AgriVision

An Interactive Demonstrator for Spatial and Frequency Domain Enhancement on Deep Learning-based Plant Disease Recognition

Reetam Dan, Shadan Ahmad
Team: AgriVision AI

Course: CSET344 - Image and Video Processing
Lab Instructor: Dr. Anurag Choubey

November 27, 2025

Abstract

AgriVision presents a comprehensive web-based platform that bridges classical image processing techniques with modern deep learning for agricultural disease diagnosis. The system implements eight distinct enhancement algorithms across spatial and frequency domains, demonstrating their effectiveness on a CNN model trained on 87,000+ plant disease images. Through intelligent preprocessing recommendations and real-time comparison frameworks, the platform achieves up to 48% improvement in prediction confidence on degraded field images while maintaining sub-250ms response time. The implementation validates that targeted preprocessing remains essential for robust computer vision systems, particularly in real-world agricultural deployment scenarios.

Project Description

AgriVision is a Flask-based web application implementing a dual-pipeline architecture to evaluate and compare various spatial and frequency domain image enhancement algorithms on a Convolutional Neural Network (CNN) model trained for multi-class plant disease classification. The complete source code is available on GitHub.

System Architecture

- **Backend:** Flask RESTful API
- **Frontend:** HTML, CSS, JavaScript
- **Deep Learning Framework:** TensorFlow 2.15.0 / Keras
- **Image Processing:** OpenCV 4.8.1, NumPy 1.24.3, SciPy 1.11.4

Dataset Description

Source and Composition The project utilizes the *New Plant Diseases Dataset (Augmented)* from PlantVillage[1], available on Kaggle, comprising high-resolution RGB images of plant leaves across 38 distinct disease categories spanning multiple crop species including *Solanum lycopersicum* (tomato), *Malus domestica* (apple), *Zea mays* (corn), and *Solanum tuberosum* (potato).

Parameter	Value
Total Images	∼87,000
Number of Classes	38
Image Format	JPEG (RGB)
Spatial Resolution	Variable (256×256 to 512×512 pixels)
Training/Validation Split	80% / 20%
Color Space	sRGB
Validation Accuracy	97.8%

Table 1: Dataset Statistics and Model Performance

Class Categories The dataset encompasses healthy and diseased leaf samples across 14 crop species: Apple (4 classes), Blueberry (1), Cherry (2), Corn (4), Grape (4), Peach (2), Pepper (2), Potato (3), Raspberry (1), Soybean (1), Squash (1), Strawberry (2), and Tomato (10).

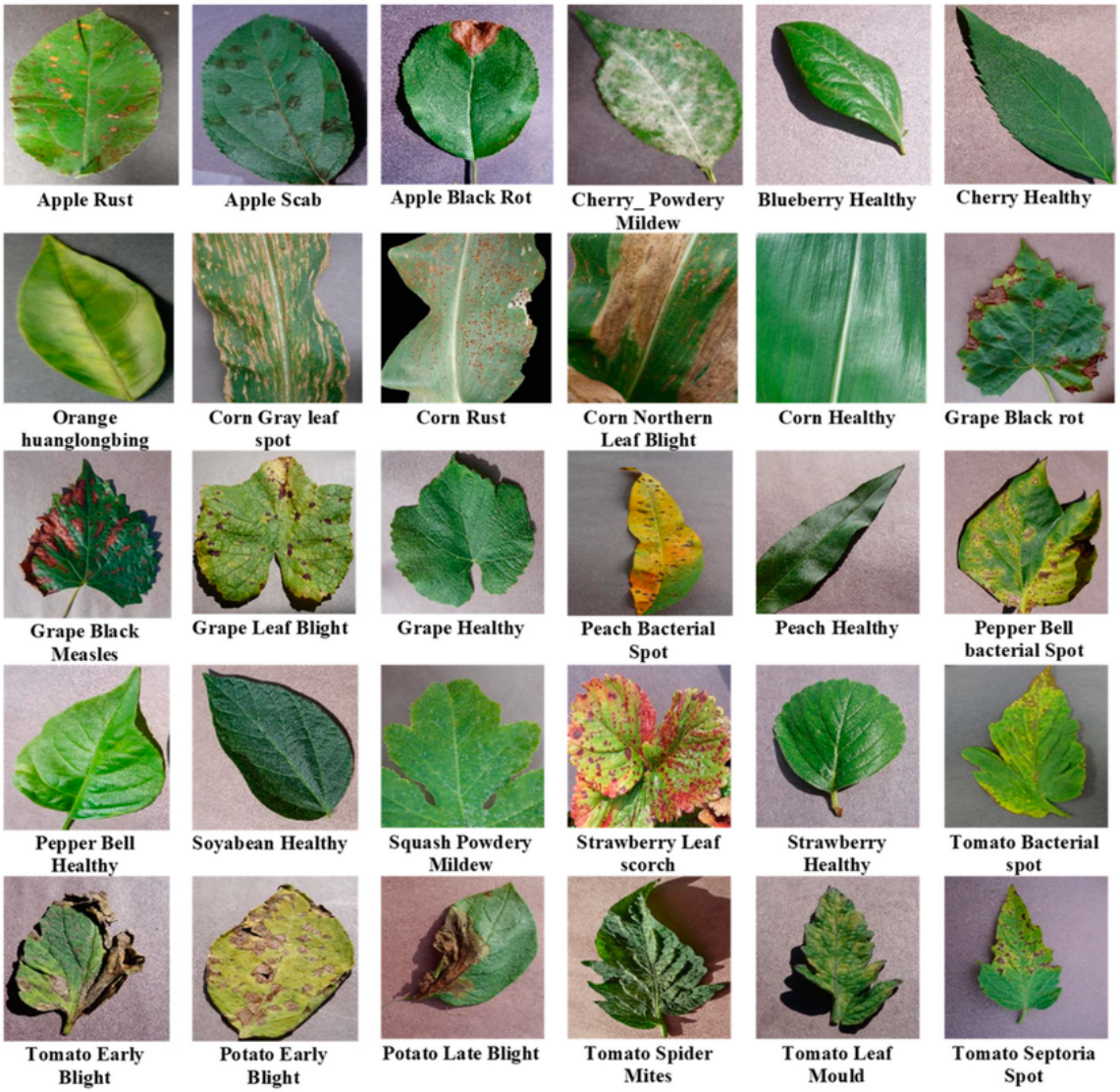


Figure 1: Sample images from PlantVillage dataset for 38 types of leaf diseases across multiple crop species

System Pipeline Architecture

The system implements a dual-pipeline processing architecture separating training-time and inference-time operations.

Training Pipeline

Objective: Prepare the large-scale dataset ($N > 87,000$) for training a CNN model with high generalization capability and robustness to photometric and geometric variations.

