# Deep Learning for Tomato Disease Detection with YOLOv8

**Hafedh Mahmoud Zayani**
Department of Electrical Engineering, College of Engineering, Northern Border University, Saudi Arabia
hafedh.zayani@nbu.edu.sa (corresponding author)

**Ikhlass Ammar**
Computer Science Department, Faculty of Sciences of Tunis (FST), University of Tunis El Manar, Tunisia | OASIS Laboratory, National Engineering School of Tunis, University of Tunis El Manar, Tunisia
ikhlass_ammar@yahoo.fr

**Refka Ghodhbani**
Department of Computer Sciences Faculty of Computing and Information Technology, Northern Border University, Saudi Arabia
refka.ghodhbani@nbu.edu.sa

**Albia Maqbool**
Department of Computer Sciences Faculty of Computing and Information Technology, Northern Border University, Saudi Arabia
albia.alam@nbu.edu.sa

**Taoufik Saidani**
Department of Computer Sciences Faculty of Computing and Information Technology, Northern Border University, Saudi Arabia
taoufik.saidan@nbu.edu.sa

**Jihane Ben Slimane**
Department of Computer Sciences, Faculty of Computing and Information Technology, Northern Border University, Saudi Arabia
jehan.saleh@nbu.edu.sa

**Amani Kachoukh**
Department of Information Systems, Faculty of Computing and Information Technology, Northern Border University, Saudi Arabia
amani.khasookh@nbu.edu.sa

**Marouan Kouki**
Department of Information Systems, Faculty of Computing and Information Technology, Northern Border University, Saudi Arabia
marouan.kouki@nbu.edu.sa

**Mohamed Kallel**
Department of Physics, Faculty of Sciences and Arts, Northern Border University, Saudi Arabia
mohamed.kallel@nbu.edu.sa

**Amjad A. Alsuwaylimi**
Department of Information Technology, College of Computing and Information Technology, Northern Border University, Saudi Arabia
amjad.alsuwaylimi@nbu.edu.sa

**Sami Mohammed Alenezi**
Department of Information Technology, College of Computing and Information Technology, Northern Border University, Saudi Arabia
sami.m.alenezi@nbu.edu.sa

*Received: 13 February 2024 | Revised: 28 February 2024 | Accepted: 29 February 2024*

## ABSTRACT

**Tomato production plays a crucial role in Saudi Arabia, with significant yield variations due to factors such as diseases. While automation offers promising solutions, accurate disease detection remains a challenge. This study proposes a deep learning approach based on the YOLOv8 algorithm for automated tomato disease detection. Augmenting an existing Roboflow dataset, the model achieved an overall accuracy of 66.67%. However, class-specific performance varies, highlighting challenges in differentiating certain diseases. Further research is suggested, focusing on data balancing, exploring alternative architectures, and adopting disease-specific metrics. This work lays the foundation for a robust disease detection system to improve crop yields, quality, and sustainable agriculture in Saudi Arabia.**

***Keywords-disease detection tomato; deep learning; YOLOv8***

## I. INTRODUCTION

Tomatoes, grown worldwide and yielding approximately 170 million tons annually, are essential in vegetable crop production [1]. Despite the harsh environmental conditions in Saudi Arabia, characterized by arid and hot climates, the Greenhouse Horticulture Business Unit of Wageningen University & Research (WUR) has achieved remarkable success, boasting an annual tomato yield of 80 kg/m$^2$ [2]. This achievement is attributed to advanced greenhouse technology and precise irrigation methods. In Saudi Arabia, tomato cultivation spans 1260 hectares of greenhouse area, exhibiting varied productivity ranging from 8-10 kg/m$^2$ under uncontrolled conditions to 35-45 kg/m$^2$ in managed greenhouses, notably lower than in countries such as the Netherlands where yields can exceed 80-90 kg/m$^2$. The disparity in crop yields is influenced by agricultural practices, climate, water quality, and tomato cultivar selection, factors that still lack precise understanding despite the introduction of several tomato varieties annually.

In addition, tomato harvesting faces challenges with damage rates as high as 10% [3-4]. However, advances in automation technology offer promising solutions [5], focusing on image recognition, precise positioning, and efficient picking mechanisms. Traditionally, early fruit recognition relied on complex machine learning algorithms, but the advent of Convolutional Neural Networks (CNNs) has significantly improved efficiency and precision [6-9]. Object detection algorithms, crucial for automating tomato disease detection, include two-stage algorithms such as R-CNN [10] and one-stage algorithms such as SSD [11-12] and YOLO [13], with various iterations of YOLO (v3 [14-15], v4 [16], v5 [17-20], v7 [21], and v8 [22]) that improve object detection capabilities.

This study proposes a deep learning-based approach for tomato disease detection, leveraging a YOLO-based algorithm trained on a custom dataset to improve tomato quality and increase crop yields in remote sensing technologies. The key contributions of this study are:

- Prepare a custom dataset and apply the YOLOv8 algorithm to improve disease detection in tomatoes and potentially increase crop yields.
- Present a carefully trained model that uses a wide array of images, including those collected from databases. The effectiveness of the model is thoroughly evaluated on validation and testing sets, as well as on external photos.

