Tomato Plant Disease Classification Based on Leaf Images Using ResNet-50 Feature Extraction and Random Forest

Rinaldi Fransius Simbolon[1], Budi Nugroho[2], Anggraini Puspita Sari[3]
Informatics Study Program, Faculty of Computer Science, Universitas Pembangunan Nasional Veteran Jawa Timur, Surabaya, Indonesia

Leaf diseases are among the factors believed to destabilize tomato yields, while manual field identification is slow, expert-dependent, and error-prone. This study develops an image-based tomato leaf disease classification system using transfer learning: a frozen ImageNet-pretrained ResNet-50 serves as a feature extractor (2,048-dimensional vectors) and Random Forest serves as the classifier. A public dataset of 11,000 leaf images in ten classes (nine diseases and one healthy class) was resized to 224 × 224 pixels. Three train–test ratios (90:10, 80:20, 70:30) and three forest sizes (100, 300, 500 trees) were evaluated. The best Random Forest configuration, a 70:30 split with 500 trees, reached 90.21% accuracy, yet training accuracy was 100% in every scenario, indicating overfitting. A Dense classifier (512 units, two hidden layers) trained on the same features reached 97.12% test accuracy, with a training–validation gap of only 1.49–3.19 percentage points across configurations. Errors concentrated on Early Blight, Late Blight, and Target Spot, whose symptoms look alike. The Dense Classifier model was deployed as a Streamlit web prototype. The findings show that ResNet-50 features are highly informative for tomato leaf diseases, but the choice of classifier strongly determines generalization.

Keywords: Tomato Leaf Disease Classification, Resnet-50, Random Forest, Transfer Learning, Feature Extraction, Digital Images.

This is an open access article under the CC BY-NC license

Corresponding Author:
Rinaldi Fransius Simbolon
Informatics Study Program, Faculty of Computer Science, Universitas Pembangunan Nasional Veteran Jawa Timur, Surabaya, Indonesia
21081010046@student.upnjatim.ac.id

1. Introduction

Tomatoes are an important horticultural commodity in Indonesia. According to Statistics Indonesia (Badan Pusat Statistik/BPS), national tomato production reached 1,114,339 tons in 2021, increased by 4.88% to 1,168,744 tons in 2022, and subsequently decreased by 2.14% to approximately 1.14 million tons in 2023 [1]. East Java ranked third among Indonesia's tomato-producing regions, with production reaching 102,099 tons in 2022 and approximately 104,090 tons in 2023, while Malang Regency served as the province's largest production center [1]. Although these production fluctuations cannot be attributed exclusively to plant diseases, empirical studies have established that disease infections can substantially reduce tomato productivity. Fatah et al. (2025), for example, reported that natural Crinivirus infection in tomato plantations in Cianjur, Indonesia, resulted in yield losses ranging from 5.51% to 63.29%, depending on disease severity and tomato variety. Their findings also demonstrated a strong positive correlation between disease severity and yield loss [16]. Similarly, Santosa et al. (2024) investigated bacterial spot disease caused by *Xanthomonas campestris* pv. *vesicatoria* and established a relationship between disease intensity and tomato yield

reduction [17]. These findings demonstrate the importance of effective disease identification as part of broader efforts to maintain tomato productivity.

Disease identification in agricultural fields is still largely performed manually. This approach requires considerable time, labor, expert knowledge, and specialized skills, potentially resulting in identification errors when conducted by insufficiently trained observers [2]. The process is further complicated by the diversity of pathogens affecting tomato plants, including bacteria, fungi, viruses, and mites, which can produce visually similar symptoms. Variations in lighting conditions, leaf orientation, and physical characteristics also increase the difficulty of accurate disease identification. Consequently, automated image-based classification has considerable potential to support faster and more consistent preliminary disease identification.

Advances in deep learning, particularly Convolutional Neural Networks (CNNs), have enabled automatic extraction of visual characteristics, including texture, color, and shape, from plant leaf images [3]. Nevertheless, training CNN architectures from scratch may require substantial computational resources and large training datasets to achieve reliable generalization. When training data are limited, these models may also experience overfitting. Transfer learning provides an alternative by utilizing pretrained CNN architectures as feature extractors, allowing previously learned visual representations to be applied to specific classification tasks. Furthermore, combining CNN-based feature extraction with conventional machine learning classifiers offers an opportunity to investigate different classification strategies without repeatedly training the entire feature extraction network.