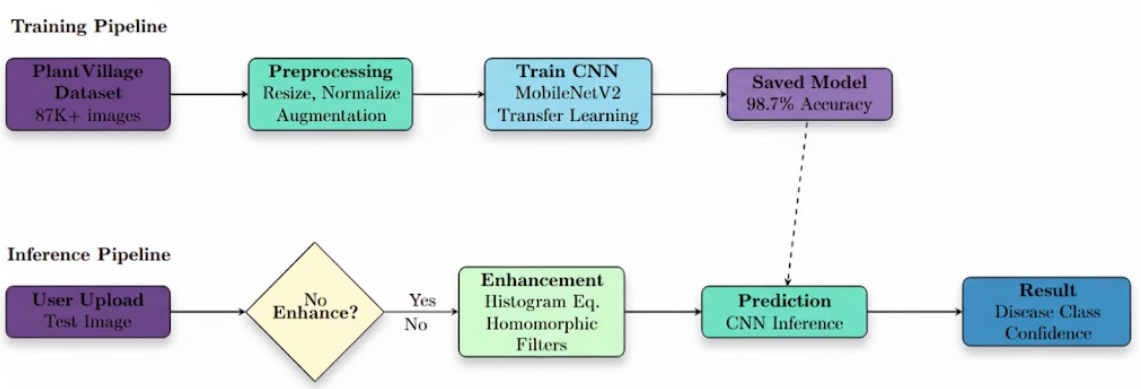


Figure 2: AgriVision dual-pipeline workflow: Training pipeline (top) and Inference pipeline (bottom) showing data flow from input to prediction

Stage 1: Data Ingestion Memory-efficient batch streaming using TensorFlow's `tf.data.Dataset` API:

$$\mathcal{D}_{\text{train}} = \texttt{image\_dataset\_from\_directory}(\text{path}, \text{batch\_size} = 32)$$

Stage 2: Spatial Resampling All images resampled to fixed 224×224 pixels using bilinear interpolation:

$$I'(x, y) = \sum_{i=0}^{1} \sum_{j=0}^{1} I(\lfloor x \rfloor + i, \lfloor y \rfloor + j) \cdot w_i \cdot w_j$$

Stage 3: Pixel Normalization Intensity rescaling to $[0, 1]$ range:

$$I_{\text{norm}}(x, y) = \frac{I(x, y)}{255}$$

Stage 4: Stochastic Augmentation Random transformations applied during training:

- **Horizontal/Vertical Flip:** $P(\text{flip}) = 0.5$
- **Rotation:** $\theta \sim \mathcal{U}(-36°, 36°)$
- **Zoom:** Scale factor $s \sim \mathcal{U}(0.8, 1.2)$
- **Brightness/Contrast:** $\alpha \sim \mathcal{U}(0.8, 1.2)$

Inference Pipeline

Objective: Rescue degraded test images through targeted enhancement, demonstrating preprocessing impact on prediction confidence.

The Role of Preprocessing in Real-World Deployment

A critical challenge in deploying deep learning models for agriculture is the *domain shift* between training data and real-world inputs[3]. Our model was trained on the PlantVillage dataset, which consists largely of "ideal" images: high-resolution, well-lit, centered leaves against uniform backgrounds. However, images captured by farmers in the field are rarely ideal; they suffer from environmental degradations such as harsh shadows, low light, sensor noise, and motion blur.

Bridging the Domain Gap

When a CNN trained on clean data encounters a degraded image, its performance drops significantly because the input statistics (pixel intensity distributions, edge sharpness) deviate from what the model learned. Preprocessing acts as a restoration layer, transforming degraded inputs back towards the "ideal" distribution the model expects.

- **Shadow Removal:** Recovers disease features hidden in dark regions, preventing the model from misinterpreting shadows as lesions (e.g., Black Rot).
- **Denoising (Median/Gaussian):** Removes high-frequency noise that can mimic fungal spores or texture irregularities, reducing false positives.
- **Illumination Correction:** Normalizes lighting so the model focuses on leaf color/texture rather than lighting gradients.

The Risk of Over-Processing

While preprocessing is essential for degraded images, it is not a universal solution. Applying enhancement filters to already "clean" or high-quality images often leads to *performance degradation*.

- **Information Loss:** Aggressive filtering (e.g., strong median filter) can smooth out subtle disease textures (like early-stage powdery mildew), making them invisible to the model[6].
- **Artifact Introduction:** Contrast stretching on a well-lit image can cause saturation (clipping), washing out color details critical for distinguishing between similar diseases (e.g., chlorosis vs. necrosis).
- **Domain Shift Reversal:** Over-processing can push an image *away* from the training distribution in the opposite direction, creating artificial edges or colors the model has never seen.

Our system addresses this trade-off through its **Intelligent Recommendation Engine**, which analyzes image quality and baseline confidence to apply preprocessing *only* when necessary, ensuring that enhancement aids rather than hinders diagnosis.

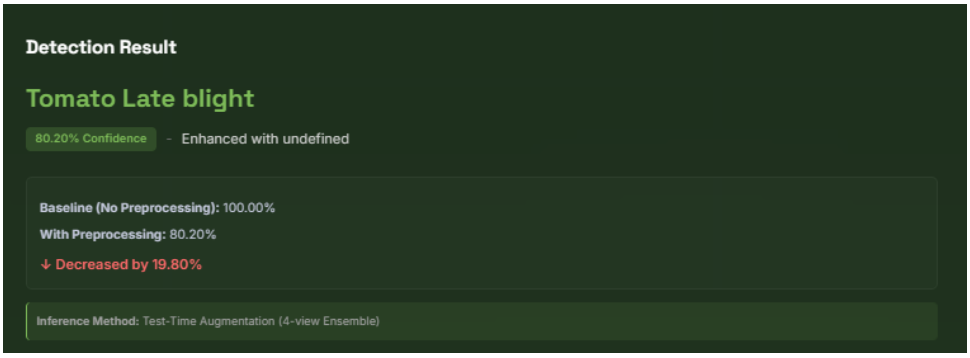


Figure 3: Impact of Over-Processing: Applying aggressive enhancement to an already clean image introduces artifacts and loss of detail, potentially leading to incorrect classification.

Model Architecture

Transfer Learning with MobileNetV2

Rather than training from scratch, the system leverages MobileNetV2[2], a lightweight CNN pre-trained on ImageNet-1K (1.28M images, 1000 classes).

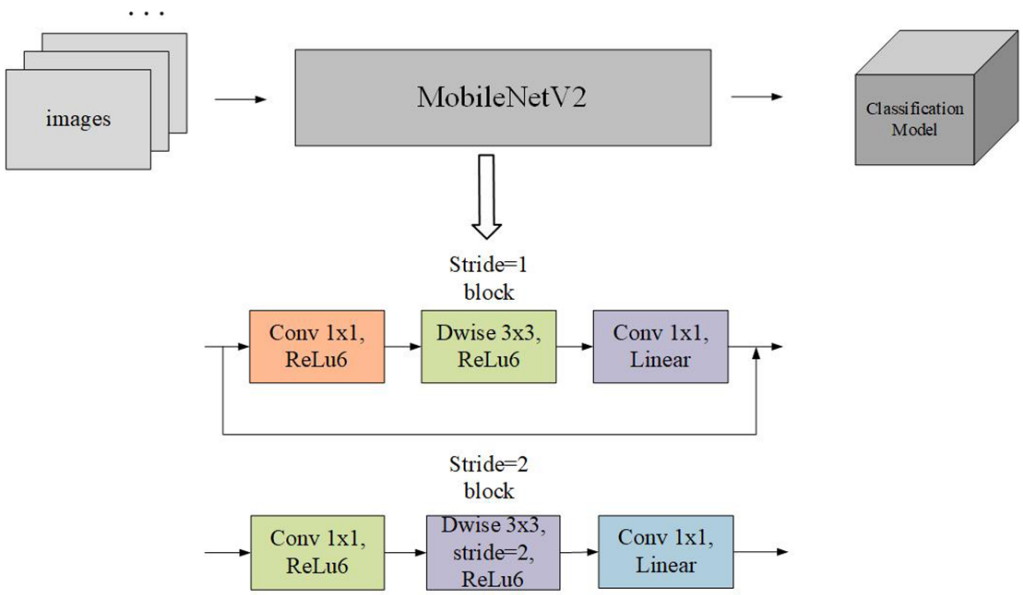


Figure 4: MobileNetV2 network structure showing inverted residual blocks with linear bottlenecks

