Optimizing Agricultural Sustainability: AI-Based Yield Prediction Bot for Precision Agriculture

Sangeeta Kurundkar, Kuldeep Aher, Shrirang Mahankaliwar, Swayamprakash Mahale, Divya Lothe, Sahil Mandhare

Vishwakarma Institute of Technology, Pune, India

Corresponding author: Kuldeep Aher, Email: kuldeep.aher212@vit.edu

Yield estimation is a critical task in modern agriculture, as it enables farmers to optimize crop management strategies and offers substantial benefits to stakeholders to optimize their supply chains with informed decisions.In this paper, authors propose a novel approach for yield estimation using image processing techniques, combined with state-of-the-art tracking algorithms and object detection algorithms such as YOLO v8 (You Only Look Once), and BoT-SORT (Deep Learning-based Object Tracking). This research utilizes both image processing and deep learning methods, providing a cost-effective and efficient solution for crop yield estimation. The experiments conducted have demonstrated promising results across various metrics, making it a valuable resource for farmers. By bridging the gap between technology and agriculture, this research not only improves crop yields but also contributes to the sustainability of the food supply in a rapidly growing world population.

Keywords: Crop-Yield estimation, Computer vision, YOLO v8, Object detection.

1. Introduction

Agriculture has been one of the most crucial sectors in the world since the stone age. The Food and Agriculture Organization (FAO) forecasts that the global population will reach 9.7 billion by 2050. Increasing the demand for global food production by 70%.Efficient and effective agricultural practices can help us meet this growing demand for food. The yield estimating bots can transform the agrarian sector by offering precise yield estimations, thereby empowering farmers to formulate decisions based on data. Additionally, helping stakeholders like grocery stores, grocery giants and merchants to optimize and enhance their supply chain.

In precision agriculture, fruit detection and counting accuracy are critical components that help farmers track crop growth, evaluate potential yield, and make well-informed decisions about harvesting and resource allocation. The scalability and efficiency of traditional manual counting methods are limited by their labor-intensive and time-consuming nature.To tackle this challenge, this study introduces an innovative method that uses computer vision techniques, a remotely controlled (RC) bot, and advanced algorithms to automate the processes of fruit identification and counting.

The primary objective of this research is to design a system proficient in detecting and counting fruits. By harnessing the capabilities of cutting-edge computer vision algorithms, this manuscript endeavors to streamline the procedure of fruit identification and counting, reducing human involvement, and augmenting accuracy.

It's essential to recognize the global environmental impact of conventional farming practices. This approach enables the targeted allocation of resources, thereby reducing the ecological impact while ensuring that yields remain adequate to cater to the needs of the burgeoning population.

2. Related Work

Automated fruit detection in farms using robots is a topic of interest in agricultural research. Several papers discuss the development of robots for fruit detection and harvesting. Wang et al. propose a Transformer-based mask R-CNN model for tomato detection and segmentation, achieving high performance in detecting and segmenting tomatoes [1]. Borkar et al. present a small farmer bot prototype that demonstrates fruit harvesting by detecting ripe fruits for plucking [2]. The CROPS project focuses on developing high-tech robots for site-specific spraying and selective harvesting of fruit and fruit vegetables, with the ability to detect fruit, sense ripeness, and gently detach ripe fruit [3]. Yoshida et al. propose a method to detect cutting points on tomato peduncles using a harvesting robot equipped with an RGB-D camera [4]. Ceres et al. present a robot prototype for aided fruit-harvesting in unstructured environments, where the operator detects fruits using a laser rangefinder and the computer performs precise location and picking sequence computations [5].

The authors in [6] present a system that uses an accelerometer-based gesture recognition technique to control the movements of a robotic arm. This allows for more intuitive and natural control of the arm based on the gestures of human hands.The system utilizes wireless control through the Zig-bee protocol, enabling remote operation of the robotic arm. The model proposed by [7] focuses on extracting fruit form and shade to establish unique function units for every fruit variety. The version comprises 3 stages: pre-processing (PP), function extraction (FE), and trying out. Within the PP level, photo resizing is performed, followed by the FE stage where color, form functions, and scale invariant function transform are used to build function vectors. In the testing phase, the okay-nearest neighborhood type algorithm is utilized for fruit identity. Importantly, the version no longer recognizes fruits robotically, but additionally offers information about their calorie content material.