## II. RELATED WORKS

Since the advent of DL algorithms, significant advances have been made in plant disease detection, particularly in leaf disease identification and diagnosis, contributing to the progress of precision agriculture. These developments, driven by computer vision techniques and neural networks, have led to the creation of machine learning and deep learning models tailored to accurately diagnose leaf diseases such as early blight, late blight, bacterial spot, and powdery mildew in various plant species, including tomatoes. Using image processing techniques and CNNs, these models analyze leaf images and classify diseases according to visual symptoms. In addition, spectral and hyperspectral imaging techniques have been explored for early disease detection, bolstering plant health monitoring. Although disease detection efforts have shown high accuracy rates, there is a lag in tomato fruit disease detection. Although many models have been developed to identify fruit diseases such as blossom end rot, fruit rot, and sunscald, the focus remains primarily on leaf diseases. This discrepancy underscores the need to expand research efforts toward fruit disease detection to advance tomato disease management and overall crop productivity. Currently, datasets for tomato fruit disease detection are limited, typically containing fewer than five classes, with less emphasis compared to leaf disease datasets, reflecting the research imbalance between the two domains.

In [23], a 15-layer CNN served as the backbone of the Single-Shot Detector (SSD) to improve the detection of healthy tomato fruits and three classes of tomato fruit diseases. The proposed CNN-SSD model outperformed state-of-the-art models, achieving a significantly higher detection accuracy. In [24], the detection and classification of tomato plant diseases was automated using a Raspberry Pi. This study used image processing techniques and CNN-based classification to successfully detect diseases such as late blight, gray spot, and bacterial canker. In [25], YOLOv5m was used to classify tomato fruits into ripe, immature, and damaged categories. These models demonstrated high prediction accuracy, particularly in ripe and immature tomatoes, offering potential applications in automated tomato fruit harvesting. In [26], an improved YOLOv5 tomato detection algorithm, called CAM-YOLO, was introduced that incorporated a Convolutional Block Attention Module (CBAM) to improve accuracy in detecting small and overlapped tomatoes, achieving an average precision of 88.1%. These studies show ongoing efforts to improve the detection of tomato fruit diseases, paving the way for advances in agricultural practices and crop management.

## III. MATERIALS AND METHODS

### *A. Data Improvement*

This study used a dataset called "balanceddata Dataset" [27] in Roboflow, published in October 2023, and contained three types of tomato diseases. Figure 1 depicts an illustrative image of a class. This dataset was augmented to improve the model performance. Although the original dataset contains 906 images, the augmented dataset consists of 2152 images. The augmented dataset was divided into three sets for training, validation, and testing tasks. Specifically, 87% or 1872 images were allocated for the training set, 9% or 188 images for the validation set, and 4% or 92 images for the testing set. The dataset was partitioned into these subsets with an average ratio of 87:9:4. Figure 2 shows the number of labels corresponding to each class and the size of the labels used in the dataset. YOLOv8 was used as the foundational architecture for the proposed model, as it offers enhanced efficiency and flexibility and addresses three key computer vision tasks:

- Classification: This task involves the identification of a predominant class within an input image, with the model outputting a class index along with a confidence score. Classification proves to be beneficial for discerning the presence of specific classes in an image. YOLOv8 offers five sizes, namely yolov8n (nano), yolov8s (small), yolov8m (medium), yolov8l (large), and yolov8x (extra large), each tailored for specific purposes [28].
- Detection: Building on classification, detection involves identifying various classes present in an image and precisely locating them using bounding boxes.
- Segmentation: Positioned beyond object detection, segmentation delves into identifying individual pixels belonging to objects, offering a more precise understanding compared to object detection alone. This technique finds diverse applications and provides detailed insight into objects within an image.

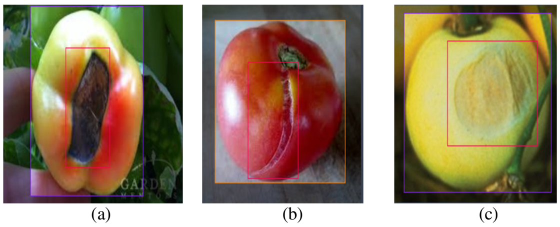

Fig. 1. Dataset classes: (a) Blossom end rot rotation, (b) splitting rotation, (c) sun scaled rotation.

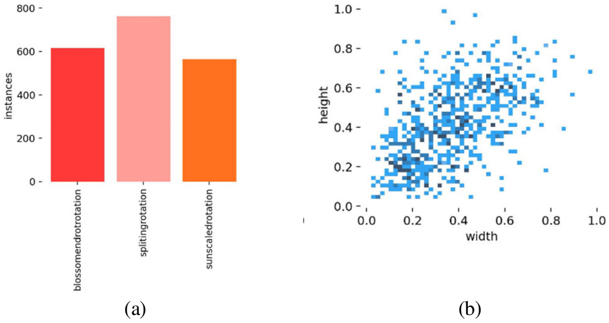

Fig. 2. (a) Number of labels in a class, (b) and different sizes of the labels.

