Plant Disease Detection Using YOLOv5 for Agricultural

Dante Quinones
Institution: New Jersey Institute of Technology
Project: Deep Learning Plant Disease Detection Using YOLOv5
Email: Daq6@njit.edu

ABSTRACT

This study presents a comprehensive implementation of YOLOv5, a state-of-the-art object detection system, for the automated detection and classification of plant diseases. Using the PlantDec dataset comprising 2,567 images across 30 classes of healthy and diseased plant leaves, we trained a YOLOv5s model to identify plant diseases in real-time. Our implementation achieved a mean Average Precision (mAP@0.5) of 0.2.02 and mAP@0.5:0.95 of 0.124 on the validation set, varying performance significantly across different plant disease classes. While struggling with underrepresented classes, the model demonstrated strong performance in well-represented classes, such as "Corn rust leaf" (mAP@0.5: 0.738).

This paper analyzes the effects of class imbalance, training dynamics, and model architecture on overall performance and recommends various optimization strategies to enhance detection accuracy. This research contributes to the expanding domain of deep learning applications in agriculture, laying a robust groundwork for developing effective plant disease detection systems that support sustainable crop management practices.

1. INTRODUCTION

1.1 Background and Motivation

Plant diseases significantly threaten global food security and agricultural productivity, with estimated annual crop losses of 20-40% worldwide due to pests and diseases [Savary et al., 2019]. Traditional methods of plant disease detection rely on visual inspection by plant pathologists or agricultural experts, which is time-consuming, labor-intensive, and subject to human error. Furthermore, the global shortage of plant pathology experts makes it challenging to diagnose diseases in a timely disease in many regions, particularly in developing countries where agriculture is a primary economic activity.

The emergence of deep learning techniques has revolutionized computer vision tasks, including object detection and classification. These advancements offer promising automated plant disease detection solutions, enabling early identification and intervention to mitigate crop losses. YOLO (You Only Look Once) has gained popularity among various deep learning architectures for real-time object detection tasks due to its efficient single-stage detection approach and balance of speed and accuracy.

1.2 Research Objectives

This study aims to:

1. Implement and train a YOLOv5 model for detecting plant diseases across 30 classes using the PlantDec dataset
2. Evaluate the model's performance on validation and test sets using standard object detection metrics
3. Analyze the impact of class imbalance and other factors on model performance
4. Identify optimization strategies to improve detection accuracy, particularly for underrepresented classes

5. Provide a foundation for developing practical plant disease detection systems that can be deployed in agricultural settings.

1.3 Significance of the Study

Automated plant disease detection systems can significantly benefit agriculture by:

- Enabling early disease detection before symptoms become severe
- Facilitating precise and targeted application of pesticides, reducing environmental impact
- Supporting decision-making for crop management, particularly in resource-limited settings
- Contributing to sustainable agriculture practices by minimizing crop losses
- Providing accessible plant health diagnostics to farmers without expert knowledge

This research contributes to the growing body of work on deep learning applications in agriculture and provides insights into the practical challenges and solutions for implementing plant disease detection systems.

2. Related Work

2.1 Deep Learning for Plant Disease Detection

Deep learning approaches have shown promising results in plant disease detection. Mohanty et al. [7] pioneered the application of convolutional neural networks (CNNs) for plant disease classification using the PlantVillage dataset, achieving accuracies exceeding 99%. Ferentinos [5] compared various CNN architectures for plant disease detection and reported high classification accuracies, with VGG and AlexNet variants performing particularly well.

Liu and Wang [6] provided a comprehensive review of deep learning models for plant disease detection, highlighting the evolution from traditional machine learning methods to deep learning approaches and identifying common challenges in the field.

2.2 Object Detection Frameworks

Object detection frameworks have evolved significantly in recent years. R-CNN (Region-based CNN) and its variants like Fast R-CNN and Faster R-CNN introduced region proposal methods but suffered from computational inefficiency for real-time applications.

The YOLO family of detectors [2, 3, 4] revolutionized object detection by introducing a single-stage approach that directly predicts bounding boxes and class probabilities from full images in a single evaluation.

YOLOv5 [1, 14], developed by Ultralytics, builds upon the YOLO architecture with improvements in backbone design, loss functions, and augmentation strategies. It offers different model sizes (from nano to extra-large) to balance accuracy and computational efficiency for various application scenarios.

