Evaluation of Standard Machine Learning for Plant Leaf Disease Recognition on a Small Field Image Dataset

Narsaiah Battu[1]
Research Scholar
Department of Computer Science & Engineering
Telangana University
Dichpally, Nizamabad, Telangana, India

Arathi Chitla[2]
Professor, Department of Computer Science & Engineering
Telangana University
Dichpally, Nizamabad, Telangana, India

***Abstract*—**Most image-based plant disease studies train deep networks on large laboratory datasets, yet many practical projects have only a few hundred field photographs and no GPU. This paper evaluates eight standard machine learning classifiers on a small field image dataset of 1,532 leaf photographs labelled Healthy, Powdery (powdery mildew) and Rust, using 129 hand-crafted colour, texture and shape features. Classifiers were compared by nested, repeated stratified cross-validation on the 1,382 training and validation images, and the best model was tested once on the untouched official 150-image test set. Gradient boosting achieved the highest cross-validated accuracy (92.04% ± 1.41%) and significantly outperformed random forest (89.87%; corrected resampled t-test, $p = 0.021$). On the official test set it reached 89.33% accuracy (95% CI 83.4–93.3%), a macro F1-score of 89.37% and Cohen's $\kappa = 0.84$, within four points of a published DenseNet121 result on the same data. Colour and texture features were the most informative; adding shape features reduced k-nearest-neighbour accuracy from 87.37% to 66.79%. Random forest was the most data-efficient model, reaching 68.4% test accuracy with only five training images per class. The commonest error was a diseased leaf predicted as healthy (10 of 100 diseased test leaves). A bias control using only eight border pixels per image reached 58.1% in cross-validation but 38.7% on the test set (chance: 33.3%), revealing that capture conditions shared within the training split inflate cross-validated estimates. Classical machine learning is therefore a strong, low-cost baseline for field disease recognition, provided performance is reported on an independent test set.

***Index Terms*—**plant disease recognition, powdery mildew, rust, field images, machine learning, hand-crafted features, gradient boosting, random forest, dataset bias

I. Introduction

Plant diseases cause major crop losses, and image-based diagnosis is one of the most active applications of machine learning in agriculture [1], [2]. Since Mohanty *et al.* [3] reached 99.35% accuracy with a convolutional neural network (CNN) on the PlantVillage collection [4], most studies have trained deep networks on large datasets. PlantVillage images, however, show single detached leaves on uniform laboratory backgrounds, and models trained on them generalise poorly to field photographs [3], [5]. Field images, with cluttered backgrounds, variable lighting and several leaves per frame, are harder and far scarcer [5], [6].

Two practical problems follow. First, many agricultural groups have only a few hundred labelled field photographs and no GPU, which limits deep learning. Second, small image datasets often contain hidden shortcuts: photographs of one class may share a background, camera or session, so a model can learn the capture conditions instead of the disease [7]. Standard machine learning on hand-crafted features is fast, interpretable and needs little data, but its performance on small field disease datasets, and the effect of capture bias on its evaluation, are rarely measured rigorously.

This paper evaluates eight standard classifiers on a public field dataset of 1,532 leaf photographs with three classes: Healthy, Powdery (powdery mildew) and Rust [8]. The contributions are:

1) a comparison of eight standard classifiers on 129 hand-crafted features using nested cross-validation and statistical significance tests;
2) a single, untouched evaluation on the dataset's official test set, compared with published deep-learning results on the same data [9], [10];
3) a feature-group ablation and a learning-curve analysis from 5 to 160 training images per class;
4) an error analysis framed as disease detection (sensitivity and specificity); and
5) a background-pixel bias control evaluated both in cross-validation and on the test set, which shows how shared capture conditions inflate cross-validated accuracy.

Section II reviews related work, Section III describes the data and methods, Section IV reports the results, Section V discusses them and Section VI concludes.

II. Related Work

Classical machine learning. Before deep learning, plant disease recognition combined colour and texture descriptors with standard classifiers. Arivazhagan *et al.* [11] used colour co-occurrence texture features and an SVM and reported 94% accuracy on about 500 leaf images. Such pipelines remain

attractive for small datasets because they train in seconds and their features (lesion colour, surface texture) can be related to visible symptoms.

Deep learning and laboratory data. On PlantVillage, CNNs reach 99.35% [3] or higher, up to 99.75% for DenseNet-121 [12], [13]. Mohanty *et al.* [3], however, observed accuracy falling to about 31% on images from other sources, and Barbedo [5], [14] showed that dataset size, diversity and capture conditions strongly limit what CNNs learn for plant pathology. Field datasets such as Plant Pathology 2020 [6] were created to address this gap.