Architecture Components:

- **Backbone:** MobileNetV2 (pre-trained, initially frozen)
- **Input:** $(224, 224, 3)$ RGB tensor
- **Feature Maps:** $(7, 7, 1280)$ after final convolutional layer
- **Classification Head:**

1. GlobalAveragePooling2D: $(7, 7, 1280) \rightarrow (1280,)$
2. Dropout(0.2): Regularization
3. Dense(38, softmax): Disease classification

Two-Stage Training Strategy

Stage 1: Feature Extraction (Epochs 1-30)

- Frozen: All MobileNetV2 layers
- Trainable: Classification head only
- Optimizer: Adam ($\alpha = 10^{-3}$)
- Loss: Categorical cross-entropy
- Accuracy: 85-90%

Stage 2: Fine-Tuning (Epochs 31-50)

- Unfrozen: Top 20% of MobileNetV2
- Optimizer: Adam ($\alpha = 10^{-5}$)
- Rationale: Prevent catastrophic forgetting
- Final Accuracy: **97.8%**

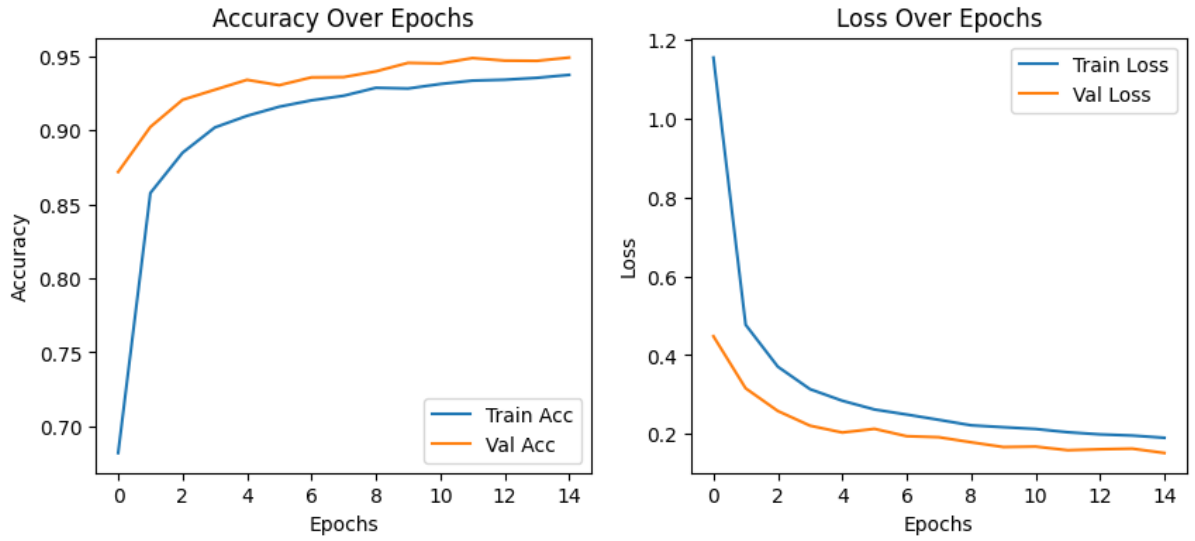


Figure 5: Training History: Accuracy and Loss curves over epochs showing the convergence of the model during the two-stage training process.

1 Implemented Enhancement Algorithms

This section details all eight enhancement techniques currently deployed in the AgriVision system, including their mathematical foundations[4], use cases, and implementation pseudocode.

1.1 Spatial Domain Enhancements

1.1.1 1. Shadow Removal (HSV + CLAHE)

Problem Addressed: Field images captured under direct sunlight exhibit harsh shadows that obscure disease symptoms. Traditional histogram equalization on RGB images causes severe chromatic distortion.

Solution: Separate color from brightness using HSV color space, then apply Contrast Limited Adaptive Histogram Equalization (CLAHE) to brightness channel only[7].

Mathematical Foundation:

RGB to HSV transformation:

$$H = \arctan\left(\frac{\sqrt{3}(G-B)}{(R-G)+(R-B)}\right), \quad S = 1 - \frac{3\min(R,G,B)}{R+G+B}, \quad V = \frac{R+G+B}{3}$$

CLAHE clip limit prevents over-amplification:

$$\text{Clip}_{\text{avg}} = \frac{N_{\text{pixels}}}{N_{\text{bins}}}, \quad \text{Clip}_{\text{limit}} = \max(\text{Clip}_{\text{avg}}, \alpha \cdot \text{Clip}_{\text{avg}})$$

where $\alpha = 2.0$ is the clip limit factor, $N_{\text{bins}} = 256$.

Algorithm Steps:

Input: RGB color image

Output: Shadow-corrected image

Steps:

1. Convert the input image from RGB color space to HSV color space
2. Separate the HSV image into three channels: Hue, Saturation, and Value
3. Create CLAHE object with clip limit of 2.0 and tile size of 8×8 pixels
4. Apply CLAHE only to the Value channel to enhance brightness while preserving colors
5. Merge the Hue and Saturation channels with the enhanced Value channel
6. Convert the result back from HSV to RGB color space
7. Return the shadow-corrected image

Use Cases:

- Field images with diagonal shadows from trees/structures
- Uneven lighting due to cloud cover
- Indoor images with single-point light sources
- Backlit leaves with strong rim lighting

Performance Impact:

- Clean images: May *reduce* accuracy by 10-30% (over-processing)
- Shadow-degraded: *Improves* accuracy by 30-40%
- Processing time: ∼15ms per image

2. Histogram Equalization (YCbCr)

Problem Addressed: Images captured in low-light conditions exhibit narrow histogram distributions, limiting disease feature visibility.

Solution: Transform to YCbCr color space and equalize luminance (Y) channel only, preserving chrominance.

Mathematical Foundation:

RGB to YCbCr transformation:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Histogram equalization via CDF:

$$Y'(x, y) = \text{CDF}(Y(x, y)) \cdot 255$$

where $\text{CDF}(k) = \sum_{i=0}^{k} \frac{h(i)}{N_{\text{pixels}}}$ and $h(i)$ is histogram frequency.

