variety, consistently account for about half of total output. Earlier data from 2019 placed Davao's share at 3.43 million metric tons, or roughly 37% of national production. Other major producing regions include Northern Mindanao and SOCCSKSARGEN,

contributing around 21% and 13% of national output, respectively. Production systems vary depending on market orientation. Export-driven plantations across Mindanao primarily cultivate Cavendish bananas, while varieties such as Saba and Lakatan are more commonly grown for local consumption and processing markets (PCAARRD, n.d.).

While national banana production in the Philippines has remained relatively stable in recent years, temporary declines were observed in 2020 and 2021, primarily due to the COVID-19 pandemic and outbreaks of plant diseases. A slight production drop of 0.5% in Davao in 2023 was attributed to weather disturbances and disease pressure (Philippine Statistics Authority, 2024).

Cavendish Banana Export Volume, Value, and Markets

According to data from the Philippine Statistics Authority, as cited by Arcalas (2024), Cavendish bananas remain the Philippines' top fresh fruit export, generating over $1.0 billion annually. In 2023, banana exports reached 2.354 million metric tons, reflecting a 3.5% increase from 2022, and generated approximately $1.22 billion in export revenue. The Davao Region accounted for nearly all of these shipments, earning $1.19 billion from banana exports in 2023, which represented around 48% of the region's total exports (Tantanud, 2024). Based on Lagare's (2025) report, while export volumes dipped slightly to 2.278 million metric tons in 2024 due to disease outbreaks and adverse weather conditions, the Philippines maintained its position as Asia's leading banana exporter and ranked fourth globally, according to the FAO (2024).

Cavendish banana exports make up roughly 25% of total national banana production, with the rest consumed domestically. The country's major export destinations include Japan, China, South Korea, and several Middle Eastern nations (DAFF, 2024b).

Japan remained the top destination, with imports totaling $759 million. Although Japan previously sourced as much as 94 percent of its bananas from the Philippines, this share declined to approximately 75 percent by 2024 (Cayon, 2025). China was the second-largest market, purchasing bananas valued at $423 million. South Korea ranked next with imports worth $165 million. Demand also increased in the Middle East, particularly in Saudi Arabia and Iran, which imported $40.1 million and $28.6 million worth of bananas, respectively (The Observatory of Economic Complexity, n.d.).

Postharvest Handling of Cavendish Bananas in the Philippines

The postharvest handling of Cavendish bananas in the Philippines follows strict protocols outlined in the Philippine National Standards on Good Agricultural Practices (GAP) (PNS/BAFPS 129:2013) and grading standards (PNS/BAFS 64:2008). Physical handling is tightly controlled to prevent bruising and contamination. Harvested bunches must not touch the ground, and all tools, trays, or crates used must be clean, padded, and sanitized. To minimize mechanical damage, many farms utilize cableways or roller systems to transport banana bunches directly to the packing area. When bananas are dehanded in the field, they are placed directly into foam-lined crates or trays to avoid surface injury (BAFPS, 2008; 2013).

The GAP code also recommends positioning packing houses near the production area to reduce the distance and time between harvest and packing. This minimizes the risk of jostling and mechanical damage during transport. Cuts and bruises sustained during handling can accelerate ripening and microbial decay, leading to postharvest losses. Philippine quality standards emphasize that Extra Class and Class I bananas must be free of bruises, scratches, and other mechanical injuries to qualify for export (BAFPS, 2013).

Once in the packing house, bananas undergo washing, sorting, and grading under hygienic conditions. Philippine GAP mandates that only potable, chlorinated water be used for washing, sometimes with additional agents such as alum or soap. High-pressure sprays or water immersion are used to remove latex and dirt. After cleaning, bananas are manually sorted and graded by size, form, and appearance. Undersized, deformed, or bruised fingers are culled to ensure that only mature, well-formed, and defect-free fruit move on to packing (BAFPS, 2013).

Packing is done with care to preserve quality. Banana clusters may be wrapped in polyethylene and placed inside corrugated cartons with cushioning materials. Each box is labeled with traceability codes including packing date, grower ID, and site code. Boxes are then stacked on pallets, secured, and often ventilated with liners to maintain humidity. Pre-cooling within 24 hours of harvest is standard to remove field heat and slow down fruit respiration. The recommended cold chain temperature for transport is 13–14°C, which delays ripening without causing chilling injury (BAFPS, 2013).

Hygiene and food safety are emphasized throughout storage and transport. Packing house floors and equipment must be easy to clean and regularly sanitized. Waste and rejected fruit are kept in separate areas to avoid contamination. Personnel must follow Good Hygienic Practices (GHP), including proper handwashing, use of protective clothing, and restrictions on working while ill. Transport vehicles must be cleaned, sanitized, and inspected before loading. Bananas are never transported with chemicals or other produce that may compromise food safety (BAFPS, 2013).

Ripeness Indicators and Grading Standards

Banana ripeness is defined by both physiological development and market readiness. In the Philippines, the *Philippine National Standard for Fresh Bananas (PNS/BAFS 64:2008)* defines a *mature* banana as one that can ripen acceptably postharvest, and a *ripe* banana as being in its most desirable condition for consumption (BAFPS, 2008). Practically, this corresponds to full finger development, initial peel color change, and sufficient starch-to-sugar conversion. International standards, such as the Codex Standard for Bananas (CXS 205-1997), similarly require that bananas be harvested at a stage of physiological maturity suitable for ripening during transit and distribution (Food and Agriculture Organization & World Health Organization, 1997/2022).

Maturity assessment prior to harvest relies on a combination of empirical indices to support uniform grading. These include finger size, evaluated using calipers; fruit shape, where a three-quarters to fully rounded cross-section signifies advanced maturity; and observable changes in peel color. Maturity timelines vary by cultivar and propagation method, with bananas from tissue culture reaching harvestable stage within approximately 9 to 10 months, and traditional varieties maturing over 11 to 14 months. (Wayua et al., 2024). These field-based indicators ensure consistent harvest maturity, which is essential for predictable ripening outcomes. The PNS grading system classifies fruit into Extra, Class I, and Class II, all of which must be mature and firm at harvest (BAFPS, 2008).

Ripeness is further assessed through physical and biochemical traits. Visual indicators include peel color progression (green to yellow), while firmness declines as starch breaks down. Sugar levels, measured as total soluble solids (TSS), rise from 4–7 °Brix (unripe) to 18–25 °Brix (fully ripe), accompanied by increased sweetness and

reduced acidity. These characteristics are critical for both domestic and export markets: locally, later-stage ripeness is targeted for immediate consumption; for export, mature-green fruit (e.g., USDA color stage 2–3) is harvested to allow ripening in transit. Uniform application of these ripeness standards ensures consistent quality, reduces spoilage, and aligns with market expectations across supply chains (United States Department of Agriculture, 2004); PNS/BAFS, 2008; Chiwate et al., 2023).

Factors in uneven ripening

Bananas within a single bunch often ripen unevenly due to natural differences in fruit development, position, and maturity. One of the primary factors influencing this variability is the position of the fruit on the stalk. Typically, bananas located on the upper (proximal) hands of the bunch are larger and more developed, having received assimilates earlier during growth. In contrast, those on the lower (distal) hands tend to be smaller, with fewer cells and less maturity at harvest. Studies have shown that basal fingers contain more dry matter and starch compared to their distal counterparts, contributing to their tendency to ripen more quickly when exposed to ripening agents. As a result, in a mixed-size bunch, the larger, more mature fruits may reach peak ripeness earlier than the smaller, less-developed ones (Satekge & Magwaza, 2021).

Harvesting practices also contribute to uneven ripening, as bananas are often collected in whole bunches that contain fruits at different developmental stages. Although bunches are later sorted by finger size, initial variations in maturity remain, and even under controlled ripening conditions, some fruits will lag behind others. Research involving 1-methylcyclopropene (1-MCP) shows that less mature, smaller fingers often respond more

weakly to the treatment, resulting in inconsistent color development across the bunch (Satekge & Magwaza, 2021).

External environmental factors further exacerbate ripening variability. For example, ethylene released from decaying plant material or nearby ripening fruits can cause localized ripening in adjacent bananas. Uneven air circulation and temperature in ripening rooms can also create microclimates, which lead to different ripening rates among fruits in the same bunch. In practice, even small sources of ethylene, such as decomposing banana leaves, can accelerate ripening in certain areas of a bunch, resulting in a non-uniform ripening pattern (Nelson, n.d.).

Limitations of traditional grading methods

Traditional grading methods for Cavendish bananas, which primarily rely on visual inspection, remain the standard in many postharvest workflows. However, these approaches have significant limitations that reduce grading accuracy, consistency, and scalability. One concern is subjectivity. Human graders rely on perception to evaluate quality, so different inspectors may grade the same banana differently, which compromises standardization and reliability (Ucat & Dela Cruz, 2019; Cahya et al., 2023). Mao et al. (2023) emphasize that subjectivity can lead to errors, especially in detecting minor surface defects or slight color variations.

Another limitation is the inability of manual grading to detect fine or densely distributed surface defects. According to Prihantoro et al. (2023), this can result in inaccurate grading and the export of substandard fruit, increasing the risk of shipment rejection. The accuracy of visual grading is also affected by lighting, which influences the

perception of color, gloss, and surface irregularities. Inconsistent lighting across facilities contributes to grading errors (Cahya et al., 2023).

Manual grading is also labor-intensive and lacks scalability. As demand for Cavendish bananas grows, dependence on human labor creates bottlenecks. Scaling up requires more labor, which may not be feasible due to economic or logistical constraints (Ucat & Dela Cruz, 2019). This inefficiency hinders the ability to meet export demands while maintaining quality standards.

Computer Vision Applications in Agriculture

As a transformative tool in agricultural automation, computer vision enables machines to process and interpret visual data, replicating human-like decision-making to enhance productivity, precision, and sustainability. According to Ghazal et al. (2024), the integration of computer vision with artificial intelligence and machine learning has given rise to vision-based intelligent systems that support a digital life cycle of crops, from image acquisition to farm planning.

Computer vision encompasses a wide range of applications within the agricultural domain. Tian et al. (2020) categorize its use across six key areas: monitoring crop health, preventing and controlling diseases and pests, automating harvesting, grading and quality inspection, managing modern farms, and monitoring farmland with Unmanned Aerial Vehicles (UAVs). For instance, image-based monitoring has been shown to detect nutrient deficiencies or plant diseases more rapidly than traditional methods, facilitating timely interventions that are important for yield and quality optimization.

Moreover, imaging technologies such as RGB, multispectral, hyperspectral, and thermal imaging have significantly expanded the capability of computer vision systems.

As Ghazal et al. (2024) note, these imaging modalities enable the detection of plant characteristics like water stress, chlorophyll content, and vegetation density which are parameters that are difficult to assess manually. Coupled with advanced machine learning techniques, these imaging systems empower automated decision-making processes that enhance accuracy and reduce labor costs.

Despite these advancements, challenges remain in deploying computer vision technologies across diverse agricultural settings. Variability in environmental conditions, the need for robust datasets, and the demand for technical expertise pose barriers to widespread adoption. Tian et al. (2020) emphasize the importance of developing adaptable and scalable solutions capable of performing reliably in complex, real-world farming environments, and highlight the need for interdisciplinary collaboration to address technical, economic, and infrastructural challenges.

Image Segmentation

Image segmentation is a foundational process in computer vision that involves partitioning an image into distinct regions based on features such as color, intensity, or texture. This process simplifies visual data and facilitates subsequent analytical tasks, including object detection, classification, and pattern recognition. Segmentation has a wide range of applications across various fields. In object recognition and surveillance, it enables the separation of objects from backgrounds and supports motion tracking. In medical imaging, segmentation allows for the identification and delineation of anatomical structures, aiding diagnosis and treatment planning. It also plays a role in autonomous systems by supporting obstacle detection and navigation, and is used in video compression,

visual effects, and augmented reality to manipulate visual content or overlay digital information in real-time (Patel, 2024).

Segmentation methods are typically classified as traditional or deep learning based. Traditional techniques such as thresholding, edge detection, region growing, and clustering rely on low-level image features and are computationally efficient but often struggle with complex scenes, lighting variability, and noisy data. In contrast, deep learning approaches, particularly convolutional neural networks (CNNs), have significantly enhanced segmentation performance. Models such as Fully Convolutional Networks (FCNs), U-Net, Mask R-CNN, and DeepLab extract hierarchical features from images and provide more accurate and precise segmentation outputs. These models use encoder-decoder architectures, instance segmentation branches, and multi-scale context modules to improve localization and boundary precision. However, challenges still persist including high computational demands. Preprocessing and post-processing techniques can help address these limitations, but transparency in model decision-making remains an active area of research (Patel, 2024).

Segmentation methods in fruits

K-means clustering algorithm. [1] In the study by Prananda et al. (2024), image segmentation was implemented using the K-Means clustering algorithm on grayscale-converted blueberry images. Thresholding was applied by calculating an average of the minimum and maximum grayscale pixel values in segmented 4×4 pixel blocks. The goal was to simplify the image to binary format, allowing segmentation based on pixel intensity values representing varying maturity levels. Once the images were thresholded, K-Means clustering was used to divide the images into two distinct classes: ripe and unripe

blueberries. The number of clusters was set to K=2, and the centroids were derived from the grayscale pixel intensities. Pixel classification was then based on proximity to the nearest cluster center. The segmentation output was shown in processed binary images, with blueberries appearing as white foreground objects against a black background. The study presented successful segmentation results showing clear separation of blueberries. However, the study did not report quantitative performance metrics to evaluate segmentation performance. The discussion of segmentation was primarily visual and descriptive, with results interpreted based on sample outputs.

SCNet50. [2] Tang et al. (2023) conducted instance segmentation of strawberries in natural field environments using a Mask R-CNN-based model enhanced with self-calibrated convolutions. The dataset used for training included 860 manually annotated images, with 240 images used for testing. The objective was to detect and segment individual strawberries under varying occlusion levels and background conditions.

The model architecture was based on Mask R-CNN with a ResNet50 backbone, where self-calibrated convolution was embedded to enhance the receptive field and improve boundary information. This convolution block was used to recalibrate features at different scales and depths. The input image was passed through the backbone network to generate feature maps, which were then processed by the Region Proposal Network (RPN) to generate region proposals. These proposals were refined and classified using ROIAlign, and corresponding masks were predicted for each detected object (Tang et al., 2023).

