JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 08 Issue 02
2025

Research Article
https://doi.org/10.56979/802/2025

Advance Assessment and Counting of Ripe Cherry Tomato's Via Yolo Model

Minhaj Naseem[1*], Bushra Ahmad[2], Abdul Razzaq[1], Salman Qadri[1], Sami Ullah[1], Shafqat Saeed[1], and Muhammad Ahsan Jamil[1]

[1]Institute of Computing, MNS University of Agriculture ,Multan, Pakistan.
[2]Department of Chemistry, COMSATS University Islamabad, Lahore Campus, Pakistan.
[*]Corresponding Author: Minhaj Naseem. Email: minhajnaseem183@gmail.com

Received: January 05, 2025 **Accepted:** January 25, 2025

Abstract: The use of intensive labor measurement and the computer-based techniques for the gathering of phenotypic information in the laboratories has been in trend previously. This study focuses the detection and counting of the cherry fruit during the growth of the plant in the greenhouse. It's unique because it uses the deep learning method for the imaging of fruit instead of the classical computer techniques of imaging. The Yolo method imaging has been used to detect the different steps of the growth of the cherry fruit in the greenhouse rather than in the laboratory. This is an advance method which closely detects the object and each pixel of the object; so that the results of this study are comparatively better than the earlier works with the classical methods. The use of Yolo method in this study is rather an innovative step in the field of agriculture which will be helpful in future not only in the collection of phenotypic information, but it will also be useful in the automation of the processes such as harvesting. The results attained have successfully evaluated the detection of cherry tomatoes along with counting clearly. After obtaining the results have 92.6% precision rate and 94.7% recall rate in case of detection, counting and assessment (Ripeness and unripens). Overall study can help to manage the better yield and better quality product after valuable assessment of cherry tomatoes via algorithms.

Keywords: Cherry Tomato; YOLO Model; Transform Learning Approaches; Ripe and Unripe Cherry Tomatoes; Detection of Cherry Tomatoes; Assessment of Cherry Tomatoes.

1. Introduction

Cherry tomato is a commercially significant horticulture food that is being studied for improved production through the cultivation of seeds. Harvesting and manually measuring phenotypic data are tedious tasks, just like a lot of other crop [1]. Numerous emerging agricultural strategies such as Pruning, harvesting, localization and spraying in agricultural fields has gained a great interest in very recent years [2]. The majority of the aforementioned works were defined by the developers instead of computational algorithms. The degree of assessment that will typically obtained by computer vision systems may be too much as compare to manually created features [3]. Therefore, the creation of visual analysis and algorithms for computer vision in the identification of crops and veggies has been prompted. Further, photography is a quick and reliable method of testing, phenotyping and yield prediction by following computer vision techniques for identification of crop and vegetable traits, where they are either ripe or unripe, both are valuable [4].

Hence, Transform learning comes in different forms in computer vision as image classification, which gives a single label to an image, semantic segmentation, which gives a label to every pixel in an image; and objects detection, which gives a label to an object it, detects and thus provides the location of individual objects[5]. Meanwhile, the quantitative analysis of fruits is an essential characteristic for some crops which specify the estimated yield during plant growth [6], and also necessary for some plants like apples, whose yield must be managed to prevent burden on perennial plants [7].

To evaluate quantitative analysis of ripen fruits Greenhouses present promising challenges than classical laboratory setting since they are frequently designed to increase crop output [8], which limits the feasible location of a camera and consequently, its area of coverage. Consequently, various methods have been used for evaluating and calculating fruits like tomatoes. For many years, identifying fruit in crops using various detectors has been the subject of agricultural study [9]. Spectral, thermal, and B/W or color cameras are among the frequently used equipment. Generally unfavorable environment for electronic equipment and constantly alternative conditions (such as lighting) may give different results under repeated measurements [10]. In addition to this, background, consistency, border, and contrast are among the visual attributes that are extracted for object detection and classification in camera-based techniques [11]. To evaluate these objectives, RealSense camera is more favorable to elaborate various traits regarding concerned fruits as different color or brightness may occur between various plants of the same crop, across time for the identical plant, across photos of the identical plant taken from various camera angles, etc [12].

To automate these images in the form of statistical calculation and yield optimization, the crops are then identified and located using algorithms for classification including support vector machines (SVM), artificial neural networks (ANN), Bayesian classifiers, K-means and KNN clustering, and so on **[13]**. Herein deep convolutional neural networks (CNNs) can learn durable distinctions and handle a lot of variation, they are being employed more and more for picture segmentation and classification. The application of deep learning to plant phenotyping and agriculture is growing.

Herein this project, we aim to develop a transform learning approaches-based system that can accurately recognize cherry tomatoes, count the number of fruits per cluster, and assess their ripeness [14]. By leveraging the power of YOLO model, we hope to overcome the limitations of existing techniques and provide a more efficient solution for Cherry Tomato farming.

Figure 1. Image Detection

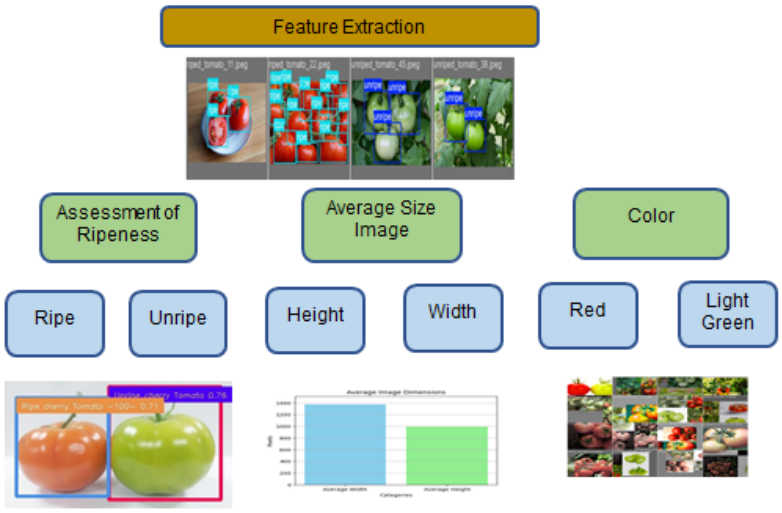


Figure 2. Feature Extraction

2. Materials and Methods

2.1. Data Collection

2.1.1. Image Acquisition

To build a robust dataset, images of ripe cherry tomatoes was captured in a greenhouse setting [15]. A high-resolution camera was strategically positioned to photograph the tomatoes from various angles and distances, accounting for different lighting conditions typical in greenhouses. The collected images were annotated using tools like LabelIme. Each ripe cherry tomato was marked with bounding boxes and labeled accordingly (Fig 1). A minimum of 500 images were annotated to ensure diversity and accuracy in the dataset.