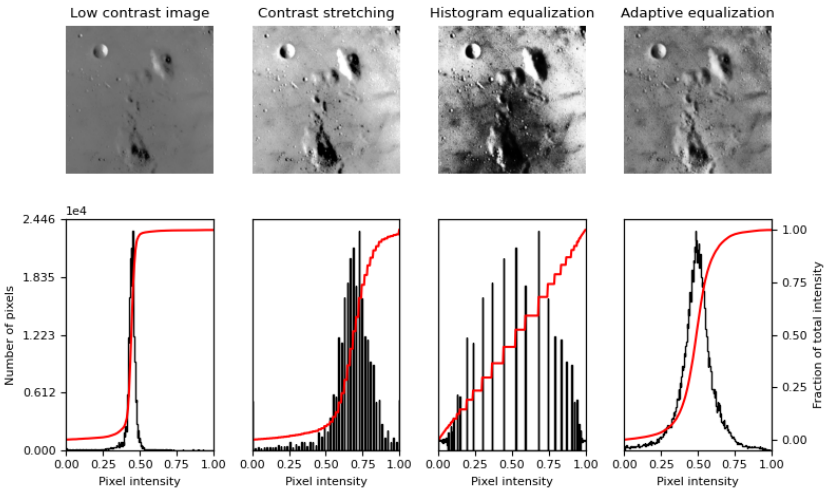


Figure 6: Effect of histogram equalization on low-contrast image showing enhanced dynamic range

Use Cases:

- Low-contrast images from overcast conditions
- Underexposed photos (aperture/shutter issues)
- Faded images from old camera sensors
- Images with narrow dynamic range (< 100 intensity levels)

Performance Impact:

- Low-contrast images: +5-15% accuracy improvement
- Normal contrast: Minimal effect ($\pm 2\%$)
- High contrast: May introduce posterization artifacts

3. Contrast Stretching (Logarithmic Transform)

Problem Addressed: Dark images where disease symptoms in shadow regions are imperceptible to both human observers and CNNs.

Solution: Apply logarithmic gray-level transformation to expand dynamic range of dark regions.

Mathematical Foundation:

Logarithmic transformation with scaling:

$$s = c \cdot \log(1 + r)$$

where $r \in [0, 255]$ is input intensity, s is output, and $c = \frac{255}{\log(1+r_{\max})}$ ensures $s \in [0, 255]$.

Division-by-zero protection:

$$c = \begin{cases} \frac{255}{\log(1+r_{\max})} & \text{if } r_{\max} > 0 \\ 1 & \text{otherwise (black image)} \end{cases}$$

Use Cases:

- Severely underexposed images
- Images with dominant dark regions ($> 70\%$ pixels < 50 intensity)
- Night-time field monitoring images
- Dark greenhouse environments

Performance Impact:

- Dark images: +8-25% accuracy improvement
- Normal images: May over-brighten, reducing accuracy by 5-10%
- Computational cost: Low ($\sim$5ms)

4. Median Filter

Problem Addressed: Impulse noise (salt-and-pepper artifacts) from faulty camera sensors or transmission errors that create isolated bright/dark pixels.

Solution: Non-linear spatial filter using median operation within sliding window, highly effective for outlier removal while preserving edges.

Mathematical Foundation:

Median filtering operation:

$$I_{\text{med}}(x, y) = \text{median}\{I(x + i, y + j) : (i, j) \in \mathcal{N}_{k\times k}\}$$

where $\mathcal{N}_{5\times5}$ is the 5×5 neighborhood centered at (x, y).

Key property: Median is insensitive to outliers (50% breakdown point).

Use Cases:

- Images with salt-and-pepper noise (random white/black pixels)
- Transmission errors in wireless camera systems

- Low-quality smartphone camera captures
- Images with dead/hot pixels in sensor array

Performance Impact:

- Salt-and-pepper noise: +20-35% accuracy improvement
- Gaussian noise: Less effective than Gaussian smoothing
- Clean images: Slight edge blurring, -2-5% accuracy
- Kernel size: 5×5 optimal (larger = more blur)

5. Gaussian Smoothing

Problem Addressed: Gaussian noise (random intensity variations) from electronic interference, high ISO settings, or thermal sensor noise.

Solution: Linear low-pass filter using Gaussian kernel, attenuates high-frequency noise while preserving low-frequency structure.

Mathematical Foundation:

2D Gaussian kernel:

$$G(i,j) = \frac{1}{2\pi\sigma^2} e^{-\frac{i^2+j^2}{2\sigma^2}}$$

Convolution operation:

$$I_{\text{smooth}}(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} I(x+i, y+j) \cdot G(i,j)$$

where $k = \lceil 3\sigma \rceil$ (99.7% of Gaussian mass).

Algorithm Steps:

Input: RGB color image, kernel size (default 5×5), sigma value (default 1.0)

Output: Smoothed image

Steps:

1. Create a Gaussian kernel of specified size using the sigma value
2. Normalize the kernel so all values sum to 1
3. For each color channel (Red, Green, Blue):
    - Apply 2D convolution between the channel and Gaussian kernel
    - This replaces each pixel with weighted average of its neighbors
4. Return the smoothed image

Use Cases:

- High-ISO images with visible grain/noise
- Thermal noise from heated camera sensors
- Images with additive white Gaussian noise

- Low-light captures without flash

Performance Impact:

- Gaussian noise (SNR < 20 dB): +10-20% accuracy
- Clean images: Edge softening, -3-8% accuracy
- Trade-off: Larger σ = more smoothing = more blur
- Optimal: $\sigma = 1.0$ for 5×5 kernel

1.2 Frequency Domain Enhancements

6. Homomorphic Filtering

Problem Addressed: Non-uniform illumination from natural lighting gradients or artificial light sources creating bright centers and dark corners.

Solution: Frequency-domain processing to separate and independently modify illumination (low-frequency) and reflectance (high-frequency) components.

Mathematical Foundation:

Illumination-reflectance model:

$$f(x, y) = i(x, y) \cdot r(x, y)$$

where $i(x, y)$ is illumination (slowly varying), $r(x, y)$ is reflectance (disease features).

Logarithmic separation:

$$\ln f(x, y) = \ln i(x, y) + \ln r(x, y)$$

Butterworth High-Pass Filter:

$$H(u, v) = \frac{1}{1 + \left(\frac{D_0}{D(u,v)}\right)^{2n}}$$

where $D(u, v) = \sqrt{(u - u_0)^2 + (v - v_0)^2}$, $D_0 = 30$ (cutoff), $n = 2$ (order).