Segmentation performance was evaluated using Average Precision (AP), AP@0.50, and AP@0.75, with the self-calibrated ResNet50 model (referred to as SCNet50) achieving an AP of 0.937, AP@0.50 of 0.979, and AP@0.75 of 0.969.

Compared to the standard ResNet50 backbone, SCNet50 showed improvements of 0.039 (AP), 0.021 (AP@0.50), and 0.032 (AP@0.75). To evaluate robustness under occlusion, segmentation results were grouped by occlusion ratios: 0–20%, 20–50%, and over 50%. SCNet50 outperformed ResNet50 in the 0–20% and 20–50% groups, with IoU improvements of 0.031 and 0.056, respectively. However, it was reported that the model still generated missed and duplicate detections, especially under severe occlusion or when strawberry color closely matched the background (Tang et al., 2023).

Modified SOLOv2. [3] Zhao et al. (2023) developed a one-stage anchor-free instance segmentation model specifically for peach detection in orchard settings. The model builds upon the SOLOv2 instance segmentation structure and introduces a detection head that comprises a feature branch and a kernel branch. The feature branch integrates the Convolutional Block Attention Module (CBAM), combining spatial and channel attention mechanisms to enhance feature selectivity and focus. The kernel branch incorporates CoordConv and Deformable Convolution Networks (DCN) to increase spatial sensitivity and geometric adaptability.

During inference, the feature maps from the feature branch are multiplied with dynamic kernels from the kernel branch to generate instance masks for each object grid. The model performs segmentation directly at the instance level using full mask supervision, and it does not rely on region proposal networks (Zhao et al., 2023).

Training and testing were conducted on the NinePeach dataset, consisting of 4599 annotated images categorized by three ripeness stages. Performance was measured using mean Average Precision (mAP). Using a Swin-T backbone, the proposed model achieved a mAP of 72.12%, outperforming Mask R-CNN with the same backbone (69.91%). The

results showed robustness in detecting large and overlapping peaches and high precision in ripe and semi-ripe fruit classes (Zhao et al., 2023).

The study reported limitations in the form of false detections where background regions were incorrectly segmented as fruit and missed detections under heavy occlusion or poor lighting. Additionally, the model's parameter count was 46.17 million with a computational cost of 213.4 GFLOPs, although it had faster inference time and lower memory usage compared to Mask R-CNN (Zhao et al., 2023).

Mask R-CNN. [4] In the study conducted by Liu et al. (2019), instance segmentation technique was used to detect cucumber fruits in greenhouse environments, where occlusions, overlapping objects, and cluttered backgrounds often complicate the task of accurate fruit recognition.

The study applied the Mask R-CNN (Mask Region-Based Convolutional Neural Network) architecture as the core model for cucumber fruit segmentation. Mask R-CNN extends the Faster R-CNN framework by incorporating a parallel branch for pixel-level mask prediction, in addition to bounding box regression and object classification. This architecture allows the model to not only detect objects but also segment each detected instance with a binary mask that outlines the shape of the object within its region of interest (RoI) (Liu et al., 2019).

The network was built upon a ResNet-101 backbone, which served as the feature extractor for generating deep convolutional features from the input images. These features were then fed into the Region Proposal Network (RPN) to generate candidate object regions. For each region, the model produced three outputs: a class label, a bounding box,

and a segmentation mask. The segmentation branch used a small Fully Convolutional Network (FCN) to predict a mask of fixed resolution for each RoI (Liu et al., 2019).

To train and evaluate the model, the authors compiled a dataset consisting of 522 images of cucumbers taken from a real greenhouse setting. The dataset was manually annotated with both bounding boxes and pixel-level masks. The dataset was split into 419 images for training, 52 for validation, and 51 for testing. The test set was used to evaluate the model's generalization performance on unseen data, particularly under conditions with varying degrees of occlusion, lighting, and overlapping fruits (Liu et al., 2019).

The segmentation performance of the Mask R-CNN model was assessed using standard evaluation metrics for instance segmentation. The model achieved a precision of 90.68%, recall of 88.29%, and F1-score of 89.47%. These results indicate that the model could generate accurate segmentation masks that closely matched the ground truth annotations, even when fruits were partially hidden or overlapping with one another. It was also noted that instance segmentation offered advantages over traditional object detection in this context (Liu et al., 2019).

Table 1. Summary of segmentation performance across different fruits and methods.

Study and Year	Subject	Method Used	Dataset Size	Performance
[1] 2024	Blueberries	K-Means Clustering Algorithm	Not specified (grayscale images of blueberries)	No quantitative metrics reported; segmentation assessed visually by binary outputs showing ripe/unripe distinction.
[2] 2023	Strawberries	SCNet50 (Mask R-CNN + Self-Calibrated Convolutions)	1100 images	AP: 0.937 AP_{50}: 0.979 AP_{75}: 0.969

[3] 2023	Peaches	Modified SOLOv2 (CBAM, CoordConv, DCN)	4599 annotated images	With Swin-T backbone: mAP: 72.12%
[4] 2019	Cucumbers	Mask R-CNN	522 images	Precision: 90.68% Recall: 88.29% F1-score: 89.47%

Segmentation methods in bananas

Multiple threshold method. [5] Hu et al. (2015) proposed an automatic segmentation algorithm for bananas in crates using a multiple threshold method based on color features extracted from different color spaces. The objective was to segment bananas accurately from complex crate backgrounds to support future non-destructive quality evaluation methods. The dataset consisted of twelve crate images containing 7 to 10 hands of Cavendish bananas at ripening stage 1. Among the twelve images, one was used for algorithm development, one for qualitative validation, and the remaining ten for quantitative evaluation.

The segmentation process began with manual annotation using Photoshop to segment seven specific components in the images: banana, plastic bag, shadow, bubble paper, overlap of plastic bag and bubble paper, overlap of banana and bubble paper, and overlap of banana and shadow. A large number of discrete pixel points were randomly selected from each category, and nine color features were extracted from the RGB, HSV, and CIE L*a*b* color spaces specifically R, G, B; H, S, V; and L*, a*, b*. The extracted values were analyzed using statistical methods to determine threshold criteria that could distinguish banana pixels from the background (Hu et al., 2015).

Based on this analysis, three specific thresholds were selected for automatic segmentation: 70 for the B channel, 34 for the b* channel, and 52 for the L* channel. These values were applied to differentiate bananas from shadows, plastic bag reflections, and background overlap. The automatic segmentation process consisted of reading the RGB image, converting it to HSV and L*a*b* color spaces, extracting the B, b*, and L* channels, applying threshold segmentation, enhancing the image using median filtering, and applying binary morphological operations including inflation (dilation) and corrosion (erosion) (Hu et al., 2015).

In terms of results, qualitative validation showed that the automatic segmentation results had high visual similarity to the manual segmentation results, with some instances where the automatic method provided better contour delineation, particularly in reflective regions of the plastic bag. Quantitative validation was carried out by comparing the area ratio (A) between the manually and automatically segmented banana regions. The area ratio was calculated using the formula presented in Equation 1.

$$A = \left(\frac{Ps}{Pm}\right) \times 100 \quad [1]$$

where:

P_s is the pixel set of the automatic segmentation

P_m is the pixel set from manual segmentation

Across ten test images, the average area ratio exceeded 80%, with a maximum of 91%, indicating a high level of agreement between the two segmentation approaches. Discrepancies were attributed to human error in manual annotation and inconsistencies due to object complexity and boundary uncertainties. For a crate image sized 256 × 253 pixels, the segmentation process took 0.12 seconds on a Pentium T4200 2.0 GHz processor.

Limitations included the restriction of the algorithm's application to only green (stage 1) bananas, controlled lighting conditions using structured D65 illumination, and the potential impact of water droplets on plastic bags affecting segmentation accuracy (Hu et al., 2015).

Otsu's global thresholding algorithm. [6] In the DeRiBa-Fuz system, segmentation of banana images was implemented through a preprocessing pipeline followed by Otsu's global thresholding. The goal of the segmentation process was to isolate the banana object from the background, enabling accurate extraction of RGB and texture features for ripeness classification (Rusmarsidik and Risnandar, 2022).

Banana images were captured using a smartphone camera under indoor conditions with natural sunlight, and each fruit was photographed from multiple sides. Images were stored in JPG format at a resolution of 4000×6000 pixels. Before segmentation, a three-step preprocessing routine was applied. First is edge filtering to emphasize object boundaries based on color transitions. Second, RGB color transformation, to enhance the color characteristics relevant to ripening stages. And lastly, morphological filtering, which included dilation to connect edge gaps, filling to address pixel voids, and erosion to remove noise and irrelevant texture data (Rusmarsidik and Risnandar, 2022).

The segmentation process used Otsu's global thresholding algorithm, which converts the grayscale banana image into a binary format. Otsu's method calculates a threshold that minimizes intra-class variance and is applied to distinguish the banana object from the background. Pixels with intensity values below the computed threshold were classified as foreground (banana object), while those above were classified as background. The binary output image retained only the segmented banana surface, discarding

background components to ensure clean feature extraction (Rusmarsidik and Risnandar, 2022).

The segmentation process in the DeRiBa-Fuz system was visually assessed through image samples showing the step-by-step results of preprocessing and binary thresholding. The effectiveness of segmentation based on visual clarity and the system's ability to produce clean, distinct banana objects for feature extraction. However, no objective accuracy metrics or comparative evaluations were included in the study (Rusmarsidik and Risnandar, 2022).

Table 2. Summary of segmentation performance using different methods on bananas.

Study and Year	Method Used	Dataset Size	Performance/Remarks
[5] 2015	Multiple Threshold Method	12 crate images (7-10 hands each)	The average area ratio exceeded 80%, with a maximum of 91%.
[6] 2022	Otsu's Global Thresholding Algorithm	Not specified (multi-view images of bananas)	Visually assessed based on clarity and successful feature extraction; no quantitative metric reported.

Image Classification

Image classification is one of the most widely used tasks in computer vision that involves automatically categorizing images into predefined classes. Traditionally, this process consisted of three main stages: region of interest (ROI) selection, feature extraction, and classifier modeling. Among these, feature extraction plays a critical role, as the quality and relevance of extracted features directly influence the performance of the classification system. In earlier approaches, features such as color histograms, texture descriptors, and shape patterns were manually engineered and combined with machine

learning algorithms like support vector machines (SVMs) or decision trees. However, these traditional methods often suffered from limited scalability and reliance on expert-driven feature selection, which constrained their performance in real-world, complex datasets (Wu, 2024).

The emergence of deep learning, particularly Convolutional Neural Networks (CNNs), has significantly transformed image classification by enabling end-to-end learning of hierarchical features directly from raw image data. CNNs have demonstrated remarkable capability in automatically capturing spatial hierarchies, patterns, and contextual relationships within images. Layers within CNNs, such as convolutional, pooling, and fully connected layers, contribute to feature abstraction from low-level edges and textures to high-level semantic structures. For example, models like VGGNet and Inception-v3 have achieved high accuracy in domains such as industrial defect detection, with reported F1-scores exceeding 95%. These architectures also benefit from model fine-tuning and transfer learning, where pre-trained models on large-scale datasets such as ImageNet are adapted to specific tasks using smaller, domain-specific datasets. This approach reduces training time and data requirements while improving classification performance (Wu, 2024).

Another important component in image classification is data augmentation, which addresses the risk of overfitting by artificially increasing the diversity of training data. Techniques such as rotation, flipping, cropping, and color jittering allow models to generalize better by simulating various real-world conditions. In addition, advancements such as semi-supervised learning and multimodal fusion have further expanded the application scope of image classification. For instance, in medical imaging, combining

labeled and unlabeled data with adversarial training allows for more robust learning even with limited annotations. Similarly, integrating text and image modalities in e-commerce applications enhances classification accuracy by enriching feature representation (Wu, 2024).

Despite the progress, several challenges remain. These include the high cost of manual data labeling, lack of interpretability in deep learning models, and vulnerability to adversarial attacks. To address these, current research is exploring directions like self-supervised learning, model compression, and cross-domain adaptation. These innovations aim to reduce dependence on labeled data, improve model efficiency for deployment in low-resource environments, and ensure reliable performance across varying domains (Wu, 2024).

Classification in fruits

K-NN and multi-class SVM. [7] In the study by Israel et al. (2024), a multi-step approach to fruit classification was implemented, including image preprocessing, feature extraction, and classification. The dataset consisted of 1200 images from 12 fruit classes, resized to 200x200 pixels for consistency. Preprocessing included Gaussian smoothing to reduce noise and Otsu thresholding for precise segmentation, followed by morphological operations to refine object boundaries.

Feature extraction was performed using Local Binary Patterns (LBP) and Principal Component Analysis (PCA). LBP captures local texture by comparing each pixel to its neighbors, generating 59-bin histograms that reflect uniform and non-uniform patterns, making it robust to grayscale variations. In contrast, PCA reduces data dimensionality by transforming high-dimensional data into a smaller set of uncorrelated components,

preserving the majority of data variance while reducing computational complexity, though potentially discarding discriminative features (Israel et al., 2024).

The classification phase utilized K-Nearest Neighbor (K-NN) and Multi-Class Support Vector Machine (SVM) algorithms. K-NN, a non-parametric, instance-based learning algorithm, was configured with k=1, meaning each test sample was classified based on the closest single training sample, measured using Euclidean distance. This approach effectively stores the entire training set and relies on majority voting among the nearest neighbors for classification decisions. While straightforward, this method is prone to overfitting, as it can be heavily influenced by noisy or misclassified training samples (Israel et al., 2024).

The SVM classifier, in contrast, employed a one-versus-one (1V1) strategy for multi-class classification, wherein a separate binary classifier was trained for each possible pair of classes. This approach, though computationally intensive, allows the SVM to handle multiple classes effectively by constructing numerous hyperplanes, each optimized to separate two distinct classes. A linear kernel was used, which attempts to find an optimal hyperplane that maximizes the margin between data points from different classes. However, this linear approach can struggle with non-linearly separable data, potentially limiting classification accuracy in more complex datasets (Israel et al., 2024).

