

Optimizing Classification-Based Machine Learning Models for Bacterial Spot Disease Detection in Tomato Plants

Siya Sharma

Reservoir High School

Fulton, MD, USA

siya.sharma94409@gmail.com

Abstract

Plant pests and diseases destroy up to 40% of global crop production each year, causing more than US\$220 billion in losses [4]. Tomatoes are among the most widely consumed crops, with approximately 160 million tons consumed each year [13]. In Florida, bacterial spot, caused mainly by *Xanthomonas perforans* [9, 11] and favored by hot, humid, rainy weather, cost tomato growers almost \$900,000 in the 2007–2008 season [2]. We hypothesized that infected tomato leaves differ measurably from healthy leaves in color and texture, and tested whether machine learning could detect these differences. Using 3,769 PlantVillage leaf images, we compared Logistic Regression, Ridge Classifier, Decision Tree, Random Forest, and Support Vector Classifier (SVC) models with a lightweight convolutional neural network (CNN). The tuned SVC performed best among the classical models (99.34% accuracy, 99.44% F1, ROC-AUC 1.00) and matched the CNN (McNemar’s test, $p = 1.00$). However, its accuracy fell sharply on rotated or brightened images, so field performance is likely lower. Machine learning shows promise for early detection of bacterial spot, but it must be validated on field images before practical use.

Keywords

Plant disease detection, bacterial spot, tomato, image classification, machine learning, support vector classifier, convolutional neural network, PlantVillage, robustness to image changes

ACM Reference Format:

Siya Sharma. 2026. Optimizing Classification-Based Machine Learning Models for Bacterial Spot Disease Detection in Tomato Plants. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 4)*. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.67149/yhjs2024.5/5k51m878>

1 Introduction

Plant disease reduces crop yield, raises production costs, and threatens food security [4, 8, 9]. Tomatoes are among the most widely produced and consumed crops in the world [13]; in the United States alone, fresh-market and processing tomatoes grossed over \$2.5 billion in 2008 [9]. Bacterial spot, caused by four species of *Xanthomonas*, produces small, water-soaked leaf lesions that darken and can merge, leading to yellowing, leaf loss, and reduced yield [7, 8]. In Florida, where *Xanthomonas perforans* is the dominant cause [9, 11],

losses from the disease were estimated at almost \$900,000 in the 2007–2008 growing season [2].

The disease spreads through contaminated seed and transplants, splashing rain and irrigation water, tools, and the handling of wet plants, especially in warm, humid conditions [7, 11], and the international seed trade can carry it between regions [9]. Pathogen-free seed and transplants are the primary defense, while copper sprays have become less effective because copper-tolerant strains now dominate in Florida [11]. Detecting infection early lets growers remove infected material, adjust irrigation, and isolate affected plants before the disease spreads. Image-based classification is therefore valuable as a screening or expert-assistance tool rather than a replacement for expert diagnosis [1].

Machine learning suits this task because leaf images can be converted into numerical features that capture color, texture, shape, and spatial patterns. Prior studies have used ResNet models to recognize 26 plant diseases from more than 54,000 leaf images [5], a convolutional neural network (CNN) to classify nine tomato leaf diseases with 98.49% accuracy [13], and ResNet transfer learning to classify strawberry leaf disease with 88% test accuracy [3]. CNNs are widely used because they learn spatial features directly from pixels [1], but they can require large training sets, long training times, and careful regularization, and a review of 19 CNN-based studies identified several issues that still limit their practical use [1]. Classical models can remain competitive: in a study of soybean disease, a support vector machine was more accurate than a deep learning classifier [12]. Because scikit-learn classifiers such as the Support Vector Classifier require tabular input [10], each image in this study was resized and converted into a fixed-length numerical feature vector.

This study used the PlantVillage dataset, a large labeled collection of leaf images that includes healthy and bacterial spot-infected tomato leaves [6]. Like datasets in prior work [3, 5], its images were captured under controlled lighting against plain backgrounds, which is useful for model development but limits conclusions about field performance, where images include uneven lighting, shadows, overlapping leaves, soil, blur, and varied camera angles. We therefore included cross-validation, confidence intervals, and robustness tests rather than relying on a single 80/20 train-test split.