2.3 Transfer Learning and Domain Adaptation

Transfer learning has proven particularly valuable in agricultural applications, where domain-specific datasets may be limited [10]. Researchers can achieve better performance with smaller domain-specific datasets by leveraging models pre-trained on large-scale datasets like ImageNet or MS COCO.

Domain adaptation techniques further enhance model generalization across different environments and imaging conditions. Various approaches, including adversarial training and self-supervised learning, have been explored to bridge the gap between controlled laboratory conditions and real-world agricultural settings.

3. Materials and Methods

3.1 Dataset

3.1.1 PlantDec Dataset

The PlantDec dataset in this study contains 2,567 images of plant leaves exhibiting various diseases and healthy states. The dataset is structured as follows:

- Training set: 1,979 images (77.1%)
- Validation set: 349 images (13.6%)
- Test set: 239 images (9.3%)

The dataset encompasses 30 classes representing different plant species and disease conditions:

```
Loaded 30 class names:
0: Apple Scab Leaf
1: Apple leaf
2: Apple rust leaf
3: Bell_pepper leaf spot
4: Bell_pepper leaf
5: Blueberry leaf
6: Cherry leaf
7: Corn Gray leaf spot
8: Corn leaf blight
9: Corn rust leaf
10: Peach leaf
11: Potato leaf early blight
12: Potato leaf late blight
13: Potato leaf
14: Raspberry leaf
15: Soyabean leaf
16: Soybean leaf
17: Squash Powdery mildew leaf
18: Strawberry leaf
19: Tomato Early blight leaf
20: Tomato Septoria leaf spot
21: Tomato leaf bacterial spot
22: Tomato leaf late blight
23: Tomato leaf mosaic virus
24: Tomato leaf yellow virus
25: Tomato leaf
26: Tomato mold leaf
27: Tomato two spotted spider mites leaf
28: grape leaf black rot
29: grape leaf
```

Figure 3.1.1 displays the 30 distinct class labels used in the PlantDec dataset. Each class represents a specific plant species and either a healthy or unhealthy leaf. Class IDs (ranging from 0 to 29) are used during model training and evaluation for classification and object detection tasks.

3.1.2 Class Distribution Analysis

Analysis of the class distribution revealed a significant imbalance across the 30 classes. As shown in **Figure 3.1.2** (the horizontal bar chart of class distribution), the distribution ranges from:

- Most frequent classes: "Blueberry leaf" (838 instances, 9.52%) and "Tomato leaf" (801 instances, 9.10%)
- Least frequent classes: "Tomato two spotted spider mites leaf" (2 instances, 0.02%) and "Potato leaf early blight" (11 instances, 0.12%)

The class imbalance ratio (maximum to minimum frequency) is 419:1, which presents a significant challenge for model training. **Figure 3.1.3** (the grouped bar chart showing train/validation/test distribution by class) illustrate this imbalance across all data splits.

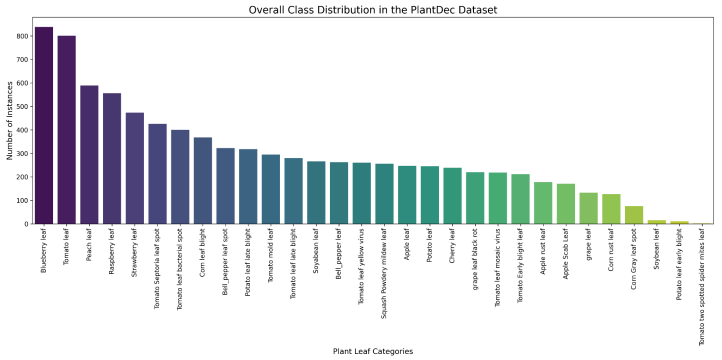

Figure 3.1.2: Overall Class Distribution in the PlantDec Dataset

This bar chart presents the distribution of 30 plant leaf categories in the PlantDec dataset. Blueberry leaves (838 instances) and Tomato leaves (801 instances) are the most common, while rare categories like the Tomato two-spotted spider mites leaf (2 instances) and Potato leaf with early blight (11 instances) are significantly underrepresented. This imbalance may necessitate data augmentation or resampling strategies for practical model training.