The performance of the classifiers was evaluated using standard metrics, including accuracy, precision, recall, and F1-score, calculated from the confusion matrix. The study reported the following results: with LBP features, K-NN achieved 100% accuracy, while multi-class SVM achieved 89.44% accuracy. In contrast, with PCA features, K-NN achieved 86.38% accuracy, and multi-class SVM achieved 85.83% accuracy. These results

indicate that the K-NN algorithm significantly outperformed the multi-class SVM when paired with LBP features, achieving perfect classification accuracy. This performance advantage likely reflects LBP's superior ability to capture fine-grained local texture details, compared to PCA's dimensionality reduction approach, which may inadvertently discard critical discriminative features (Israel et al., 2024).

Despite these promising results, the reliance on LBP and PCA means the classifier's performance is heavily dependent on the chosen feature set, which may not capture all relevant fruit characteristics. The one-versus-one SVM strategy may require training multiple binary classifiers, significantly increasing computational requirements for large, multi-class datasets (Israel et al., 2024).

Decision tree algorithm. [8] Han (2023) used the Decision Tree algorithm as one of three supervised learning methods to perform image classification on a curated subset of the Fruit-360 dataset. This dataset consists of 90,483 images representing 131 classes of fruits and vegetables, each with a resolution of 100×100 pixels. To optimize processing time and focus the comparative evaluation, only 10 fruit classes were selected for the classification task.

Images were preprocessed by resizing to a uniform scale and generating histograms. The dataset was split into an 80:20 training-to-testing ratio, and the classification labels were preserved, making this a supervised learning setup. Image data were loaded into Python using custom scripts that extracted class labels from folder names (Han, 2023).

The Decision Tree model used in this study was applied in its supervised form. The training process constructs the tree using known class labels to guide splits, enabling it to

generalize classification rules based on the training images. The algorithm then traverses this structure during prediction, routing new images through branches of the tree based on their feature values to arrive at a classification outcome (Han, 2023).

The results of the Decision Tree classifier showed strong classification performance on the selected dataset. The model achieved an overall accuracy of 94 percent. The macro average and weighted average F1-scores were both reported as 0.94. Evaluation metrics for individual classes showed that the lowest F1-score was 0.89, while the highest reached 0.97. Class-specific precision and recall values also remained consistently high across the 10 fruit categories. The confusion matrix further confirmed that the predictions were evenly distributed, and no particular class suffered from disproportionately high misclassification. Additionally, the training and testing of the model were computationally efficient, with a total execution time recorded at 25.34 seconds (Han, 2023).

Image noise was identified in the study as a factor that affected the accuracy of classification. The presence of noise in the dataset was attributed to environmental factors such as lighting conditions and hardware-related inconsistencies. In addition, the limited pixel resolution of the images was noted as a constraint that may reduce the classification performance of the Decision Tree algorithm. It was suggested that future research could focus on improving feature extraction methods, specifically by incorporating additional descriptors such as texture and shape (Han, 2023).

TL-MobileNetV2. [9] Fruit classification was conducted using a modified deep learning architecture called TL-MobileNetV2. The model is based on the MobileNetV2 convolutional neural network architecture, which was specifically chosen due to its memory efficiency and suitability for mobile and resource-constrained environments. To

enhance performance, the original classification layer of MobileNetV2 which originally contains 1000 nodes was replaced with a customized head consisting of five layers. These include an average pooling layer with a (7 × 7) size, a flatten layer, a dense layer activated by the ReLU function, a dropout layer set to 0.5 to mitigate overfitting, and a softmax layer containing 40 output nodes corresponding to the 40 fruit classes in the dataset (Gulzar, 2023).

Transfer learning was used to further improve classification performance. The training process followed a hybrid approach: initially, only the newly added layers were trained while the pre-trained layers from MobileNetV2 remained frozen. After the 20th iteration, the previously frozen layers were unfrozen to allow for slight weight adjustment based on the fruit dataset. Model tuning techniques were also applied, including data augmentation to address the limited sample size, adaptive learning rate scheduling to accelerate convergence, model checkpointing to preserve optimal weights, and dropout regularization to prevent overfitting (Gulzar, 2023).

The dataset used for training and evaluation consisted of 26,149 images, which were extracted from 20-second videos taken while fruits were mounted on a rotating shaft at 3 rmp. Backgrounds were algorithmically removed to maintain uniformity. Data augmentation was applied using Keras functions, which generated ten variants per image through random zooming, height and width shifts, and rotation (Gulzar, 2023).

The model was trained over 100 iterations. During training, accuracy increased rapidly from 44 percent to 100 percent by the 30th iteration, where it remained until the final iteration. The corresponding training loss decreased progressively, reaching 0.3 by the end of training. Validation results followed a similar trend: validation accuracy rose to

100% within 10 iterations and remained stable, while validation loss decreased from a high initial value to 0.35 by the 100th iteration (Gulzar, 2023).

On the validation dataset, the TL-MobileNetV2 model achieved precision, recall, and F1-score values of 0.99 for most fruit classes. Slightly lower scores were observed in a few classes such as Apple Golden 1, Apple Red 3, and Banana Lady Finger, which had F1-scores ranging between 0.96 and 0.99. On the testing dataset composed of unseen images, the model achieved an overall accuracy of 99%. Testing results showed that most classes maintained high precision and recall scores, with minor drops in a few classes due to visual similarities among varieties, such as between Golden 1 and Golden 2 or Red 3 and Red 2 (Gulzar, 2023).

The TL-MobileNetV2 model outperformed other well-known CNN architectures such as AlexNet, VGG16, InceptionV3, and ResNet in all evaluation metrics. Specifically, it achieved 99% for precision, recall, and F1-score, compared to 96% accuracy from the unmodified MobileNetV2 and lower results from other baseline models (Gulzar, 2023).

ANN and CNN. [10] The study conducted by Wanthong et al. (2024) aimed to enhance the classification of mangoes (*Mangifera indica* L.), particularly those intended for export, through the application of artificial intelligence (AI) and image processing techniques. The classification process was divided into two key components: mango size categorization and quality assessment, with a focus on identifying anthracnose disease, a common fungal infection affecting export-quality fruit.

To build the dataset, mango images were captured using a Logitech Brio webcam installed 55.5 centimeters above a conveyor belt. Lighting was carefully controlled using dimmable light bulbs and measured using a lux meter, maintaining two lighting conditions

(430–450 lux and 1000–1020 lux) to ensure image consistency. Image resolution was set initially to 640 × 480 pixels, then cropped to 620 × 300 pixels for preprocessing. The Binary Algorithm was employed to convert images into black-and-white representations for simplified pixel analysis and segmentation (Wanthong et al., 2024).

The dataset was divided into training (70%), testing (15%), and validation (15%) subsets, with additional unseen data used for evaluating model generalization. For size classification, the dataset included four categories which are M, 1L, 2L, and 3L where each contained 150 training, 30 test, 30 validation, and 10 unseen samples, resulting in a total of 880 images. For quality classification, the dataset included 176 training, 38 test, 38 validation, and 20 unseen samples for each class (Good vs. Not Good), totaling 544 images (Wanthong et al., 2024).

The ANN architecture used for size classification included four convolutional layers and one fully connected layer. ReLU activation functions were used in the convolutional layers, while the output layer employed a SoftMax function. Dropout was implemented to reduce overfitting. Meanwhile, the CNN model consisted of five convolutional layers and two fully connected layers with similar activation functions and dropout strategies (Wanthong et al., 2024).

Both ANN and CNN models were trained on the same dataset. Classification accuracy of mango sizes was measured using confusion matrices and performance metrics. The ANN model achieved only 25.0% training accuracy and 25.0% test accuracy, indicating poor performance. In contrast, the CNN model achieved significantly better results, with a training accuracy of 99.16% and a test accuracy of 88.0%. The CNN model was also applied to assess mango quality, specifically detecting anthracnose disease.

Training accuracy reached 91.0%, while test accuracy on unseen data was 88.0%. Misclassification was observed in cases where anthracnose symptoms were minor or unclear in the images (Wanthong et al., 2024).

Table 3. Summary of classification performance across different methods for different fruits.

Study and Year	Subject	Method Used	Dataset Size	Performance
[7] 2024	12 fruit classes	KNN	1200 images from Fruit-360 database	LBP-KNN: Accuracy:100% PCA-KNN: Accuracy: 86.38%
[7] 2024	12 fruit classes	Multi-class SVM	1200 images from Fruit-360 database	LBP-SVM: Accuracy: 89.44% PCA-SVM: Accuracy: 85.83%
[8] 2023	10 fruit classes	Decision Tree Algorithm	10 classes from 90,843 images (subset of Fruit-360 database)	Accuracy: 94% Macro Average and Weight Average F1-score: 0.94
[9] 2022	40 fruit types	TL-MobileNetV2	Augmented video-derived images of 40 fruit classes	Training: Accuracy: 100% Validation: Accuracy: 100% Precision: 100% Recall: 100% F1-score: 100% Testing: Accuracy: 99% Precision: 99% Recall: 99% F1-score: 99%

[10] 2024	Mangoes	ANN	880 images (size classification)	Training: Accuracy: 25% Testing: Accuracy: 25%
[10] 2024	Mangoes	CNN	880 images (size classification) and 544 images (quality classification)	Size Classification: Training: Accuracy: 99.16% Testing: Accuracy: 88% Quality Classification: Training: Accuracy: 91% Testing: Accuracy: 88%

Classification methods in bananas

Naive Bayes and SVM. [11] Kosasih et al. (2023) developed a classification system for identifying six banana ripeness levels using statistical features extracted from grayscale images and applied two supervised machine learning algorithms: Naive Bayes and Support Vector Machine (SVM). The research focused on processing banana images to extract first-order statistical features from histograms and using these features as inputs to the classification models.

The classification process began with converting RGB images of bananas into grayscale using the standard weighted formula. Each image was resized to a resolution of 150 × 200 pixels to maintain uniformity during processing. Four histogram-based features which are mean, skewness, energy, and smoothness are extracted from the grayscale

images. These features were selected as they represent first-order statistics that summarize the pixel intensity distribution of the grayscale images (Kosasih et al., 2023).

A total of 45 banana images were used in the study. These were split into 30 training images and 15 testing images. The dataset was labeled according to six predefined ripeness levels. The Naive Bayes classifier used conditional probability derived from the probability density function of each feature per class to classify the images. The SVM model used hyperplane separation in a higher-dimensional space to distinguish between the ripeness categories (Kosasih et al., 2023).

The performance of both models was evaluated using accuracy, precision, recall, and a confusion matrix. According to the results, the Naive Bayes classifier achieved an accuracy of 86.67 percent, with a weighted average precision (WAP) of 83.6 percent and a weighted average recall (WAR) of 86.7 percent. In contrast, the SVM model obtained an accuracy of 80 percent, a WAP of 85.6 percent, and a WAR of 80 percent (Kosasih et al., 2023).

The results indicated that the Naive Bayes model classified a larger number of test samples correctly across multiple ripeness levels compared to the SVM. It was concluded that Naive Bayes outperformed SVM in this specific experimental setup, which involved a small dataset and relied on grayscale statistical features. One of the limitations noted in the study was the limited number of image samples. Increasing the dataset size and exploring the use of deep learning models for future research to potentially improve the classification performance was recommended (Kosasih et al., 2023).

Mask R-CNN. [12] Le et al. (2019) implemented a deep learning-based classification system using Mask Region-Based Convolutional Neural Network (Mask R-

CNN) to classify banana fruit tiers as either normal or reject. The classification was part of a larger objective to develop a noninvasive method for automated postharvest sorting of bananas using a single side-view image, thereby minimizing the need for multi-view imaging and manual measurements.

The system used the Matterport implementation of Mask R-CNN, which integrates a Region Proposal Network (RPN) with a Feature Pyramid Network (FPN) for multi-scale feature extraction and segmentation. The model was initialized using weights pre-trained on the COCO dataset to leverage pre-learned object features. Transfer learning was applied by fine-tuning this pre-trained model with a custom dataset of banana tiers (Le et al., 2019).

The dataset consisted of 194 side-view images of banana fruit tiers, categorized into 139 normal and 55 reject samples. All images were resized to a resolution of 640 × 640 pixels and annotated using VGG Image Annotator (VIA). The annotations, encoded in JSON format, included polygon coordinates for the segmentation masks and bounding box labels for classification (Le et al., 2019).

Two training setups were conducted. In the first, the original 194-image dataset was split into 70% training and 30% testing data, with 5-fold cross-validation to ensure robustness. The second setup involved a data augmentation pipeline, in which the original images were transformed using rotation (±30°), translation (horizontal and vertical shifts), and scaling (±30%). This process increased the training set size by approximately 20 times, providing greater diversity for model learning (Le et al., 2019).

Model training was performed using a batch size of 2 images, with the number of training epochs set to 30. The learning rate was initialized at 0.001, and Stochastic Gradient Descent (SGD) was used as the optimization algorithm with momentum = 0.9 and weight

decay = 0.0001. The model used binary cross-entropy loss for classification, smooth L1 loss for bounding box regression, and binary mask loss for segmentation (Le et al., 2019).

The performance of the classification task was evaluated using average precision (AP) at two Intersection over Union (IoU) thresholds: 0.50 and 0.75. In the first training scenario (without augmentation), the model achieved an average classification accuracy of 97.56% for the normal class and 92.5% for the reject class, resulting in a weighted average accuracy of 96.14%. In the augmented scenario, the classification accuracy for the reject class improved to 93.75%, and the weighted average increased to 96.49%. Both training setups showed stable convergence, with training and validation loss curves stabilizing by the 10th epoch (Le et al., 2019).

It was concluded that the Mask R-CNN model was effective in classifying banana tiers from a single image. However, it was also acknowledged that the limited dataset size posed a constraint. Although data augmentation expanded the training set and improved the reject class accuracy slightly, it was noted that the improvement was not significant. A larger and more diverse real-image dataset was suggested to enhance generalization and performance reliability in real-world applications (Le et al., 2019).

VGG19. [13] Pushpa et al. (2024) utilized the VGG19 architecture to classify banana images into three categories: unripe, ripe, and overripe. The VGG19 model consists of 16 convolutional layers arranged into five blocks, with each block followed by a max pooling layer. The convolutional layers used a 3×3 kernel size, and the number of filters increased progressively from 64 to 512 across the network. The convolutional layers were followed by three fully connected layers, each containing 4096 neurons, and a final output layer responsible for classification.