### *B. Architecture of YOLOv8l*

The YOLOv8 design consists of 53 convolutional layers and is implemented as a CNN. In particular, it integrates cross-stage partial connections, enabling improved information transmission between layers for enhanced performance. YOLOv8 stands out for its ability to efficiently detect objects at multiple scales because it employs a feature pyramid network. This network uses multiple layers specifically engineered to detect objects of different sizes and scales within an image, guaranteeing extensive coverage across various object dimensions. The structure of YOLOv8 can be dissected into three main elements [28].

#### *1) Backbone*

This component consists of a pre-trained CNN that is responsible for extracting low-, medium-, and high-level feature maps from the input images. Figure 3 shows the dimensions of the input images in four different sizes. Functionally, this component of YOLOv8 largely approximates that of YOLOv5, with the main divergence being the mosaic data augmentation technique. Mosaic data augmentation includes the simultaneous presentation of many resized images to the model, improving its ability to recognize objects from different viewpoints and in situations where they are partially obscured by merging four images. However, it has been observed that the implementation of mosaic data augmentation in YOLOv8 tends to reduce performance, terminating its discontinuation during the last 10 training epochs. To boost model efficiency, YOLOv5's C3 module was replaced with the C2f module, inspired by YOLOv7's ELAN concept. C2f, shown in Figure 4, unlike C3, which uses only the last output, uses all the outputs of the choke unit. The C2f structure consists of two convolution blocks (this process reduces the number of channels in the feature map usually by half) and three bottleneck convolution blocks. The structural bottleneck blocks, shown in Figure 5, are composed of two convolutional blocks and a contact. This block reduces the calculations and improves the training speed. C2f optimizes the network structure by regulating gradient routes, improving trainability. Additionally, it effectively learns multiscale features and expands receptive fields through feature diversion and multilevel convolution, all without altering the network's shortest and longest gradient paths. This leads to a lighter model with preserved feature richness. Figure 6 shows a backbone block, which comprises a sequence of convolutional layers responsible for extracting significant features from the input image at different sizes. Toward the end of the backbone, the Spatial Pyramid Pooling Fusion (SPPF) layer, followed by convolutional layers, processes features across various scales, while upsample layers improve the resolution of feature maps.

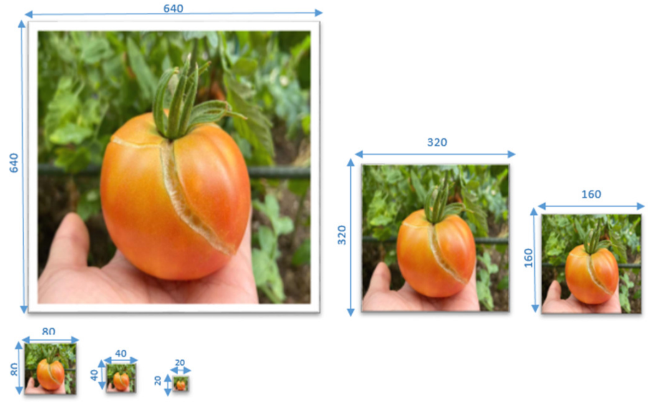

Fig. 3. The different sizes of an image in backbone.

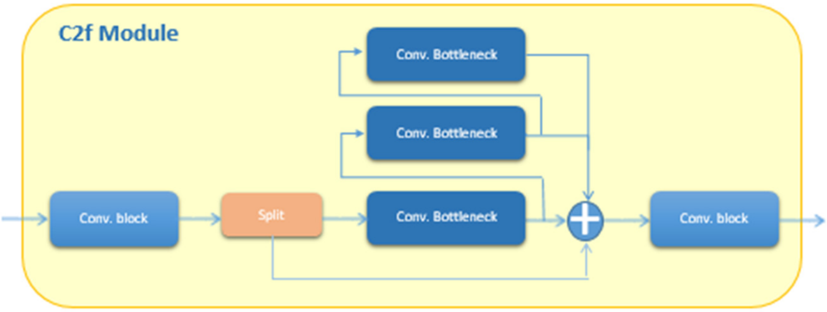

Fig. 4. The structure of the C2f convolutional block.

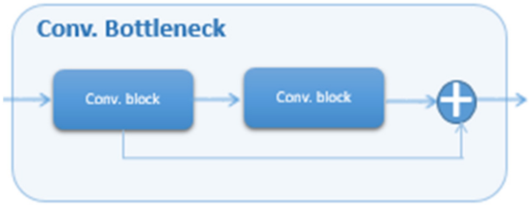

Fig. 5. The structure of the convolutional bottleneck block.

### 2) *Neck*

The neck is the bridge between the backbone's raw feature maps and the head's final predictions. He refines and merges information from the backbone's raw features before feeding it to the head for final predictions. It employs path aggregation blocks, such as FPNs, to optimize multiscale feature integration [29], further enhanced by upsampling and numerous C2f modules. Figure 7 shows three components: the upsampling block that aims to increase the spatial resolution of the feature maps extracted from the backbone, the c2f module that manipulates the feature maps to extract more relevant information for object detection, and the conventional block. These components can extract higher-level features, reduce noise, and improve the model's ability to differentiate between different objects. YOLOv8 also decouples confidence and regression boxes in the later stages of the neck, increasing accuracy. Through these mechanisms, the neck prepares high-quality data for the head, ensuring accurate object detection.