Dataset bias. Noyan [7] trained a model on only eight background pixels per PlantVillage image and obtained 49.0% accuracy across 38 classes (chance 2.6%), demonstrating strong capture bias. Such shortcuts inflate reported accuracy whenever training and test images share capture conditions.

Prior work on this dataset. Purba [9] trained a sequential DenseNet121 model on the same Healthy/Powdery/Rust dataset and reported 93.33% test accuracy. Al-Zubi [10] described the images as apple-orchard leaves and reported 91.34% overall accuracy for a six-layer sequential CNN. To our knowledge, no study has evaluated standard machine learning algorithms or measured capture bias on this dataset.

III. Materials and Methods

A. Dataset

The Plant disease recognition dataset [8] is distributed on Kaggle under a CC0 public-domain licence with an official split into Train, Validation and Test folders. The copy used here contained 1,532 RGB photographs (the dataset description states 1,530) of leaves in natural field conditions with cluttered backgrounds (Fig. 1). Classes are nearly balanced (Table I).

TABLE I
Dataset Split (Images per Class)

Class	Train	Validation	Test	Total
Healthy	458	20	50	528
Powdery	430	20	50	500
Rust	434	20	50	504
Total	1,322	60	150	1,532

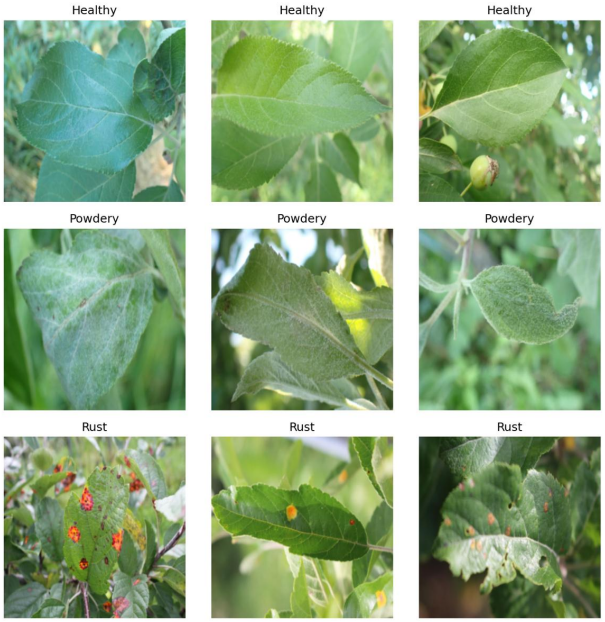


Fig. 1. Example field images from the dataset (three per class), shown after resizing.

Model selection used the Train and Validation images together (1,382 images, the *development set*), because the official Validation set of 60 images is too small to compare eight classifiers reliably. The official Test set (150 images) was used only once, in the final evaluation.

B. Pre-processing and Feature Extraction

Each image was decoded at reduced resolution and resized to 128 × 128 pixels. Four groups of hand-crafted features were computed (Table II): colour (HSV histograms and colour moments), texture (local binary patterns [15] and grey-level co-occurrence statistics [16]), and shape (Hu moments [17] of a saturation-based leaf mask and histograms of oriented gradients, HOG [18]). Standardisation and the PCA that reduces HOG from 1,764 to 50 values were fitted inside each training fold only, giving 129 features per image without information leakage.

TABLE II
Hand-Crafted Feature Set

Group	Descriptor and settings	Dim.
Colour	HSV histograms (16 + 8 + 8 bins); mean, std. dev. and skewness of RGB and HSV channels	50
Texture	Uniform LBP (P = 8, R = 1), 10 bins; GLCM (32 levels, distances 1 and 3, 4 angles): contrast, dissimilarity, homogeneity, energy, correlation, ASM	22
Shape	Hu moments of leaf mask (saturation > 0.25), log-scaled	7
Shape	HOG: 9 orientations, 16 × 16 cells, 2 × 2 blocks, reduced by PCA	50
All		129

C. Classifiers

Eight classifiers from scikit-learn [19] were compared, each tuned by grid search with inner stratified 3-fold cross-validation (Table III).