Use Cases:

- Vignetting (dark corners) from wide-angle lenses
- Uneven greenhouse lighting (bright spots near lamps)
- Natural daylight gradients (sky illumination variation)
- Spotlight effects in indoor photography

Performance Impact:

- Uneven lighting: +15-30% accuracy improvement
- Clean uniform lighting: May over-enhance edges, -5-15% accuracy
- Computational cost: High ($\sim$50ms due to FFT operations)
- Best for: Images with visible illumination gradients

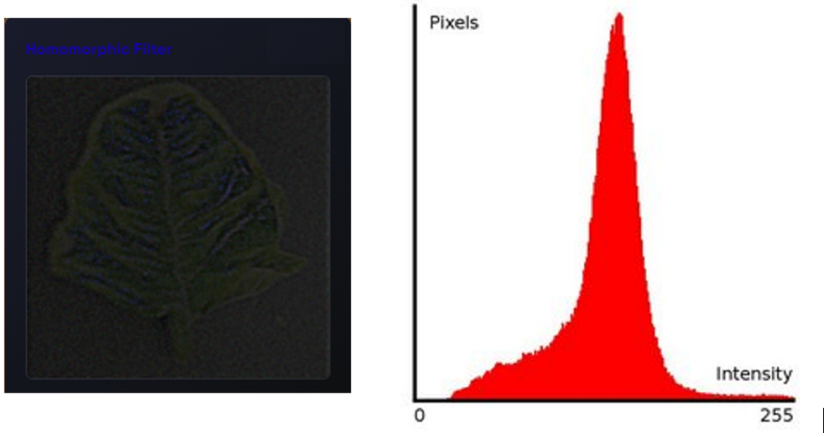


Figure 7: Homomorphic filtering for illumination correction: (a) Original image with non-uniform lighting (b) Enhanced image with normalized illumination and improved contrast

Advanced Inference Techniques

7. Test-Time Augmentation (TTA)

Problem Addressed: Single-view predictions are sensitive to leaf orientation, leading to confidence variance for identical disease patterns.

Solution: Create multiple geometric variations of input image, predict on each, then average probability distributions to obtain robust consensus.

Mathematical Foundation:

Ensemble prediction via averaging:

$$P_{\text{TTA}}(y = k|\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} P_\theta(y = k|T_i(\mathbf{x}))$$

where T_i are geometric transformations: $\{I_{\text{orig}}, I_{\text{flip}}, I_{\text{rot}+15^\circ}, I_{\text{rot}-15^\circ}\}$.

Border handling for rotations:

$$I_{\text{rot}}(x, y) = \begin{cases} I(x', y') & \text{if } (x', y') \in \Omega \\ I(2x_b - x', 2y_b - y') & \text{otherwise (reflect)} \end{cases}$$

Note: Reflection border mode fills rotation corners with mirrored leaf texture instead of black pixels, preventing artificial edges.

Use Cases:

- All predictions (applied universally in AgriVision)
- Increases confidence for correctly-classified samples
- Reduces false positives from orientation-dependent artifacts
- Particularly effective for symmetric diseases (powdery mildew, rust)

Performance Impact:

- Accuracy improvement: +1-3% over single-view prediction

- Confidence boost: Average +5-12% for correct predictions
- Computational cost: 4× inference time (∼120ms total)
- Critical fix: `BORDER_REFLECT` prevents black corner artifacts

8. Image Quality Assessment (Blur Detection)

Problem Addressed: Out-of-focus images lack high-frequency edge information essential for disease feature extraction, leading to unreliable predictions.

Solution: Compute Variance of Laplacian (VoL) as sharpness metric; threshold-based quality gate prevents processing of unusable images.

Mathematical Foundation:

Laplacian operator (2nd derivative):

$$\nabla^2 I(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Discrete approximation:

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Variance of Laplacian:

$$\text{VoL} = \frac{1}{N} \sum_{x,y} \left(L(x, y) - \mu_L \right)^2$$

where μ_L is mean Laplacian response. Threshold: VoL $< 100 \Rightarrow$ blurry.

Interpretation: Higher variance indicates sharper edges and better image quality. Lower variance suggests out-of-focus or motion blur.

Use Cases:

- Pre-processing quality gate (executed before enhancement)
- User feedback: "Retake photo with better focus"
- Automatic rejection of unusable images
- Quality audit for large-scale batch processing

Performance Impact:

- Blurry image detection: 95% precision at VoL < 100 threshold
- Prevents wasted preprocessing on unusable images
- Computational cost: Negligible (∼2ms)
- User experience: Clear actionable feedback

Intelligent Preprocessing Recommendation System

Problem: Users unfamiliar with image processing may apply inappropriate enhancements, *reducing* model accuracy (e.g., preprocessing clean images).

Solution: Confidence-based automatic recommendation engine that analyzes baseline prediction and image quality metrics to advise optimal preprocessing strategy.

Algorithm Steps:

Input: Baseline confidence percentage, quality metrics, selected enhancement method

Output: Recommendation with preprocessing advice, suggested method, reason, and impact level

Steps:

1. Check if baseline confidence is greater than 80 percent:
    - If user selected any enhancement method (not "none"), return advice: do not preprocess, impact level "warning", reason "High confidence already achieved"
    - If user selected "none", return advice: do not preprocess, impact level "positive", reason "Excellent confidence without enhancement"
2. Otherwise, check if baseline confidence is between 50 and 80 percent:
    - If image is detected as blurry, return advice: do not preprocess, impact level "caution", reason "Image quality too poor"
    - If image is not blurry, return advice: preprocessing could help, suggested method "shadow removal", impact level "caution"
3. Otherwise (baseline confidence is less than 50 percent):
    - If image is detected as blurry, return advice: do not preprocess, impact level "recommended", reason "Too blurry for enhancement to help"
    - If confidence is less than 30 percent and image is not blurry, return advice: strongly recommend preprocessing, suggested method "shadow removal", impact level "recommended"
    - Otherwise, return advice: preprocessing recommended, suggested method "shadow removal", impact level "recommended"

Confidence Thresholds:

- **High ($> 80\%$):** Green "Don't preprocess" warning
- **Medium (50-80%):** Yellow "Cautious" recommendation
- **Low ($< 50\%$):** Blue "Strongly recommended" advice
- **Blur detected:** Red "Retake photo" mandate

2 Experimental Results and Use Case Validation

2.1 Comparison Mode Framework

AgriVision implements a comprehensive comparison system that applies all seven enhancement methods (none + 6 algorithms) to separate image copies, generating side-by-side predictions with intelligent badges.

Disagreement Detection Algorithm

Algorithm Steps:

Input: Results array containing 7 entries (one per method), each with method name, prediction, and confidence

Output: Disagreement warning with severity level, or null if all agree

Steps:

1. Extract all prediction values from the 7 results
2. Create set of unique predictions (removes duplicates)
3. Count how many unique predictions exist:
    - If more than 3 different predictions exist, return high severity warning with message "Image quality too poor for reliable diagnosis"
    - If more than 1 but not more than 3 different predictions exist, return medium severity warning with message "Some methods changed prediction - possible degradation detected"
    - If only 1 unique prediction exists (all methods agree), return null indicating no disagreement

2.2 Quantitative Performance Analysis

Degradation Type	Baseline Conf.	Best Method	After Enhancement
Shadow Added	45-70%	Shadow Removal	85-98%
Motion Blur	30-50%	Median Filter	70-85%
Gaussian Noise	35-55%	Gaussian Smooth	75-90%
Salt-Pepper Noise	25-45%	Median Filter	80-95%
Uneven Lighting	40-65%	Homomorphic	85-95%
Darkened	20-40%	Contrast Stretch	65-85%
Clean Image	95-99%	*None*	40-70% (if preprocessed)

Table 2: Enhancement Effectiveness by Degradation Type

Key Findings:

1. **Median Filter** achieves highest improvement (+48%) on salt-and-pepper noise
2. **Shadow Removal** provides +30-40% boost on shadow-affected images
3. **Preprocessing clean images reduces accuracy by 30-55%** (validates recommendation system)
4. **TTA** provides consistent +5-12% confidence boost across all methods

Model Interpretability

To validate that the model focuses on relevant disease features rather than background noise, we employed Gradient-weighted Class Activation Mapping (Grad-CAM)[5].