The dataset used for training and testing consisted of 687 banana images, which included both single bananas and banana bunches. The input images were resized to 255×255 pixels and converted to RGB color space. To enhance model generalization, data augmentation techniques were applied, including rotation, flipping, and zooming. The model was initialized with pre-trained ImageNet weights using transfer learning and fine-tuned on the banana dataset (Pushpa et al., 2024).

The training process was conducted with an 80:20 train-test split, using sparse categorical cross-entropy as the loss function and the Adam optimizer with a learning rate of 0.001. The model was trained for 20 epochs. During training, the loss and accuracy metrics were monitored (Pushpa et al., 2024).

The performance of the VGG19 model was evaluated using classification accuracy and confusion matrix analysis. The model achieved an accuracy of 98.1% on the test set. Specifically, it classified 98% of single banana images and 98.2% of banana bunch images correctly. The confusion matrix showed that 674 out of 687 images were correctly classified, with 12 misclassifications (Pushpa et al., 2024).

InceptionV3. [13] The InceptionV3 model was also used by Pushpa et al. (2024) for the same classification task. This architecture requires input images resized to 299×299 pixels and converted to BGR color space. Like VGG19, InceptionV3 was implemented using transfer learning with pre-trained ImageNet weights, and the model was fine-tuned on the banana dataset.

The same dataset of 687 banana images was used, and the 80:20 train-test split was maintained. The training conditions, including the optimizer and loss function, were

consistent with the VGG19 model: Adam optimizer, learning rate of 0.001, and sparse categorical cross-entropy loss. The model was trained for 20 epochs (Pushpa et al., 2024).

InceptionV3 achieved a classification accuracy of 97.5%. The model correctly classified 97.4% of single banana images and 97.6% of banana bunch images. The confusion matrix indicated 670 correctly classified images and 16 misclassifications (Pushpa et al., 2024).

EfficientNet-B7. [14] Nikhilesh et al. (2023) proposed a deep learning approach to automate banana grading, specifically focusing on the classification of Nendran banana images into four ripeness categories: unripe, semi-ripe, ripe, and overripe. The model utilized for classification was EfficientNet-B7 which began with the collection of a curated dataset containing labeled images of Nendran bananas at various ripeness stages, including differences in size and the presence of external flaws. Each image was annotated by experts to ensure accurate labeling of the four ripeness classes.

The dataset was preprocessed using scaling and normalization techniques and split into training and validation sets. The EfficientNet-B7 model was initialized with pre-trained weights from the ImageNet dataset, and transfer learning was employed to adapt the model to the banana grading task. The final classification layers of the network were modified to accommodate the four output classes specific to Nendran banana ripeness (Nikhilesh et al., 2023).

To improve the model's generalization performance, data augmentation techniques were applied during training. These included random cropping, rotation, and flipping, which helped introduce variability and mitigate overfitting by exposing the model to diverse image conditions (Nikhilesh et al., 2023).

The model was trained on batches of augmented banana images using standard optimization procedures, although specific hyperparameter values (e.g., learning rate or batch size) were not detailed in the paper. Hyperparameter tuning was performed to determine the optimal configuration for classification (Nikhilesh et al., 2023).

The performance of the model was evaluated using accuracy, precision, recall, and F1-score on a validation set. The EfficientNet-B7 model achieved an overall classification accuracy of 97%. For the overripe category, the model obtained a precision of 0.97, recall of 0.95, and an F1-score of 0.96 based on 41 test images. For ripe bananas, the model achieved a precision of 0.94, recall of 0.97, and an F1-score of 0.96 using 35 images. In the semi-ripe class, the model recorded a precision of 1.00, recall of 0.92, and an F1-score of 0.96 across 13 samples. Finally, for the unripe class, the model achieved a precision of 0.97, recall of 1.00, and an F1-score of 0.99 from 39 images. These results demonstrate consistent classification performance across all ripeness categories (Nikhilesh et al., 2023).

Table 4. Summary of classification performance across different methods for banana ripeness classification.

Study and Year	Method Used	Dataset Size	Ripeness Annotation Type	Performance
[11] 2023	Naive Bayes	45 images	Tier-based (6 ripeness stages)	Accuracy: 86.67% WAP: 83.6% WAR: 86.7%
[11] 2023	SVM	45 images	Tier-based (6 ripeness stages)	Accuracy: 80% WAP: 85.6% WAR: 80%

[12] 2019	Mask R-CNN	194 images (increased by 20 times using data augmentation)	Tier-based (Normal/ Reject)	Without Augmentation: Weighted Average Accuracy: 96.14% Average Accuracy (normal class): 97.56% Average Accuracy (reject class): 92.5% With Augmentation: Weighted Average Accuracy: 96.49% Average Accuracy (normal class): 97.56% Average Accuracy (reject class): 93.75%
[13] 2024	VGG19	687 images	Tier-based (Unripe, Ripe, Overripe)	Accuracy: 98.1%
[13] 2024	InceptionV3	687 images	Tier-based (Unripe, Ripe, Overripe)	Accuracy: 97.5%

[14] 2023	EfficientNet-B7	Not specified (per-class counts provided)	Tier-based (Unripe, Semi-ripe, Ripe, Overripe)	Accuracy:	0.97
				Overripe:	
				Precision:	0.97
				Recall:	0.95
				F1-score:	0.96
				Ripe:	
				Precision:	0.94
				Recall:	0.97
				F1-score:	0.96
				Semi-ripe:	
				Precision:	1.00
				Recall:	0.92
				F1-score:	0.96
				Unripe:	
				Precision:	0.97
				Recall:	1.00
				F1-score:	0.99

Evolution of YOLO Models

The You Only Look Once (YOLO) framework is a family of single-stage object detectors that have become widely known for their ability to perform real-time object detection with a high balance of speed and accuracy. Unlike two-stage detectors that separate region proposal and classification steps, YOLO treats detection as a regression problem, predicting bounding boxes and class probabilities directly from full images in a single forward pass. This design significantly reduces computational complexity, making YOLO suitable for practical applications like surveillance, medical imaging, autonomous vehicles, and robotics. Since its first release in 2015, YOLO has undergone multiple iterations, each introducing architectural innovations, optimization techniques, and improved training strategies to address challenges such as detecting small or overlapping

objects, adapting to various domains, and supporting segmentation tasks (Ali & Zhang, 2024).

YOLOv1, the original version, was known for its fast and real-time detection performance due to its simplified CNN backbone and direct bounding box prediction method. However, it used a basic training strategy involving geometric transformations and hue jitter and relied on Mean Squared Error (MSE) for bounding box regression and Binary Cross Entropy (BCE) for classification. While efficient, its detection accuracy was significantly lower than that of two-stage detectors (Ali & Zhang, 2024).

YOLOv2 improved performance by using a deeper DarkNet-19 backbone and introducing anchor boxes with K-means clustering for better bounding box prediction. Training included fine-tuning and pre-trained weights, and optimization was done using SGD with momentum and hyperparameter tuning. Despite these changes, YOLOv2 still faced accuracy limitations (Ali & Zhang, 2024).

YOLOv3 adopted a more complex backbone, DarkNet-53, and implemented multi-scale detection and residual connections. Training involved mix-up, noise, and standard data augmentation, and the loss functions included Generalized IoU (GIoU) or Distance IoU (DIoU) for bounding boxes and Cross Entropy (CE) for classification. The accuracy improved, but the architecture became more complex and slower (Ali & Zhang, 2024).

YOLOv4 introduced PANet for better feature fusion, CSPDarkNet-53 as the backbone, and techniques like mosaic data augmentation. It used CIoU for bounding boxes and BCE or Focal Loss for classification. YOLOv4 became more flexible due to its adaptability with different heads and backbones, although this made it less user-friendly for practical use (Ali & Zhang, 2024).

YOLOv5 focused on a lighter architecture using CSPNet and dynamic anchor refinement. It employed early stopping, Mosaic, and CutMix for training, while optimization involved post-training quantization and filter pruning. Loss functions included CIoU and Focal Loss, making this version faster and more accurate than YOLOv3, though it was less adaptable than YOLOv4 (Ali & Zhang, 2024).

YOLOv6 and **YOLOv7** continued the trend of improving speed and reducing computational demands. YOLOv6 used PANet and CSPDarkNet53, introduced domain-specific augmentation, and used VariFocal Loss. YOLOv7 incorporated dynamic label assignment and adversarial patch detection and introduced NAS (Neural Architecture Search) with quantization. Although both achieved accuracy and speed improvements, YOLOv7 had higher complexity and risk of overfitting (Ali & Zhang, 2024).

YOLOv8 used Dynamic Kernel Attention and a Path Aggregation Network (PAN). It retained CIoU and added Distributed Focal Loss (DFL) and used adversarial training with post-training quantization. While lighter and faster than YOLOv7, its optimizations were highly specific to certain use cases (Ali & Zhang, 2024).

YOLOv9 targeted real-time use cases by reducing model size. It introduced a multi-level auxiliary feature extraction system and used L1 loss for bounding boxes. It was ideal for resource-constrained environments but was criticized for focusing too narrowly on specific objects (Ali & Zhang, 2024).

YOLOv10 removed non-maximum suppression (NMS) during training and focused on separating spatial and channel transformations. It used a dual label assignment strategy and relied on coordinate and confidence loss. Its distinct architecture aimed to boost downsampling efficiency (Ali & Zhang, 2024).

Finally, **YOLOv11** brought significant innovations. It introduced the C3k2 block in the backbone for better multi-scale feature extraction and C2PSA (Channel and Spatial Attention) to strengthen attention mechanisms. YOLOv11 used mix-up and adaptive gradient clipping in training and was optimized through stochastic gradient descent (SGD) and quantization. For loss, it employed an IoU-based loss for bounding box regression and combined BCE, Focal Loss, and CloU for classification. These changes collectively enhanced object detection and set the stage for improved instance segmentation and object delineation (Ali & Zhang, 2024).

YOLO-based Ripeness Classification in Bananas

YOLOv5 model ripeness classification

[15] Cahya et al. (2023) developed a deep learning-based classification system using the YOLOv5 object detection architecture to classify the ripeness levels of multiple Cavendish bananas within a single image. It aimed to address the limitations of previous approaches that focused on single-object classification and did not reflect the practical scenario in which bananas are commonly sold and distributed in bunches.

To construct the dataset, a total of 600 high-resolution images of Cavendish bananas at different ripeness stages were collected using an automated storage testbed that recorded environmental parameters. The bananas were photographed during their natural ripening process under standardized lighting and background conditions. Each image contained four bananas, and all were manually annotated with bounding boxes and labeled according to their ripeness stage. Five ripeness categories were defined: stage 4, stage 5,

stage 6, stage 7, and overripen. Due to logistical constraints in sample acquisition, the dataset did not include stages 1 to 3 (Cahya et al., 2023).

The images were augmented using several transformations to improve the generalization capability of the model. The applied augmentations included horizontal and vertical flips, rotations between ±20°, shearing up to ±21°, brightness variation of ±10%, and mosaic augmentation. The augmented dataset was expanded to 2,400 images and resized to 640×640 pixels to match the input requirement of the YOLOv5 model (Cahya et al., 2023).

The YOLOv5 architecture used comprises three main components: a backbone for feature extraction, a neck for aggregating multi-scale features, and a head for performing bounding box regression and class prediction. The model was trained using COCO-pretrained weights and optimized over 200 epochs. The dataset was split into 85% for training and 15% for validation (Cahya et al., 2023).

Model performance was evaluated using standard object detection metrics: mean Average Precision (mAP), precision, recall, and class-wise average precision. The trained model achieved a mAP of 98.8%, precision of 90.5%, and recall of 92.6% on the validation set. The class-wise average precision scores were as follows: 98.4% for stage 4, 98.7% for stage 5, 99.1% for stage 6, 100% for stage 7, and 98.7% for the overripen category. Loss curves for classification loss, objectness loss, and box loss all showed decreasing trends, indicating proper convergence during training (Cahya et al., 2023).

Several limitations were identified and first is the dataset was restricted to ripeness stages 4 to overripen due to difficulties in sourcing early-stage bananas during distribution. This limitation affects the generalizability of the model across the full ripeness spectrum.

Second, the model was trained using data from a single-camera setup under uniform lighting, limiting its robustness to real-world variations in background, lighting, and camera angle. Third, the system faced challenges in handling partially visible bananas, a common scenario in commercial settings. It was suggested that incorporating additional segmentation techniques such as Segment Anything Model (SAM) or CNN-Mask models could improve object localization and ripeness prediction accuracy. They also proposed that future enhancements could involve integrating environmental parameters as additional input features to track ripening trends and enable more accurate predictions over time (Cahya et al., 2023).

YOLOv8 model ripeness classification

Two recent studies have explored the use of YOLOv8 for banana ripeness detection and classification, highlighting the model's capacity to support real-time and edge-computing applications.

[16] In the study conducted by Baldovino et al. (2024), the researchers developed a banana ripeness detection system using the YOLOv8-Small (YOLOv8-S) model, integrated into a webcam-based real-time detection environment. The training data was sourced from the "Fruit Ripening Process Dataset," which contained 7,546 images with a total of 15,053 bounding box annotations. These annotations represented six ripeness stages: unripe, ripe, overripe, rotten, fresh-ripe, and fresh-unripe. The images were labeled and processed for use in YOLO format and resized to 640×640 pixels. Training was conducted using Google Colab for 25 epochs, with the YOLOv8-S variant selected for its balance between accuracy and computational efficiency. After training, the model was

exported and deployed locally using a Python script with OpenCV integration for webcam-based classification.

Performance evaluation was conducted through both static image testing and real-time webcam inference. The model achieved a classification accuracy of 89.5% on the static test dataset, while real-time detection accuracy reached approximately 80%. Confusion matrix analysis revealed that the model had difficulties in distinguishing between similar ripeness stages, particularly ripe and overripe classes, and occasional misclassification of background objects as bananas. In individual sample tests, the model correctly classified all ripe bananas but had some difficulty with unripe and overripe samples. The authors identified that the performance degradation in real-world conditions was influenced by background noise and sample overlap. They recommended further fine-tuning of model hyperparameters and enhancement of the training dataset to improve generalizability (Baldovino et al., 2024).