TABLE III
CLASSIFIERS AND HYPER-PARAMETER GRIDS

Classifier	Grid
Logistic regression	C ∈ {0.1, 1, 10}
k-nearest neighbours	k ∈ {1, 3, 5, 9}; uniform or distance weights
Gaussian naive Bayes	—
Decision tree (CART)	max. depth ∈ {none, 10, 20}; min. leaf ∈ {1, 3}
Random forest [20]	300 trees; max. features ∈ {√d, 0.3d}
Gradient boosting [21], [22]	histogram-based, 200 iterations; learning rate ∈ {0.05, 0.1}
SVM, RBF kernel [23]	C ∈ {1, 10, 100}; γ ∈ {scale, 0.01}
Multilayer perceptron	hidden units ∈ {128, 256}; L2 ∈ {10^{-4}, 10^{-2}}; early stopping

D. Evaluation Protocol

1) Nested cross-validation: On the development set, the outer loop was stratified 5-fold cross-validation repeated twice (10 test folds of about 276 images), with the inner grid search inside each outer training set to avoid optimistic tuning bias [24]. Accuracy and macro-averaged precision, recall and F1-score were recorded.

2) Significance tests: The Friedman test compared all eight classifiers over the 10 folds [25]. The two best were compared with the corrected resampled t-test [26], which accounts for overlapping training sets:

$$t = \frac{\bar{d}}{\sqrt{\left(\frac{1}{k} + \frac{n_{\text{test}}}{n_{\text{train}}}\right)\hat{\sigma}_d^2}} \quad (1)$$

where $\bar{d}$ and $\hat{\sigma}_d^2$ are the mean and variance of the per-fold accuracy differences, k = 10 and $n_{\text{test}}/n_{\text{train}} = 1/4$.

3) Feature ablation: Logistic regression, k-NN, random forest and SVM were re-evaluated with the same protocol on five feature sets: colour, texture, shape, colour + texture, and all features.

4) Learning curve: Six classifiers with fixed default settings were trained on 5, 10, 20, 40, 80 or 160 development images per class (five random draws each) and evaluated on the official Test set.

5) Final test: The best classifier was tuned and trained on the full development set and evaluated once on the Test set. A 95% Wilson confidence interval was computed for its accuracy.

6) Background-bias control: Following Noyan [7], each image (resized to 256 × 256) was reduced to eight border pixels (four corners and four edge midpoints; 24 RGB values). A random forest was evaluated on these alone, both by the cross-validation of step 1 and by training on the development set and testing on the Test set. Accuracy above the 33.3% chance level indicates that classes differ in capture conditions rather than leaf appearance.

Experiments ran in Python with scikit-learn [19] and scikit-image [27] on a Google Colab CPU runtime without a GPU.

IV. RESULTS

A. Comparison of Classifiers

Tree ensembles performed best (Table IV): gradient boosting reached 92.04% and random forest 89.87% mean cross-validated accuracy. Logistic regression (87.63%) and the RBF SVM (85.71%) followed, while k-NN was clearly worst (66.79%). Precision, recall and F1-score gave the same ranking.

TABLE IV
NESTED CROSS-VALIDATION RESULTS ON THE DEVELOPMENT SET (1,382 IMAGES, ALL 129 FEATURES)

Classifier	Acc. (%)	Prec. (%)	Rec. (%)	F1 (%)	Mean rank	Time/fold (s)
Gradient boosting	**92.04 ± 1.41**	**92.16**	**92.04**	**92.04**	**1.00**	31.55
Random forest	89.87 ± 1.62	90.05	89.89	89.85	2.00	37.39
Logistic regression	87.63 ± 1.01	87.70	87.64	87.63	3.15	4.04
SVM (RBF)	85.71 ± 1.73	85.84	85.71	85.68	4.20	6.38
MLP	84.01 ± 1.30	83.96	83.99	83.85	4.95	6.60
Decision tree	82.49 ± 1.71	82.75	82.49	82.50	5.85	6.45
Gaussian naive Bayes	79.56 ± 2.25	79.96	79.48	78.91	6.85	0.54
k-NN	66.79 ± 3.44	71.54	66.73	67.03	8.00	5.91

Acc.: accuracy, mean ± std. dev. over 10 outer folds; Prec., Rec., F1: macro-averaged precision, recall and F1-score; MLP: multilayer perceptron; k-NN: k-nearest neighbours. Time includes the inner grid search (Colab CPU).

The Friedman test rejected equal performance ($\chi^2 = 67.34$, df = 7, $p = 5.1 \times 10^{-12}$). Gradient boosting ranked first in every fold (mean rank 1.00), and its 2.17-point advantage over random forest was significant under the corrected resampled t-test (t = 2.78, p = 0.021).