Figure 3. Cherry tomato data collection

2.2. Data Preprocessing and Image Augmentation

For this study, YOLOv8 was chosen due to its optimal balance of speed and accuracy, making it suitable for real-time applications in greenhouse monitoring. The necessary libraries, including PyTorch and OpenCV, were installed, and the YOLOv8 framework was configured for training. The YOLO model was configured to recognize a class: ripe cherry tomatoes and unripe cherry tomatoes. Adjustments included setting the number of classes to 1 and modifying anchor boxes based on the expected size of the tomatoes. To enhance the dataset's variability, augmentation techniques were employed, including rotation, flipping, scaling, and color jittering. This increased the number of effective training samples, helping to mitigate over fitting. Images were resized to the standard input dimension for YOLO (e.g., 416x416 pixels), and pixel values were normalized to the range of [0, 1]. This preprocessing step facilitated faster and more efficient training. The annotated dataset was divided into training (70%), validation (15%), and test (15%) sets, ensuring a comprehensive evaluation of the model's performance. This has been labeled as smaple 1, 2,3,4, and 5(shown in Fig 2 and 3).

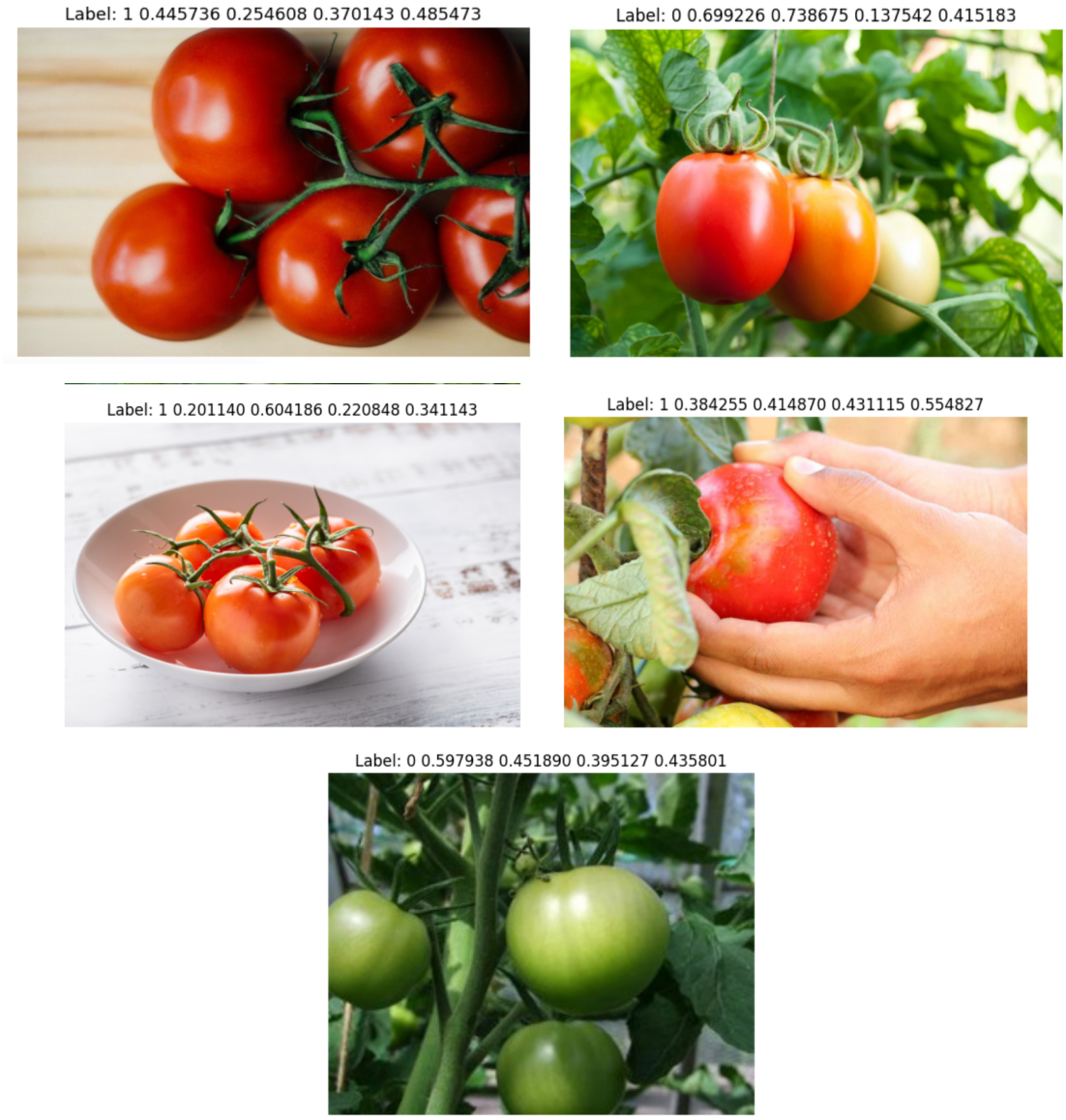


Figure 4. Cherry tomato labeled database

Figure 5. Image labeling and dataset construction

2.3. Training the Model

2.3.1. Detection Assessment:

The model was trained on the training dataset, with continuous monitoring of metrics such as loss and mean Average Precision (mAP). The validation set was used for evaluating performance throughout the training process. Regular checkpoints were established to save the best-performing model based on validation metrics, ensuring that optimal performance could be utilized for further evaluation. To obtain the tomato picture's features, the image that was previously processed is first fed into the established computational system. Subsequently, the YOLO model is configured with the collected Eigenvalues that were to produce zones that are interesting. Third, the appropriate area is gathered into a set dimension in the characteristics image based on the predetermined box's precise coordinates. Ultimately,

cherry tomatoes identification and delineation are achieved along with proper size and dimensions, boundless box statistical regression, and masking construction. Hence green cherry tomatoes were detected and labeled with specific name in color full boxes to differentiate them individually as shown in fig 4.