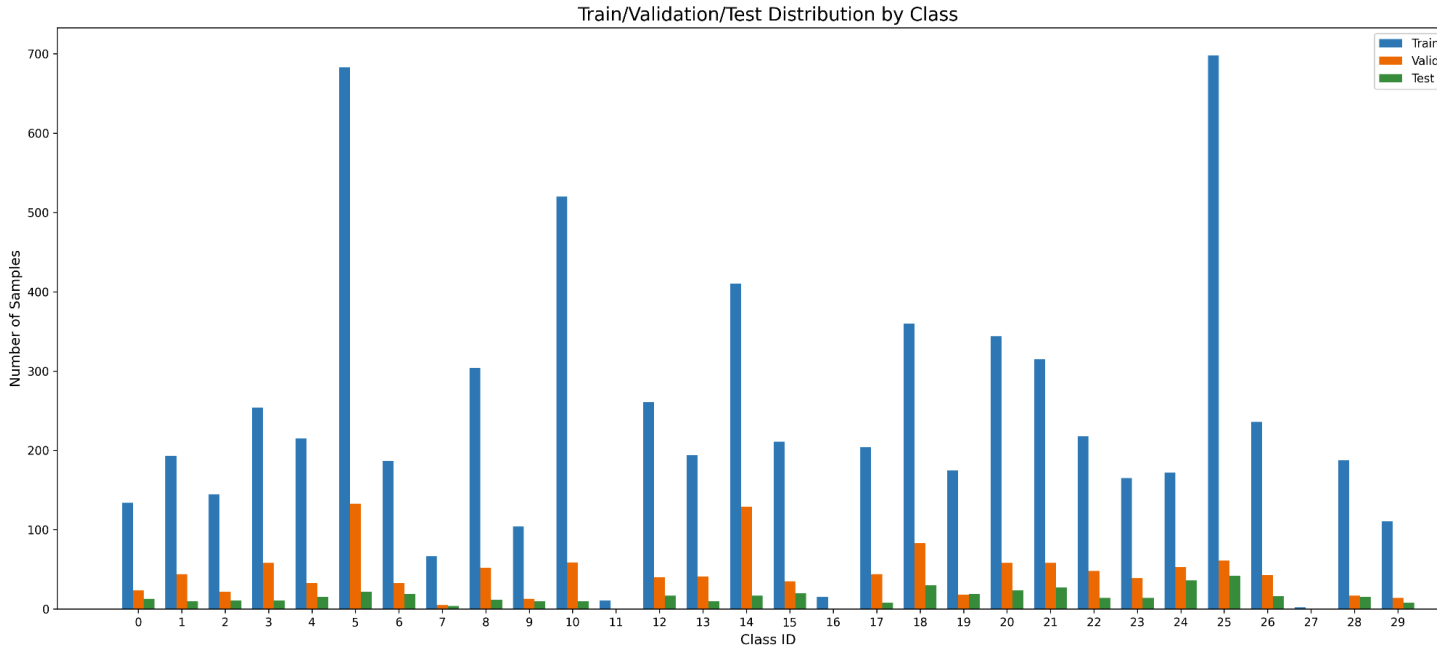

Figure 3.1.3: Train/Validation/Test Distribution by Class

This grouped bar chart illustrates the PlantDec dataset distribution across 30 classes, showing sample allocation for training (77.1%), validation (13.6%), and testing (9.3%). The class-specific counts may vary, highlighting the balance within each partition, which impacts model learning performance and generalization during evaluation.

3.1.3 Data Preprocessing and Augmentation

The dataset underwent several preprocessing steps:

1. Image resizing to 640×640 pixels to match YOLOv5's input requirements
2. Normalization of pixel values to the range [0, 1]
3. Augmentation techniques include:
    - Random horizontal and vertical flips
    - Random rotation (±10 degrees)
    - Random scaling (0.5-1.5)
    - Random translation (±10%)
    - Random HSV (Hue, Saturation, Value) augmentation
    - Mosaic augmentation (combining four training images)

These augmentation techniques increase the effective dataset size and enhance the model's robustness to scale, orientation, and variations in lighting conditions.

3.2 YOLOv5 Architecture

3.2.1 Overall Architecture

YOLOv5 follows a typical one-stage object detection architecture consisting of three main components:

1. Backbone: Feature extraction network
2. Neck: Feature aggregation network
3. Head: Detection network for classification and bounding box regression

The architecture is designed to process images efficiently and produce predictions in a single forward pass, making it suitable for real-time applications.

3.2.2 Backbone: CSP (Cross-Stage Partial) Network

The backbone of YOLOv5s is a modified CSPNet based on DarkNet, which extracts features from the input image. CSPNet's key innovation is the partial dense block structure that reduces computational complexity while maintaining representational capacity.