### 3) *Head*

The head of YOLOv8 acts as the decision-maker, taking the refined information from the neck and turning it into actionable results. Unlike previous iterations, the head no longer combines classification and regression tasks. Instead, it performs these tasks independently, resulting in enhanced model performance. The feature maps are transmitted to the head, where objects are classified, and predictions for bounding boxes, objectness scores, and class probabilities are made for detected objects within an image. The head comprises multiple convolutional layers followed by a sequence of fully connected layers, as shown in Figure 8. YOLOv8 takes object detection up a notch with its innovative "anchor-free" approach. Ditching the limitations of predefined shapes, it predicts bounding boxes directly, seamlessly adapting to diverse object sizes and shapes. However, YOLOv8 incorporates a self-attention mechanism that mimics this focus, allowing it to prioritize relevant image features for even more precise detection. This powerful combination of independent analysis and keen attention to detail makes the YOLOv8 head a critical player in delivering accurate results.

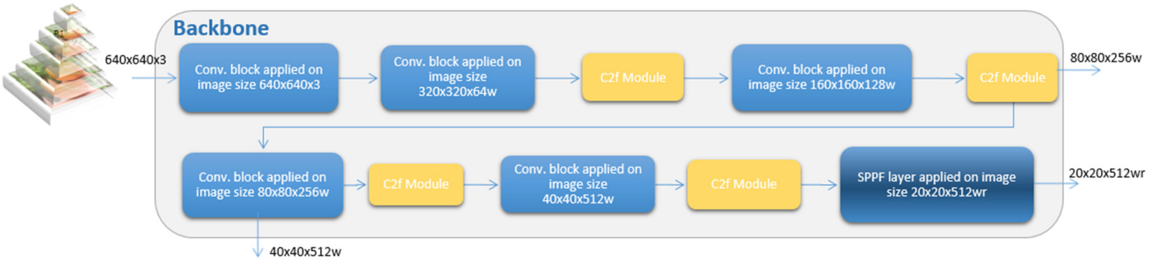
Backbone
640x640x3
Conv. block applied on image size 640x640x3
Conv. block applied on image size 320x320x64w
C2f Module
Conv. block applied on image size 160x160x128w
C2f Module
80x80x256w
Conv. block applied on image size 80x80x256w
C2f Module
Conv. block applied on image size 40x40x512w
C2f Module
SPPF layer applied on image size 20x20x512wr
20x20x512wr
40x40x512w

Fig. 6. The structure of the backbone block.

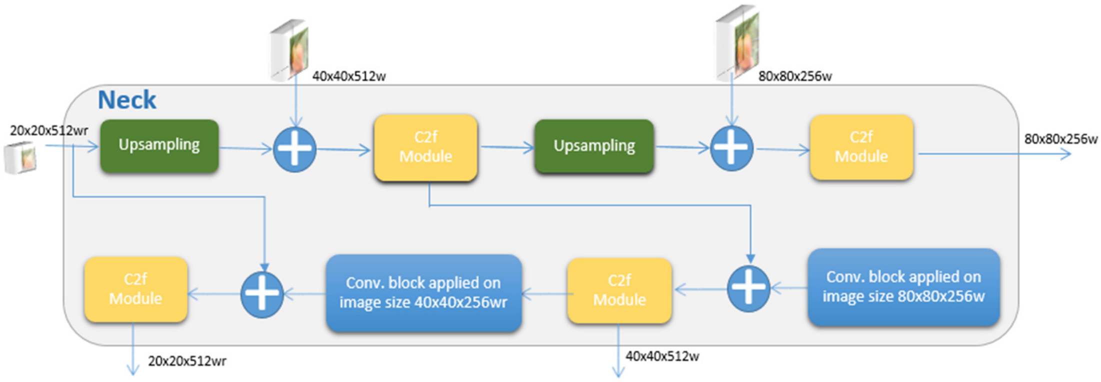

Fig. 7. The structure of the neck.