To overcome the time-consuming strategies currently deployed, an automated fruit counting system with the use of imaging and prescient techniques is proposed in this work. The device makes use of a minimal Euclidean distance-based total segmentation technique to extract fruit areas from input pix. Circle overlaying is then finished on the fruit regions, and fruit counting is finished based totally on the centroids of those areas [8].

The system proposed in [9] demonstrates correct detection and counting of apples in looking at pictures, imparting a promising solution for efficient crop control and productivity enhancement in the agricultural enterprise. An investigation into the use of digital image analysis for distinguishing between fruit and other elements within Cabernet Sauvignon canopies revealed a strong correlation. Specifically, the ratio of 'fruit' pixels to the total image pixels accounted for 85% of the yield variation. These findings have significant implications for the future advancement of automated and spatially aware techniques for predicting vineyard yields.

Deep learning-based approaches have gained interest in fire detection in smart cities. Al-Turjman et al. proposed a hybrid approach combining CNN and RNN for fire detection, achieving high accuracy and low false alarm rates. The system is compiled of YOLOv8-based approach for fire detection in outdoor scenes, achieving an accuracy of 92.8% The proposed YOLOv8-based smart fire detection system (SFDS) in this paper improves accuracy, reduces false alarms, and can be extended to detect other objects of interest in smart cities [10]. In other research, author's used k-means segmentation on orange tree images to minimize shadow effects and extract oranges using blob detection and size calculation. The object separation step is used to separate overlapping fruit, reduce undesired edges, and segment the object in the image using a particular RGB value. Results show that manually counting and marking all the orange fruit in all the input images produces a ground truth [11].

The YOLO family has undergone numerous modifications since its conception, each improving upon the ones before it to address flaws and improve performance. With an image size of 640 pixels, YOLOv5x received an AP of 50.7% when tested on the MS COCO dataset test-dev 2017. On an NVIDIA V100, it can run at a speed of 200 FPS with a batch size of 32[12]. The proposed system of yield estimation by [13] is divided into two phases: GUI development and image acquisition. Charge-coupled device (CCD) is used to capture images of tree fruits, image processing is performed to modify the image, and image segmentation is used to separate input images into areas that strongly relate to objects or areas of interest. The implementation of crop yield detection involves finding the coordinates of edge pixels, calculating the mean of pixels, and calculating the center of clusters.

In this paper, the authors propose modifications to enhance the state-of-the-art YOLO-V8 model, focusing on achieving speed and reliability in drone detection. The introduced enhancements involve the incorporation of Multi-Scale Image Fusion and the integration of the P2 Layer into the medium-size model (M-model) of YOLO-V8. The proposed model underwent evaluation in the 6th WOSDETC challenge, providing a practical assessment of its performance in real-world scenarios [14].

Yiting Li, Qingsong Fan, Haisong Huang, Zhenggong Han, and Qiang Gu wrote- about improving drone target identification methods in their piece. They target challenges like missing small targets and flawed detection in aerial shots. Their strategy enlists BiPAN-FPN, used for featuring and GhostblockV2 to minimize information loss. The VisDrone2019 dataset was used to test their model and various other tests. Theseshowed the model's worth, suggesting a fresh path for deep learning in drone target detection [15].