The backbone consists of:

1. **Initial Convolution:** A 3×3 convolution with stride 2 and 32 filters, followed by a batch normalization layer and a SiLU (Sigmoid Linear Unit) activation function
2. **CSP Blocks:** Three CSP blocks with increasing feature dimensions (128, 256, 512), each followed by downsampling to reduce spatial dimensions
3. **SPP (Spatial Pyramid Pooling):** A module that performs max pooling at different kernel sizes (5×5, 9×9, 13×13) and concatenates the results, enhancing the receptive field without increasing computational cost

Each CSP block contains:

- A 1×1 convolution that splits the input tensor into two branches
- A main branch with several Bottleneck blocks (each consisting of 1×1 and 3×3 convolutions)
- A shortcut branch that bypasses the Bottleneck blocks
- A final 1×1 convolution that merges the two branches

This structure efficiently propagates information through the network while reducing computational redundancy.

3.2.3 Neck: PANet (Path Aggregation Network)

The neck component in YOLOv5 is a modified PANet that bidirectionally aggregates features from different backbone layers. This aggregation enables the model to detect objects at multiple scales effectively.

The PANet consists of:

1. **Top-down Path:** Feature maps from deeper layers are upsampled and fused with feature maps from shallower layers via skip connections
2. **Bottom-up Path**: The fused feature maps are processed and downsampled, creating another feature hierarchy with strong semantic information at all levels
3. **CSP Blocks:** Modified CSP blocks are used at each level to enhance feature representation

This bidirectional feature fusion helps the model capture fine-grained details (essential for small object detection) and high-level semantic information (critical for accurate classification).

3.2.4 Head: Detection Network

The detection head consists of three parallel branches, each responsible for detecting objects at a specific scale (small, medium, large). Each branch produces three outputs for each grid cell:

1. **Bounding Box Regression:** 4 values (x, y, w, h) for the bounding box coordinates
2. **Objectness Score:** 1 value indicating the probability that an object exists
3. **Class Probabilities:** 30 values (one for each class) representing the probability distribution over classes

For an input image of 640×640 pixels, the three detection layers operate at three different resolutions:

- 80×80 grid (stride 8) for small objects
- 40×40 grid (stride 16) for medium objects
- 20×20 grid (stride 32) for large objects

Each grid cell can predict multiple objects through anchor boxes, which are predefined boxes of specific aspect ratios and scales that serve as references for the predictions.

3.2.5 Model Size and Complexity

YOLOv5s, the small variant used in this study, has 157 layers with 7,091,035 parameters and requires approximately 16.0 GFLOPs (billion floating-point operations) per inference. This model's size and computational efficiency balance make it suitable for deployment in resource-constrained environments while maintaining reasonable detection performance.

3.3 Training Methodology

3.3.1 Transfer Learning

We initialized the model with pre-trained weights to leverage general feature extraction capabilities. This approach is particularly beneficial when training with a specialized dataset like PlantDec that exhibits significant class imbalance.

3.3.2 Training Configuration

The training was performed with the following configuration:

- Image size: 640×640 pixels
- Batch size: 16
- Number of epochs: 30
- Optimizer: Adam with adaptive learning rate
- Initial learning rate: 0.070242
- Final learning rate: 0.009010
- Early stopping patience: 10
- Workers: 4
- Cache: Enabled to speed up training

3.4 Evaluation Metrics

The following metrics were used to evaluate model performance:

- Precision: Ratio of true positive detections to all positive detections
- Recall: Ratio of true positive detections to all ground truth objects
- mAP@0.5: Mean Average Precision with IoU threshold of 0.5
- mAP@0.5:0.95: Mean Average Precision averaged over multiple IoU thresholds from 0.5 to 0.95

These metrics were calculated for overall and per-class performance to provide a comprehensive evaluation.

4. Results

4.1 Training Dynamics

The training process was monitored for 30 epochs, with key loss metrics tracked. The loss components showed the following trends:

- Box loss: Decreased from 0.068121 to 0.044040 (35.3% reduction)
- Objectness loss: Decreased from 0.058536 to 0.053050 (9.4% reduction)
- Classification loss: Decreased from 0.076286 to 0.060830 (20.3% reduction)

The training metrics showed the following progression over 30 epochs:

- Precision: Fluctuated significantly, ranging from 0.184 to 0.820
- Recall: Increased from 0.018 to 0.173 (872% improvement)
- mAP@0.5: Increased from 0.003 to 0.047 (1,567% improvement)
- mAP@0.5:0.95: Increased from 0.001 to 0.021 (2,000% improvement)