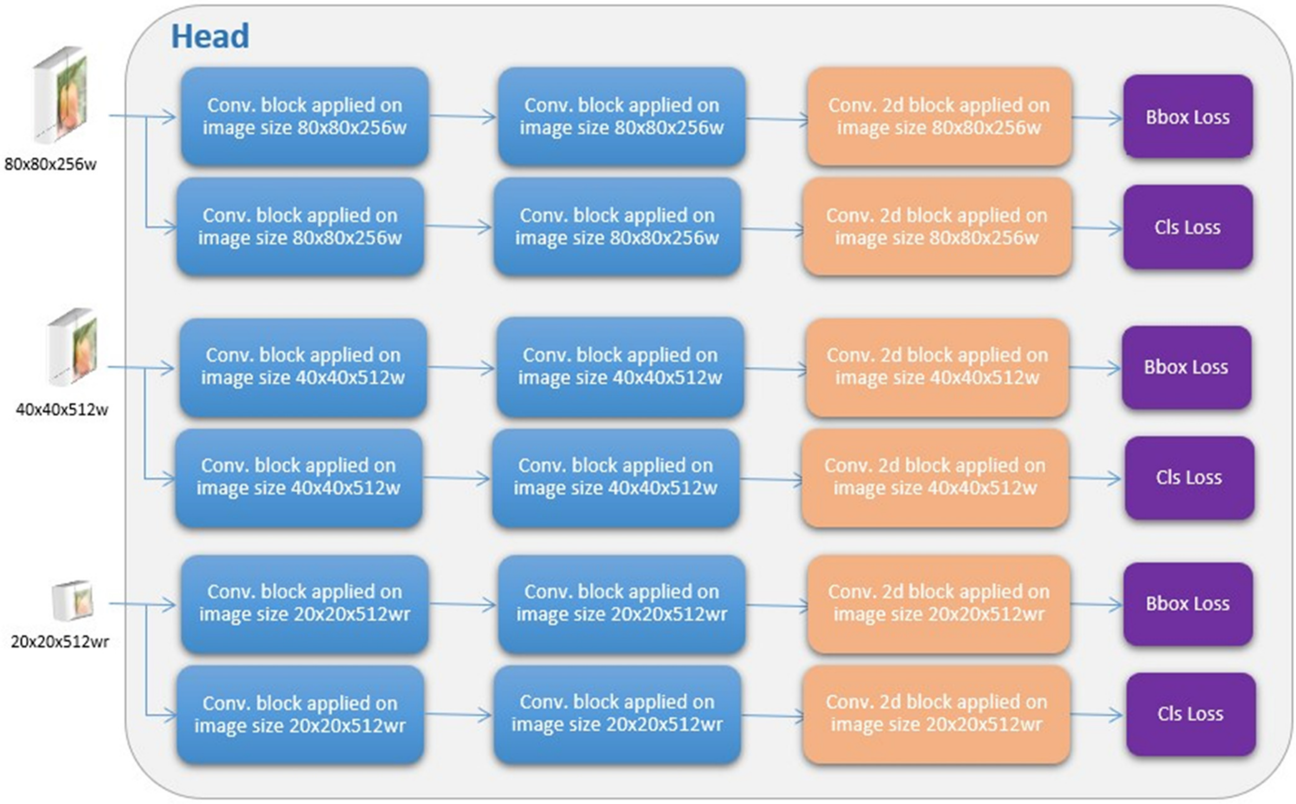
Head
80x80x256w
Conv. block applied on image size 80x80x256w
Conv. block applied on image size 80x80x256w
Conv. 2d block applied on image size 80x80x256w
Bbox Loss
Conv. block applied on image size 80x80x256w
Conv. block applied on image size 80x80x256w
Conv. 2d block applied on image size 80x80x256w
Cls Loss
40x40x512w
Conv. block applied on image size 40x40x512w
Conv. block applied on image size 40x40x512w
Conv. 2d block applied on image size 40x40x512w
Bbox Loss
Conv. block applied on image size 40x40x512w
Conv. block applied on image size 40x40x512w
Conv. 2d block applied on image size 40x40x512w
Cls Loss
20x20x512wr
Conv. block applied on image size 20x20x512wr
Conv. block applied on image size 20x20x512wr
Conv. 2d block applied on image size 20x20x512wr
Bbox Loss
Conv. block applied on image size 20x20x512wr
Conv. block applied on image size 20x20x512wr
Conv. 2d block applied on image size 20x20x512wr
Cls Loss

Fig. 8. The structure of the head block.

## IV. EXPERIMENTAL RESULTS ANALYSIS

### A. Hardware Environment

The experiments were conducted using Google Colab, a free cloud-based platform that offers a convenient notebook environment for research because of its seamless integration with popular machine learning libraries such as TensorFlow and PyTorch. Google Colab provides the flexibility to choose between different hardware options: CPUs, GPUs, and even TPUs, each offering a 12-hour continuous execution window. To maximize performance, the 12GB NVIDIA Tesla T4 GPU was used, achieving significant computational power at no cost. The models were trained for 50 epochs, processing small batches of 3 images at a time. Each image was resized to 640 pixels to balance model performance with training efficiency.

### B. Results

Figure 10 shows the effectiveness of the proposed YOLO-based tomato disease detection algorithm evaluated using precision, recall, and mean Average Precision (mAP). Precision ensures minimal false alarms by measuring the proportion of true positives in detections, reflected by a consistently high precision curve. Recall measures the efficiency in identifying actual diseased plants, depicted by a high recall curve. Finally, mAP combines both, providing a holistic view of performance across various diseases. These metrics were used to objectively evaluate the model's ability to accurately and efficiently detect diseases, minimizing both false positives and missed infections.