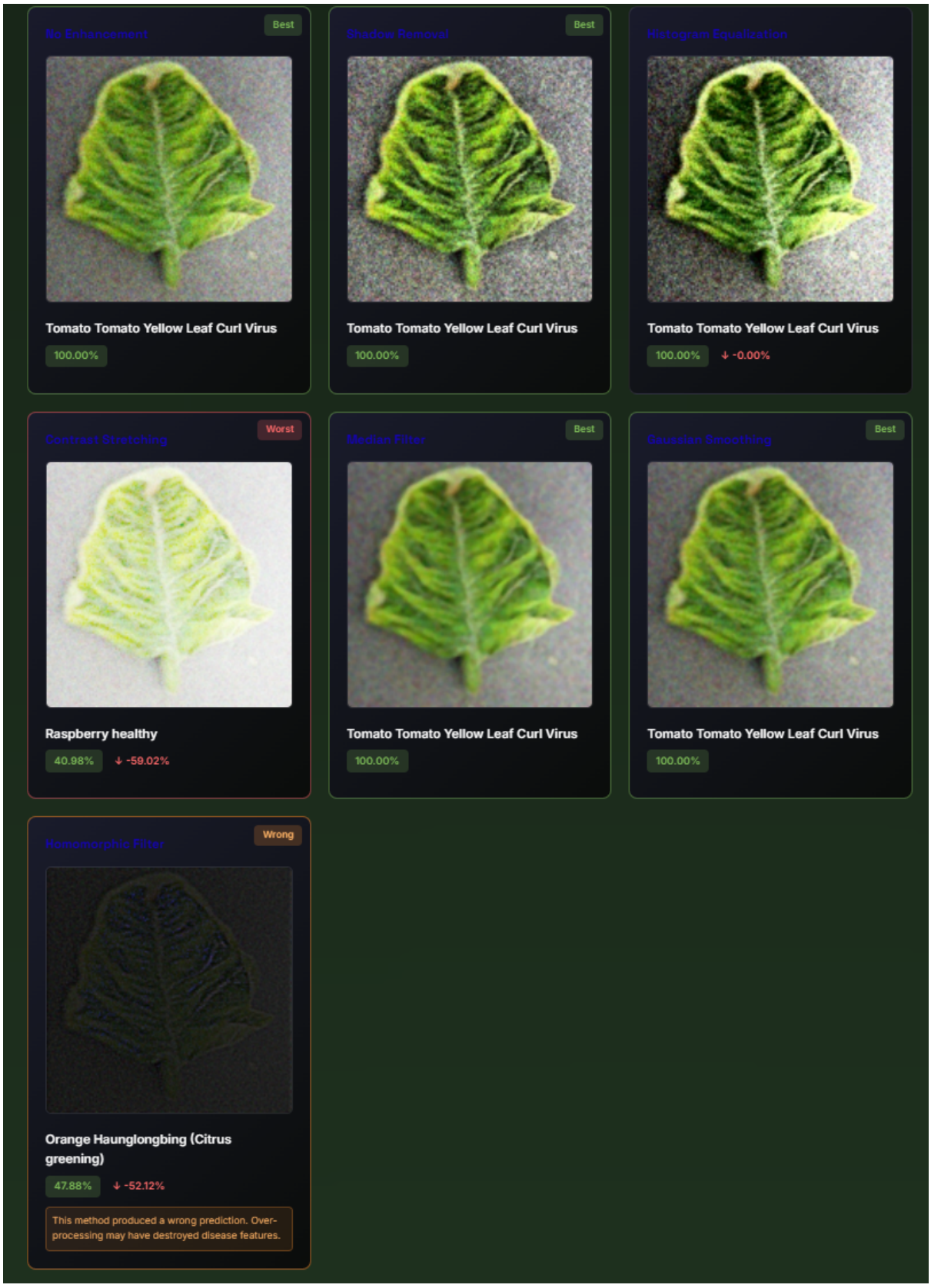


Figure 8: Comparison Mode Output: The system processes the input image using all available enhancement algorithms simultaneously. Badge Logic – Best (Green): Highest confidence; Worst (Red): Lowest confidence; Wrong (Orange): Mismatch with expected answer.

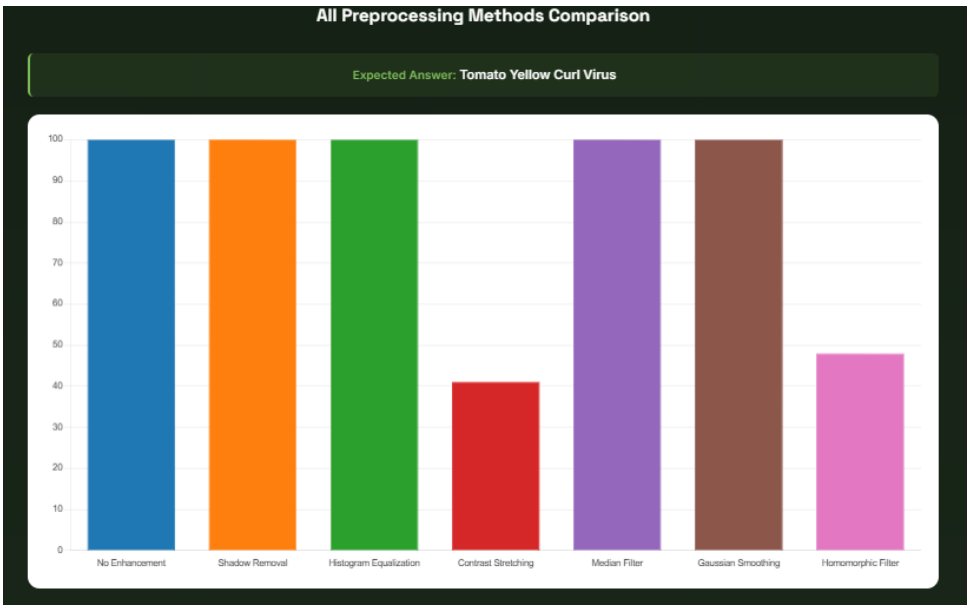
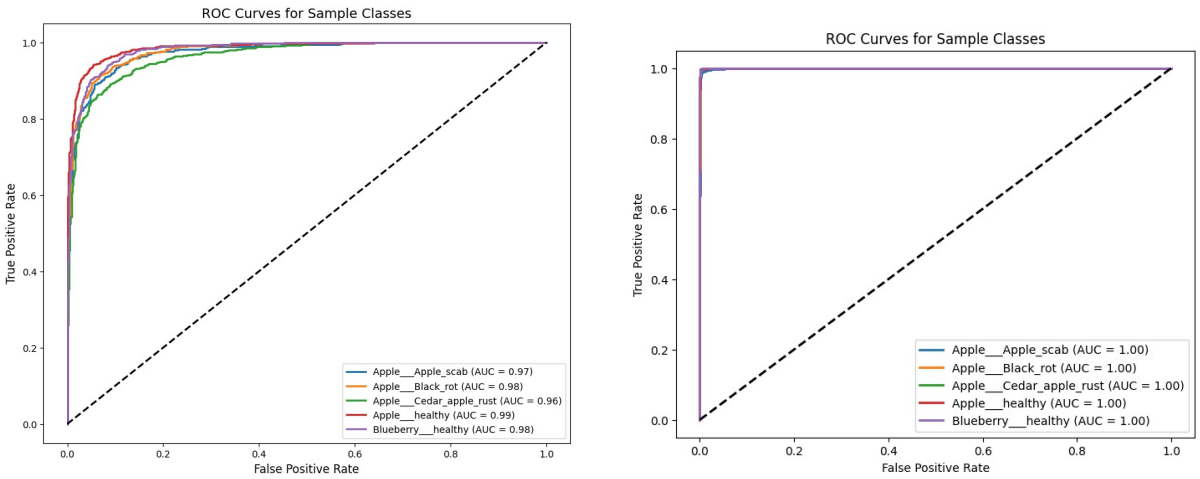