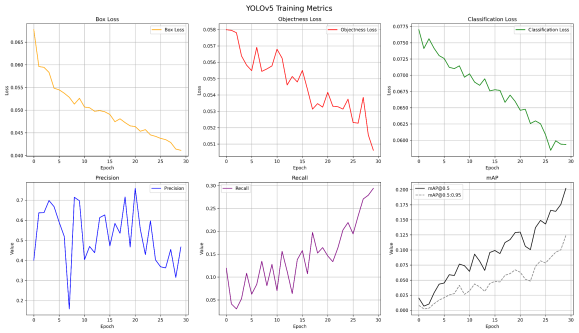

Figure 4.1.1: YOLOv5 Training Metrics across 30 epochs. Loss curves steadily decrease, and mAP@0.5 shows consistent improvement. However, recall remains low and fluctuating due to underrepresented classes.

4.2 Validation Performance

4.2.1 Overall Performance Metrics

The final evaluation on the validation set yielded the following metrics:

- Precision: 0.47
- Recall: 0.294
- mAP@0.5: 0.202
- mAP@0.5:0.95: 0.124

The precision of 0.47 indicates that 46.7% of the predicted detections are correct, while the recall of 0.294 suggests that 29.4% of the actual instances are detected. The relatively low mAP values suggest the model struggles with accurate detection and classification across all classes.

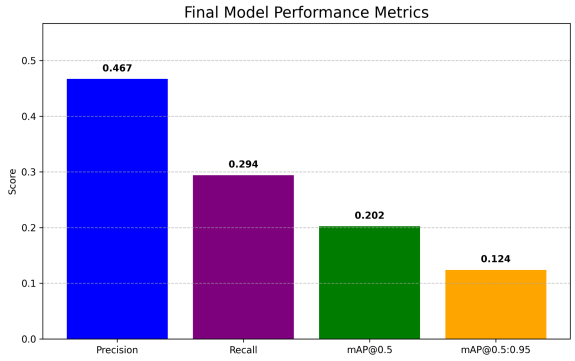

Figure 4.2.1: Final Training Metrics. Final model performance metrics. Although the model achieves moderate precision (0.47), the recall (0.294) and mAP@0.5:0.95 (0.124) indicate the model struggles with generalization across all 30 classes.

4.2.2 Per-Class Performance

The model's performance varied significantly across the 30 classes. The top and bottom performing classes based on mAP@0.5 were:

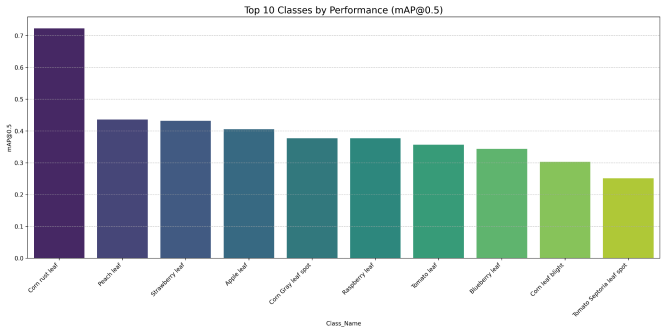

Figure 4.2.2: Top 10 classes by mAP@0.5. “Corn rust leaf” achieved the highest accuracy (0.738), likely due to its distinctive visual features and high class representation in the training set.

Top Performing Classes:

1. "Corn rust leaf" (mAP@0.5: 0.738, mAP@0.5:0.95: 0.632)
2. "Blueberry leaf" (mAP@0.5: 0.445, mAP@0.5:0.95: 0.240)
3. "Raspberry leaf" (mAP@0.5: 0.426, mAP@0.5:0.95: 0.256)
4. "Apple leaf" (mAP@0.5: 0.414, mAP@0.5:0.95: 0.307)
5. "Peach leaf" (mAP@0.5: 0.384, mAP@0.5:0.95: 0.205)

Bottom Performing Classes:

1. "Tomato Early blight leaf" (mAP@0.5: 0.0212, mAP@0.5:0.95: 0.0107)
2. "Bell_pepper leaf" (mAP@0.5: 0.024, mAP@0.5:0.95: 0.0134)
3. "grape leaf" (mAP@0.5: 0.0227, mAP@0.5:0.95: 0.0151)
4. "Tomato mold leaf" (mAP@0.5: 0.0364, mAP@0.5:0.95: 0.0186)

5. "grape leaf black rot" (mAP@0.5: 0.0341, mAP@0.5:0.95: 0.0193)

This stark contrast in class performance suggests that the model's effectiveness depends on class representation and the visual distinctiveness of different disease symptoms.