[17] In a separate work, Aishwarya and Kumar (2023) evaluated five YOLOv8 variants which are YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x on a banana ripeness classification task using a larger dataset of 18,074 images divided into six ripeness categories. The dataset was split into training (15,792 images), validation (1,525 images), and testing (757 images). All models were trained for 100 epochs using transfer learning on an NVIDIA Tesla V100 GPU. Each YOLOv8 model was evaluated using standard metrics: Precision, Recall, F1-Score, and mean Average Precision at 0.5 IoU (mAP@0.5). The YOLOv8x variant achieved the highest mAP@0.5 of 95.4%, while YOLOv8s demonstrated the best balance between accuracy and real-time feasibility with a mAP@0.5 of 95.4%, precision of 90.1%, recall of 89.2%, and F1-score of 90%.

YOLOv8s processed single images in 13.9 milliseconds on the NVIDIA Jetson Xavier AGX, confirming its suitability for edge deployment. The authors observed that all five models performed consistently across all ripeness classes, even under visually cluttered backgrounds.

Both studies affirmed YOLOv8's effectiveness for banana ripeness classification but differed in application scope. Baldovino et al. (2024) focused on lightweight deployment in webcam-based setups using YOLOv8-S, while Aishwarya and Kumar (2023) provided a broader comparative analysis of YOLOv8 variants and emphasized performance optimization for edge devices. The latter also included detailed per-class evaluation and latency benchmarks, reinforcing the operational flexibility of YOLOv8 models (Aishwarya and Kumar, 2023).

ESD-YOLO9vc model ripeness classification

[18] Wang et al. (2025) proposed a banana ripeness detection framework based on an enhanced version of the YOLOv9c architecture, referred to as ESD-YOLOv9. The model was specifically developed to handle complex, multifactorial environments commonly encountered in banana harvesting, storage, and distribution. The classification system was trained to identify four ripeness categories—unripe, ripe, overripe, and rotten—across two banana varieties (Golden Finger Banana and Small Banana). A total of 10,115 images were used, which included various conditions such as different lighting intensities, rotations, noise injections, and image distortions. The dataset was split into a training set (8092 images) and a test set (2023 images), and annotation was carried out using LabelImg, with each banana labeled using a bounding box and corresponding ripeness label.

The improved classification architecture consisted of several enhancements to the original YOLOv9c model. These included the SENetV2 attention mechanism, which enhanced feature extraction by strengthening the model's sensitivity to maturity-related visual cues; DualConv, a grouped convolution module that replaced standard downsampling to maintain spatial precision while reducing computational cost; and the EIoU loss function, which improved bounding box regression accuracy and convergence speed by optimizing center-point alignment, width, and height differences between predictions and ground truth. The final architecture incorporated all three modules and was trained without using pretrained weights, applying a stochastic gradient descent optimizer with a learning rate of 0.01 across 120 epochs (Wang et al., 2025).

The classification results revealed that the ESD-YOLOv9 model achieved a mean Average Precision (mAP) of 98.9% at IoU=0.5 and mAP50–95 of 95.9%. Class-specific performance metrics included a precision of 94.0%, recall of 98.8%, and F1-score of 96.3%. For each ripeness class, the model recorded high detection accuracy: unripe (mAP50: 98.1%), ripe (99.4%), overripe (99.2%), and rotten (98.8%). Compared to several state-of-the-art models including YOLOv5X, YOLOv7, YOLOv8n, YOLOv10n, RT-DETR, Faster-RCNN, and SSD, the ESD-YOLOv9 model consistently outperformed them in terms of mAP and F1-score, while also maintaining competitive inference speed (Wang et al., 2025).

While the ESD-YOLOv9 demonstrated high classification performance, the study also identified a number of limitations. The model size increased by 1.8 MB, and the number of parameters rose by 3.556 million, which could affect real-time deployment on low-power edge devices. Furthermore, the study was conducted on a cloud-based system,

and deployment compatibility in physical hardware environments remains to be validated. Additionally, the dataset included only two banana varieties, which may constrain the model's generalizability across broader banana types. The need for future work to include a wider range of banana cultivars, optimize the model for lightweight devices, and assess real-world performance beyond the controlled experimental setting was acknowledged (Wang et al., 2025).

Table 5. Summary of classification performance across different YOLO models for banana ripeness detection

Study and Year	Method Used	Dataset Size	Ripeness Annotation Type	Performance
[15] 2023	YOLOv5	600 (expanded to 2,400 through augmentation)	Tier-based (ripeness stage 4 to overripe)	mAP:98.8% Precision: 90.5% Recall: 92.6%
[16] 2024	YOLOv8-S	7,546 images/ 15,053 annotations	Tier-based (6 ripeness stages)	Static: Accuracy: 89.5% Real-time: Accuracy: ~80%
[17] 2023	YOLOv8 Variants (n, s, m, l, x)	18,074 images	Tier-based (6 ripeness stages)	YOLOv8n: Precision: 88.8% Recall: 90.5% F1-score: 89% mAP@0.5: 95.2% YOLOv8s: Precision: 90.1% Recall: 89.2% F1-score: 90% mAP@0.5: 95.4% YOLOv8m: Precision: 87.2%

				Recall: 91.7% F1-score: 89% mAP@0.5: 95.3% YOLOv8l: Precision: 87.3% Recall: 90.7% F1-score: 89% mAP@0.5: 95.3% YOLOv8x: Precision: 87.8% Recall: 92% F1-score: 90% mAP@0.5: 95.4%
[18] 2025	ESD-YOLOV9	10,115 images	Tier-based (4 ripeness stages)	mAP@0.5: 98.9%, mAP@0.5-0.95: 95.9% F1-score: 96.3%

Evaluation Metrics

The evaluation of object detection and segmentation models relies on specific performance metrics that quantify prediction accuracy. According to Vlăsceanu et al. (2021), these metrics allow researchers to assess the quality of model outputs based on how well they match the ground truth annotations. The most used evaluation metrics include Intersection over Union (IoU), Precision, Recall, F1-score, and mean Average Precision (mAP) are defined and explained as follows:

Intersection over Union (IoU). It is a standard metric used to assess the localization performance of object detection models. It measures the degree of overlap between a predicted bounding box and the corresponding ground truth box.

Mathematically, IoU is defined as the ratio of the area of intersection to the area of union of the two bounding boxes with the following formula shown in Equation 2:

$$IoU = \frac{Area\,(Bp \cap Bgt)}{Area\,(Bp\ Bgt)} \quad [2]$$

Higher IoU values indicate more accurate localization, with a value of 1 representing perfect alignment between prediction and ground truth (Shah, 2023).

Confusion Matrix. A tabular representation used to evaluate the performance of a classification model by comparing predicted and actual class labels. It is particularly useful for multi-class classification problems, such as ripeness classification of banana fingers. The matrix displays four key outcomes: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). These values are arranged in a square matrix format, where rows typically represent actual classes and columns represent predicted classes.

		Actual Values	
		Positive	**Negative**
Predicted Values	**Positive**	True Positive (TP)	False Positive (Type I error)
	Negative	False Negative (Type II error)	True Negative

Fig 1. Confusion Matrix (Lifted from Sathyanarayanan, 2024)

The confusion matrix provides insight not only into the number of correct predictions but also into the specific types of errors made by the model, which is particularly important in cases where certain misclassifications may carry varying levels of significance.

Precision. It is a metric in object detection that measures the proportion of correctly predicted positive instances out of all instances predicted as positive. It reflects the model's

ability to minimize false positives, or in other words, how many of the detected objects are actually relevant. In the context of object detection, this means evaluating how many of the predicted bounding boxes correspond accurately to actual objects of interest (Kundu, 2022).

Mathematically, precision is defined in Equation 3:

$$Precision = \frac{TP}{TP + FP} \quad [3]$$

where:

TP (True Positives) refers to correctly predicted objects.

FP (False Positives) refers to predicted objects that do not match any ground truth.

Recall. It is the ratio of correctly predicted positive instances to all actual positives.

$$Recall = \frac{TP}{TP + FN} \quad [4]$$

where:

FN refers to false negatives.

The metric reflects the model's ability to detect all relevant regions. It is important for evaluating how well a model identifies the complete object or region of interest (Kundu, 2022; Vlăsceanu et al., 2024).

F1 score. It is a performance metric that combines precision and recall into a single value by calculating their harmonic mean. It provides a balanced measure of a model's accuracy, especially when an uneven distribution of classes exists or when both false positives and false negatives carry significant weight. The F1 score is particularly useful for evaluating classification models that require a balance between identifying all relevant instances and minimizing incorrect positive predictions (Kundu, 2022).

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad [5]$$

Mean Average Precision (mAP). Average Precision (AP) summarizes the precision-recall curve by calculating the area under the curve for a single class. The mean Average Precision (mAP) is then computed by averaging the AP across all classes:

$$mAP = = \frac{1}{N} \sum_{i=1}^{N} APi \quad [6]$$

where:

N is the number of classes

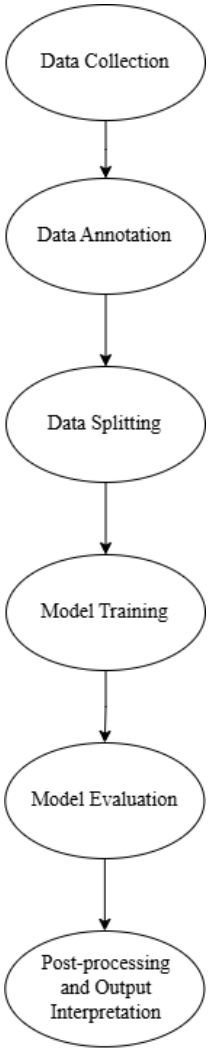
APi is the average precision for class i

mAP is often calculated over multiple IoU thresholds (e.g., 0.5 to 0.95 with 0.05 increments), offering a more reliable evaluation of a model's ability to accurately detect and classify objects under various spatial alignment criteria (Vlăsceanu et al., 2024).

MATERIALS AND METHODS

General Framework of the Study

This study will adopt a supervised, single-stage object detection approach to perform finger-level segmentation and ripeness classification of Cavendish bananas. Specifically, YOLOv11 model will be utilized as the framework which is chosen for its ability to simultaneously localize banana fingers and classify each finger's ripeness stage in one forward pass. The overall methodology covers dataset preparation, model architecture and configuration, training procedures, evaluation metrics, and post-processing of detection outputs. Emphasis is placed on maintaining a controlled experimental setup and reproducible training pipeline to ensure the credibility of results. In the following sections, the image dataset collection and annotation process are detailed, followed by a description of the YOLOv11 architecture improvements and training configuration. Finally, the training process, hardware environment, evaluation metrics, and output post-processing techniques are discussed.



on Figure 1. Proposed General Framework of the Study

Dataset Preparati

Data collection

RGB images of Cavendish bananas were collected in postharvest conditions using a controlled imaging setup. All samples were photographed under consistent conditions to ensure uniform data quality. A digital camera was mounted at a fixed height and distance from the banana samples, with uniform lighting and a neutral backdrop (plain white background) to minimize shadows and facilitate image segmentation. The camera was

positioned such that both front and back sides of banana hands were captured by turning the fruit, ensuring that multiple angles of each banana finger were obtained. Images were taken from various viewpoints (top, side, and oblique angles). This approach maximizes the coverage of each finger and mimics real-world postharvest inspection where bananas are viewed from different angles. The image resolution of the collected photos was set to a high definition (1920×1080 pixels) to capture fine texture details of banana peels (Mesa and Chiang, 2021). In some cases, images were downsampled to the model's input size during preprocessing; for YOLO-based models, a typical input resolution is a 640×640 pixel square, which balances detail with computational efficiency. This controlled acquisition procedure yields a comprehensive image dataset of Cavendish banana bunches, with clear visualization of individual fingers across the unripe, ripe, and overripe stages of ripeness.

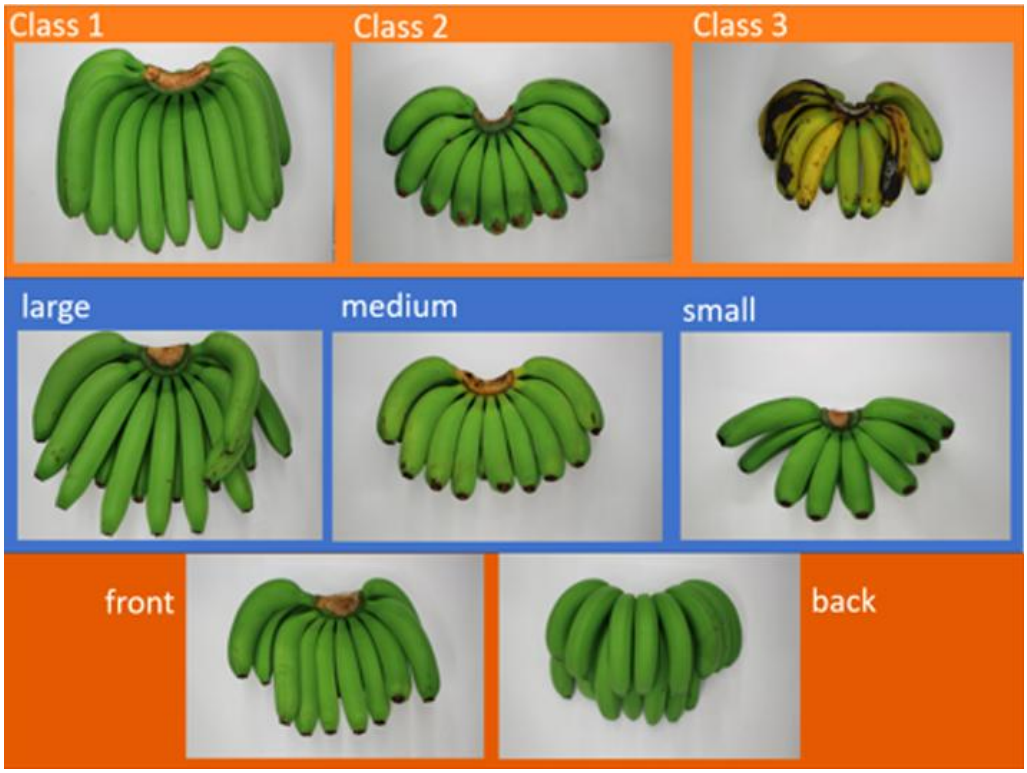


Fig 2. Collected Cavendish banana dataset (Lifted from Mesa and Chiang, 2021).