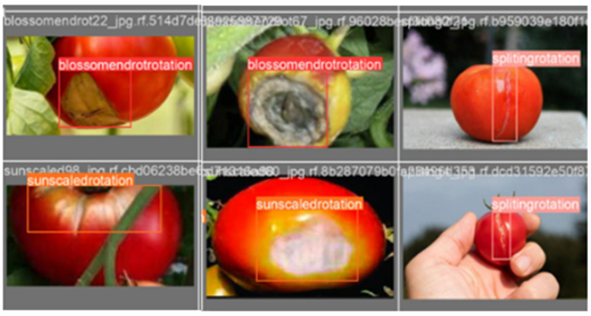

Fig. 9. Some experimental results.

Figure 11(a) shows the confidence-precision curves. Each curve shows how accurate the model is for different object types as its confidence increases. In general, higher confidence leads to higher precision, indicating that the model effectively distinguishes confident detections from less confident ones. Figure 11(b) shows the recall vs. confidence curves, indicating how many true positives the model finds at different confidence levels. Initially, it detects many diseases even with low confidence, suggesting a good sensitivity. However, as the confidence threshold increases, the model becomes stricter. Figure 11(c) shows the precision-recall curve, which visualizes the trade-off between precision and completeness (recall). The model starts by selectively prioritizing accuracy. As it tries to capture more diseases, it includes some false positives. The peak shows the optimal balance, where it identifies many diseases while maintaining good accuracy. The chosen threshold determines the specific precision-recall values. Figure 11(d) shows the F1-confidence curve, which combines precision and recall and is a measure of balanced performance at different confidence levels. The peak indicates the confidence level where the model achieves the best balance between precision and recall for this specific dataset.

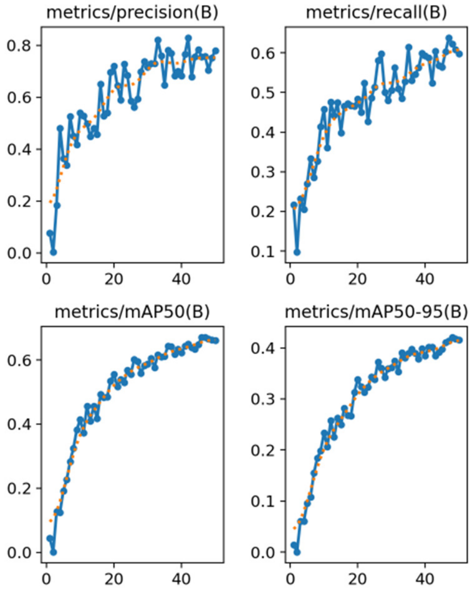

Fig. 10. Performance results.

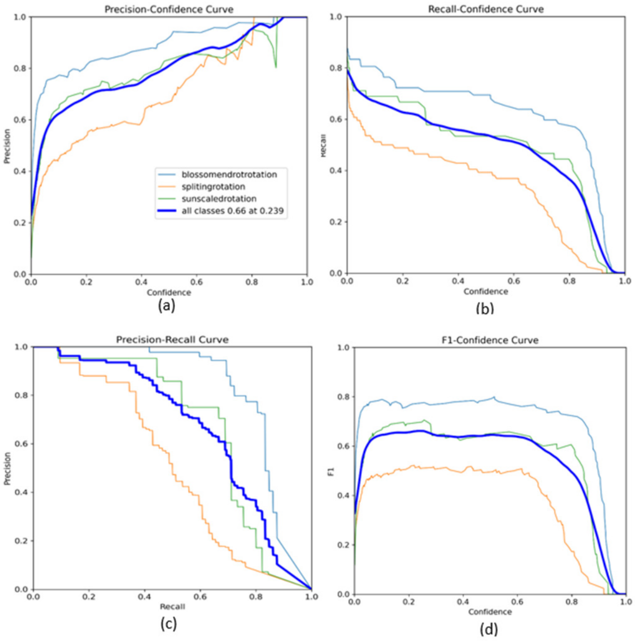

Fig. 11. (a) Precision-confidence, (b) recall-confidence, (c) precision-recall, and (d) F1-confidence curves.

The confusion matrix in Figure 12 shows the model's performance in different types of tomato diseases. Each cell shows how often the model confused one disease for another, or correctly identified it. For example, 51 "blossom end rot rotation" instances were correctly identified, while 8 were misclassified. In general, the model achieved 66.67% accuracy across all diseases. However, this masks differences in performance for each disease. Table I illustrates the precision, recall, and accuracy of each class in the chosen model.

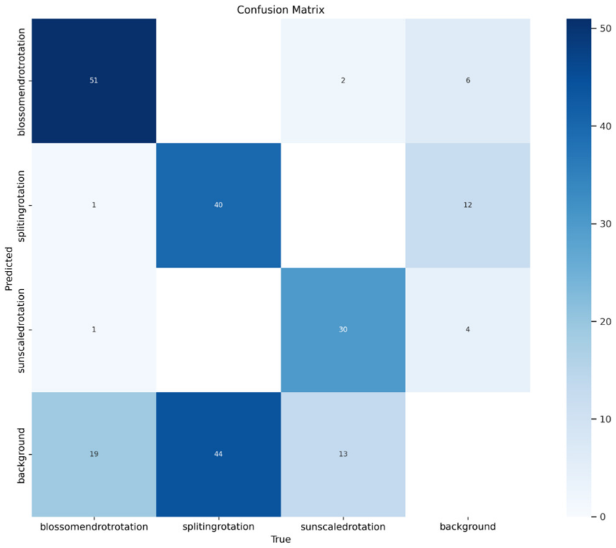

Fig. 12. Confusion matrix.