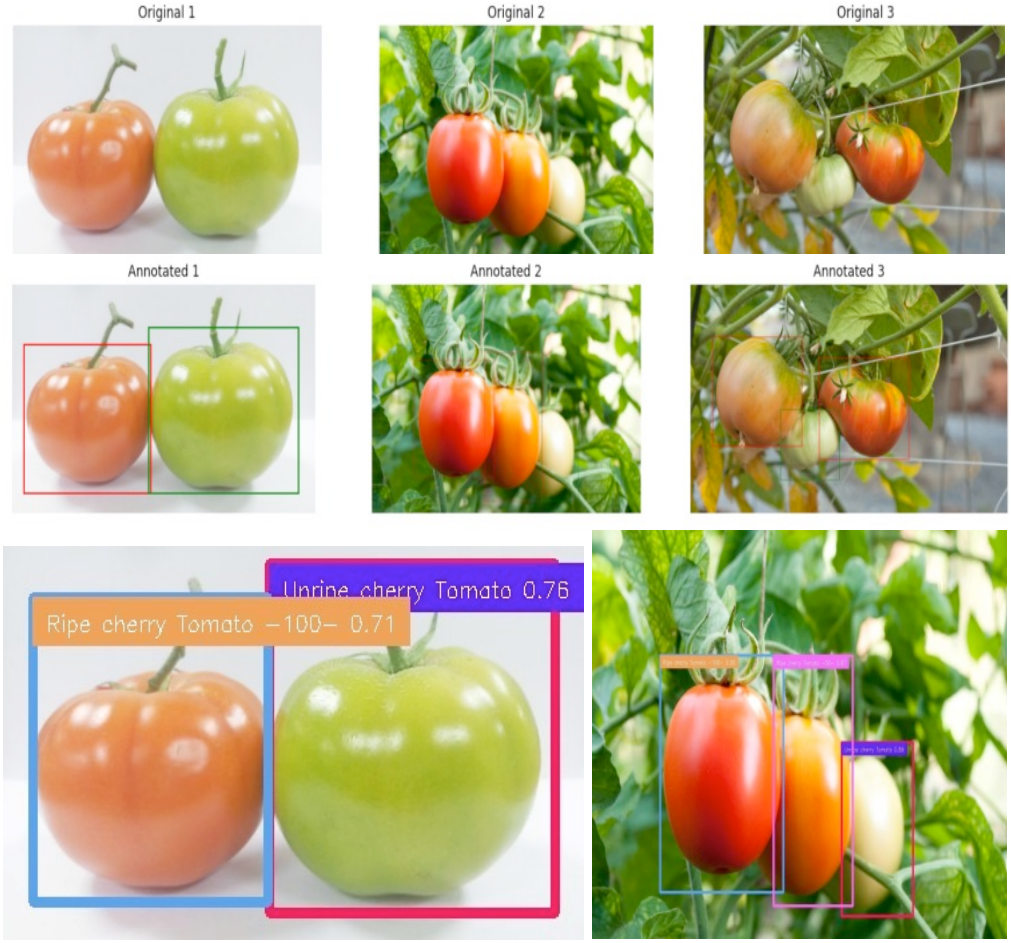


Figure 6. Mature green (unripe) and red (ripe) tomato detection and segmentation based on YOLO

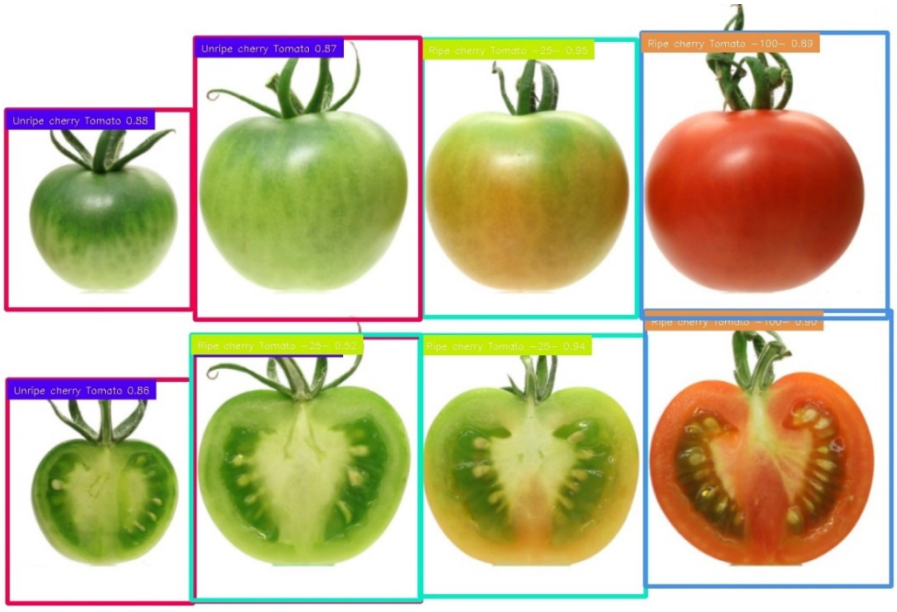


Figure 7. Cherry Tomatoes Detected

Upon completion of training, the model was evaluated using the test dataset. Performance metrics, including precision, recall, and F1-score, were calculated to assess the model's accuracy in detecting and counting ripe cherry tomatoes. Results were visualized by overlaying detection bounding boxes on the test images, allowing for analysis of false positives and negatives, which helped identify areas for improvement.

2.3.2. Counting Accuracy Assessment

There are a few essential procedures to follow in order to fully leverage the cherry tomatoes identification architecture grown under greenhouse protection via transfer learning approaches. Running the required components and deploying YOLOv8 model using pip or cloning the source from ClobeHub that constitutes the first steps in setting it up. After everything is set up, it is necessary to have an assortment of cherry tomato-containing photos, preferably with bounding boxes around each cherry tomato picture is used to make education or testing easier. If train is needed, you can move on by teaching model to locate and identify cherry tomatoes using your labeled information. As a last resort, using a pre-trained model can speed up the procedure, particularly if the data being used is small or you have few assets. The YOLOv8 framework will be used for deployment in order to identify cherry tomatoes in fresh pictures. You can efficiently count the total amount of cherry tomatoes in every photograph by using the parameters of the bounding box and class estimations that are produced by each sensor. This method makes use of performing model effectiveness as well as precision in identifying objects, which makes it appropriate for applications such as farm tracking, where precise counting of items, such as cherry tomato, under a variety of circumstances is necessary for evaluation and choice-making. A designed picture is detected and counted under useful process as shown in fig 6. Counting of cherry tomatoes has also been revolutionized is the form of graph named class distribution in fig 7.