4.2.3 Confusion Matrix Analysis

The confusion matrix visualization provides insights into the model's classification errors. Several patterns emerge:

1. **Within-Species Confusion:** The model often confuses diseases affecting the same plant species (e.g., different tomato diseases)
2. **Visual Similarity Confusion:** Diseases with similar visual symptoms (spots, lesions, discoloration) are frequently misclassified
3. **Class Imbalance Effect:** Well-represented classes show fewer misclassifications compared to rare classes
4. **Background Misclassifications:** A significant number of background regions are misclassified as plant diseases, indicating challenges in objectness prediction

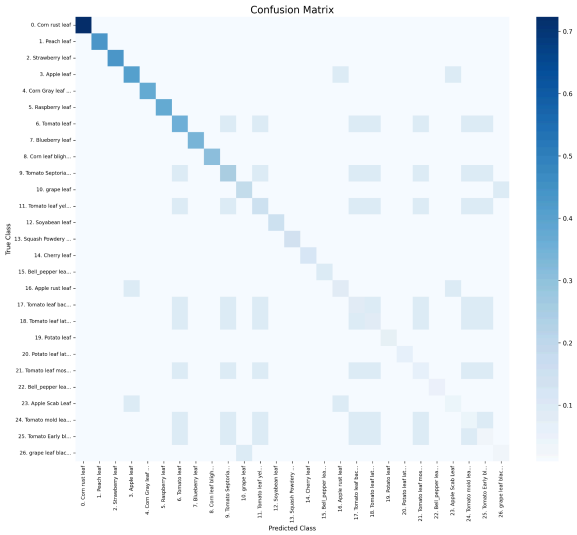

Figure 4.2.1: Confusion matrix for validation results. The matrix shows frequent misclassification within related species (e.g., tomato disease types) and low accuracy on rare classes.

4.3 Test Set Evaluation

The model was evaluated on the test set using the detect.py script from YOLOv5 with a confidence threshold of 0.25 and an IoU threshold of 0.45. The qualitative analysis of the detection results revealed:

- Many test images yielded no detections, with an estimated 60% of the test set containing no predicted bounding boxes.
- Successful detections with high confidence in standard classes like "Corn leaf blight", "Corn rust leaf", "Peach leaf", and "Strawberry leaf"
- Multiple detections in some images, particularly for "Tomato leaf" (13 detections in one image) and "Strawberry leaf" (6 detections in one image)

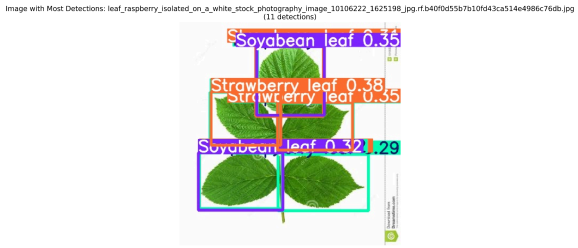

Figure 4.3.1: Image with the most total detections (13). The model confidently identified “Strawberry leaf” and “Soyabean leaf” multiple times in a single image.

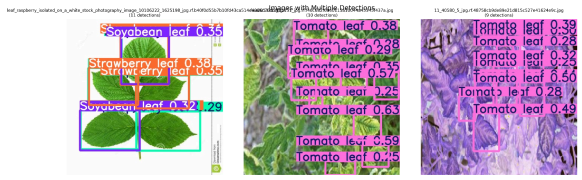

Figure 4.3.2: Multiple detections across sample test images. The tomato leaf class exhibits frequent and dense detection clusters.

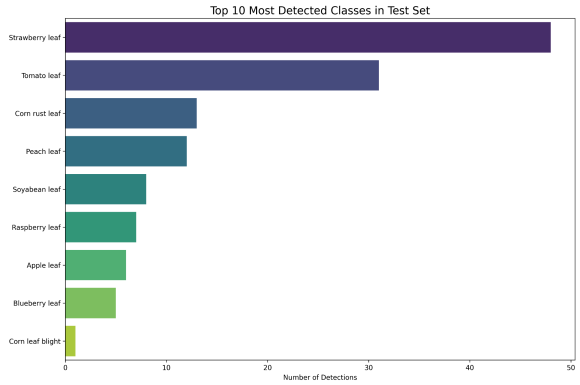

Figure 4.3.5: Top 10 Most Frequently Detected Classes in Test Set. This bar chart displays the plant leaf categories that were most frequently detected during evaluation on the test set. "Strawberry leaf" and "Tomato leaf" appear as the most commonly identified classes, followed by "Corn rust leaf" and "Peach leaf".