We tested the hypothesis that tomato leaves infected with bacterial spot show measurable color and texture differences from healthy leaves. Our objectives were to compare several supervised classifiers on the same preprocessed images, identify the strongest classical model, and assess whether its performance is likely to generalize beyond a curated dataset. We also included a lightweight

This paper is published under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC-BY-NC-ND 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

IJSCAR VOL. 3, ISSUE 4

© 2026 The Author(s). Published by IJSCAR.

CNN baseline to test whether the simpler model remained competitive with an image-specific neural network, rather than to claim that SVCs outperform deep learning in general.

2 Materials and Methods

2.1 Dataset and Labels

The experiment used tomato leaf images from the PlantVillage dataset available through Kaggle. The two selected image categories were `Tomato_Bacterial_spot` and `Tomato_healthy`. The bacterial spot category contained images labeled as tomato leaves infected with bacterial spot disease, while the healthy category contained images labeled as healthy tomato leaves. Each image was assigned a binary label for supervised learning, where 0 represented a healthy leaf and 1 represented a leaf infected with bacterial spot disease. The final dataset contained 3,769 images: 2,177 bacterial spot images and 1,592 healthy images.

2.2 Image Preprocessing

Before model training, all images were resized to 128×128 pixels. Resizing ensured that every image had the same input dimensions and reduced computational cost during training and testing. This step also removed variation caused by different original image sizes, allowing each classifier to process a standardized input shape.

The RGB pixel values were then normalized from the original 0–255 scale to a 0–1 scale by dividing each pixel value by 255. Normalization ensured that all features were on a similar numerical scale, which is especially important for models such as Logistic Regression, Ridge Classifier, and Support Vector Classifier. These normalized RGB values were flattened into a one-dimensional vector, meaning each image became one row in the feature matrix and each pixel-channel value became one feature. The input matrix X therefore contained one row per image and 49,152 columns representing normalized RGB pixel values. The output vector y contained the corresponding binary labels, with 0 for healthy leaves and 1 for bacterial spot infected leaves.

No additional handcrafted feature extraction was performed; the flattened, normalized pixel vectors were used directly as input to all classical models. This conversion was necessary because the Support Vector Classifier and the other scikit-learn models classify numerical feature arrays rather than image files. Because the classifiers learned from raw pixel values and their positions rather than from handcrafted features such as lesion size or texture measures, they could draw on any visual information in the image, including information unrelated to disease. As a result, classifier performance depends on how closely new images match the framing, background, and lighting of the training images.

2.3 Train-Test Split and Model Training

The dataset was randomly divided into training and testing subsets using an 80/20 split (`random_state = 42`). The training set contained 3,015 images (1,725 bacterial spot and 1,290 healthy), and the test set contained 754 images (452 bacterial spot and 302 healthy). The training set was used to teach each model how to distinguish healthy and infected leaves, while the testing set was held out to evaluate how well each trained model classified new images that were not used during training. All classifiers were trained using

the same preprocessed feature matrix to ensure a fair comparison. The models evaluated were Logistic Regression, Ridge Classifier, Decision Tree Classifier, Random Forest Classifier, and Support Vector Classifier.

2.4 Classifier Model Comparison

Logistic Regression was used as a baseline classifier that estimates the probability that an image belongs to the infected class. A decision threshold of 0.5 was used to convert probability outputs into binary predictions. The Ridge Classifier was used as a regularized linear classifier. Its regularization term penalizes large coefficients and can help reduce overfitting when many correlated input features are present. The Decision Tree Classifier classified images by repeatedly splitting the feature space into smaller regions based on decision rules. This model can capture nonlinear patterns but may overfit if the tree becomes too complex. The Random Forest Classifier created decision trees and predicted the class by aggregating the outputs of those trees. Because each tree is trained using random subsets of data and/or features, the Random Forest model can reduce some of the overfitting risk associated with a single Decision Tree. The Support Vector Classifier was used to identify a decision boundary that best separates healthy and infected leaves in a high-dimensional feature space. The SVC focuses on the data points closest to the decision boundary, called support vectors, and attempts to maximize the margin between classes. This model was especially relevant because flattened image data produces a high-dimensional input matrix.