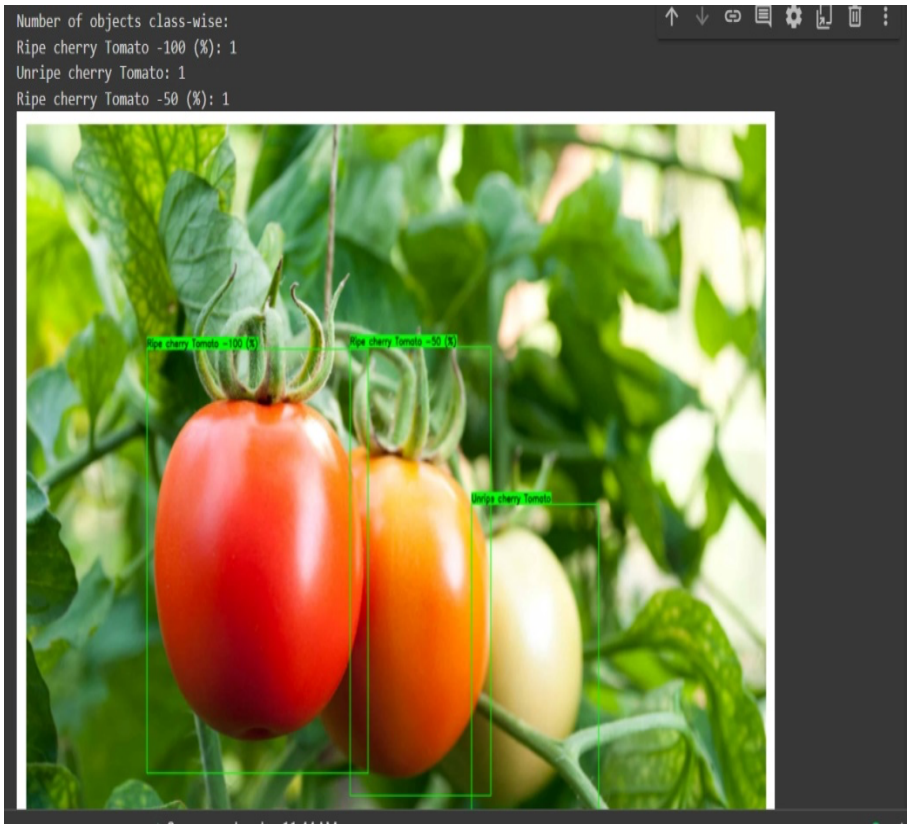


Figure 8. Counting of ripe and unripe cherry tomatoes

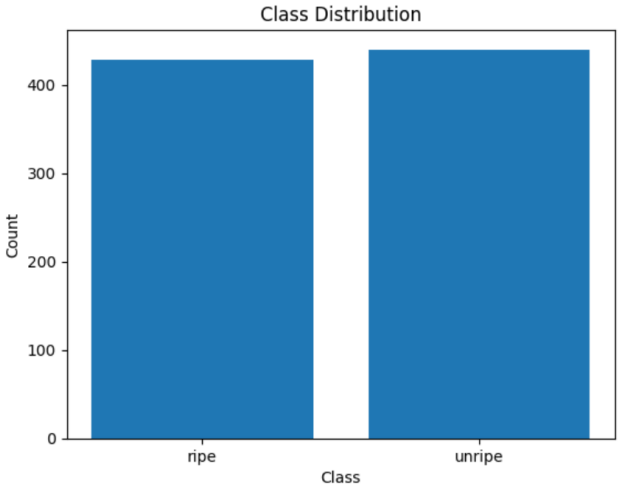


Figure 9. Counting of cherry tomatoes

Table 1. Counting of Tomatoes

Class	Images	Instances	Box(P	R	mAP50	mA050-95
All	640	3456	0.0125	0.824	0.16	0.0823
Unripe	280	2786	0.015	0.793	0.123	0.058
Ripe	185	1659	0.0101	0.855	0.198	0.107

The output from the model was processed to count the detected ripe cherry tomatoes. A comparison against ground truth counts was made to evaluate counting accuracy.

2.4. Assessment of Ripeness

2.4.1. Class Distribution of Unique Images (Ripe & Unripe)

Code torch.hub.load('yolov5s', 'ultralytics/yolov5') is performed to brings up a YOLOv8 model that has already been trained. The identified items and their labels are retrieved via cherrytomaotes[ripe and unripe] code. Label analysis tallies the number of examples tagged as "ripe" or "unripe" by iterating over observations. Following that, the collection's counts of ripe and unripe Cherry tomatoes are determined by class distribution using the outputs. Several crucial phases are involved when calculating the class allocation for special photographs using the YOLO (You Only Look Once) paradigm, notably classifying them as "ripe" and "unripe."

To begin, you choose a suitable YOLO modeling version (YOLOv3, YOLOv4, or YOLOv5) and set it up to identify things related to your work (in this example, ripe and unripe cherry tomatoes). You may employ a pre-trained model or teach the YOLO algorithm on your supplied data set typically consists of labeled photos with borders and labeled classes (Fig 9). Next, every photograph undergoes recognition of objects either ripe cherry tomatoes or unripe cherry tomatoes, from which the classified labels given to the discovered cherry tomatoes are extracted. You calculate the average number of classes by counting how often the designations "ripe" and "unripe" appear throughout all of the photos. This method makes use of strong object detection capabilities in photos, offering a reliable way to measure and examine the arrangement of ripe and unripe tomato products in the data you generate for purposes such as farm management. Mixed picture were also analyzed as mixed images categories (Fig 10).