TABLE I. PRECISION AND RECALL FOR EACH CLASS IN THE MODEL

|  | Blossom end rot rotation | Splitting rotation | Sun scaled rotation |
|---|---|---|---|
| Precision | 86.44% | 74.07% | 85.71% |
| Recall | 70.84% | 90.90% | 66.67% |
| Accuracy | 82.50% | 50.00% | 67.10% |

## V. CONCLUSION

This study trained the YOLOv8 model to automatically detect three tomato fruit diseases. Although the overall accuracy was 66.67%, discriminatory disease detection occurred. Although the splitting rotation was accurately identified, the visual similarity between blossom end rot and sun-scaled rotation posed difficulties. Although the precision for blossom end rot rotation remained satisfactory at 86.44%, the recall of 70.84% indicates the possibility of overlooking genuine instances of the disease. The recall and precision for sun-scaled rotation were 66.67% and 85.71%, respectively, suggesting that there is potential to improve the ability to classify this category with greater accuracy. The model performed exceptionally well in identifying dividing rotation with a high recall of 90.90% and a precision rate of 74.07%. The limitations mentioned can be ascribed to intrinsic visual similarities among diseases, which can complicate the precision of the identification process.

Future work should focus on addressing data imbalance by adding healthy class, data collection or augmentation, exploring alternative architectures or incorporating domain knowledge, and evaluating disease-specific metrics for a more practical perspective. These improvements can pave the way for a robust and reliable automated tomato disease detection system, enhancing crop monitoring and contributing to sustainable agriculture.

## ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA, for funding this research work through project number NBU-FFR-2024-1563-02.

## REFERENCES

[1] J. M. Costa and E. Heuvelink, "The global tomato industry," in *Tomatoes*, CABI, 2018, pp. 1–26.

[2] "Huge increase in tomato production in Saudi Arabia due to targeted watering," Jan. 20, 2020. https://www.wur.nl/en/research-results/research-institutes/plant-research/greenhouse-horticulture/show-greenhouse/huge-increase-in-tomato-production-in-saudi-arabia-due-to-targeted-watering.htm.

[3] I. Tiwari *et al.*, "Post-harvest practices and loss assessment in tomato (Solanum lycopersicum L.) in Kathmandu, Nepal," *Journal of Agriculture and Natural Resources*, vol. 3, no. 2, pp. 335–352, Oct. 2020, https://doi.org/10.3126/janr.v3i2.32545.

[4] M. Benavides, M. Cantón-Garbín, J. A. Sánchez-Molina, and F. Rodríguez, "Automatic Tomato and Peduncle Location System Based on Computer Vision for Use in Robotized Harvesting," *Applied Sciences*, vol. 10, no. 17, 2020, https://doi.org/10.3390/app10175887.

[5] Y. Bai, S. Mao, J. Zhou, and B. Zhang, "Clustered tomato detection and picking point location using machine learning-aided image analysis for automatic robotic harvesting," *Precision Agriculture*, vol. 24, no. 2, pp. 727–743, Apr. 2023, https://doi.org/10.1007/s11119-022-09972-6.

[6] Y. Gulzar, "Fruit Image Classification Model Based on MobileNetV2 with Deep Transfer Learning Technique," *Sustainability*, vol. 15, no. 3, 2023, https://doi.org/10.3390/su15031906.

[7] M. Afonso *et al.*, "Tomato Fruit Detection and Counting in Greenhouses Using Deep Learning," *Frontiers in Plant Science*, vol. 11, 2020, https://doi.org/10.3389/fpls.2020.571299.

[8] G. Moreira, S. A. Magalhães, T. Pinho, F. N. dos Santos, and M. Cunha, "Benchmark of Deep Learning and a Proposed HSV Colour Space Models for the Detection and Classification of Greenhouse Tomato," *Agronomy*, vol. 12, no. 2, Feb. 2022, Art. no. 356, https://doi.org/10.3390/agronomy12020356.

[9] Y. Mu, T. S. Chen, S. Ninomiya, and W. Guo, "Intact Detection of Highly Occluded Immature Tomatoes on Plants Using Deep Learning Techniques," *Sensors*, vol. 20, no. 10, 2020, https://doi.org/10.3390/s20102984.

[10] J. Wu, Z. Kuang, L. Wang, W. Zhang, and G. Wu, "Context-Aware RCNN: A Baseline for Action Detection in Videos," in *Computer Vision – ECCV 2020*, Glascow, UK, Aug. 2020, pp. 440–456, https://doi.org/10.1007/978-3-030-58595-2_27.