2.5 Model Settings

All models were implemented in scikit-learn using a fixed random seed (`random_state = 42`) where applicable, and all settings not listed here were left at scikit-learn defaults. Logistic Regression used L2 regularization ($C = 1.0$) with a maximum of 5,000 iterations. The Ridge Classifier used a regularization strength of $\alpha = 1.0$. The Decision Tree Classifier used the Gini criterion with no maximum depth. The Random Forest Classifier used 100 trees. The Support Vector Classifier used a radial basis function (RBF) kernel with gamma set to “scale,” and C was selected as described below.

2.6 Hyperparameter Tuning

The Support Vector Classifier was further evaluated using hyperparameter tuning. The C hyperparameter controls how strongly the model penalizes misclassified training points. A smaller C value allows a wider margin and tolerates more training errors, while a larger C value penalizes misclassification more strongly and may fit the training data more closely.

The C hyperparameter was selected by grid search over $C \in \{0.01, 0.1, 1, 10, 100, 1000\}$ using stratified 5-fold cross-validation on the training set only, with accuracy as the selection criterion. The selected model was then refit on the full training set, and the test set was used only once for final evaluation. Mean cross-validation accuracy ($\pm$ SD) for each value of C is shown in Figure 1.

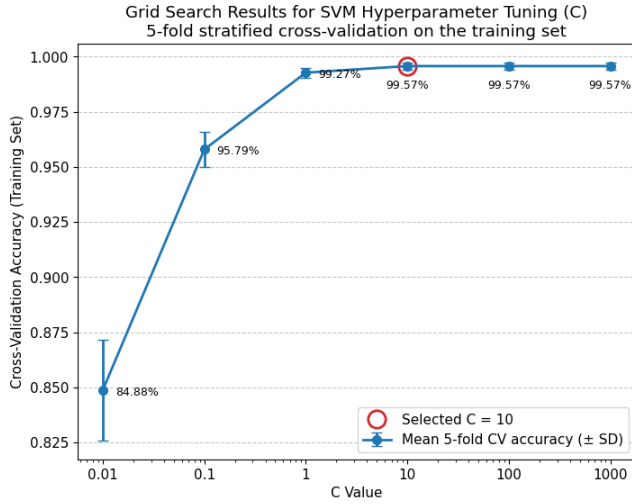


Figure 1: Selecting the SVC parameter C . Mean 5-fold cross-validation accuracy ($\pm$ SD) on the training set for each value of C . The circled point marks the selected value ($C = 10$). The test set was not used for tuning.

2.7 Lightweight CNN Baseline Comparison

Because we compare classical machine learning methods with neural-network approaches, a lightweight CNN baseline was included to provide a direct comparison under the same experimental conditions. The CNN baseline used the same two classes, *Tomato_Bacterial_spot* and *Tomato_healthy*, the same 128×128 RGB image inputs, the same 80/20 train-test split, and the same evaluation metrics as the classical classifiers.

The CNN baseline was implemented in TensorFlow/Keras and consisted of three convolutional layers with 16, 32, and 64 filters (3×3 kernels, ReLU activation), each followed by 2×2 max pooling. The output was flattened and passed to a 64-unit dense layer with ReLU activation, followed by dropout (rate 0.3) and a single sigmoid output unit for binary classification. The model was trained with the Adam optimizer and binary cross-entropy loss for 20 epochs with a batch size of 16, with the random seed set to 42. Of the 3,015 training images, the last 20% (603 images) were held out as a validation set, and the validation accuracy reported in the Discussion refers to these images. The CNN never saw the 754 test images during training. Unlike the SVC model, which used flattened pixel vectors, the CNN processed the $128 \times 128 \times 3$ image structure directly, allowing it to learn spatial features such as edges, lesions, color patterns, and texture differences.