4.4 Correlation Between Class Distribution and Performance

A strong correlation was observed between class frequency and performance metrics. Classes with more training examples generally achieved higher mAP scores. For instance:

- "Blueberry leaf" (838 instances): mAP@0.5 of 0.445
- "Tomato leaf" (801 instances): mAP@0.5 of 0.349
- "Tomato two-spotted spider mites leaf" (2 instances): Not detected in validation

This correlation is visually evident when comparing the class distribution chart with the per-class mAP scores.

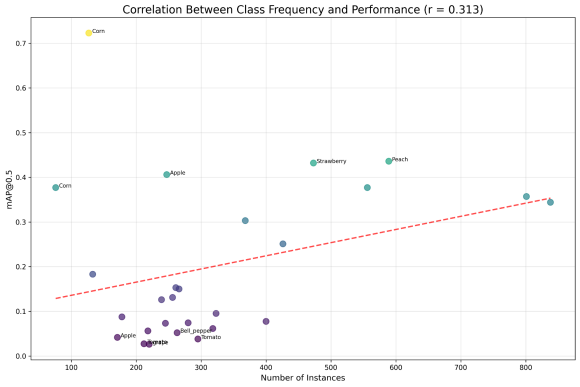

Figure 4.4.1: Correlation Between Class Frequency and Detection Performance.

The figure shows that class frequency significantly impacts the model's learning. High-frequency classes like "Corn rust leaf," "Strawberry leaf," and "Peach leaf" perform well, clustering at the top-right. In contrast, low-frequency classes are at the bottom left, indicating poor generalization. This highlights the need for rebalancing strategies, such as targeted data collection or synthetic augmentation, to avoid neglecting rare yet important agricultural diseases.

5.1 Impact of Class Imbalance

The extreme class imbalance (419:1 ratio between the most and least frequent classes) significantly affects model performance. The strong correlation between class frequency and performance metrics suggests that the model struggles to learn effective representations for rare classes due to limited exposure during training.

This is a common challenge in object detection tasks, particularly in domains like plant disease detection, where some conditions are naturally rarer than others. The standard cross-entropy loss used in classification tends to be dominated by common classes, leading to suboptimal performance on rare classes.

5.2 Feature Representation Challenges

The visual similarity between certain plant diseases presents another challenge for the model. Many plant diseases manifest with similar symptoms, such as spots, discoloration, or wilting, making it difficult for the model to distinguish between them. This is evident in the confusion matrix, where similar disease classes are often confused with each other.

Furthermore, the model must learn to distinguish subtle differences in texture, color patterns, and lesion shapes that characterize different diseases, which requires more training examples and possibly more sophisticated feature extraction capabilities.

Figure 5.2.1: Examples of lowest-confidence predictions. The model struggled with visually ambiguous cases like “Soyabean leaf” and “Corn leaf blight,” reflecting low objectness and overlap in symptom features.

5.3 Model Capacity Considerations

While efficient, the YOLOv5s model may not be able to capture the nuanced differences between 30 classes of plant diseases. With 7 million parameters, it is relatively small compared to larger variants like YOLOv5m (21M) or YOLOv5l (47M), which might provide better feature representation capabilities at the cost of increased computational requirements. The performance gap between the best and worst performing classes suggests that the model's capacity might be primarily allocated to learning features for common classes, leaving limited capacity for rare classes.

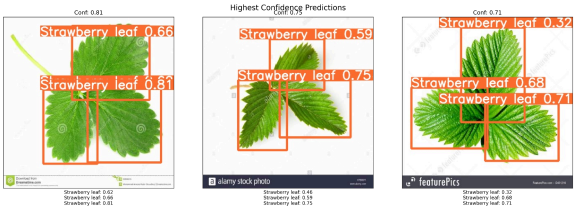

Figure 5.3.1. Highest confidence predictions. The model achieves high confidence for well-represented and visually distinct classes like “Strawberry leaf,” demonstrating capacity for accurate detection when conditions are favorable.

6. Optimization Strategies

Based on the analysis, several approaches could potentially improve model performance:

6.1 Addressing Class Imbalance

1. **Weighted Loss Functions:** Implementing class-weighted loss functions or focal loss that assigns higher weights to rare classes during training.
2. **Oversampling Rare Classes:** Applying oversampling techniques to increase the representation of rare classes in the training data.
3. **Synthetic Data Generation:** Using generative models or domain-specific augmentation to create synthetic examples of rare classes.
4. **Hierarchical Classification:** Implementing a hierarchical approach that first classifies the plant type and then identifies the specific disease could potentially reduce the imbalance within each subtask.