Previous studies have demonstrated the effectiveness of machine learning and deep learning approaches for plant disease classification. Setiawan et al. reported that integrating CNN and Random Forest improved classification accuracy from 98.16% to 98.78% in plant leaf disease identification [4]. However, this finding does not necessarily establish that Random Forest consistently outperforms other classifiers across different datasets and feature representations. Sun et al. [5] also demonstrated the effectiveness of combining CNN and Random Forest for hydrogen sensor fault diagnosis, although their research focused on industrial sensor classification rather than agricultural image recognition. In tomato leaf disease classification, Khultsum and Subekti achieved 96% accuracy using Random Forest combined with feature extraction across ten disease classes [6]. Meanwhile, Kotta et al. applied CNN-based classification to identify diseases in tomato leaf images [7]. Although these studies demonstrate the potential of automated tomato disease classification, their different feature extraction and classification configurations make direct comparisons between classification algorithms difficult.

Research employing ResNet-50 has also demonstrated promising classification performance in agricultural applications. Suprihanto et al. achieved maximum accuracies of 92.68% for binary classification and 88.98% for multiclass classification of Robusta coffee leaf diseases [8]. Similarly, Kulsum and Cherid reported 91% accuracy when applying ResNet-50 to classify healthy and diseased apple leaves [9]. Nevertheless, these investigations focused on different plant species and classification tasks, limiting their direct applicability to multiclass tomato leaf disease identification. More importantly, the existing studies discussed above do not establish a systematic comparison between Random Forest and a Dense Classifier using an identical set of frozen CNN-derived features under a controlled experimental framework. This creates a methodological gap concerning how different classifier architectures utilize the same visual representations and how their classification performance, generalization, and computational requirements differ.

To address this gap, the present study employs ImageNet-pretrained ResNet-50 [10] as a frozen feature extractor to generate 2,048-dimensional feature vectors from ten tomato leaf classes, comprising nine disease categories and one healthy category. Random Forest and a Dense Classifier are subsequently evaluated using the same extracted feature representations. This comparative design is important because both classifiers receive identical input features, allowing differences in classification performance to be examined primarily in relation to their respective learning mechanisms rather than differences in feature extraction. Random Forest utilizes ensemble decision trees and majority voting, whereas a Dense Classifier applies trainable weights and nonlinear activation functions to learn relationships among extracted features. Evaluating these approaches within a shared feature extraction framework therefore provides a more consistent basis for investigating their respective classification capabilities, overfitting tendencies, and computational efficiency.

The contributions of this study are as follows:

1. Development of a hybrid ResNet-50 and Random Forest classification pipeline for identifying ten tomato leaf classes, including nine disease categories and one healthy category.
2. Investigation of the effects of different training-to-testing data-splitting ratios and the number of decision trees on Random Forest classification accuracy, overfitting tendencies, and training time.
3. Comparative evaluation of Random Forest and a Dense Classifier using identical ResNet-50-derived feature representations to examine differences in classification performance, generalization, and computational efficiency.
4. Development of a web-based application prototype to facilitate image-based tomato leaf disease classification.

2. Methods

Research Workflow

The research was conducted in six stages: (1) dataset collection; (2) image preprocessing (resizing and normalization); (3) data splitting; (4) feature extraction using ResNet-50; (5) classification using Random Forest and a dense classifier; and (6) evaluation and application implementation. The experiments were conducted systematically, with each parameter variation evaluated and compared using accuracy, classification reports, and confusion matrices.

Dataset

The dataset was obtained from the public Kaggle repository (kaustubhb999/tomatoleaf) and consists of 11,000 tomato leaf images distributed across ten classes: Bacterial Spot, Early Blight, Late Blight, Leaf Mold, Septoria Leaf Spot, Two-Spotted Spider Mites, Target Spot, Mosaic Virus, Yellow Leaf Curl Virus (TYLCV), and Healthy. The class distribution was balanced; under the 70:30 split, each class contained 330 test images. The images were stored in class-specific folders, which served as their labels.

Image Preprocessing and Data Splitting

All images were resized to 224 × 224 pixels to match the standard input dimensions of ResNet-50. They were then normalized using the built-in ResNet-50 preprocess_input function in Caffe mode, which converts RGB channels to BGR and subtracts the ImageNet pixel mean values ([103.939, 116.779, 123.68]). No data augmentation was performed. Stratified data splitting was applied using random_state

= 42 to preserve class proportions and ensure reproducibility. The three data-splitting schemes are presented in Table 1.