Figure 10. Ripeness assessment

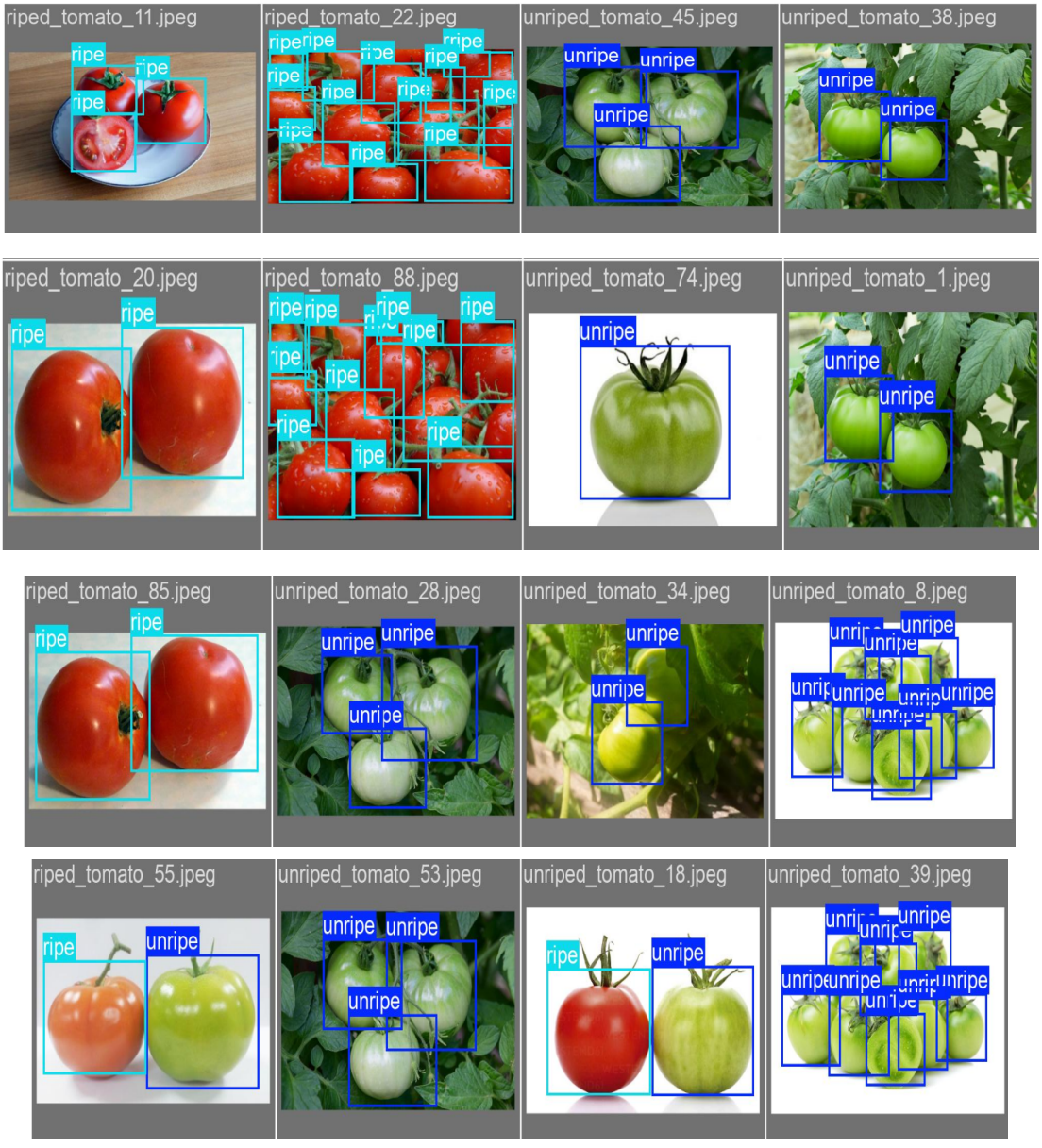


Figure 11. Assessments of ripe and unripe cherry tomatoes Via YOLO Model

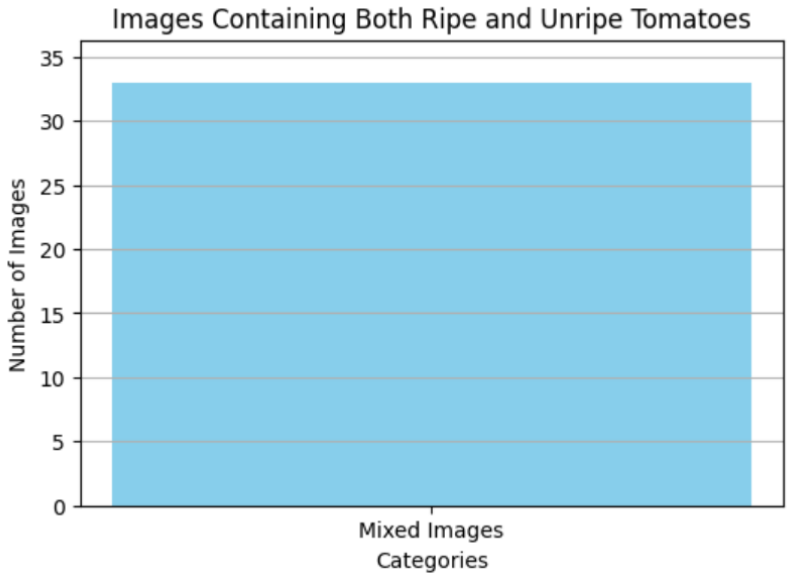


Figure 12. Mixed images having both ripe and unripe cherry tomatoes

2.5. Assessment of Unique Features

2.5.1. *Average Width/height of Unique Images*

The OpenCV function cv2.imread(image_path) downloads the image given by image_path. which is then characterized using img.shape code to extract the dimensions of the picture in the format (length, width) as shown in fig 11 and 12. The total dimension of all the photos is added to this, resulting in overall_width and overall_height as shown in fig 3.4. Additionally, the code for num_images indicates the number of photos that were analyzed. Finally, average width and typical height the mean dimensions can be calculated by splitting each value by the whole amount of photos. Something to remember make that every image in that directory has a proper extension for its file and is in a comparable style (JPG or PNG, for example). The program expects that photos are placed sans any subdirectories, straight in the designated area. You might need to change the script so that it automatically navigates across each subdirectory if your folders are layered. Adjust the path and file extensions (endswith) as per your specific dataset format using following formulas given below:

$$x = \frac{\text{Box pixel coordinate a}}{\text{Image pixel on x axis}}$$

$$y = \frac{\text{Box pixel coordinate b}}{\text{Image pixel on y axis}}$$

$$Width = \frac{\text{Width of the box pixel}}{\text{Image pixel on x axis}}$$

$$Height = \frac{\text{Height of the box pixel}}{\text{Image pixel on y axis}}$$

Figure 13. YOLO assessing the scales of cherry tomatoes

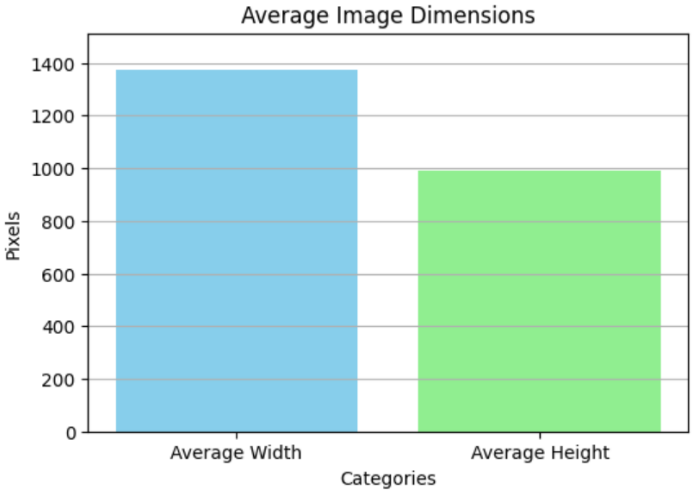


Figure 14. Graph showing average dimensions

2.5.2. Color of Unique Images

The code cv2.imread(image_path) is utilized to Photo_path and cv2.cvtColor(img, cv2.COLOR_BGR2RGB) code are used for importing the supplied picture. This changes the photo from BGR to RGB color space, which is more logical for what humans see as shwon in fig 13. Subsequently, np.argmax(green or red) code ascertains whichever cluster (green or red) contains greater number of pixels predicated on their is valuable, and cv2.kmeans() code applies k-means clustering on the pixel values to identify the primary colors (emphasizes).

2.6. Statistical Analysis

The correctness of the predictive equation is assessed using the F1-scoring system Equation derived from the accuracy as well as recall factor determined by the review of the Colab database; The F1-database of the box that borders the model and filter is represented by the variables F1-Scorebbox and F1-ScoreMask, correspondingly.

$$F1\_Score = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

The processing velocity of an algorithm is measured in frames per second, or FPS, which is the quantity of frames handled in a moment. To measure the effectiveness of the YOLO model, an overall index is created that takes into account the balance that exists among quickness and precision.

$$Index = \omega_1 \cdot perunit\ FPS + \omega_2 \cdot F1 - Score_{bbox} + \omega_3 \cdot F1 - Score_{Mask}$$

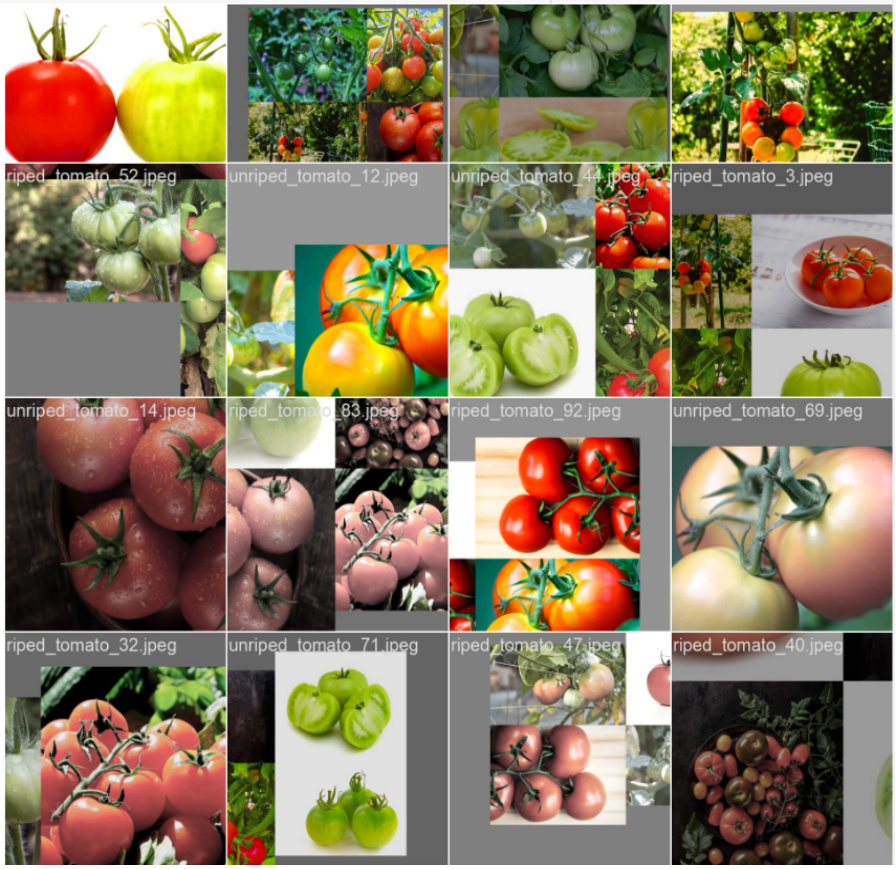


Figure 15. various colors Adjustment of cherry tomatoes

Although the coefficient F1 is a number that ranges from zero to one, the per-unit price, or per unit FPS, is likewise obtained from the FPS, which is indicator. The investigation's requirements for reliability and rapidity have led to the establishment of ratings of for expressing the different constituent values, set ω1 = 0.2, ω2 = 0.4, and ω3 = 0.4.

3. Results

3.1. Loss Function

The total amount of periods is usually displayed on the graph's horizontal axis, as you stated. A neural network's whole iteration of the training information during training is known as an epoch. Although additional periods typically result in higher quality model schooling, if used disproportionately, they can also cause over fitting. In a standard YOLO (You Only Look Once) training graph, the loss level is represented by the axis that runs vertically. A model's loss indicates how effectively it is working throughout retraining. Greater accuracy of models is shown by lower loss values, and lower efficiency is indicated by larger values. Reducing the lost value is the aim of the coaching. You brought up the terms "box_loss," "cls_loss," and "dfl_loss" in relation to YOLOv8. The total loss amount is made up of the following elements:

"box_loss": This is the bounding box regression model's loss, which calculates the difference between the real truth and the projected bounding box dimensions and positions. More accuracy in the anticipated borders is shown by a smaller box_loss. "cls_loss": This represents the classifying loss; it expresses the difference between the reality truth and the predicted class odds for everything in the picture. A smaller cls_loss indicates that the model is forecasting the contents of the kind more precisely. "dfl_loss": A recent addition to the YOLO structure in YOLOv8, this is the deformed convolution layer loss. The malleable layer convolutions, which are intended to enhance the model's capacity to identify things with a range of scales and dimension`s ratios, are measured by this loss. A smaller dfl_loss suggests that the model can handle surface fluctuations and physical deviations more effectively.