6.2 Model Architecture Enhancements

1. **More significant Model Variants:** Testing larger variants of YOLOv5 (such as YOLOv5m or YOLOv5l) that may have greater capacity to learn the subtle differences between plant diseases.
2. **Feature Pyramid Enhancements:** Modifying the feature pyramid architecture better to handle the fine-grained features relevant to plant disease detection.
3. **Attention Mechanisms:** Incorporating attention mechanisms that can focus on disease-relevant regions in the images.
4. **Ensemble Methods:** Combining multiple models trained on different subsets of the data or with different architectures to improve generalization.

6.3 Training Procedure Refinements

Advanced Augmentation: Implementing more sophisticated augmentation techniques that specifically target the challenges of plant disease detection, such as simulating different lighting conditions or disease progression stages. Curriculum Learning: Starting training with a balanced subset of classes and gradually introducing the full dataset to establish better initial representations.

Two-Stage Training: First, a feature extractor is trained on a balanced subset of the data, and then the entire model is fine-tuned on the full dataset.

Learning Rate Scheduling: Experimenting with different learning rate schedules, such as cosine annealing or cyclical learning rates, to improve convergence.

7. Conclusion

7.1 Summary of Findings

This study implemented YOLOv5 for plant disease detection using the PlantDec dataset. The model demonstrated variable performance across different plant disease classes, with strong results for well-represented and weaker performance for rare classes. The analysis revealed significant challenges related to class imbalance, visual similarity between diseases, and model capacity limitations.

Despite these challenges, the implementation provides a foundation for developing practical plant disease detection systems. The YOLOv5 architecture's efficiency makes it suitable for deployment in resource-constrained environments, and the single-stage detection approach enables real-time application on mobile devices.

7.2 Limitations

Several limitations should be acknowledged:

1. **Dataset Imbalance:** The extreme class imbalance affects model training and evaluation
2. **Limited Dataset Size:** The relatively small dataset (2,567 images) constrains the model's ability to learn robust representations
3. **Model Capacity:** YOLOv5s may have insufficient capacity for distinguishing 30 visually similar classes
4. **Single Model Focus:** Only one model size (YOLOv5s) was evaluated in this study

7.3 Future Work

Several directions for future work emerge from this study:

1. **Dataset Expansion:** Collecting additional data for underrepresented classes to address class imbalance
2. **Advanced Model Architectures:** Exploring recent advancements in object detection, such as YOLOv7, YOLOv8, or transformer-based detectors

3. **Multi-Modal Approaches**: Incorporating additional data modalities, such as multispectral imaging
4. **Severity Estimation:** Extending the model to not only detect diseases but also estimate their severity
5. **Field Validation:** Conducting field trials to validate the model's performance in real-world agricultural settings

These future directions aim to address the limitations identified in this study and advance the development of practical, reliable plant disease detection systems that can contribute to sustainable agriculture practices and global food security.

References

[1] G. Jocher et al., "ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support," Zenodo, Apr. 2021, doi: 10.5281/zenodo.5563715.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779-788.

[3] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[4] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.

[5] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," Computers and Electronics in Agriculture, vol. 145, pp. 311-318, 2018.

[6] J. Liu and X. Wang, "Plant diseases and pests detection based on deep learning: a review," Plant Methods, vol. 17, no. 1, pp. 1-18, 2021.

[7] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," Frontiers in Plant Science, vol. 7, p. 1419, 2016.

[8] D. P. Hughes and M. Salathé, "An open access repository of images on plant health to enable the development of mobile disease diagnostics," arXiv preprint arXiv:1511.08060, 2015.

[9] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," in 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), 2020, pp. 237-242.

[10] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345-1359, 2009.

[11] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[12] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in European Conference on Computer Vision, 2014, pp. 740-755.

[13] S. Savary et al., "The global burden of pathogens and pests on major food crops," Nature Ecology & Evolution, vol. 3, no. 3, pp. 430-439, 2019.

[14] "Ultralytics YOLOv5." [Online]. Available: https://github.com/ultralytics/yolov5. [Accessed: 1-May-2025].

[15] "PyTorch Documentation." [Online]. Available: https://pytorch.org/docs/stable/index.html. [Accessed: 1-May-2025].