Data annotation

After image collection, annotation will be performed to both delineate each individual banana finger and assign its corresponding ripeness class label. Manual annotation will be carried out using Roboflow, which supports instance segmentation. For each banana finger, a tight polygon mask will be manually drawn to trace the visible contour of the fruit, capturing its shape at the pixel level. Alongside the mask, a bounding box will also be automatically generated to enclose the polygon, as required by YOLOv11's segmentation format.

Each image may contain multiple banana fingers, and each will be annotated as a separate instance, regardless of overlap, to ensure accurate per-finger segmentation and classification. Banana fingers located at the edge of the image will be annotated if the visible portion is sufficient for ripeness labeling. Conversely, any fruit that is severely occluded or blurred to the extent that ripeness cannot be confidently determined will be excluded to minimize label noise.

Each segmented instance will be assigned one of three ripeness classes which are unripe, ripe, or overripe based on peel coloration and texture. The annotations will be exported in YOLOv11-compatible instance segmentation format, which includes class labels, bounding box coordinates, and segmentation polygons in normalized coordinates. This structured annotation approach ensures compatibility with YOLOv11's segment task and enables the model to learn both spatial localization and detailed shape features necessary for ripeness classification at the instance level.

Data splitting

The annotated dataset will be divided into training, validation, and testing subsets to support supervised learning and model evaluation. A 70%–20%–10% split will be employed: 70% of the images and their annotations will be allocated to the training set, 20% to the validation set used for hyperparameter tuning and early stopping checks, and the remaining 10% to the test set for final performance evaluation. This split ensures that the model is trained on the majority of data while holding out a portion for unbiased evaluation of generalization. The distribution of ripeness classes will be kept approximately stratified across these subsets where each set contains a similar proportion of unripe, ripe, and overripe instances to prevent class imbalance issues during training and testing.

Various data augmentation techniques will be applied to the training images to increase the model reliability and reduce overfitting. The augmentation techniques to be used are shown in Table 6.

Table 6. Data augmentation techniques applied to the collected dataset.

Data Augmentation Technique	Parameter Values/Description
Rotation	±15°
Horizontal Flip	True
Vertical Flip	True
Brightness Adjustment	Random brightness and contrast adjustments will be applied to simulate varying lighting conditions. The pixel intensities will be scaled or offset within a safe range to avoid color distortion.
Resizing	±10%
Noise Injection	Some training images will have a small amount of random noise added or slight blur, to mimic sensor noise or motion blur.

Mosaic Augmentation	Four images will be randomly combined into one mosaic image during training.

Through these augmentation techniques, the original training set will be expanded several-fold, and the model will be exposed to a wide range of geometric and photometric variations of banana appearances. In total, the training dataset increased from N original images to over M images after augmentation (where $M \gg N$ due to multiple augmentations per original image). The validation and test sets will be not augmented, as they represent real-world data for unbiased performance assessment. This augmented training set helps ensure the YOLOv11 model learns features that generalize well to new banana images.

Model Selection

YOLOv11 architecture

For banana finger segmentation and ripeness classification, the YOLOv11 object detection architecture will be employed. YOLOv11 is one of the latest generations in the "You Only Look Once" family of single-stage detectors, designed for high accuracy and speed in real-time computer vision tasks. Like its predecessors, YOLOv11 follows the one-stage detection paradigm that directly predicts bounding boxes and class probabilities in one unified network pass, rather than using a two-stage proposal-refinement approach. This unified design means the model has a single neural network that transforms input images into output detections (locations and classes) in an end-to-end differentiable manner. By integrating localization and classification in one stage, YOLO models can be trained end-to-end and achieve fast inference (Khanam and Hussain, 2024).

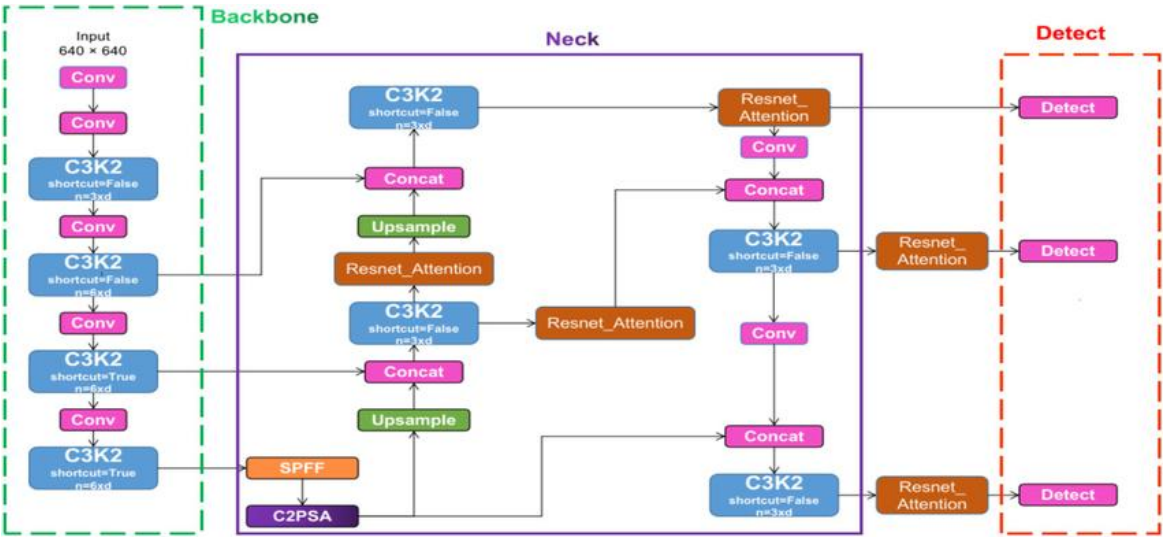


Fig 3. YOLOv11 architecture (Lifted from Hameed et al., 2025).

Backbone. YOLOv11's backbone is a deep convolutional neural network responsible for feature extraction from input images. The backbone takes the raw image (resized to a fixed resolution) and produces a hierarchy of feature maps at multiple scales. YOLOv11 introduces an improved C3k2 module in its backbone in place of the earlier CSP (Cross Stage Partial) bottlenecks. The C3k2 block divides a standard convolution into two smaller consecutive convolutions (hence "k2"), which achieve the same receptive field with fewer parameters and less computation. This design increases the backbone's efficiency by using two small-kernel convolutions instead of one large kernel, thereby speeding up processing while preserving feature extraction capability. The prefix "C3" indicates that this module is a variant of the CSP concept with three convolutional layers internally (one for downsampling and two for bottlenecking) optimized for faster execution. In essence, the YOLOv11 backbone is leaner and faster due to C3k2 blocks, yet it retains strong representational power to capture banana shape, color, and texture features relevant to ripeness (Khanam and Hussain, 2024).

Additionally, YOLOv11 incorporates an attention mechanism in the backbone to enhance significant features. After the initial convolutional layers and an SPPF (Spatial Pyramid Pooling-Fast) block (which pools features at multiple scales), YOLOv11 adds a novel C2PSA module. C2PSA stands for Cross Stage Partial with Spatial Attention, a block that applies spatial attention to the feature maps. The C2PSA module enables the network to focus on important regions in the image by weighting spatial locations that likely contain objects of interest (bananas) while suppressing background regions. By pooling and re-weighting feature maps spatially, C2PSA highlights banana-specific features such as the shape outline or color patches corresponding to certain ripeness, improving the detector's sensitivity to those areas. This attention mechanism is particularly useful for detecting smaller or partially occluded bananas, as it guides the model to concentrate on the relevant parts of the scene. The inclusion of C2PSA distinguishes YOLOv11 from earlier versions like YOLOv8, which lack this spatial attention module. The backbone's combination of C3k2 modules and the C2PSA attention block yields a stronger feature representation for the subsequent network stages, contributing to higher accuracy without sacrificing speed (Khanam and Hussain, 2024).

Neck. The neck of YOLOv11 is an intermediate set of layers that takes the multi-scale feature maps from the backbone and fuses them to prepare for object prediction. It typically involves upsampling and concatenation operations often referred to as a feature pyramid or PANe to combine high-level semantic information with lower-level fine details. YOLOv11's neck also leverages the efficient C3k2 blocks for feature aggregation. In the neck, the model upsamples higher-level features and merges them with lower-level features (skip connections from the backbone) to ensure that both large and small bananas can be

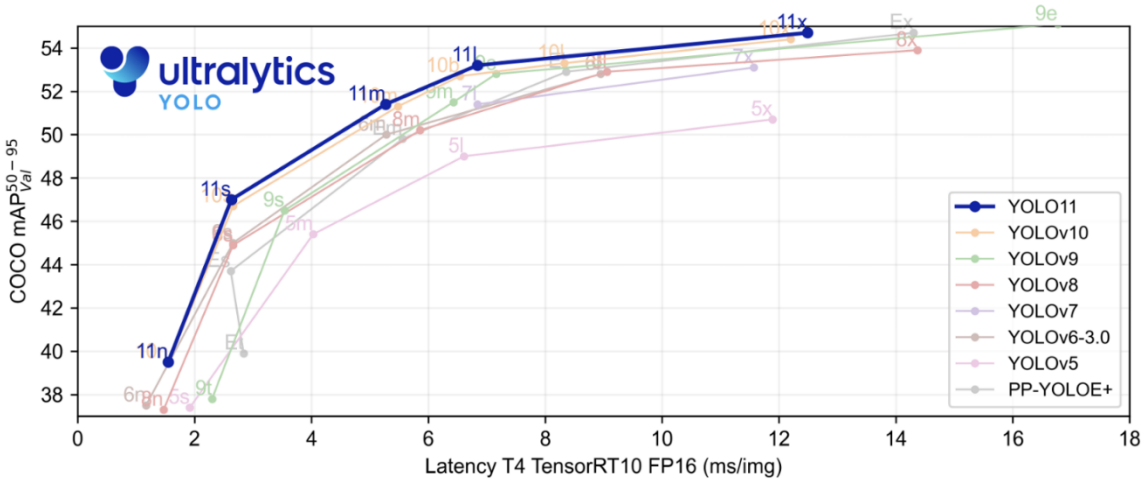
detected. After merging, a C3k2 block processes these fused features, which streamlines the neck computations and speeds up the feature fusion process. The neck may also include additional attention or normalization layers to further refine the combined features, but the key innovation remains the use of the faster C3k2 modules. By the end of the neck, YOLOv11 has prepared a set of rich feature maps at different scales, each ready to be fed to the detection head for final predictions (Khanam and Hussain, 2024).

Detection Head. The head is the final component of YOLOv11 that produces bounding box coordinates, objectness scores, and class probabilities for each detection. YOLOv11's head follows the YOLO family's grid-based prediction approach: it projects the processed feature maps onto a grid and predicts multiple bounding boxes per grid cell, each with a confidence score and class label. In YOLOv11, multiple C3k2 blocks are deployed within the head to progressively refine features before the final outputs. These repeated C3k2 modules in the head allow the model to handle multi-scale predictions, ensuring that bananas of different sizes (from close-up single fingers to smaller distant ones) are all detected. The head is configured to predict one bounding box per banana instance along with a ripeness class. Since we have three ripeness classes, the head outputs three class probabilities for each predicted box (one for each class, e.g., P(unripe), P(ripe), P(overripe)). YOLOv11 retains the mechanism of using anchor boxes or anchor-free layouts similar to YOLOv8. In our configuration, YOLOv11 was set in anchor-free mode (automatically determining bounding box priors), simplifying the detection of banana fingers which tend to have consistent aspect ratios (Khanam and Hussain, 2024).

Importantly, YOLOv11 extends its capabilities beyond object detection to include instance segmentation support. YOLOv11's architecture is inherently capable of producing

segmentation outputs (object masks). This is facilitated by the multi-task design of the network and the rich feature maps it generates. Segmentation capability implies the model is learning fine-grained pixel-level features, which can indirectly benefit the accuracy of bounding box localization (Khanam and Hussain, 2024).

Overall, the YOLOv11 architecture is well-suited to the problem for its accuracy, speed, and its ability to perform detection and classification in one stage. These characteristics ensure that banana ripeness can be determined at the individual finger level quickly and reliably (Khanam and Hussain, 2024).



Model configuration

Table 7. Configuration parameters.

Configuration Parameter	Setting	Description
Input image size	640 × 640	All training and validation images will be resized to 640×640 pixels before being fed into the network. This size provides a balance between preserving detail in small objects and maintaining computational efficiency.

Batch size	16	A batch of 16 images were used during training. This was optimal for the available 6 GB VRAM, ensuring smooth training while preserving gradient stability.
Epochs	100	The model will be trained for 100 complete passes over the dataset to allow the model to converge reliably in preliminary tests.
Early stopping (patience)	10	Training will be terminated early if validation loss fails to improve after 10 consecutive epochs. However, the model will still complete all 100 epochs during actual training.
Optimizer	auto (Adam selected)	The YOLOv11 framework's 'auto' optimizer setting selected Adam with momentum, suitable for fast convergence in object detection tasks.
Learning rate	0.001	The initial learning rate will be 0.001 with cosine decay scheduling applied to gradually reduce the learning rate to fine-tune the model in later epochs.
Number of classes	3	The classification head will be configured to output scores for three ripeness stages: unripe, ripe, and overripe.
Loss functions	IoU-based for localization; BCE for classification	YOLOv11 uses composite loss: an IoU-based loss (e.g., CIOU or EIOU) for bounding box accuracy, and binary cross-entropy loss for class probabilities.
Precision mode	Mixed precision	Automatic mixed-precision training will be enabled through CUDA AMP to improve training speed and reduce memory load.
Default augmentations	Enabled	YOLOv11's built-in augmentations will be used alongside custom augmentations specific to this study.