Table 1. Dataset Splitting Schemes

Training-to-Test Ratio	Training Data	Test Data
90:10	9,900	1,100
80:20	8,800	2,200
70:30	7,700	3,300

For experiments involving three subsets, the training portion should subsequently be divided into training and validation data using a stratified procedure. The exact validation proportion and resulting sample counts should be reported. The same partition indices must be applied to both classification algorithms to ensure a consistent comparison.

Feature Extraction Using ResNet-50

This study employed transfer learning [11], whereby visual knowledge from a model pretrained on ImageNet [12] was reused for a new task. ResNet-50 [10] introduces shortcut connections within residual blocks, enabling deep networks to be trained effectively while mitigating the vanishing gradient problem. The output of a residual block is expressed as $H(x) = F(x) + x$ represents the block input and $F(x)$ denotes the nonlinear transformation within the block. The architecture consists of an initial convolutional layer (7×7, stride 2) followed by max pooling, four stages of bottleneck blocks containing 3, 4, 6, and 3 blocks, with output filter dimensions of 256, 512, 1,024, and 2,048, respectively, followed by average pooling.

The pretrained ImageNet weights were frozen, and no fine-tuning was performed. The final classification layer was removed (include_top = False) and replaced with Global Average Pooling, representing each image as a 2,048-dimensional feature vector. All feature vectors and their corresponding labels were saved in features.npy and labels.npy, respectively, eliminating the need to repeat feature extraction for each classification experiment.

Classification Using Random Forest

Random Forest [13] is an ensemble learning method that constructs multiple decision trees, each trained on bootstrap samples from the training data (bagging) while considering only a random subset of features at each split (feature bagging). The final prediction is determined through majority voting: $\hat{y} = \text{mode}(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)$, where $\hat{y}_i$ denotes the prediction of the i-th tree. The variables investigated in this study were the data-splitting ratio (90:10, 80:20, and 70:30) and the number of trees (100, 300, and 500), resulting in nine experimental scenarios.

Dense Classifier as a Comparative Model

A Dense Classifier was implemented as the second classification approach using the same 2,048-dimensional feature vectors generated by ResNet-50. Unlike Random Forest, which relies on decision-tree ensembles, the Dense Classifier learns nonlinear relationships among extracted features through trainable weights and activation functions. Five architectural configurations were investigated by varying the number of neurons in the first Dense layer and the number of hidden layers. The evaluated configurations included 128, 256, and 512 neurons, with one to three hidden layers. Dropout with a rate of 0.5 and batch normalization were incorporated to reduce overfitting and improve training stability.

The model was optimized using the Adam optimizer, with early stopping implemented to prevent unnecessary training after validation performance stopped improving. The maximum training duration was limited to 100 epochs. The complete experimental configuration, including batch size, initial learning rate, loss function, Adam parameters, early-stopping monitoring variable, patience, and restoration of the best model weights, should be specified to ensure reproducibility. The selected architecture was determined using validation performance rather than independent test accuracy. Following model selection, the final configuration was evaluated using the same independent testing subset applied to Random Forest.

Evaluation Metrics

Model performance was evaluated using accuracy, precision, recall, F1-score, and confusion matrices. For multiclass classification, precision, recall, and F1-score were calculated for individual classes and summarized using macro and weighted averages [14], as follows:

Accuracy = (TP + TN) / (TP + TN + FP + FN)
Precision = TP / (TP + FP)
Recall = TP / (TP + FN)
F1-score = 2 × (Precision × Recall) / (Precision + Recall)

Training time was also recorded to compare computational efficiency between the two classification approaches. The difference between training and validation accuracy was examined to identify potential overfitting..

Application Implementation

The selected Dense Classifier was deployed through a web application developed using Streamlit. The application accepts JPG and PNG images, resizes the uploaded image to 224 × 224 pixels, applies the ResNet-50 preprocessing procedure, and extracts the corresponding 2,048-dimensional feature representation.The extracted features are subsequently processed by the trained Dense Classifier to generate predicted disease categories and their associated confidence scores. The application was developed using Python 3.11 and the TensorFlow/Keras and scikit-learn libraries.

3. Result And Discussion

Effect of Data-Splitting Ratios and Number of Trees

Table 2 summarizes the results of the nine Random Forest experimental scenarios.