The paper "An Improved Fire Detection Approach Based on YOLO-v8 for Smart Cities" by Fatma M. Talaat and Hanaa Zinedine proposes a Smart Fire Detection System (SFDS) utilizing the YOLOv8 algorithm. The SFDS, designed for smart cities, enhances fire detection accuracy in real-time while reducing false alarms and maintaining cost-effectiveness. The proposed smart city framework integrates Fog and Cloud computing with the IoT layer, ensuring swift data collection and processing for faster responses, minimizing property damage and human risks [16].Anand Koirala and his team came- up with an academic piece. It's a summary of how deep learning can play a part in spotting fruits and guessing the- produce. They say we should use common measures to compare models. Then they suggest we get our images of fruits in orchards from open sources. They also say that transfer learning may be- helpful. Their main idea is practical tips for finding fruits. The-y talk about issues like hidden fruits. Be-sides, they also investigate how many fruits we might ge-t from orchards judging by the number of fruits in the picture-s of trees. To help newcomers in fruit detection research using deep learning [17]. Stephen Nuske and his team present a new, nonharmful method for estimating vineyard yield through computer vision. They identify and count grape berries by their shape and look. This could be a great help to big vineyards. They test this syste-m with 224 vines of two kinds of grapes. The results match the real harvest yield very closely, with only a 9.8% error. Thus, this method is more inclusive and efficient than the old, harmful methods usually used to predict vineyard yield [18].In Christine De-wi et al's work [19] aim is simple: better traffic sign detection. They use Convolutional Neural Networks (CNN), tackling issues that go with limited datasets. The team applies Generative- Adversarial Networks (GAN) such as DCGAN, LSGAN, and WGAN to create synthetic training images. Quality of these images is measured by Structural Similarity Inde-x (SSIM) and Mean Square Error (MSE). Surprisingly, these synthetic pictures are very accurate. When Yolo V4 uses LSGANsynthesized images, thereis an accuracy of 89.33%.

In the paper by Muhammad Hussain the evolution of YOLO object detectors from YOLO-v1 to the latest YOLO-v8 is explored, emphasizing their rapid growth since 2015. The YOLO variants prioritize real-time and high-classification performance with efficient computational parameters. The review delves into architectural advancements across iterations and highlights YOLO's compatibility with industrial manufacturing, particularly in surface defect detection, showcasing its applicability for fast detection, high accuracy, and deployment on constrained edge devices [20].Growing global interest in cooperative robotics in agriculture, addressing crucial challenges such as food production for a growing population, environmental sustainability, and resilience during crises. The emergence of "Agriculture 4.0" and "Agriculture 5.0" technologies highlight the need for effective human-robot interaction models in human-robot interfaces, coordination, and the application of collaborative robots in livestock handling [21].

3. Methodology

The literature review indicates computer vision-based techniques are best suited to detect and count fruit. In this research, authors propose RC-controlled bot mounted with a wide-angle camera. The feed from the camera is processed to detect the fruits. For generalization, only pomegranates are used in this research.

3.1 Data Collection

To train a computer vision model to detect and count pomegranates, a robust dataset of images was gathered. The authors utilized two primary sources for data acquisition:

1. **Publicly available YouTube videos -** Publicly accessible YouTube videos offered an array of visual conditions, including varying light levels, angles, and stages of fruit maturity. A frame extraction method was used to convert them into a set of still images. The videos were split approximately at the rate of 30 frames per second, yielding a considerable number of images.
2. **Original footage taken directly from various farms -** Simultaneously, images were captured from various pomegranate farms. A diverse array of images representing different scenarios that the RC bot could encounter in real-world conditions were captured.

From the amassed collection, a selection process was carried out to choose the most suitable images. Discarding images that did not meet standards due to poor lighting, blurriness, or the absence of pomegranates. After this rigorous process, the dataset was narrowed down to approximately 5,000 images.

3.2 Data Preprocessing

In the preprocessing stage, the following steps were carried out:

Auto-Orient - To ensure the correct orientation of all images, an auto-orientation function was applied.
Resize - The images were resized to 1280x1280 resolution.
Tile - Each image was divided into a grid of 2x2, resulting in four smaller images.

Post-preprocessing, data augmentation was done. Augmentation is the most effective technique used to artificially expand the diversity and size of the dataset by applying various transformations to the images. Here are the augmentations performed:

Rotation - Each image was rotated within a range of -14° to +14°.
Saturation - The color saturation of each image was adjusted, varying it by -30% to +30%.
Brightness - The brightness of each image was also varied within the range of -30% to +30%.
Exposure - Exposure levels were adjusted between -25% and +25%.