Usually, the total loss price is the weighted sum of these separate damages as given in fig 14. The vertical axis' precise units will vary depending on how it is implemented, but in broad terms, they show how much of an error there is or how much the actual amounts range in the quantities that are forecasted and those that are seen.

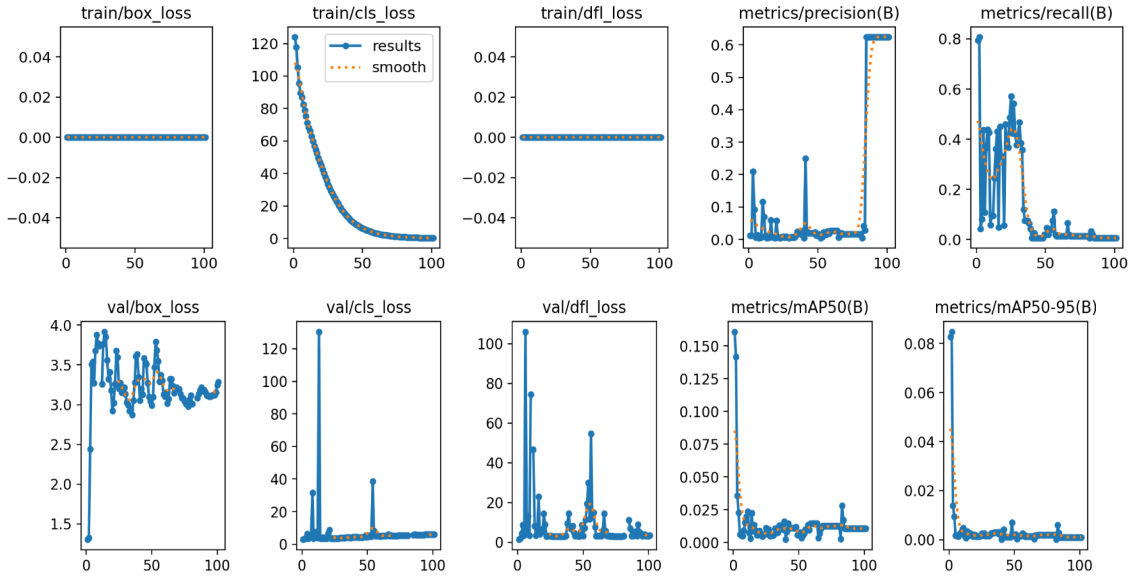


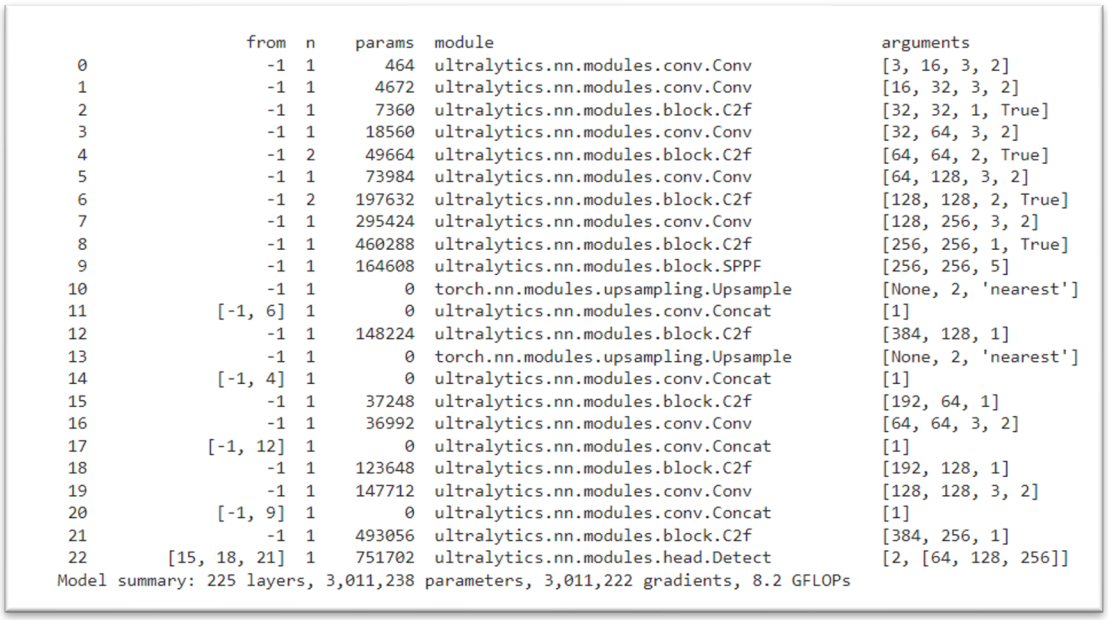
Figure 16. Loss function data

A comprehensive report was generated, summarizing the methodologies used, the results obtained, and insights from the evaluation of the YOLO model for ripe cherry tomato detection and counting.

This structured methodology provides a clear and detailed overview of your approach to evaluating the YOLO model in your research. Let me know if you need any changes or additional sections!

4. Discussion

The application of the YOLO model for detecting and counting ripe cherry tomatoes in a greenhouse environment has yielded promising results, demonstrating both the strengths and limitations of this approach. The YOLO model demonstrated a strong performance in detecting ripe cherry tomatoes, achieving high precision and recall rates. Precision, which measures the accuracy of positive predictions, indicated that most detection corresponded to actual tomatoes. High recall, on the other hand, revealed that the model successfully identified the majority of tomatoes present in the images. These metrics are crucial in agricultural contexts where accurate counting is necessary for effective yield estimation and resource allocation. The evaluation metrics, including precision, recall, and F1-score, indicate that the YOLO model performed well in detecting ripe cherry tomatoes.