Table 2. Evaluation Results of ResNet-50 + Random Forest

Data Split Ratio	Number of Trees	Training Accuracy	Validation Accuracy	Training Time
90:10	100	100%	88.09%	41.89 s
90:10	300	100%	88.73%	120.80 s
90:10	500	100%	89.45%	198.91 s
80:20	100	100%	88.14%	35.61 s
80:20	300	100%	89.09%	103.94 s
80:20	500	100%	89.36%	173.16 s
70:30	100	100%	88.21%	29.27 s
70:30	300	100%	89.70%	87.67 s
70:30	500	100%	90.21%	146.12 s

Table 2 presents the evaluation results of nine Random Forest experimental scenarios using three data-splitting ratios and three different numbers of trees. The highest validation accuracy of 90.21% was obtained using a 70:30 training-to-test ratio and 500 trees. In all scenarios, the model achieved 100% training accuracy, whereas validation accuracy ranged from 88.09% to 90.21%. This difference indicates a substantial gap between training and validation performance, suggesting that the Random Forest model experienced overfitting when processing the 2,048-dimensional ResNet-50 feature representations.

The results also show that increasing the number of trees improved validation accuracy across all data-splitting ratios, although the improvement became smaller as the number of trees increased. For the 70:30 ratio, increasing the number of trees from 300 to 500 improved validation accuracy by 0.51 percentage points, while the recorded training time increased from 87.67 to 146.12 seconds. Meanwhile, differences in accuracy among the three data-splitting ratios remained relatively small. These findings indicate that the number of trees had a more noticeable effect on model performance than the data-splitting ratio within the experimental configurations examined. However, each experimental scenario was conducted only once using random_state = 42. Therefore, the observed differences should be interpreted as results from the specific data partition used in this study rather than evidence of consistent performance across different random data allocations.

Class-Wise Performance Analysis

For the best configuration (70:30 ratio, 500 trees), the macro and weighted averages of precision, recall, and F1-score were all 0.90, indicating relatively balanced performance across classes. Table 3 presents the classes with the highest and lowest performance.

Table 3. Performance of Selected Classes (70:30 Ratio, 500 Trees; 330 Test Images per Class)

Class	Result
Tomato Healthy	F1-score: 0.98
Tomato Yellow Leaf Curl Virus	F1-score: 0.97; 311 out of 330 images correctly classified
Tomato Mosaic Virus	F1-score: 0.96
Two-Spotted Spider Mites	305 out of 330 images correctly classified; 25 misclassified as Target Spot, Leaf Mold, and other classes
Tomato Early Blight	Recall: 0.78; F1-score: 0.81; 256 out of 330 images correctly classified

The class-wise evaluation of the best Random Forest configuration (70:30 ratio and 500 trees) showed differences in classification performance among the ten tomato leaf classes. The macro and weighted averages of precision, recall, and F1-score were approximately 0.90, indicating relatively balanced overall performance. Tomato Healthy achieved the highest reported F1-score of 0.98, followed by Tomato Yellow Leaf Curl Virus (TYLCV) at 0.97 and Tomato Mosaic Virus at 0.96.

In contrast, Early Blight showed comparatively lower classification performance, with a recall of 0.78 and an F1-score of 0.81. The model correctly classified 256 out of 330 Early Blight images. Misclassifications were distributed among Bacterial Spot, Late Blight, and Healthy categories. Errors were also identified in the Two-Spotted Spider Mites class, where 25 images were incorrectly classified into Target Spot, Leaf Mold, and other categories.

These classification errors may be associated with similarities in the visual characteristics of certain diseases. Early Blight and Target Spot, for example, can produce brown necrotic lesions with concentric patterns, while Late Blight may exhibit dark lesions that complicate visual identification. These overlapping characteristics may reduce the ability of the extracted features to distinguish between visually similar disease categories. The findings suggest that classification performance depends not only on the overall quality of the extracted features but also on the distinctiveness of the symptoms represented in each class.

The class-wise results presented in this study are limited to the reported precision, recall, F1-score, and selected misclassification observations. A complete confusion matrix and detailed per-class metrics for all ten categories were not available in the reported results. Consequently, a comprehensive comparison of misclassification patterns across all classes cannot yet be established.

These errors are plausible from a plant pathology perspective. Early Blight and Target Spot both produce brown necrotic lesions with concentric ring patterns [15], while Late Blight causes dark leaf lesions that can be difficult to distinguish visually during the early stages of infection. The concentration of misclassifications among classes with similar symptoms suggests that more discriminative features are needed to distinguish these classes, potentially through fine-tuning or the addition of training data.