The purpose of this CNN baseline was not to claim that SVC is better than neural networks. Instead, it allowed the study to determine whether the simpler SVC model remained competitive against a small image-specific neural network trained under comparable conditions.

2.8 Evaluation Metrics

Model performance was evaluated using accuracy, precision, recall, F1 score, ROC-AUC, and confusion-matrix counts (true negatives,

false positives, false negatives, and true positives), with bacterial spot as the positive class. Accuracy measures the overall proportion of correct predictions. Precision measures how many images predicted as infected were truly infected. Recall measures how many truly infected images were correctly detected. F1 score summarizes the balance between precision and recall. ROC-AUC was computed from the SVC’s continuous decision-function scores rather than from predicted class labels (Figure 2).

Recall is especially important for plant disease detection because false negatives may allow diseased plants to remain undetected and continue spreading infection. Precision is also important because false positives could lead to unnecessary treatment or inspection. For this reason, accuracy alone is not sufficient for evaluating the model.

2.9 Cross-Validation and Robustness

To address the limitation of relying on a single 80/20 split, stratified 5-fold cross-validation was performed on the SVC ($C = 10$) using the full dataset of 3,769 images. Because this analysis used all images, it served as a check of stability across splits rather than as an independent test. Stratification preserved the proportion of healthy and infected leaves in each fold. In each fold, the model was trained on four folds and evaluated on the remaining fold, and accuracy, precision, recall, and F1 score were recorded for each fold.

To evaluate whether the SVC’s performance depended on the controlled visual conditions of the PlantVillage dataset, the trained model was evaluated, without retraining, on modified versions of the held-out test images. Five perturbations were tested, including additive Gaussian noise (standard deviation 25, fixed random seed), a 30% decrease in brightness, a 30% increase in brightness, a 15-degree rotation, and Gaussian blur (5×5 kernel). Accuracy and F1 score were recalculated for each perturbed test set and compared against the model’s performance on the unperturbed test images.

2.10 Statistical Analysis and Overfitting

Two methods were used to calculate 95% confidence intervals. For the cross-validated metrics (accuracy, precision, recall, and F1 score), confidence intervals were computed using a t -distribution across the five fold-level scores. For the held-out test set, confidence intervals for accuracy, precision, recall, F1 score, and ROC-AUC were computed using percentile bootstrapping: the 754 test predictions were resampled with replacement 1,000 times (random seed 42), each metric was recalculated for every resample, and the 2.5th and 97.5th percentiles were reported. The SVC and CNN were compared on the same test images using McNemar’s exact test.

Overfitting was assessed by comparing each model’s accuracy on the training set with its accuracy on the test set, where a large gap would suggest overfitting. For the CNN, training accuracy was also compared with validation accuracy across epochs.

3 Results

Classical models used normalized 128×128 RGB images flattened into 49,152-feature vectors, while the lightweight CNN baseline used the full $128 \times 128 \times 3$ image structure.

On the held-out 80/20 test split, the tuned Support Vector Classifier ($C = 10$) achieved 99.34% accuracy and 99.44% F1 score. In 5-fold

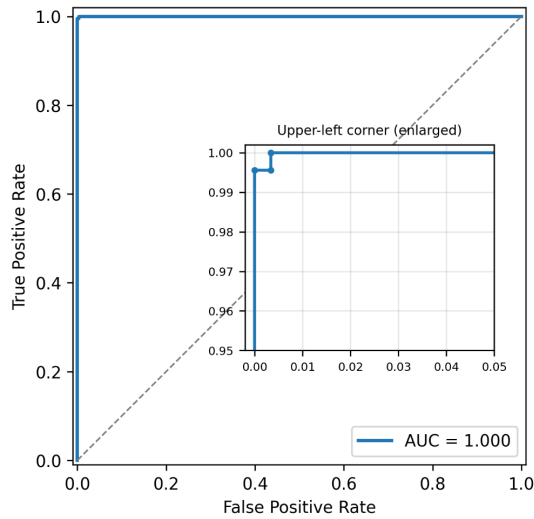


Figure 2: ROC curve of the tuned SVC ($C = 10$) on the held-out test set, computed from decision-function scores. The inset enlarges the upper-left corner. AUC = 1.00.