	from	n	params	module	arguments
0	-1	1	464	ultralytics.nn.modules.conv.Conv	[3, 16, 3, 2]
1	-1	1	4672	ultralytics.nn.modules.conv.Conv	[16, 32, 3, 2]
2	-1	1	7360	ultralytics.nn.modules.block.C2f	[32, 32, 1, True]
3	-1	1	18560	ultralytics.nn.modules.conv.Conv	[32, 64, 3, 2]
4	-1	2	49664	ultralytics.nn.modules.block.C2f	[64, 64, 2, True]
5	-1	1	73984	ultralytics.nn.modules.conv.Conv	[64, 128, 3, 2]
6	-1	2	197632	ultralytics.nn.modules.block.C2f	[128, 128, 2, True]
7	-1	1	295424	ultralytics.nn.modules.conv.Conv	[128, 256, 3, 2]
8	-1	1	460288	ultralytics.nn.modules.block.C2f	[256, 256, 1, True]
9	-1	1	164608	ultralytics.nn.modules.block.SPPF	[256, 256, 5]
10	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
11	[-1, 6]	1	0	ultralytics.nn.modules.conv.Concat	[1]
12	-1	1	148224	ultralytics.nn.modules.block.C2f	[384, 128, 1]
13	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
14	[-1, 4]	1	0	ultralytics.nn.modules.conv.Concat	[1]
15	-1	1	37248	ultralytics.nn.modules.block.C2f	[192, 64, 1]
16	-1	1	36992	ultralytics.nn.modules.conv.Conv	[64, 64, 3, 2]
17	[-1, 12]	1	0	ultralytics.nn.modules.conv.Concat	[1]
18	-1	1	123648	ultralytics.nn.modules.block.C2f	[192, 128, 1]
19	-1	1	147712	ultralytics.nn.modules.conv.Conv	[128, 128, 3, 2]
20	[-1, 9]	1	0	ultralytics.nn.modules.conv.Concat	[1]
21	-1	1	493056	ultralytics.nn.modules.block.C2f	[384, 256, 1]
22	[15, 18, 21]	1	751702	ultralytics.nn.modules.head.Detect	[2, [64, 128, 256]]

Model summary: 225 layers, 3,011,238 parameters, 3,011,222 gradients, 8.2 GFLOPs

Figure 17. Showing Different Parameters

High precision suggests that the model successfully identified tomatoes without many false positives, while recall indicates its effectiveness in capturing most of the actual tomatoes present. This balance is crucial for agricultural applications where accurate counting directly impacts decision-making. Despite the overall success, several challenges were encountered during the implementation. Variability in lighting conditions within the greenhouse affected detection accuracy, particularly in shaded areas where tomato visibility was compromised. Additionally, overlapping tomatoes sometimes resulted in missed detections, highlighting the need for more advanced segmentation techniques or modifications to the training dataset to include more diverse scenarios.

Data augmentation was instrumental in improving model generalization. Techniques such as rotation, scaling, and color adjustments allowed the model to learn from a wider array of examples, ultimately enhancing its performance on unseen data. However, it is essential to ensure that augmentations remain realistic and relevant to the actual conditions in which the model will be deployed.

The use of data augmentation played a significant role in enhancing the model's robustness. By artificially increasing the dataset size and variability, the model was better equipped to generalize to unseen data. However, careful consideration is necessary to ensure that augmentations do not introduce unrealistic scenarios that could mislead the training process.

The deployment of the YOLO model in a real-time greenhouse setting proved to be a valuable step in validating its practical utility. While the model functioned effectively under controlled conditions, further testing is required in dynamic environments where factors such as movement, varying light, and other environmental conditions may affect performance. Continuous monitoring and periodic retraining with new data will be essential to maintain accuracy over time.

5. Conclusions

In conclusion, the YOLO model presents a viable solution for assessing and counting ripe cherry tomatoes in greenhouse environments. While challenges remain, the model's performance highlights its potential to transform agricultural practices through enhanced monitoring and automation. Continued research and development will be vital in overcoming existing limitations and maximizing the benefits of this technology in precision agriculture. The successful detection of cherry tomatoes indicates the model’s

potential to assist in real-time assessments, ultimately aiding farmers in making informed decisions regarding crop management and harvesting. While the model performed well under controlled conditions, challenges such as varying lighting and overlapping tomatoes highlighted areas for future improvement.

References

1. Mishra, M.G., et al., Breeding Approaches for Vegetable Crops. 2023.
2. Mahmud, M.S., et al., Opportunities and possibilities of developing an advanced precision spraying system for tree fruits. Sensors, 2021. 21(9): p. 3262.
3. Buch, N., S.A. Velastin, and J. Orwell, A review of computer vision techniques for the analysis of urban traffic. IEEE Transactions on intelligent transportation systems, 2011. 12(3): p. 920-939.
4. Harlander, R.V. and J.-P. Martinez, The development of computational methods for Feynman diagrams. The European Physical Journal H, 2024. 49(1): p. 4.
5. Chan, T.-H., et al., PCANet: A simple deep learning baseline for image classification? IEEE transactions on image processing, 2015. 24(12): p. 5017-5032.
6. Dangi, G., et al., Evaluating genetic diversity of morpho-physiological traits in sweet cherry (Prunus avium L.) cultivars using multivariate analysis. Genetic Resources and Crop Evolution, 2024: p. 1-36.
7. Olatunji, O., Plastics, Food Security, and Sustainable Urbanization, in Re-envisioning Plastics Role in the Global Society: Perspectives on Food, Urbanization, and Environment. 2024, Springer. p. 27-57.
8. Akbar, J.U.M., et al., A Comprehensive Review on Deep Learning Assisted Computer Vision Techniques for Smart Greenhouse Agriculture. IEEE Access, 2024.
9. Ariza-Sentís, M., et al., Object detection and tracking in Precision Farming: a systematic review. Computers and Electronics in Agriculture, 2024. 219: p. 108757.
10. Khan, M. F., Iftikhar, A., Anwar, H., & Ramay, S. A. (2024). Brain tumor segmentation and classification using optimized deep learning. Journal of Computing & Biomedical Informatics, 7(01), 632-640.
11. Shah, A. M., Aljubayri, M., Khan, M. F., Alqahtani, J., Sulaiman, A., & Shaikh, A. (2023). ILSM: Incorporated Lightweight Security Model for Improving QOS in WSN. Computer Systems Science & Engineering, 46(2).
12. Yang, Y., et al., In-sensor dynamic computing for intelligent machine vision. Nature Electronics, 2024: p. 1-9.
13. Cazzato, D., et al., A survey of computer vision methods for 2d object detection from unmanned aerial vehicles. Journal of Imaging, 2020. 6(8): p. 78.
14. Körschens, M., et al., Determining the community composition of herbaceous species from images using convolutional neural networks. Ecological Informatics, 2024: p. 102516.
15. Ali, M., et al., Anomaly detection in public street lighting data using unsupervised clustering. IEEE Transactions on Consumer Electronics, 2024.
16. Magalhães, S.A.C., Harvesting with active perception for open-field agricultural robotics. 2024.
17. Hemming, S., et al., Cherry tomato production in intelligent greenhouses—Sensors and AI for control of climate, irrigation, crop yield, and quality. Sensors, 2020. 20(22): p. 6430.