This configuration follows YOLOv11's configuration defaults and builds on them with specific parameters designed for banana ripeness detection. Hyperparameter settings such as image size, batch size, and learning rate were chosen to balance performance and

hardware constraints. The use of mixed precision will allow for more efficient training on limited GPU memory. Classification will be performed across three ripeness classes, with appropriate loss functions selected for object detection. Data augmentations will be applied both through YOLOv11's internal pipeline and study-specific methods to increase generalization. This configuration will ensure a reproducible and optimized training pipeline for the target application (Ultraltyics, 2025).

Training and Validation

Instance segmentation setup

This study will implement instance segmentation as a preparatory step to enable precise ripeness classification of individual Cavendish banana fingers. The YOLOv11 model was configured to operate in segmentation mode (task=segment), enabling it to output both bounding boxes and pixel-level masks for each banana finger. The segmentation variant yolo11-medium-seg was selected for its balanced performance in speed and accuracy among the available model sizes (Ultralytics, 2025).

Annotation of the dataset was performed using Roboflow, which supports YOLO-compatible segmentation formats. Each annotated object includes a class label (unripe, ripe, or overripe), normalized bounding box coordinates (x, y, width, and height), and a series of normalized (x, y) points outlining the segmentation polygon. These annotations will be exported in the YOLOv11 segmentation format to ensure compatibility with the training pipeline.

The segmentation model enables the identification of each banana finger as a distinct instance, preserving both the object's contour and class label. YOLOv11's segmentation head integrates the C2PSA attention block and C3k2 bottleneck modules,

which enhance spatial attention and multi-scale feature extraction. These features support the model's ability to generate accurate binary masks for each object, which are used in subsequent ripeness classification tasks (Khanam and Hussain, 2024).

Training procedure

Model training will be conducted using Python v3.10+ with the PyTorch deep learning framework (version 1.x) as the foundation for implementing YOLOv11. Training will be performed on a workstation equipped with an NVIDIA GPU, utilizing the CUDA backend for hardware acceleration. An existing PyTorch implementation of YOLOv11 will be adopted to construct the training pipeline.

Prior to training, the annotated dataset will be loaded and parsed. Each image and its corresponding YOLO-format segmentation label will be read into memory. Data augmentation transformations will be applied on-the-fly during each training epoch to introduce variability and reduce overfitting. The data loader will shuffle the training samples per epoch and generate mini-batches of size 16. Each input image will be normalized (pixel values scaled to [0,1]) as required by the YOLOv11 backbone.

YOLOv11's segmentation variant (yolo11-medium-seg) operates in anchor-free mode, so there is no need to compute anchor boxes during training (Ultralytics, 2025). The model simultaneously predicts bounding boxes and corresponding instance masks using a unified architecture.

Training will be conducted end-to-end for 100 epochs. For each training epoch, the following steps will occur.

Forward Pass. The model receives a batch of input images and generates predictions that include class labels, bounding boxes, and segmentation masks. The

backbone network extracts multi-scale features, which are aggregated by the neck and passed to the segmentation head.

Loss Computation. The predicted outputs will be compared to the ground truth annotations. Loss is calculated using a combination of segmentation loss (Dice + BCE loss for masks), localization loss (based on Intersection over Union), and classification loss (typically using cross-entropy). These will be weighted according to YOLOv11 defaults.

Backpropagation. The optimizer (Adam) backpropagates the computed gradients through the network to update the model parameters and minimize the total loss. Adaptive learning rate adjustments and momentum are handled internally.

Learning Rate Scheduling. A learning rate scheduler will gradually reduce the learning rate during training to stabilize convergence. Both training and validation losses will be monitored after each epoch.

At the end of each epoch, the model will be evaluated using the validation set (20% of the data), with no augmentations applied. Metrics such as validation loss and mean Average Precision (mAP) will be recorded to monitor performance and identify overfitting. The model typically converges before or around the 100th epoch. The final model will be selected based on the highest mAP score achieved on the validation set.

Throughout training, periodic checkpointing and logging will be employed. Model weights will be saved every 10 epochs. The best model checkpoint, determined by validation mAP, will be retained for final evaluation on the independent test set (10%).

In summary, the training procedure involves optimizing the YOLOv11 segmentation network to accurately detect and segment individual banana fingers. This

enables precise ripeness classification at the instance level, with continuous validation to ensure generalization to unseen data.

Model Evaluation

To quantitatively evaluate the performance of the trained model, we used a set of standard object detection metrics: Intersection over Union (IoU), mean Average Precision (mAP) at a specified IoU, confusion matrix, precision, recall, and F1-score. These metrics capture different aspects of detection quality, from localization accuracy to classification reliability. Table 8 will define each metric and explain how it was used in this study.

Table 8. Evaluation metrics.

Evaluation Metric	Formula	Description
Intersection over Union (IoU)	$IoU = \frac{Area\,(Bp \cap Bgt)}{Area\,(Bp\ Bgt)}$	Used to assess localization accuracy of detected banana fingers. A detection is counted as correct if IoU ≥ 0.5 with ground truth. Average IoU of matched detections was reported to measure precision in localization.
Confusion Matrix	<table><tr><td colspan="2"></td><td>Actual Values</td><td></td></tr><tr><td colspan="2"></td><td>+</td><td>-</td></tr><tr><td rowspan="2">Predicted Values</td><td>+</td><td>TP</td><td>FP</td></tr><tr><td>-</td><td>FN</td><td>TN</td></tr></table>	Presents the number of correct and incorrect predictions made by the model, organized by actual and predicted classes.
Precision	$Precision = \frac{TP}{TP + FP}$	Proportion of predicted positive instances that are actually correct. Calculated per class and overall. High precision means fewer false alarms.
Recall	$Recall = \frac{TP}{TP + FN}$	Proportion of actual positives that are correctly identified by the model. Calculated per class and overall. High recall means the model missed very few bananas in the test set.

F1-Score	$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$	Harmonic mean of precision and recall. High F1 indicates both low false positives and low false negatives. Reported per class and macro-averaged.
Mean Average Precision	$mAP = = \frac{1}{N} \sum_{i=1}^{N} APi$	Primary detection metric. Computed as the area under the precision-recall curve for each class and averaged. Reported at IoU=0.5 and IoU thresholds from 0.5–0.95. Higher mAP reflects better overall localization and classification performance.

Using these metrics, the model will be evaluated on the test set (the 10% of images reserved and never seen during training/validation). For each test image, the YOLOv11 model outputs a set of detections. These are matched to ground truth annotations using the IoU threshold (0.5) to determine TP, FP, FN counts per class. Then precision, recall, and F1 are computed from those counts, and AP is computed by taking the precision-recall curve (by varying the confidence threshold for detections) and finding the area under the curve. We also computed the overall mAP and class-specific APs. Additionally, the average IoU of correct detections was also recorded to measure localization accuracy explicitly.

Hardware and Software Requirements

Table 9. Hardware Specifications

Component	Specification
GPU	NVIDIA GeForce RTX 4050 Laptop GPU, 6 GB GDDR6 VRAM
CPU	Intel Core i7-12650HX (12-core, base frequency 2.0 GHz)

RAM	16 GB system RAM (12 GB usable during model training due to OS overhead)
Operating System	Windows 11, 64-bit

Table 10. Software Environment

Component	Specification
Python v3.10+	Programming language used for model development and running the training pipeline.
PyTorch	Deep learning framework (CUDA-enabled) used to implement and train the YOLOv11 model with GPU acceleration and FP16 mixed precision.
Ultralytics YOLO	Core framework for YOLOv11 configuration, training, validation, and inference. Provides model definition, utilities, and built-in evaluation tools.
CUDA Toolkit	Enables GPU-based tensor operations, necessary for accelerating PyTorch and YOLOv11 training on the NVIDIA RTX 4050 GPU.
NumPy	Used for numerical operations, label processing, and custom metric computations.
Pandas	Used for handling tabular data such as metric logs, performance summaries, and result outputs.
Matplotlib	Visualization libraries used for plotting training curves (loss, mAP) and confusion matrices.
Roboflow	Image annotation tool used to label individual banana fingers and ripeness stages. Supports YOLO-format export and version-controlled datasets.
PyTorch DataLoader	Manages efficient data loading and batch creation with built-in multi-threaded preprocessing during training.
Conda Environment	Used to isolate and manage dependencies, ensuring reproducibility across training runs and experiments.
Seed Setting Utilities	Ensures consistent training behavior and results by fixing the random seed during model initialization.

Post-Processing and Output Interpretation

After the YOLOv11 model generates predictions for a given image during validation, testing, or deployment, a post-processing step is applied to refine these outputs. As the model is configured for instance segmentation, the output for each detected object includes a bounding box, a class label (unripe, ripe, or overripe), a confidence score, and a segmentation mask that outlines the pixel-level contour of each banana finger.

To reduce redundancy in object detection, Non-Maximum Suppression (NMS) will be applied. YOLO models may produce multiple overlapping predictions for the same object due to adjacent grid cells generating similar outputs. NMS addresses this by retaining only the prediction with the highest confidence score among overlapping boxes (Bhalerao, 2023). An IoU threshold of 0.45 will be typically used; overlapping boxes with an IoU greater than 0.45 will be considered redundant, and the one with the lower score will be suppressed. The NMS will be implemented using the built-in function provided by the YOLOv11 PyTorch-based framework.

Following suppression, the model outputs will be visualized for interpretation. Each prediction will include bounding box (normalized or mapped back to the original image resolution), segmentation mask, which highlights the detected banana finger at pixel level, predicted class label, and confidence score.

To assist in qualitative review, a visualization step is conducted using OpenCV. Each banana finger instance is displayed with its segmentation mask overlay and a color-coded bounding box: green for unripe, yellow for ripe, and red for overripe. The predicted class label and associated confidence percentage will also be rendered beside each detection.

For evaluation on the test set, predictions will be considered correct if the predicted class matches the ground truth and the IoU between the predicted mask and the annotated mask is ≥ 0.50. In cases where multiple detections remain for a single object, any extra predictions will be counted as false positives, while missed instances will be counted as false negatives. Failure cases will be manually reviewed by inspecting output visualizations, especially when segmentation masks do not tightly align with object contours or when ripeness is misclassified due to visual ambiguity.

This post-processing pipeline ensures that final outputs are interpretable, compact, and ready for quantitative evaluation through metrics such as mAP, precision, recall, and F1-score.

LITERATURE CITED

2023 fruit crops situationer: Davao Region | Philippine Statistics Authority. (2024, August 28). Available at: https://rsso11.psa.gov.ph/content/2023-fruit-crops-situationer-davao-region#:~:text=,in%202023%20and%202022%2C%20respectively (Accessed: 1 June 2025).

A Deep Dive Into Non-Maximum Suppression (NMS). (n.d.). Built In. Available at: https://builtin.com/machine-learning/non-maximum-suppression (Accessed: 23 May 2025).

Aditya Putra Prananda, Pardede, A.M.H. and Rahmadani. (2024) Segmentation Algorithm K – Means Based On The Maturity Level Of Blueberries. *j. of artif. intell. and eng. appl.*, 3, pp. 584–589. doi: 10.59934/jaiea.v3i2.433.

Ali, M.L. and Zhang, Z. (2024) The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection. *Computers*, 13(12), p. 336. doi: 10.3390/computers13120336.

Arcalas, J. E. (2024) Banana exports reverse sagging trend, rise in 2023. *Philstar.com*. Available at: https://www.philstar.com/business/2024/02/22/2335080/banana-exports-reverse-sagging-trend-rise-2023#:~:text=MANILA%2C%20Philippines%20%E2%80%94%20Philippine%20banana%20exports,previously%20affected%20by%20Panama%20disease. (Accessed: 1 June 2025).

Baldovino, R.G., Lim, R.A.U., Salvador, P.R.R. and Tiamzon, E.A.P. (2024) Real-Time Banana Ripeness Detection and Classification using YOLOv8. In: *2024 9th International Conference on Mechatronics Engineering (ICOM)*. Kuala Lumpur, Malaysia: IEEE, pp. 219–223. doi: 10.1109/ICOM61675.2024.10652438.

Bananas | Markets and Trade | Food and Agriculture Organization of the United Nations. (n.d.). *MarketsAndTrade*. Available at: https://www.fao.org/markets-and-trade/commodities-overview/bananas-tropical-fruits/bananas/en (Accessed: 5 May 2025).

Bananas from the Philippines - DAFF. (2024, November 11). Available at: https://www.agriculture.gov.au/biosecurity-trade/policy/risk-analysis/plant/banana-philippines#:~:text=The%20Philippines%E2%80%99%20banana%20industry%20is,of%20production%20is%20Cavendish (Accessed: 1 June 2025).

Behnam Israel, N., Ismail Al-Sulaifanie, A. and Khorsheed Al-Sulaifanie, A. (2024) A Recognition and Classification of Fruit Images Using Texture Feature Extraction and Machine Learning Algorithms. *ACAD J NAWROZ UNIV*, 13, pp. 92–104. doi: 10.25007/ajnu.v13n1a1514.

Bureau of Agriculture and Fisheries Product Standards. (2008) *Philippine national standard: Fresh fruits – Banana (PNS/BAFPS 64:2008)*. Department of Agriculture, Republic of the Philippines.

Bureau of Agriculture and Fisheries Product Standards. (2013) *Philippine National Standard: Code of good agricultural practices (GAP) for banana production (PNS/BAFPS 129:2013)*. Department of Agriculture, Republic of the Philippines.

Cahya, D.E., Cahya, Z., Taufiqurrohman, H., Destika, H., Alfa, M.N. and Putri, T.E. (2023) Ripeness Classification of Cavendish Bananas using Multi-object Detection Approach. In: *2023 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*. Bandung, Indonesia: IEEE, pp. 146–151. doi: 10.1109/ICRAMET60171.2023.10366571.

Cavendish banana - Musa acuminata “Dwarf Cavendish” | Plants | Kew. (n.d.). Available at: https://www.kew.org/plants/cavendish-banana (Accessed: 31 May 2025).

Cavendish Banana Market Size, Growth, Trends and Forecast. (n.d.). Available at: https://www.globalinsightservices.com/reports/cavendish-banana-market/ (Accessed: 1 June 2025).