cross-validation on the training set, mean accuracy rose from 95.79% to 99.57% and did not improve at $C = 100$ or 1,000, so $C = 10$ was selected (Figure 1). The SVC correctly classified all 302 healthy test images and 447 of the 452 infected images, giving 302 true negatives, 0 false positives, 5 false negatives, and 447 true positives. Training accuracy for the tuned SVC was 100.0%. The ROC-AUC, computed from decision-function scores, was 0.9999 (95% Confidence Interval: 0.9996–1.0000; Figure 2). In stratified 5-fold cross-validation, per-fold accuracy ranged from 99.47% to 99.87% (mean = 99.63%, 95% Confidence Interval: 99.41%–99.84%), per-fold F1 score ranged from 99.54% to 99.88% (mean = 99.68%, 95% Confidence Interval: 99.49%–99.86%), precision ranged from 99.77% to 100.0% (mean = 99.95%, 95% Confidence Interval: 99.83%–100.0%), and recall ranged from 99.08% to 99.77% (mean = 99.40%, 95% Confidence Interval: 99.08%–99.73%). Bootstrap 95% Confidence Intervals (1,000 resamples) on the test set were 98.67%–99.87% for accuracy, 100.0%–100.0% for precision, 97.85%–99.77% for recall, and 98.91%–99.89% for F1.

The lightweight CNN baseline, trained under identical conditions, achieved 99.34% accuracy, 99.45% F1 score, 99.56% precision, and 99.34% recall, with 300 true negatives, 2 false positives, 3 false negatives, and 449 true positives. Both models misclassified 5 of the 754 test images, and none of their errors overlapped. McNemar’s exact test found no difference between the two models ($p = 1.00$).

Table 1 compares all models on the same test set. Among the classical models, the tuned SVC had the highest accuracy (99.34%), followed by Random Forest (96.55%), Logistic Regression (96.29%), Ridge Classifier (92.84%), and Decision Tree (92.04%). The CNN baseline reached the same accuracy as the SVC. Logistic Regression and Ridge Classifier detected most infected leaves (recall 98.45% and 98.01%) but produced more false positives (21 and 45), whereas Random Forest produced fewer false positives but missed 20 infected leaves. All classical models reached 100% training accuracy.

SVC performance changed substantially under some image modifications, as seen in Table 2. Accuracy remained high under Gaussian blur but fell under Gaussian noise and brightness changes and dropped to 40.05% under a 15-degree rotation, where the model classified every test image as healthy ($F1 = 0$).

4 Discussion

The tuned Support Vector Classifier was the strongest classical model in this study, outperforming Random Forest, Logistic Regression, Ridge Classifier, and Decision Tree by 2.8 to 7.3 percentage points in test accuracy (Table 1), and it matched the accuracy of the lightweight CNN baseline. The tuned Support Vector Classifier achieved strong performance on the PlantVillage held-out test set, with accuracy and F1 score both exceeding 99%, and this result was confirmed as stable across five stratified cross-validation folds. The narrow confidence intervals around each cross-validated metric, together with an independently computed bootstrap confidence interval for test-set accuracy, indicate that the reported performance is statistically consistent rather than the product of a single favorable split.

The lightweight CNN and the tuned SVC reached the same test accuracy (99.34%; McNemar’s $p = 1.00$) but made different errors: the SVC produced no false positives and missed 5 infected leaves, while the CNN produced 2 false positives and missed 3. This result indicates that, for this controlled binary classification task, flattened RGB pixel features gave the SVC no substantial advantage or disadvantage relative to a small image-specific convolutional neural network. This finding should not be interpreted as evidence that SVCs are generally superior or inferior to deep learning approaches for plant disease detection. It reflects only that both approaches were able to reach near-ceiling performance under the same controlled dataset conditions used in this study. This is consistent with earlier work in which an SVM classified plant diseases more accurately than a deep learning classifier [12], although that study used measured plant attributes rather than images.