Dense Classifier Results

Table 4. Evaluation Results of ResNet-50 + Dense Classifier

Configuration	Units	Layers	Epochs	Training Accuracy	Validation Accuracy	Test Accuracy	Training Time
1	128	1	50	99.68%	96.49%	97.06%	22.96 s
2	256	1	54	99.89%	96.75%	96.85%	34.31 s
3	256	2	56	98.56%	96.82%	96.94%	40.38 s
4	512	2	39	98.62%	96.69%	97.12%	48.74 s
5	512	3	49	97.79%	96.30%	96.85%	61.74 s

All configurations demonstrated high and consistent performance, with validation accuracy ranging from 96.30% to 96.82% and test accuracy ranging from 96.85% to 97.12%. The best-performing configuration used 512 units and two hidden layers, achieving a test accuracy of 97.12% and a test loss of 0.0766.

The differences between training and validation accuracy ranged from 1.49 to 3.19 percentage points, substantially smaller than those observed with Random Forest. Early stopping terminated training between epochs 39 and 56, well before the maximum limit of 100 epochs. Increasing model complexity did not consistently improve performance: Configuration 5, which used 512 units and three hidden layers, required the longest training time (61.74 seconds) but achieved the lowest validation accuracy. Configuration 1, the simplest model with 128 units and one hidden layer, achieved a test accuracy only 0.06 percentage points below that of the best configuration, while requiring just 22.96 seconds to train.

Comparison of Random Forest and Dense Classifier

Table 5. Summary of Model Comparison

Classifier	Best Accuracy	Training–Validation Accuracy Gap	Training Time
Random Forest (70:30 ratio, 500 trees)	90.21%	Approximately 9.8 percentage points	146.12 s

Classifier	Best Accuracy	Training–Validation Accuracy Gap	Training Time
Dense classifier (512 units, 2 layers)	97.12% (test)	1.49–3.19 percentage points across configurations	48.74 s

The dense classifier outperformed Random Forest by approximately seven percentage points in accuracy while also requiring less training time. Several plausible explanations, which remain hypotheses and were not directly tested in this study, may account for this difference. (1) Dense layers process all 2,048 features simultaneously through weighted linear combinations, enabling them to capture interactions among features, whereas decision trees split on one feature at a time using axis-aligned decision boundaries. (2) The dense classifier employs multiple regularization mechanisms, including dropout, batch normalization, and early stopping, whereas fully grown Random Forest trees can readily memorize training data. (3) Gradient-based optimization may be better suited to the continuous representations generated by CNNs. (4) The high feature-to-sample ratio may make it more difficult for individual decision trees to select relevant features at each split.

This comparison is indicative rather than definitive for two reasons: Random Forest hyperparameters other than the number of trees were not optimized, and the data-splitting protocols used for the two methods were not entirely identical. Nevertheless, the approximately seven-percentage-point accuracy gap and the differences in overfitting behavior suggest that, when using frozen ResNet-50 features, the dense classifier is more suitable than Random Forest. This finding also demonstrates that combining a CNN with Random Forest does not automatically provide superior performance; positive results reported in other studies [4], [5] may not generalize to every dataset and feature extraction configuration.

Application Implementation and Functional Testing

The application provides an image-upload interface and a results panel displaying the predicted class and confidence percentages for each class. Black-box functional testing was conducted using five test images, as presented in Table 6.

Table 6. Application Functional Testing Results

No.	Actual Class	Predicted Class	Confidence	Status
1	Bacterial Spot	Bacterial Spot	79.8%	Successful
2	Early Blight	Early Blight	70.2%	Successful
3	Late Blight	Late Blight	42%	Successful
4	Healthy	Healthy	91%	Successful
5	Leaf Mold	Late Blight	25%	Failed

The entire workflow, from image uploading and feature extraction to classification, operated without system errors. Four out of five samples were correctly classified. The Leaf Mold sample was incorrectly predicted as Late Blight with low confidence (25%), while the correctly classified Late Blight sample received a confidence score of only 42%.

Low confidence scores can serve as indicators of uncertainty. For example, the application could display a warning or request an additional image whenever confidence falls below a predefined threshold. Five samples are insufficient to provide a statistically reliable measure of model performance; the purpose of this test was to verify application functionality. Since feature extraction is shared by both classifiers, replacing

Random Forest with the dense classifier in the application would require relatively minor modifications and is strongly recommended based on the findings in Section 3.4.