B. Feature Ablation

Colour alone gave 82.4–86.3% accuracy, texture alone 68.3–77.1% and shape alone 58.7–75.1% (Table V). Combining colour and texture was the best feature set for logistic regression (88.75%), k-NN (87.37%) and the SVM (90.56%), the best single-classifier result after the tree ensembles. Adding the 57 shape features lowered accuracy for these three classifiers, most sharply for k-NN (87.37% → 66.79%), because the 50 HOG components dominate the Euclidean distance. Random forest was the only classifier that gained slightly from all features (89.33% → 89.87%).

TABLE V
MEAN ACCURACY (%) BY FEATURE SET, NESTED CROSS-VALIDATION

Feature set	Dim.	LR	k-NN	RF	SVM
Colour	50	82.45	82.42	85.53	86.25
Texture (LBP+GLCM)	22	73.08	68.31	72.65	77.10
Shape (HOG+Hu)	57	75.07	58.68	70.99	73.98
Colour + texture	72	**88.75**	**87.37**	89.33	**90.56**
All features	129	87.63	66.79	**89.87**	85.71

LR: logistic regression; RF: random forest. Best value per column in bold.

C. Learning Curve

Random forest was the most data-efficient classifier at every training size (Fig. 2). With only five images per class it reached

68.4% test accuracy, 12–27 points above the other classifiers, and it rose to 85.1% with 160 images per class. Logistic regression (79.5%) and the SVM (76.3%) improved steadily but needed far more data, naive Bayes and the decision tree plateaued near 73% from about 40 images per class, and k-NN stayed below 53%.

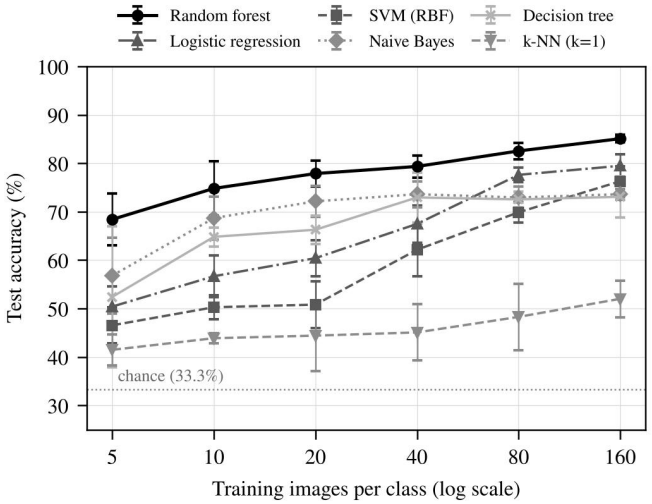


Fig. 2. Learning curves: accuracy on the official 150-image Test set versus training images per class (mean ± std. dev. over five random draws).

D. Final Evaluation on the Official Test Set

Trained on all 1,382 development images (selected learning rate 0.05), gradient boosting classified the 150 test images with 89.33% accuracy (95% CI 83.4–93.3%), a macro F1-score of 89.37% and Cohen's $\kappa = 0.84$ (Table VI, Fig. 3).

TABLE VI
PER-CLASS RESULTS OF GRADIENT BOOSTING ON THE OFFICIAL TEST SET

Class	Precision (%)	Recall (%)	F1 (%)	Images
Healthy	82.5	94.0	87.9	50
Powdery	93.8	90.0	91.8	50
Rust	93.3	84.0	88.4	50
Macro average	89.8	89.3	89.4	150

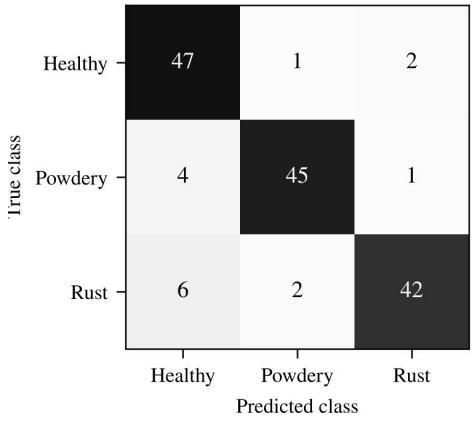


Fig. 3. Confusion matrix of gradient boosting on the official Test set (50 images per class).