The SVC’s training accuracy (100.0%) was only 0.7 percentage points higher than its test accuracy (99.34%; 95% CI: 98.67%–99.87%), and accuracy on held-out cross-validation folds was similarly high (mean 99.63%). Together, these results indicate little overfitting to the training images. The large drops under rotation and brightness changes reflect a different problem: sensitivity to imaging conditions not represented in PlantVillage, rather than memorization of the training set. The CNN showed a similar pattern, with 99.73% training accuracy and 99.34% test accuracy.

4.1 Limitations

The near-perfect performance metrics should be interpreted with caution rather than as evidence of a generally robust disease-detection model. The PlantVillage dataset used in this study consists of images captured under highly controlled conditions, including consistent lighting, uniform blank backgrounds, and leaves centered and isolated from surrounding foliage, which likely makes the classification task easier than real-world detection would be. Because the SVC was trained directly on flattened pixel values rather than on disease-specific handcrafted features, it is plausible that the model partly learned to exploit incidental patterns in

Table 1: Test-set performance of all models. The test set has 754 images, with bacterial spot as the positive class. TN, FP, FN and TP are true negatives, false positives, false negatives and true positives.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	ROC-AUC	Train acc. (%)	TN	FP	FN	TP
SVC (RBF, $C = 10$)	99.34	100.00	98.89	99.44	0.9999	100.00	302	0	5	447
CNN baseline	99.34	99.56	99.34	99.45	0.9999	99.73	300	2	3	449
Random Forest	96.55	98.63	95.58	97.08	0.9934	100.00	296	6	20	432
Logistic Regression	96.29	95.49	98.45	96.95	0.9950	100.00	281	21	7	445
Ridge Classifier	92.84	90.78	98.01	94.26	0.9767	100.00	257	45	9	443
Decision Tree	92.04	94.75	91.81	93.26	0.9210	100.00	279	23	37	415

Table 2: SVC robustness to image modifications. Test accuracy and F1 of the tuned SVC after each modification, without retraining.

Condition	Accuracy (%)	F1 (%)
Original (no modification)	99.34	99.44
Gaussian blur (5×5 kernel)	98.54	98.80
Gaussian noise (SD = 25)	89.12	90.02
Brightness -30%	87.67	88.97
Brightness +30%	71.88	69.36
Rotation 15°	40.05	0.00

background uniformity or image framing rather than biologically meaningful indicators of bacterial spot disease. This concern is directly corroborated by the robustness testing conducted in this study: while the model retained high accuracy under Gaussian blur, performance degraded substantially under brightness shifts and Gaussian noise and collapsed to near-random performance (accuracy 40.05%, $F1 = 0.0000$) under a modest 15-degree rotation. This sharp sensitivity to simple geometric and lighting perturbations indicates that the model’s apparent competence is tied closely to the narrow visual conditions of the training data, and that the reported metrics likely overstate how the model would perform on images collected outside a controlled setting.

4.2 Future Work

Field-image evaluation remains a priority for future work. Because PlantVillage images are captured under controlled lighting and uniform backgrounds, this study could not assess how the SVC performs on real agricultural images with natural lighting, shadows, occlusion, and varied camera angles. It is also very likely that performance on real field images would be significantly lower than performance on the PlantVillage test set. Future work should collect a labeled dataset of field or phone camera images of tomato leaves and evaluate the trained model on this external set without retraining, reporting the results separately from the PlantVillage test set, to determine whether the classifier generalizes beyond curated conditions or would require retraining on more heterogeneous data before practical deployment.

In addition to field-image evaluation, future work should investigate targeted data augmentation to determine whether the sharp performance degradation observed during robustness testing can be mitigated without requiring an entirely new field dataset. Comparing the augmented model’s robustness profile to the one reported

here would help isolate whether the original model’s sensitivity stems from the training data’s limited visual diversity or from a more fundamental limitation of the flattened-pixel feature representation. Future work could also train the classifier on grayscale versions of the images to determine how much its performance depends on color information rather than lesion texture and shape.