Bounding Box Augmentations: For the bounding boxes in each image, two key augmentations were performed: rotation (90°, both clockwise and counterclockwise) and exposure adjustment (between -25% and +25%).

Each image in the dataset was subjected to these augmentations, creating two output variations per training example.

The final dataset consisted of 32,352 images, split into a training set, a validation set, and a test set. The training set consisted of 25,000 images, accounting for 78% of the total dataset, while the validation set consisted of 5,200 images, making up 16% of the dataset. The test set consisted of the remaining 2,000 images, making up 6% of the dataset.

3.3 Model Training

The process of model training in this study focused on using the YOLOv8 object detection model. YOLO, or "You Only Look Once", is a real-time object detection system that has gained popularity due to its performance and speed. The YOLOv8 variant of this model introduces further enhancements, and in this study, the authors experimented with training the model with different pre-trained weights - YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8xl.

The first phase of model training involved training models with different pre-trained weights with 700 epochs, 1280 batch size and batch size was varied according to the models. Table 1. Shows the configuration used to train all the models, accuracy and inference time are displayed in Table 2.

Retraining with different weights allowed the model to learn from the existing patterns identified in the pre-trained models, while further adapting to the specific features of the dataset. The differences in the results underscore the significance of choosing appropriate weights and parameters for training. It demonstrates the trade-offs and fine-tuning involved in achieving the optimal balance between precision, recall, processing speed, and resource usage, specifically tailored for the task of pomegranate detection and counting.

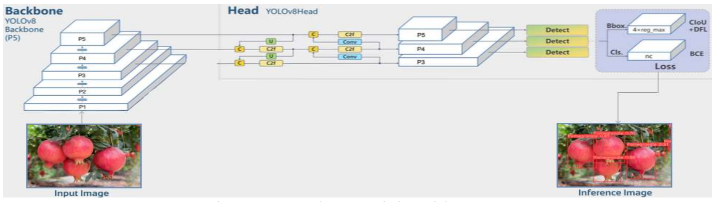

Fig. 1. Detection model architecture

3.4 Detection and Counting

A Python GUI application serves as an interface between the user and the detection system. The GUI allows the user to feed in a video or a real-time camera feed from the RC bot. This feed is then processed for fruit detection using the trained YOLOv8 model.

Fruit Detection with YOLOv8. In YOLOv8, object detection is a two-step process. First, the image is divided into a grid, where each cell is responsible for predicting multiple bounding boxes and class probabilities. These bounding boxes are evaluated by their confidence scores, which tell how likely a box contains an object and how accurate the bounding box is.

Second, the class probabilities denote the likelihood of the object belonging to a particular class (in this case, a pomegranate). This combination of bounding box prediction and class probability forms the final detection framework. The predicted bounding boxes are drawn over the original image, signifying the detection of pomegranates. Fig. 1 showcases the architecture of the yolov8 model.

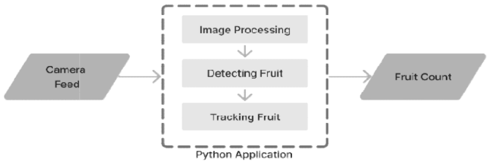

Fig. 2. Detection and counting flow

Fruit Tracking and Counting with BoT SORT. Once the fruits were detected, the authors used a tracking algorithm, specifically the BoT SORT (Breadth of Time Spatial Object Recognizer Tracker) algorithm integrated within YOLOv8 for counting the fruits. BoT SORT is a simple yet efficient approach that uses bounding box coordinates from the detection phase and tracks objects across frames by assigning unique IDs to each detected fruit.

In essence, BoT SORT uses spatial and temporal information from consecutive video frames to estimate the movement of the detected fruits. This allows the algorithm to maintain the identity of the fruits even as they move across frames or temporarily disappear.When the video feed ends or is closed, the total number of unique fruit IDs provided by BoT SORT corresponds to the total count of the fruits. The Fig. 2 describes the whole system flow, first the camera sends feed to the app and fruits are detected and tracked by the model updating the final fruit count.

3.5 RC-Controlled Bot

The proposed methodology employed a Remote Control (RC) bot, which acted as a mobile platform for the camera.