Most errors were diseased leaves predicted as healthy: 6 rust and 4 powdery leaves, that is 10 of the 100 diseased test leaves. Viewed as disease detection (diseased versus healthy), the model therefore has a sensitivity of 90% and a specificity of 94% (47 of 50 healthy leaves). Confusions between the two diseases were rare (3 images).

E. Background-Bias Control

A random forest using only eight border pixels per image reached 58.1% ± 2.3% accuracy in cross-validation on the development set, well above the 33.3% chance level. On the official Test set, the same control reached only 38.7% (95% CI 31.2–46.7%), not significantly above chance (one-sided binomial test, $p = 0.098$). The full model showed a smaller drop, from 92.04% in cross-validation to 89.33% on the test set (Fig. 4).

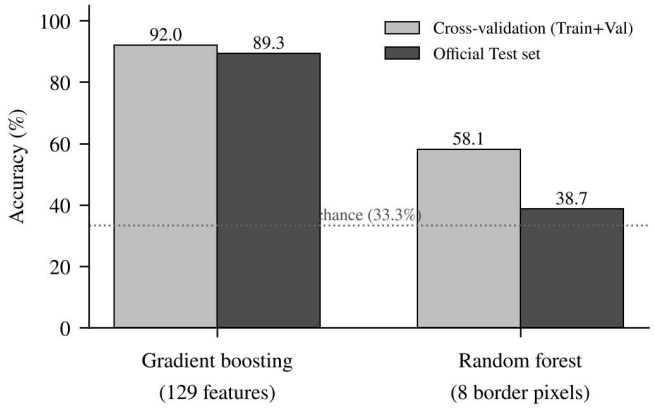


Fig. 4. Cross-validated versus official-test accuracy for the full model and for the eight-border-pixel bias control.

V. DISCUSSION

Classical machine learning is competitive on small field data. With 1,382 training images and no GPU, gradient boosting on hand-crafted features reached 89.33% on the official test set. This is within four points of the 93.33% reported for DenseNet121 on the same dataset [9], and Purba's result lies at the upper edge of our 95% confidence interval (83.4–93.3%); the 91.34% reported for a sequential CNN [10] lies inside it. Because evaluation protocols differ between studies, these comparisons are indicative rather than exact (Table VII).

TABLE VII
COMPARISON WITH PUBLISHED RESULTS ON THE SAME DATASET

Study	Model	Accuracy (%)
Purba [9]	Sequential DenseNet121	93.33 (test)
Al-Zubi [10]	Six-layer sequential CNN	91.34 (overall)
This work	Gradient boosting, 129 hand-crafted features, CPU only	89.33 (official test)

Which algorithm to choose. Gradient boosting was the most accurate and, unlike on laboratory data, significantly better than random forest. Random forest is nevertheless the better choice when very few images are available: it led the learning curve at every size and needed no tuning. Distance-based k-NN should be avoided with mixed high-dimensional features.

Colour and texture capture the symptoms. The best feature set combined colour and texture, which matches the visible symptoms: rust produces orange-brown pustules (a colour cue), and powdery mildew produces a whitish, matt coating (a colour and texture cue). Shape descriptors such as HOG mainly encode leaf outline and framing, which do not distinguish diseases and add noise for distance-based models.

Cross-validation overestimates field performance. The bias control is the key methodological finding. Eight border pixels predicted the class at 58.1% in cross-validation but only 38.7% on the test set. Training photographs of the same class evidently share capture conditions, probably because they were taken in sessions, so random cross-validation folds place near-identical conditions in both training and validation data. The official test set was evidently captured separately, so this shortcut does not transfer. The same effect explains why the full model lost 2.7 points from cross-validation to test. Studies on small field datasets should therefore report accuracy on an independent test set, or use grouped splits by capture session, rather than random cross-validation alone.

Missed disease is the costly error. All but three errors involved the Healthy class, and most were diseased leaves called healthy, probably leaves with few or small lesions. For crop protection a missed infection is more costly than a false alarm, so a deployed system should favour sensitivity, for example by lowering the decision threshold for the disease classes.

Limitations. Results come from one dataset of 1,532 images whose plant species and capture details are not documented, and the test set has only 150 images, so the confidence interval is about ±5 points. Leaves were not segmented from the background, and no deep-learning baseline was trained under our protocol; comparisons with [9], [10] rely on their reported numbers.