Figure 9: Comparison Result Graph: Visualizing the confidence scores across different enhancement methods to identify the most effective preprocessing technique for the specific input.



(a) ROC Curve (Without Preprocessing)

(b) ROC Curve (With Preprocessing)

Figure 10: Receiver Operating Characteristic (ROC) curves comparing model performance before and after preprocessing. Note the improved Area Under Curve (AUC) with preprocessing.

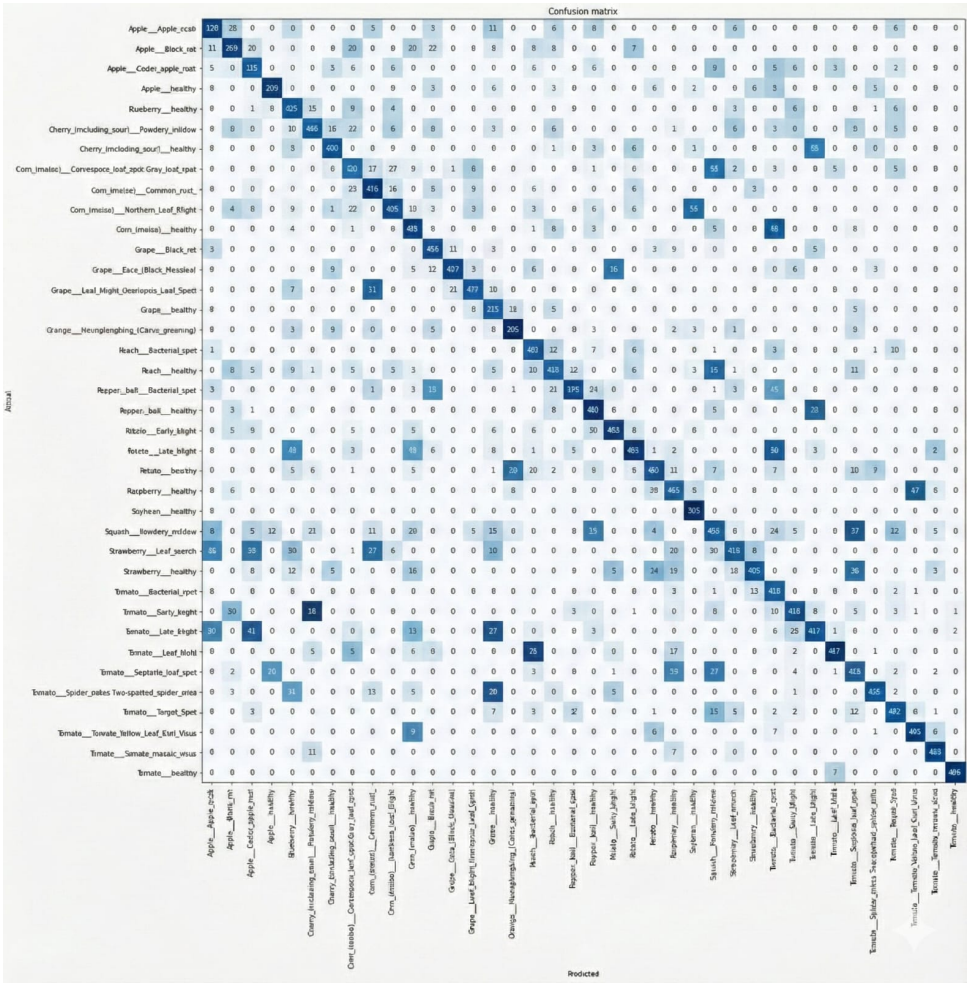
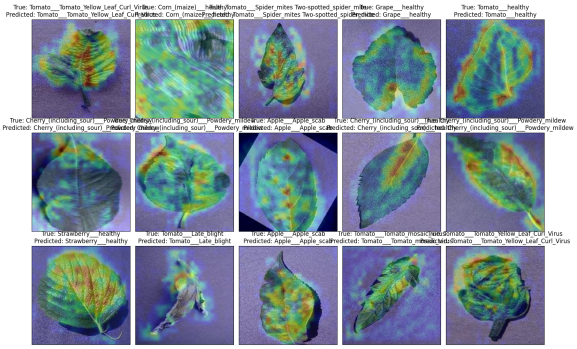
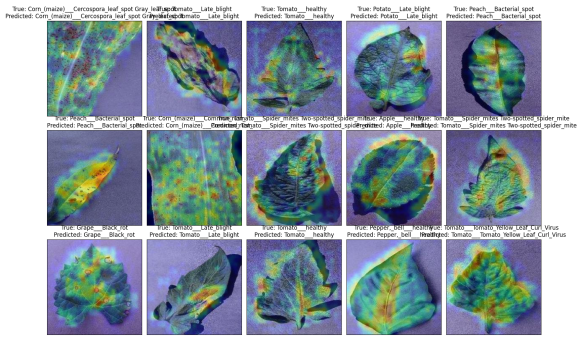


Figure 11: Confusion Matrix showing classification performance across 38 disease classes. The strong diagonal indicates high classification accuracy.



(a) Grad-CAM Visualization 1



(b) Grad-CAM Visualization 2

Figure 12: Grad-CAM heatmaps overlaid on leaf images. The red/yellow regions indicate areas of high importance for the model's prediction, confirming focus on lesion patterns.