RC System Overview. The bot operates on a radio-based control system, which consists of a transmitter and receiver. In RC technology, the transmitter sends control signals to the receiver, which then interprets and executes the respective actions. These signals are transmitted over a specific frequency, offering precise control over the bot's movements and the camera's orientation fig. 3 shows the bot along with circuit diagram in fig. 4.

Bot Design and Components. The bot was designed to ensure mobility and adaptability. Themotors are managed by an Arduino Mega microcontroller, selected for its robust I/O capabilities. The Mega received inputs from the RC receiver, processed these signals, and sent the corresponding outputs to the motors via the motor controller to the motors. A standout feature of the bot is a 3D-printed robotic arm,built to hold and control the camera's positioning. The robotic arm allowed adjustments to the camera's angle as required. Although the camera was initially connected to a laptop for image processing, the system is designed to be compatible with compact GPU boards like the NVIDIA Jetson Xavier, and Jetson Nano.

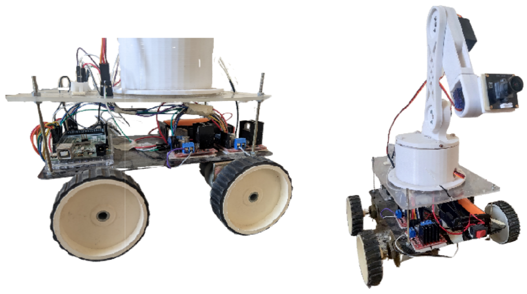

Fig. 3. Side view and front view of RC-Bot

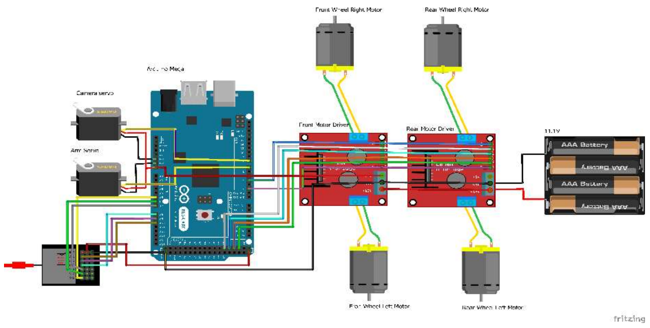

Fig. 4. RC-Bot Circuit Diagram

Operational Modes of the Bot. To optimize the bot's functionality, it is designed with two operational modes - Movement Control and Arms Control.

1. **Movement Control Mode -** In this mode, the bot focuses on navigation. The operator can command the bot to move in various directions and adjust its speed, enabling quick traversals of large spaces and careful maneuvering near pomegranate trees.
2. **Arms Control Mode -** In this mode, the operator has control over the camera's position through the 3D-printed robotic arm. This mode ensures that the camera can capture clear images of pomegranates from multiple angles and distances.

The ability to transition smoothly between these two modes increases the bot's flexibility, making it a highly capable platform for fruit detection and counting.

Table 1. Experimental environment configuration

Category	Configuration
CPU	Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz, 125GB Ram, 1.8TB Disk
GPU	Tesla V100 16GB
Framework	YOLOv8.0.145
Programming Voice	Python 3.10.12, torch-2.0.1

4. Results and Discussion

The authors obtained metrics this included box loss (box_loss), classification loss (cls_loss), distribution focal loss (dfl_loss), precision, recall, mean average precision 50 (mAP50), and mean average precision 50-95 (mAP50-95);

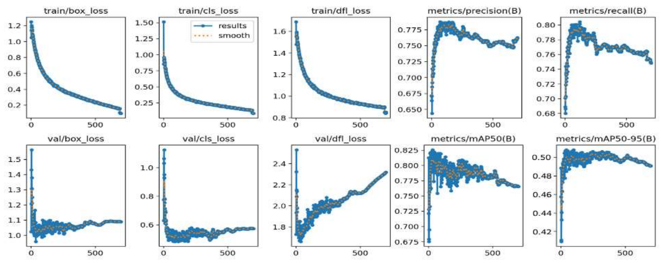

Fig. 5. Results of large model.