Limitations

First, the evaluation used a single public dataset and did not include testing on images captured directly in agricultural fields under varying backgrounds, lighting conditions, and camera angles. Second, ResNet-50 was used without fine-tuning or data augmentation. Third, each experimental scenario was executed only once, so variability across repeated experiments was not measured. Fourth, other Random Forest hyperparameters, such as maximum tree depth and the number of features considered at each split, were not optimized. Fifth, application testing involved only five samples.

4. Conclusion

This study developed a tomato leaf disease classification system using ResNet-50 for feature extraction and Random Forest and Dense Classifier algorithms for classification. The Random Forest model achieved its highest reported validation accuracy of 90.21% using a 70:30 data split and 500 trees. Meanwhile, the Dense Classifier achieved a test accuracy of 97.12% using 512 neurons and two hidden layers. However, these results were obtained from different evaluation protocols; therefore, they do not provide a direct basis for determining which classifier performs better. A controlled evaluation using the same independent test set is required to establish a valid comparison.

The application was also evaluated through black-box testing using five images, of which four were classified correctly. Although this result demonstrates that the application can perform the intended classification process, the limited number of test images does not provide sufficient evidence of its reliability in practical agricultural settings. Furthermore, the models were evaluated using a publicly available dataset rather than images collected directly from agricultural fields. Future research should therefore conduct external validation using field-collected tomato leaf images under varying environmental conditions, including differences in lighting, image backgrounds, and disease appearance. Such evaluation is necessary to assess the model's generalizability and reliability before the system can be recommended for practical agricultural disease diagnosis.

5. References

[1] Badan Pusat Statistik, *Produksi Tanaman Sayuran (Tomat), 2021–2023*. Jakarta, Indonesia: BPS.

[2] D. Iswantoro and D. Handayani UN, "Klasifikasi Penyakit Tanaman Jagung Menggunakan Metode Convolutional Neural Network (CNN)," *J. Ilm. Univ. Batanghari Jambi*, vol. 22, no. 2, p. 900, 2022, doi: 10.33087/jiubj.v22i2.2065.

[3] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Comput. Electron. Agric.*, vol. 147, pp. 70–90, 2018, doi: 10.1016/j.compag.2018.02.016.

[4] M. J. Setiawan, B. Nugroho, and A. P. Sari, "Klasifikasi Penyakit Daun Tanaman Menggunakan Algoritma CNN dan Random Forest," *Teknol. J. Ilm. Sist. Inf.*, vol. 12, pp. 1–8, 2022.

[5] Y. Sun, H. Zhang, T. Zhao, Z. Zou, B. Shen, and L. Yang, "A New Convolutional Neural Network With Random Forest Method for Hydrogen Sensor Fault Diagnosis," *IEEE Access*, vol. 8, pp. 85421–85430, 2020, doi: 10.1109/ACCESS.2020.2992231.

[6] U. Khultsum and A. Subekti, "Penerapan Algoritma Random Forest dengan Kombinasi Ekstraksi Fitur Untuk Klasifikasi Penyakit Daun Tomat," *J. Media Inform. Budidarma*, vol. 5, no. 1, p. 186, 2021, doi: 10.30865/mib.v5i1.2624.

[7] C. R. Kotta, D. Paseru, and M. Sumampouw, "Implementasi Metode Convolutional Neural Network untuk Mendeteksi Penyakit Pada Citra Daun Tomat," *J. Pekommas*, vol. 7, no. 2, pp. 123–132, 2022, doi: 10.56873/jpkm.v7i2.4961.

[8] S. Suprihanto, I. Awaludin, M. Fadhil, and M. A. Z. Zulfikor, "Analisis Kinerja ResNet-50 dalam Klasifikasi Penyakit pada Daun Kopi Robusta," *J. Inform.*, vol. 9, no. 2, pp. 116–122, 2022, doi: 10.31294/inf.v9i1.13049.

[9] U. Kulsum and A. Cherid, "Penerapan Convolutional Neural Network Pada Klasifikasi Tanaman Menggunakan ResNet50," *Simkom*, vol. 8, no. 2, pp. 221–228, 2023, doi: 10.51717/simkom.v8i2.191.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[11] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010, doi: 10.1109/TKDE.2009.191.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[13] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[14] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, 2006, doi: 10.1016/j.patrec.2005.10.010.

[15] J. B. Jones, T. A. Zitter, and S. A. Miller, "Compendium of Tomato Diseases and Pests," *Am. Phytopathol. Soc.*, 2016, doi: 10.1094/9780890544341.fm.