VI. Conclusion and Future Work

Eight standard machine learning classifiers were evaluated on a small field dataset of 1,532 leaf photographs labelled Healthy, Powdery and Rust. Gradient boosting on 129 hand-crafted features achieved 92.04% in nested cross-validation and 89.33% ($\kappa = 0.84$) on the untouched official test set, close to published deep-learning results and without a GPU. Colour and texture features carried most of the information, random forest was the most data-efficient model, and the main error was missed disease (sensitivity 90%, specificity 94%). A background-pixel control showed that shared capture conditions inflate cross-validated accuracy on this dataset, so independent test sets are essential for honest reporting.

Future work will segment leaves before feature extraction, use session-aware data splits, evaluate cost-sensitive decision thresholds to reduce missed disease, and compare the classical pipeline with lightweight CNNs trained under the same protocol.

Data and Code Availability

The dataset is publicly available on Kaggle [8]. The complete Google Colab notebook used for all experiments, together with the result files, is provided as supplementary material.

References

[1] K. G. Liakos, P. Busato, D. Moshou, S. Pearson, and D. Bochtis, "Machine learning in agriculture: A review," *Sensors*, vol. 18, no. 8, Art. no. 2674, 2018.

[2] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Comput. Electron. Agric.*, vol. 147, pp. 70–90, 2018.

[3] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Front. Plant Sci.*, vol. 7, Art. no. 1419, 2016, doi: 10.3389/fpls.2016.01419.

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

[5] J. G. A. Barbedo, "Factors influencing the use of deep learning for plant disease recognition," *Biosyst. Eng.*, vol. 172, pp. 84–91, 2018.

[6] R. Thapa, K. Zhang, N. Snavely, S. Belongie, and A. Khan, "The Plant Pathology Challenge 2020 data set to classify foliar disease of apples," *Appl. Plant Sci.*, vol. 8, no. 9, Art. no. e11390, 2020.

[7] M. A. Noyan, "Uncovering bias in the PlantVillage dataset," 2022, *arXiv:2206.04374*.

[8] R. Rahman, "Plant disease recognition dataset," Kaggle. [Online]. Available: https://www.kaggle.com/datasets/rashikrahmanpritom/plant-disease-recognition-dataset (accessed Nov. 2026).

[9] M. Purba, "Plant disease recognition based on leaf images using sequential-based DenseNet architecture," *J. Sci. Appl. Informatics (JSAI)*, vol. 9, no. 1, pp. 153–159, 2026, doi: 10.36085/jsai.v9i1.9988.

[10] A. A. Al-Zubi, "CNN-based detection of powdery mildew and rust in apple orchards for optimizing crop management," *J. Anim. Plant Sci.*, vol. 35, no. 2, pp. 381–389, 2025.

[11] S. Arivazhagan, R. Newlin Shebiah, S. Ananthi, and S. Vishnu Varthini, "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features," *Agric. Eng. Int.: CIGR J.*, vol. 15, no. 1, pp. 211–217, 2013.

[12] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Comput. Electron. Agric.*, vol. 145, pp. 311–318, 2018.

[13] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," *Comput. Electron. Agric.*, vol. 161, pp. 272–279, 2019.

[14] J. G. A. Barbedo, "Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification," *Comput. Electron. Agric.*, vol. 153, pp. 46–53, 2018.

[15] T. Ojala, M. Pietikäinen, and T. Mäenpää, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[16] R. M. Haralick, K. Shanmugam, and I. Dinstein, “Textural features for image classification,” *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, no. 6, pp. 610–621, Nov. 1973.

[17] M.-K. Hu, “Visual pattern recognition by moment invariants,” *IRE Trans. Inf. Theory*, vol. 8, no. 2, pp. 179–187, Feb. 1962.

[18] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2005, vol. 1, pp. 886–893.

[19] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[20] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[21] J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, 2001.

[22] G. Ke *et al.*, “LightGBM: A highly efficient gradient boosting decision tree,” in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017, pp. 3146–3154.

[23] C. Cortes and V. Vapnik, “Support-vector networks,” *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[24] S. Varma and R. Simon, “Bias in error estimation when using cross-validation for model selection,” *BMC Bioinformatics*, vol. 7, Art. no. 91, 2006.

[25] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.

[26] C. Nadeau and Y. Bengio, “Inference for the generalization error,” *Mach. Learn.*, vol. 52, no. 3, pp. 239–281, 2003.

[27] S. van der Walt *et al.*, “scikit-image: Image processing in Python,” *PeerJ*, vol. 2, Art. no. e453, 2014.