The Fig. 5 shows the box loss vs epochs metric graph for the training set and validation set, in the context of object detection models, during the training and validation process, the term "box loss" typically refers to the loss function that measures the distinction between the predicted bounding boxes and the actual ground truth bounding boxes of objects within an image using IoU (Intersection over Union). From the graph, it can be seen how the box loss goes down as the number of epochs for both training and validation sets is increased.

Equation used to calculate IoU.

$$IoU(A,B) = \frac{A \cap B}{A \cup B} \tag{1}$$

For class loss (cls_loss), the authors observed that the training and validating graphs are almost similar signifying model can adjust both training data and new data for validation. Class loss is a component of the overall loss function used during the training and validation of object detection and image classification CNN models. For class loss, the rate of decrease in the loss slows down as the number of epochs increases.

It was observed that Precision, Recall, mAP50, and mAP50-95 increase with the number of epochs (x), but the rate of increase slows down over time, representing the typical behavior of a model.
This research applied a range of YOLO models - Nano, Small, Medium, Large, and Extra-Large, deploying them across multiple formats for inference graphs shown in fig. 6. In terms of inference time, YOLO Nano, being the most lightweight, exhibited the fastest performance across all formats. YOLO Small

offered a commendable balance between speed and accuracy. YOLO Medium provided a slightly slower but notably more accurate inference compared to YOLO Small. YOLO Large and Extra-Large, owing to their increased complexity, demanded more time for inference but delivered superior precision.

For below graphs - Blue, orange, green, red and purple color represents results for nano, medium, large and extra-large models respectively.

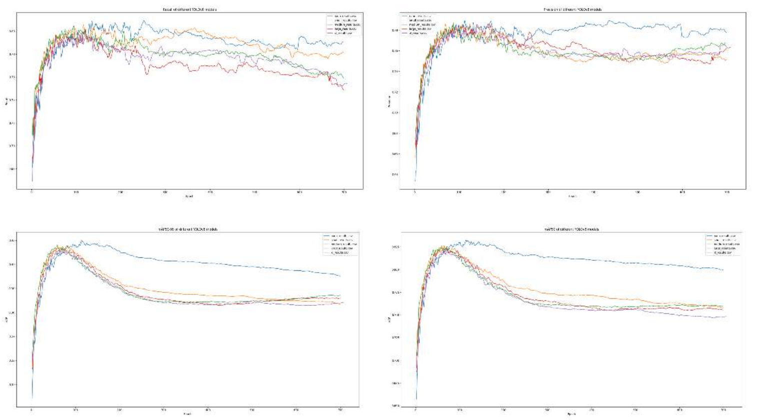

Fig. 6. Graphs for recall, precision, mAP50 and mAP50-95 respectively of all the trained models

Equations used to calculate Precision, Recall, mAP 50 and mAP 50-95

$$Precision = \frac{TP}{TP+F} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad (4)$$

For eqs. (2), (3) and (4) –
TP = True Positive,FN= False Negative, FP = False Positive and
AP = Average Precision (In this case ≈ Precision)

The distinctions among the YOLO models, as deployed in different formats, revolve around three fundamental aspects: model size, inference time, and mAP50-95. First, 'model size' refers to the complexity and architecture of the YOLO variant. Smaller models, like YOLO Nano, comprise fewer layers and parameters, resulting in a lightweight model suitable for resource-constrained devices, while larger models, such as YOLO Extra-Large, are more intricate and sophisticated, demanding substantial computational resources. Second, 'inference time' signifies the duration it takes for a given YOLO model to process and analyze an input image. Faster inference times are vital for real-time applications, and this metric depends on the model's size and the efficiency of the deployment format. Lastly, 'mAP50-95,' or mean average precision across IoU thresholds, quantifies the model's accuracy in object detection. A higher mAP score indicates that the model is better at precisely localizing and classifying objects in an image.