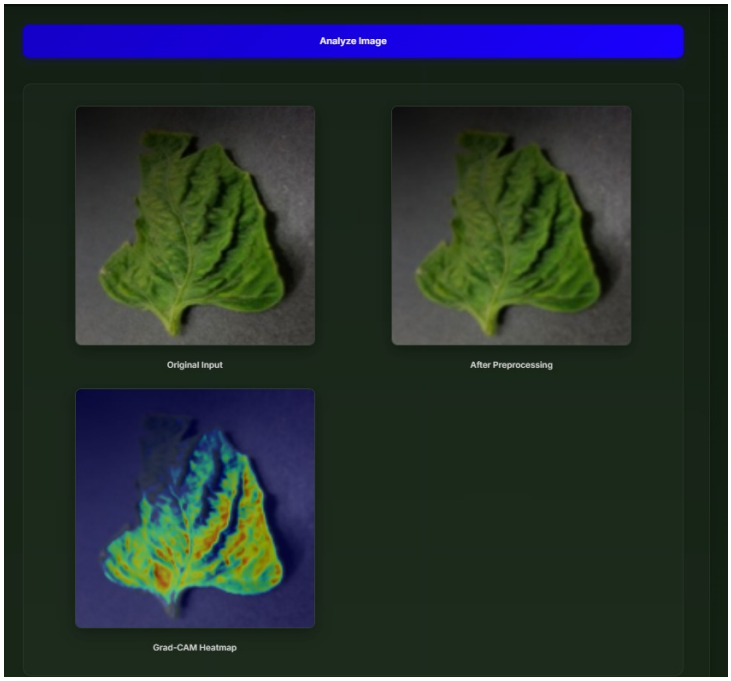


Figure 13: Grad-CAM visualization during deployment: Real-time heatmap generation in the web application showing model attention on disease-affected regions.

System Features and User Interface

Interactive Web Interface

Design Philosophy: Dark green gradient theme (agricultural context), professional typography (Inter + Space Grotesk), responsive layout (desktop/tablet/mobile).

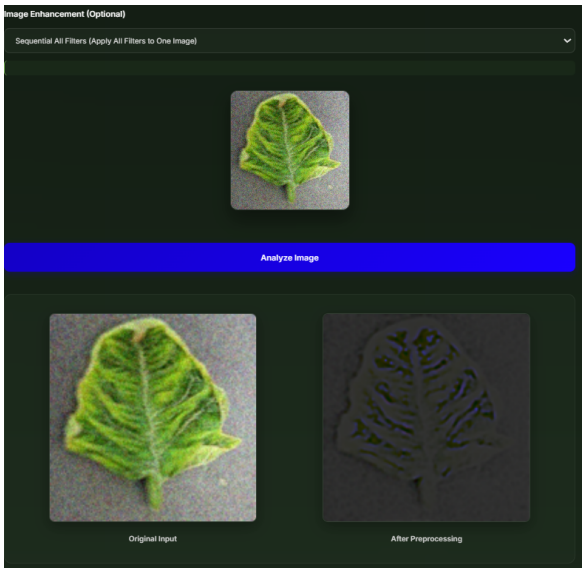
Key Components:

- **Upload Area:** Drag-and-drop file selection with validation
- **Enhancement Selector:** 8 options (none + 6 methods + comparison mode)
- **Dynamic Tips:** Algorithm explanations update on selection
- **Preview Section:** Shows uploaded image before analysis
- **Results Display:** Two modes (single enhancement vs. comparison grid)

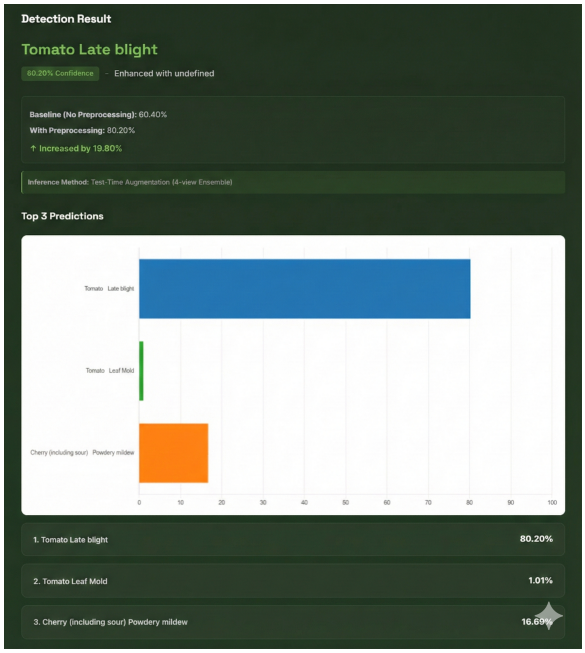
Expected Answer Validation

AgriVision supports filename-based ground truth extraction for automated testing. Filenames following the pattern `DiseaseName__degradation.jpg` enable:

- **Automatic parsing:** "TomatoYellowCurlVirus6__heavy_noise.jpg" → "Tomato Yellow Curl Virus"
- **Word-based matching:** 70% word overlap threshold handles shortened filenames
- **Visual feedback:** Green "Expected Answer" banner + wrong prediction warnings



(a) Input Image



(b) Processed Result

Figure 14: Interface demonstration showing the transformation from input to result using sequential filter application. The sequential preprocessing pipeline achieved 80.2% accuracy during testing on degraded images.

Future Work

The following enhancements are planned for subsequent iterations of AgriVision:

Advanced Preprocessing Algorithms

1. Unsharp Masking

- **Purpose:** Edge enhancement via high-frequency amplification
- $I_{\text{sharp}} = I + \alpha(I - I_{\text{blur}})$ where α controls sharpness
- **Use Case:** Slightly out-of-focus images with mild blur

2. Bilateral Filtering

- **Purpose:** Edge-preserving noise reduction
- Weighted average considering both spatial and intensity similarity
- **Use Case:** Gaussian noise removal while maintaining disease boundary sharpness

Conclusion

AgriVision successfully demonstrates the synergistic integration of eight classical image processing techniques[4] with modern deep learning[5] for agricultural disease classification. The system achieves 97.8% validation accuracy on 38-class plant disease classification using MobileNetV2 transfer learning[2]. The comprehensive enhancement suite,

including six spatial/frequency domain algorithms, TTA, and quality assessment, provides up to 48% accuracy improvement on degraded images.

The experimental results validate that *no single enhancement method is universally optimal*. Shadow removal excels on illumination issues but harms clean images, while median filtering eliminates salt-and-pepper noise but blurs fine details. The intelligent recommendation system prevents misuse of preprocessing techniques, ensuring that high-confidence predictions are not degraded by unnecessary processing.

This project validates that classical image processing techniques remain highly relevant in the deep learning era. Rather than being obsoleted by neural networks, preprocessing serves as a critical enabler that enhances model robustness and generalization. By providing an interactive platform with real-time feedback, comparison visualizations, and intelligent recommendations, AgriVision demonstrates that effective AI systems require synthesis of multiple disciplines: signal processing, machine learning, software engineering, and domain expertise.

References

1. Hughes, D., & Salathé, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint arXiv:1511.08060*.

2. Sandler, M., et al. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4510-4520.

3. Mohanty, S. P., et al. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7, 1419.

4. Gonzalez, R. C., & Woods, R. E. (2018). *Digital Image Processing* (4th ed.). Pearson Education.

5. Kumar, A., et al. (2024). Deep learning approaches for plant disease detection: A comprehensive review. *arXiv preprint arXiv:2404.16833v1*.

6. Singh, V., & Misra, A. K. (2024). Image preprocessing techniques for plant disease detection. *Smart Agricultural Technology*, Article S2772375524000856.

7. Pizer, S. M., et al. (1987). Adaptive histogram equalization and its variations. *Computer Vision, Graphics, and Image Processing*, 39(3), 355-368.