[11] F. Zeng, Y. Liu, Y. Ye, J. Zhou, and X. Liu, "A detection method of Edge Coherent Mode based on improved SSD," *Fusion Engineering and Design*, vol. 179, Jun. 2022, Art. no. 113141, https://doi.org/10.1016/j.fusengdes.2022.113141.

[12] H. Peng *et al.*, "General improved SSD model for picking object recognition of multiple fruits in natural environment.," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 34, no. 16, pp. 155–162, 2018.

[13] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, Jan. 2022, https://doi.org/10.1016/j.procs.2022.01.135.

[14] G. Liu, J. C. Nouaze, P. L. Touko Mbouembe, and J. H. Kim, "YOLO-Tomato: A Robust Algorithm for Tomato Detection Based on YOLOv3," *Sensors*, vol. 20, no. 7, 2020, https://doi.org/10.3390/s20072145.

[15] Y. Yang, J. Li, J. Nie, S. Yang, and J. Tang, "Cotton Stubble Detection Based on Improved YOLOv3," *Agronomy*, vol. 13, no. 5, 2023, https://doi.org/10.3390/agronomy13051271.

[16] R. Gai, N. Chen, and H. Yuan, "A detection algorithm for cherry fruits based on the improved YOLO-v4 model," *Neural Computing and Applications*, vol. 35, no. 19, pp. 13895–13906, Jul. 2023, https://doi.org/10.1007/s00521-021-06029-z.

[17] R. Li, Z. Ji, S. Hu, X. Huang, J. Yang, and W. Li, "Tomato Maturity Recognition Model Based on Improved YOLOv5 in Greenhouse," *Agronomy*, vol. 13, no. 2, 2023, https://doi.org/10.3390/agronomy13020603.

[18] R. Rajamohanan and B. C. Latha, "An Optimized YOLO v5 Model for Tomato Leaf Disease Classification with Field Dataset," *Engineering, Technology & Applied Science Research*, vol. 13, no. 6, pp. 12033–12038, Dec. 2023, https://doi.org/10.48084/etasr.6377.

[19] T. Saidani, "Deep Learning Approach: YOLOv5-based Custom Object Detection," *Engineering, Technology & Applied Science Research*, vol. 13, no. 6, pp. 12158–12163, Dec. 2023, https://doi.org/10.48084/etasr.6397.

[20] T. Saidani, R. Ghodhbani, A. Alhomoud, A. Alshammari, H. Zayani, and M. B. Ammar, "Hardware Acceleration for Object Detection using YOLOv5 Deep Learning Algorithm on Xilinx Zynq FPGA Platform," *Engineering, Technology & Applied Science Research*, vol. 14, no. 1, pp. 13066–13071, Feb. 2024, https://doi.org/10.48084/etasr.6761.

[21] J. Zhou, Y. Zhang, and J. Wang, "RDE-YOLOv7: An Improved Model Based on YOLOv7 for Better Performance in Detecting Dragon Fruits," *Agronomy*, vol. 13, no. 4, 2023, https://doi.org/10.3390/agronomy13041042.

[22] G. Yang, J. Wang, Z. Nie, H. Yang, and S. Yu, "A Lightweight YOLOv8 Tomato Detection Algorithm Combining Feature Enhancement and Attention," *Agronomy*, vol. 13, no. 7, 2023, https://doi.org/10.3390/agronomy13071824.

[23] B. N. E. Nyarko, W. Bin, Z. Jinzhi, and J. Odoom, "Tomato fruit disease detection based on improved single shot detection algorithm," *Journal of Plant Protection Research*, vol. 63, no. 4, pp. 405–417, Oct. 2023, https://doi.org/10.24425/jppr.2023.146877.

[24] A. Santosh, S. Bikesh, B. Bibek, and K. K. C. Saban, "Tomato Plant Diseases Detection System Using Image Processing," in *1st KEC Conference Proceedings*, Sep. 2018, pp. 81–86.

[25] Q. H. Phan, V. T. Nguyen, C. H. Lien, T. P. Duong, M. T. K. Hou, and N. B. Le, "Classification of Tomato Fruit Using Yolov5 and Convolutional Neural Network Models," *Plants*, vol. 12, no. 4, 2023, https://doi.org/10.3390/plants12040790.

[26] S. N. Appe, A. G, and B. Gn, "CAM-YOLO: tomato detection and classification based on improved YOLOv5 using combining attention mechanism," *PeerJ Computer Science*, vol. 9, Jul. 2023, Art. no. e1463, https://doi.org/10.7717/peerj-cs.1463.

[27] “balanceddata Computer Vision Project.” Roboflow Universe, Oct. 2023, [Online]. Available: https://universe.roboflow.com/research-proj-mgap9/balanceddata-y4ox0.

[28] S. Mukherjee, "YOLOv8: Pioneering Breakthroughs in Object Detection Technology," *Paperspace Blog*, Nov. 03, 2023. https://blog.paperspace.com/yolov8-a-revolutionary-advancement-in-object-detection-2/.

[29] Z. Zhang, "Drone-YOLO: An Efficient Neural Network Method for Target Detection in Drone Images," *Drones*, vol. 7, no. 8, 2023, https://doi.org/10.3390/drones7080526.