Cavendish Banana Market: Trends, Challenges, and Future Outlook. (n.d.). Available at: https://www.linkedin.com/pulse/cavendish-banana-market-trends-challenges-future-outlook-nagula-hmqtc (Accessed: 31 May 2025).

Chiwate, S. M., Jadhav, B. T., & Nikam, S. V. (2023) Analysis of physicochemical changes during the ripening of cavendish banana and velchi banana. *Current Agriculture Research Journal*, 11(1), pp. 236–243. doi: 10.12944/carj.11.1.20.

EST: Banana facts. (n.d.). Available at: https://www.fao.org/economic/est/est-commodities/oilcrops/bananas/bananafacts/en/ (Accessed: 1 May 2025).

Food and Agriculture Organization of the United Nations & World Health Organization. (1997/2022) *Codex Standard for Bananas (CXS 205-1997, amended 2022)*. Codex Alimentarius Commission. Available at: https://www.fao.org/fao-who-codexalimentarius/sh-proxy/en/?lnk=1&url=https%3A%2F%2Fworkspace.fao.org%2Fsites%2Fcodex%2FStandards%2FCXS%2B205-1997%2FCXS_205e.pdf (Accessed: 1 June 2025).

Ghazal, S., Munir, A., & Qureshi, W. S. (2024) Computer vision in smart agriculture and precision farming: Techniques and applications. *Artificial Intelligence in Agriculture*, 13, pp. 64–83. doi: 10.1016/j.aiia.2024.06.004.

Gopal, K., Govindarajulu, B., Ramana, K.T.V., Kumar, C.S.K., Gopi, V., Sankar, T.G., Lakshmi, L.M., Lakshmi, T.N. and Sarada, G. (2014) Citrus Scab (Elsinoe fawcettii): A Review. *Research & Reviews: Journal ofÂ Agriculture and Allied Sciences*, pp. 49–58.

Gulzar, Y. (2023) Fruit Image Classification Model Based on MobileNetV2 with Deep Transfer Learning Technique. *Sustainability*, 15(3), p. 1906. doi: 10.3390/su15031906.

Guo, J., Fu, H., Yang, Z., Li, J., Jiang, Y., Jiang, T., Liu, E., & Duan, J. (2021) Research on the physical characteristic parameters of banana bunches for the design and development of postharvesting machinery and equipment.[1] *Agriculture*, 11(4), p. 362. doi: 10.3390/agriculture11040362.

Hailu, M., Workneh, T.S. and Belew, D. (n.d.) Review on postharvest technology of banana fruit.

Hameed, A., Shah, S., Khan, S., Alanazi, S., Algamdi, S., 2025. Dermatology 2.0: Deploying YOLOv11 for Accurate and Accessible Skin Disease Detection: A Web-Based Approach. Int. J. Imaging Syst. Technol. 35. https://doi.org/10.1002/ima.70050

Han, X. (2023) Fruit Image Classification Based on SVM, Decision Tree and KNN. In: *Proceedings of the 1st International Conference on Data Analysis and Machine Learning*. Kuala Lumpur, Malaysia: SCITEPRESS - Science and Technology Publications, pp. 374–380. doi: 10.5220/0012815800003885.

Intersection over Union (IoU): Definition, Calculation, Code. (n.d.). Available at: https://www.v7labs.com/blog/intersection-over-union-guide (Accessed: 2 June 2025).

Khanam, R. and Hussain, M. (2024) *YOLOv11: An Overview of the Key Architectural Enhancements*. doi: 10.48550/arXiv.2410.17725.

Lagare, J. B. (2025) PH slips to No. 4 in banana export ranking. *INQUIRER.net*. Available at: https://business.inquirer.net/501321/ph-slips-to-no-4-in-banana-export-ranking (Accessed: 1 June 2025).

Le, T.-T., Lin, C.-Y. and Piedad, E.J. (2019) Deep learning for noninvasive classification of clustered horticultural crops – A case for banana fruit tiers. *Postharvest Biology and Technology*, 156, p. 110922. doi: 10.1016/j.postharvbio.2019.05.023.

Liu, X., Zhao, D., Jia, W., Ji, W., Ruan, C. and Sun, Y. (2019) Cucumber Fruits Detection in Greenhouses Based on Instance Segmentation. *IEEE Access*, 7, pp. 139635–139642. doi: 10.1109/ACCESS.2019.2942144.

Ma, C., Wang, J., Zeng, T., Liang, Q., Lan, X., Lin, S., Fu, W. and Liang, L. (2024) Banana Individual Segmentation and Phenotypic Parameter Measurements Using Deep Learning and Terrestrial LiDAR. *IEEE Access*, 12, pp. 50310–50320. doi: 10.1109/ACCESS.2024.3385280.

Maduwanthi, S. D. T., & Marapana, R. a. U. J. (2019) Induced ripening agents and their effect on fruit quality of banana. *International Journal of Food Science*, 2019,[2] pp. 1–8. doi: 10.1155/2019/2520179.

Magno, R.D.C., Bersamen, P.J.S. and Valiente, L.D. (2022) Monitoring of Banana Ripening Process and Identification of Banana type using FRCNN, LSTM, And Color Segmentation. In: *2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*. Boracay Island, Philippines: IEEE, pp. 1–6. doi: 10.1109/HNICEM57413.2022.10109517.

Manalo, M. R., Falco, M. E. M., Gomez, R. M., Dela Cruz, K. A. S., Estudillo, P. M. N., Ramos, J. M. F., Flores Jr., B. B., & Cortado, C. N. (2023) Utilization of out-of-specification unripe Cavendish banana (Musa acuminata AAA) as banana powder. *Philippine Journal of Science*, 152(5), pp. 1831–1846.

Mao, D., Wang, X., Liu, Y., Zhang, D., Wu, J. and Chen, J. (2023) YOLO-Banana: An effective grading method for banana appearance quality. *Journal of Beijing Institute of Technology*, 32(3), pp. 363–373. doi: 10.15918/j.jbit1004-0579.2023.004.

Maseko, K.H., Regnier, T., Meiring, B., Wokadala, O.C. and Anyasi, T.A. (2024) Musa species variation, production, and the application of its processed flour: A review. *Scientia Horticulturae*, 325, p. 112688. doi: 10.1016/j.scienta.2023.112688.

Mesa, A.R. and Chiang, J.Y. (2021) Multi-Input Deep Learning Model with RGB and Hyperspectral Imaging for Banana Grading. *Agriculture*, 11(8), p. 687. doi: 10.3390/agriculture11080687.

Morphology of banana plant | Improving the understanding of banana. (n.d.). *Improving the Understanding of Banana*. Available at: https://www.promusa.org/Morphology+of+banana+plant#:~:text= (Accessed: 1 June 2025).

Munir, N., Moazzam, S., & Haq, R., Naz, S. (n.d.) Detection of Fungus Causing Scab Disease of Citrus in Punjab, Pakistan.

N, A. and R, V.K. (2023) Banana Ripeness Classification with Deep CNN on NVIDIA Jetson Xavier AGX. In: *2023 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. Kirtipur, Nepal: IEEE, pp. 663–668. doi: 10.1109/I-SMAC58438.2023.10290326.

Nelson, S. C. (n.d.) *Banana ripening: Principles and practice* [PDF]. University of Hawai‘i at Mānoa, College of Tropical Agriculture and Human Resources (CTAHR). Available at: https://www.ctahr.hawaii.edu/nelsons/banana/ripeningbunchmanagement.pdf (Accessed: 1 June 2025).

Philippine Council for Agriculture, Aquatic and Natural Resources Research and Development. (n.d.) *Banana – Industry Strategic S&T Program*. Available at: https://ispweb.pcaarrd.dost.gov.ph/isp-commodities/banana/ (Accessed: 1 June 2025).

Philippine Statistics Authority. (2023) *Major fruit exports of the Philippines, 2022*. Available at: https://psa.gov.ph/ (Accessed: 1 June 2025).

Precision vs. Recall: Differences, Use Cases & Evaluation. (n.d.). Available at: https://www.v7labs.com/blog/precision-vs-recall-guide (Accessed: 2 June 2025).

Prihantoro, J., Anbiya, D.R., Prihantoro, G., Nashrullah, E., Cahya, Z. and Riza (2023) Comparison of Machine Learning Techniques for Grade Classification of Exported Cavendish Bananas. In: *2023 IEEE International Conference on Data and Software Engineering (ICoDSE)*. Toba, Indonesia: IEEE, pp. 37–42. doi: 10.1109/ICoDSE59534.2023.10291592.

Pushpa, B.R., Chirag, D.L. and Bhat, S. (2024) An Intelligent Approach to Determine Banana Ripeness Stages using Deep Learning Models. In: *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)*. New Delhi, India: IEEE, pp. 1774–1780. doi: 10.23919/INDIACom61295.2024.10498201.

Rahman, Md.M., Basar, Md.A., Shinti, T.S., Khan, Md.S.I., Babu, H.Md.H. and Uddin, K.M.M. (2023) A deep CNN approach to detect and classify local fruits through a web interface. *Smart Agricultural Technology*, 5, p. 100321. doi: 10.1016/j.atech.2023.100321.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016) 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, pp. 779–788. doi: 10.1109/CVPR.2016.91.

Rusmarsidik, H.S.P. and Risnandar (2022) DeRiBa-Fuz: Detection of Ripening Banana Using Fuzzy Logic with RGB and GLCM Extraction. In: *2022 International Conference on Advanced Creative Networks and Intelligent Systems (ICACNIS)*. Bandung, Indonesia: IEEE, pp. 1–7. doi: 10.1109/ICACNIS57039.2022.10055225.

Sangeetha, K., Raja, P.V., S, S., J, S. and S, R. (2024) Classification of Fruits and its Quality Prediction using Deep Learning. In: *2024 5th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*. Tirunelveli, India: IEEE, pp. 342–346. doi: 10.1109/ICICV62344.2024.00059.

Satekge, T. K., & Magwaza, L. S. (2021) Investigating the Effect of Fruit Size on Ripening Recovery of Banana Treated with 1-Methylcyclopropene. *Horticulturae*, 7(10), p. 357. doi: 10.3390/horticulturae7100357.

Sathyanarayanan, S. (2024) Confusion Matrix-Based Performance Evaluation Metrics. *AJBR*, pp. 4023–4031. doi: 10.53555/AJBR.v27i4S.4345.

Tang, C., Chen, D., Wang, X., Ni, X., Liu, Y., Liu, Y., Mao, X. and Wang, S. (2023) A fine recognition method of strawberry ripeness combining Mask R-CNN and region segmentation. *Front. Plant Sci.*, 14, p. 1211830. doi: 10.3389/fpls.2023.1211830.

Tanudtanud, G. (2024) Davao's new investments in banana reached P235 million: DTI XI. *Mindanao Times*. Available at: https://mindanaotimes.com.ph/davaos-new-investments-in-banana-reached-p235-million-dti-xi/ (Accessed: 1 June 2025).

The Observatory of Economic Complexity. (n.d.). *The Observatory of Economic Complexity*. Available at: https://oec.world/en/profile/bilateral-product/bananas/reporter/phl (Accessed: 1 June 2025).

Tian, H., Wang, T., Liu, Y., Qiao, X., & Li, Y. (2020) Computer vision technology in agricultural automation—A review. *Information Processing in Agriculture*, 7(1), pp. 1–19. doi: 10.1016/j.inpa.2019.09.006.

Ucat, R.C. and Dela Cruz, J.C. (2019) Postharvest Grading Classification of Cavendish Banana Using Deep Learning and Tensorflow. In: *2019 International Symposium on Multimedia and Communication Technology (ISMAC)*.[3] Quezon City, Philippines: IEEE, pp. 1–6. doi: 10.1109/ISMAC.2019.8836129.

Ukwuoma, C.C., Zhiguang, Q., Bin Heyat, M.B., Ali, L., Almaspoor, Z. and Monday, H.N. (2022) Recent Advancements in Fruit Detection and Classification Using Deep Learning Techniques. *Mathematical Problems in Engineering*, 2022, pp. 1–29. doi: 10.1155/2022/9210947.

Ultralytics. (2023) *YOLOV8*. Available at: https://docs.ultralytics.com/models/yolov8/ (Accessed: 1 June 2025).

Ultralytics. (n.d.). *Configuration*. Available at: https://docs.ultralytics.com/usage/cfg (Accessed: 23 May 2025).

United Nations. (2015) *Transforming our world: The 2030 agenda for sustainable development*. United Nations. Available at: https://sdgs.un.org/2030agenda (Accessed: 1 June 2025).

United States Department of Agriculture. (2004) *Bananas: Market inspection instructions*. Agricultural Marketing Service, Fruit and Vegetable Programs, Fresh Products Branch.

Vlăsceanu, G.V., Tarbă, N., Voncilă, M.L. and Boiangiu, C.A. (2024) Selecting the Right Metric: A Detailed Study on Image Segmentation Evaluation. *BRAIN*, 15(4), p. 295. doi: 10.70594/brain/15.4/20.

Wang, A., Qian, W., Li, A., Xu, Y., Hu, J., Xie, Y., & Zhang, L. (2024) NVW-YOLOv8s: An improved YOLOv8s network for real-time detection and segmentation of tomato fruits at different ripeness stages.[4] *Computers and Electronics in Agriculture*, 219, p. 108833. doi: 10.1016/j.compag.2024.108833.

Wang, G., Gao, Y., Xu, F., Sang, W., Han, Y., & Liu, Q. (2025) A banana ripeness detection model based on improved YOLOv9c multifactor complex scenarios. *Symmetry*, 17(2), p. 231. doi: 10.3390/sym17020231.

Wu, J. (2024) A Review of Research on Image Classification Methods. *Appl. Comput. Eng.*, 80(1), pp. 110–114. doi: 10.54254/2755-2721/80/2024CH0068.

Xiao, B., Nguyen, M., & Yan, W. Q. (2024) Fruit ripeness identification using YOLOv8 model. *Multimedia Tools and Applications*, 83, pp. 28039–28056. doi: 10.1007/s11042-023-16570-9.

Zhao, Z., Hicks, Y., Sun, X. and Luo, C. (2023) Peach ripeness classification based on a new one-stage instance segmentation model. *Computers and Electronics in Agriculture*, 214, p. 108369. doi: 10.1016/j.compag.2023.108369.