Acknowledgments

The author thanks Emily Broadhurst for her guidance throughout this research and her parents for their constant support.

References

- [1] J. Boulent, S. Foucher, J. Théau, and P.-L. St-Charles. 2019. Convolutional Neural Networks for the Automatic Identification of Plant Diseases. *Frontiers in Plant Science* 10 (2019), 941. <https://doi.org/10.3389/fpls.2019.00941>
- [2] B. Buck. 2020. Scientists to Study Crippling Tomato Disease to Lay Groundwork for Prevention. UF/IFAS News, University of Florida. <https://blogs.ifas.ufl.edu/news/2020/07/22/scientists-to-study-crippling-tomato-disease-to-lay-groundwork-for-prevention/> 22 July 2020.
- [3] P. K. Das and S. S. Rupa. 2023. ResNet for Leaf-based Disease Classification in Strawberry Plant. *International Journal of Applied Methods in Electronics and Computers* 11, 3 (2023), 151–157. <https://doi.org/10.58190/ijamec.2023.42>
- [4] Food and Agriculture Organization of the United Nations. 2025. Cooperation at All Levels and Funding “Critical” for Plant Health and Food Security, FAO Says. FAO Newsroom. <https://www.fao.org/newsroom/detail/cooperation-at-all-levels-and-funding--critical--for-plant-health-and-food-security--fao-says/en> 12 May 2025.
- [5] D. Gosai, B. Kaka, D. Garg, R. Patel, and A. Ganatra. 2022. Plant Disease Detection and Classification Using Machine Learning Algorithm. In *2022 International Conference for Advancement in Technology (ICONAT)*. 1–6. <https://doi.org/10.1109/ICONAT53423.2022.9726036>
- [6] D. P. Hughes and M. Salathé. 2015. An Open Access Repository of Images on Plant Health to Enable the Development of Mobile Disease Diagnostics. <https://doi.org/10.48550/arXiv.1511.08060> arXiv:1511.08060
- [7] M. Marks. 2024. Bacterial Spot of Tomato. Wisconsin Horticulture, Division of Extension, University of Wisconsin–Madison. <https://hort.extension.wisc.edu/articles/bacterial-spot-of-tomato/> 28 February 2024.
- [8] I. Meadows, A. Scherer, and M. Henson. 2019. Bacterial Spot of Pepper and Tomato. NC State Extension Publications. <https://content.ces.ncsu.edu/bacterial-spot-of-pepper-and-tomato> 14 February 2019.
- [9] National Institute of Food and Agriculture. 2016. Bacterial Spot of Tomato. U.S. Department of Agriculture. https://www.nifa.usda.gov/sites/default/files/resource/Tomato_Spot_Fact_Sheet.pdf
- [10] Scikit-learn Developers. [n. d.]. 1.4. Support Vector Machines. Scikit-learn 1.9.1 Documentation. <https://scikit-learn.org/stable/modules/svm.html>
- [11] A. Strayer-Scherer, Y.-Y. Liao, P. Abrahamian, S. Timilsina, M. Paret, T. Momol, J. Jones, and G. Vallad. 2019. Integrated Management of Bacterial Spot on Tomato in Florida. Ask IFAS, University of Florida. <https://edis.ifas.ufl.edu/publication/PP353> 5 November 2019.
- [12] K. Thailaynayaki and C. Joseph. 2021. Classification of Plant Disease Using SVM and Deep Learning. *Materials Today: Proceedings* 47 (2021), 468–470. <https://doi.org/10.1016/j.matpr.2021.05.029>
- [13] N. K. Trivedi, V. Gautam, A. Anand, H. M. Aljahdali, S. G. Villar, D. Anand, N. Goyal, and S. Kadry. 2021. Early Detection and Classification of Tomato Leaf Disease Using High-Performance Deep Neural Network. *Sensors* 21, 23 (2021), 7987. <https://doi.org/10.3390/s21237987>