For mAP50-95, the results were consistent with expectations. YOLO Nano, while swift, yielded lower mAP50-95 scores due to its reduced accuracy. YOLO Small offered a solid compromise, providing respectable accuracy scores across deployment formats. YOLO Medium excelled in terms of accuracy but at the cost of slightly increased inference time. YOLO Large and Extra-Large demonstrated the highest accuracy levels but necessitated more significant computational resources, resulting in slower inference times. The choice of deployment format, whether PyTorch, TorchScript, ONNX, OpenVINO, TensorFlow, TensorFlow Lite, TensorFlow.js, or Paddle Paddle, influenced both inference speed and accuracy for each YOLO model variant, reflecting the trade-offs inherent in model selection and deployment.

Table 2. Benchmark of PyTorch models on the experimental environment.

Model	Size (MB)	mAP50-95	Inference Time (ms)
Nano	12.1	0.511	8.1
Small	21.5	0.517	8.92
Medium	49.7	0.517	16.83
Large	83.6	0.513	28.8
Extra Large	130.4	0.520	43.79

In Table 2, it is observed that the inference time for yolo models was directly proportional to the size of the yolo model. For the PyTorch format, yolo nano had smallest model size. And in terms of speed the small model was better considering the size. Overall medium model serves as a good trade-off between model size and inference time.

Model size and model complexity is crucial when deploying models on devices with less computational power. Upon deployment on a Laptop equipped with Nvidia GTX 1650 4GB Graphics Processing Unit (GPU), the medium-sized model yielded optimal outcomes, with an average inference time of approximately 50 milliseconds, while maintaining an average frame rate of 38.Extra-large sized model had the worst average inference time of about 400ms with and average rate of 23.

5. Conclusion

This research presents a comprehensive framework for automated detection and counting of fruits. Through the training and testing of the YOLOv8 model, the authors demonstrated its effectiveness in accurately detecting pomegranates in images. By integrating the BoT SORT tracking algorithm, precise fruit-counting results were achieved. The medium model deployed on laptop with NVIDIA gtx1650 had inference time of about 50ms with accurately counting the fruits showing promising results.

The findings of this manuscript have significant implications for precision agriculture, offering a solution for automating fruit detection and counting. By leveraging the proposed system, corporations and grocery stores can enhance their supply chain by knowing the yield of area.Further research can explore the generalization of this framework to other fruits and automate the bot's navigation. Additionally, high-quality and diverse training data will enhance the accuracy and reliability of the models.

6. Acknowledgment

The authors would like to express their sincere gratitude to everyone who contributed to the success of this project. The authors would like to express their gratitude to Vishwakarma Institute of Technology, Pune for providing the resources to complete this research. Lastly the authors also want to express their gratefulness to Prof. Sangeeta Kurundkar for her unwavering guidance and motivation.

References

[1] Cheng, Wang., Gongping, Yang., Yuwen, Huang., Yikun, Liu., Yan, Zhang. (2023). A transformer-based mask R-CNN for tomato detection and segmentation. Journal of Intelligent and Fuzzy Systems, 44(5), 8585-8595. doi: 10.3233/jifs-222954

[2] Ganaraj, Borkar., Gourav, Kanekar., Parth, Ghaware., Pramesh, Budkuley., Prannoy, Caeiro., Flavia, Leitao., D.S., Vidhya. (2019). Raspberry Pi based Full Fledged Automated Fruit Farm. doi: 10.1109/ICCES45898.2019.9002545

[3] Jan, Bontsema., Jochen, Hemming., Erik, Pekkeriet., Wouter, Saeys., Yael, Edan., A., Shapiro., Marko, Hočevar., Thomas, Hellström., Roberto, Oberti., Manuel, Armada., Heinz, Ulbrich., J., Baur., B., Debilde., Stanley, Best., S., Evain., Wolfgang, Gauchel., Ola, Ringdahl. (2014). CROPS : high tech agricultural robots.

[4] Takeshi, Yoshida., Takanori, Fukao., Takaomi, Hasegawa. (2020). Cutting Point Detection Using a Robot with Point Clouds for Tomato Harvesting. Journal of robotics and mechatronics, doi: 10.20965/JRM.2020.P0437

[5] Ramón, Ceres., Jose, L, Pons., Antonio, Jiménez., J., M., Martin., L., Calderón. (1998). Design and implementation of an aided fruit□harvesting robot (Agribot). Industrial Robot-an International Journal, doi: 10.1108/01439919810232440

[6] S.S., Dheeban., D, V, Harish., A, Hari, Vignesh., M, Prasanna. (2018). Arduino Controlled Gesture Robot. doi: 10.1109/ROMA46407.2018.8986730

[7] Ren, S., He, K., Girshick, R., & Sun, J. (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". arXiv (Cornell University). https://doi.org/10.48550/arxiv.1506.01497

[8] P. Kunchur, V. Pandurangi and M. Hollikeri, "Building Efficient Fruit Detection Model," 2019 1st International Conference on Advances in Information Technology (ICAIT), Chikmagalur, India, 2019, pp. 277-281, doi: 10.1109/ICAIT47043.2019.8987358.

[9] A. Syal, D. Garg and S. Sharma, "Apple fruit detection and counting using computer vision techniques," 2014 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, India, 2014, pp. 1-6, doi: 10.1109/ICCIC.2014.7238364.

[10] Talaat, Fatma M., and Hanaa ZainEldin. "An improved fire detection approach based on YOLO-v8 for smart cities." Neural Computing and Applications 35, no. 28 (2023): 20939-20954. 4 https://doi.org/10.1007/s00521-023-08809-1

[11] Dunn, G., & Martin, S. R. (2008). Yield prediction from digital image analysis: A technique with potential for vineyard assessments prior to harvest. Australian Journal of Grape and Wine Research, 10(3), 196–198. https://doi.org/10.1111/j.1755-0238.2004.tb00022.x

[12] Terven, J. R., & Cordova-Esparza, D. (2023). A comprehensive review of YOLO: from YOLOV1 and beyond. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2304.00501

[13] N Niveditha A Dharun D Rohan Reddy N S G Shiva Prasad Yadav, and Hema. "Crop Yield Prediction Using Image Processing | NVEO - NATURAL VOLATILES & ESSENTIAL OILS Journal | NVEO." Crop Yield Prediction Using Image Processing | NVEO - NATURAL VOLATILES & ESSENTIAL OILS Journal | NVEO, January 1, 2021.

[14] J. -H. Kim, N. Kim and C. S. Won, "High-Speed Drone Detection Based On Yolo-V8," ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Greece, 2023, pp. 1-2, doi: 10.1109/ICASSP49357.2023.10095516.

[15] Li, Yiting, Qingsong Fan, Haisong Huang, Zhenggong Han, and Qiang Gu. "A Modified YOLOv8 Detection Network for UAV Aerial Image Recognition." Drones 7, no. 5 (2023): 304.

[16] Talaat, Fatma M., and Hanaa ZainEldin. "An improved fire detection approach based on YOLO-v8 for smart cities." Neural Computing and Applications 35, no. 28 (2023): 20939-20954.

[17] Koirala, Anand, Kerry B. Walsh, Zhenglin Wang, and Cheryl McCarthy. "Deep learning–Method overview and review of use for fruit detection and yield estimation." Computers and electronics in agriculture 162 (2019): 219-234.

[18] Nuske, Stephen, Supreeth Achar, Terry Bates, Srinivasa Narasimhan, and Sanjiv Singh. "Yield estimation in vineyards by visual grape detection." In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2352-2358. IEEE, 2011.

[19] Dewi, Christine, Rung-Ching Chen, Yan-Ting Liu, Xiaoyi Jiang, and Kristoko Dwi Hartomo. "Yolo V4 for advanced traffic sign recognition with synthetic training data generated by various GAN." IEEE Access 9 (2021): 97228-97242.

[20] Hussain, Muhammad. "YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection." Machines 11, no. 7 (2023): 677.

[21] Chris, Lytridis., Vassilis, G., Kaburlasos., Theodore, Pachidis., Michalis, Manios., Eleni, Vrochidou., Theofanis, Kalampokas., Stamatis, Chatzistamatis. (2021). An Overview of Cooperative Robotics in Agriculture. Agronomy, 11(9), 1818-. doi: 10.3390/AGRONOMY11091818