Journal of Intelligent Computing and Applied Informatics (JICAI)
Vol. 1, No. 1, December, pp. 10–17 | e-ISSN: xxxx-xxxx | p-ISSN: xxxx-xxxx
DOI: https://doi.org/10.xxxxx/jicai.vXiX.XXX
Journal homepage: https://ejournal.unuja.ac.id/index.php/jicai

RESEARCH ARTICLE

Web-Based Chili Leaf Disease Classification Using a Fine-Tuned MobileNet Convolutional Neural Network and Streamlit

Ulfa Ma'idatul Hasanah[1,*], Mochammad Faid[1], M. Noer Fadli Hidayat[1],

[1]*Department of Informatics Engineering, Faculty of Engineering, Universitas Nurul Jadid, Probolinggo, Indonesia*
***Corresponding author:** *ulfamh2004@gmail.com*

ARTICLE INFO

Article history:
Received: 27 October 2026
Revised: 10 November 2026
Accepted: 18 November 2026
Published: 01 December 2026

Keywords:
Chili leaf disease
Plant disease classification
MobileNet
Transfer learning
Streamlit web application

ABSTRACT

Chili plants are frequently damaged by leaf diseases whose early symptoms look alike, which makes visual diagnosis difficult for farmers without agronomic training. This study develops and evaluates a web-based system that classifies chili leaf images into seven categories (healthy, whitefly infestation, yellowish, leaf curl virus, leaf spot, damping off, and veinal mottle virus) using a fine-tuned MobileNet convolutional neural network. A public dataset of 4,688 images was resized to 224 × 224 pixels, normalised with the MobileNet preprocessing function, and augmented. A pre-trained MobileNet was fine-tuned by unfreezing its last 30 layers and adding global average pooling, dropout (rate 0.4), and a seven-way softmax layer; training used the Adam optimiser with a learning rate of 0.00001 for 10 epochs. On an evaluation set of 100 images the model reached an accuracy of 95%, with weighted precision, recall, and F1-score of 0.96, 0.95, and 0.95. Damping off, healthy, and veinal mottle leaves were classified without error, whereas yellowish leaves had the lowest recall (0.60) and were mostly confused with leaf curl. The trained model was deployed in a Streamlit application that returns the predicted disease, its confidence, the three most probable classes, and treatment recommendations. These results demonstrate the feasibility of a lightweight network for chili leaf disease recognition on curated images; because the evaluation set is small and the validation images were also used for monitoring, independent testing and field validation are still required before the system can be used for diagnosis.

I. INTRODUCTION

Chili is a high-value horticultural crop that is widely cultivated in Indonesia, but its yield and quality are repeatedly threatened by leaf diseases and pest infestations. The visible symptoms (colour change, spots, yellowing, and curling) are often similar across diseases, so farmers without agronomic training find it hard to identify the cause and usually react only after the damage has spread. An automatic, image-based diagnosis tool that can be used from an ordinary phone or computer browser could close this gap.

Deep learning has made image-based plant disease recognition practical [1], [2]. Early studies on chili leaves relied on hand-crafted colour and texture features fed to conventional classifiers, such as learning vector quantization [3] and a support vector machine [4], or on artificial neural networks [5]. More recent studies applied convolutional neural networks (CNNs), including MobileNet-based models [6],[7],[8], a VGG16 model [9], and plain CNNs [10], [11]. Several of these studies used between 30 and 2,000 images and three to five classes (Table 1), which limits what they can say about realistic diagnosis, and a trained model on its own does not help farmers unless it is delivered through an accessible interface.

MobileNet [12] is a suitable candidate for such a tool because it factorises standard convolutions into depthwise and pointwise convolutions, which keeps the number of parameters and the computation low. This paper presents a web-based chili leaf disease classifier built on a fine-tuned MobileNet. The model distinguishes seven leaf conditions (healthy leaves, whitefly infestation, yellowish leaves, leaf curl virus, leaf spot, damping off, and veinal mottle virus) using 4,688 images from a public dataset [13], and it is deployed in a Streamlit application [14]. Performance is measured with accuracy, precision, recall, and F1-score.

The main contributions of this paper are as follows:

- A seven-class chili leaf classifier covering healthy leaves, a pest infestation, a symptom, fungal damage, and viral diseases, obtained by fine-tuning a pre-trained MobileNet on 4,688 public images; it reaches 95% accuracy and a weighted F1-score of 0.95 on a 100-image evaluation set, which is a preliminary estimate.

© 2026 The Author(s). Published by LP3M, Universitas Nurul Jadid, Indonesia. This is an open access article under the CC BY 4.0 license (https://creativecommons.org/licenses/by/4.0/).

- A transparent per-class evaluation (precision, recall, F1-score, and confusion matrix) with Wilson confidence intervals, which shows that the main weakness is the confusion of yellowish leaves with leaf curl and leaf spot (recall 0.60), together with an explicit account of the limitations of the evaluation protocol.
- A proof-of-concept Streamlit application that shows the predicted class, its confidence, the three most probable classes, and generic first-aid control messages; it is a demonstration of deployment, not a validated decision-support tool.

The remainder of the paper reviews related work (Section II), describes the data, method, and experimental setup (Section III), reports the results (Section IV), discusses them (Section V), and concludes (Section VI).

II. RELATED WORK

Studies on chili leaf disease recognition can be grouped by how features are obtained. Methods based on hand-crafted features first convert the image to a colour or texture representation and then classify it with a conventional learner. Siswipraptini *et al.* [3] converted images from RGB to HSV, extracted edges with the Canny detector, and classified 30 images of three diseases with learning vector quantization, obtaining an accuracy of 80% and an F1-score of 81%. Nurmadinah *et al.* [4] combined gray-level co-occurrence matrix (GLCM) texture features and HSV colour statistics with an RBF-kernel support vector machine on 1,150 images of five classes and reported an accuracy of 91.30%. Such pipelines depend on segmentation, thresholding, and background-removal steps tuned to the capture conditions, and the first of them uses a very small sample.

Neural networks learn features directly from the image. Arafa *et al.* [5] trained an artificial neural network on 1,600 images of four classes and reported an accuracy of 93%. Convolutional models were applied by Firdaus and Litanianda [10], who classified four classes (healthy, whitefly, yellowish, and leaf spot) and reported 88% training and 82% validation accuracy, and by Munir *et al.* [7], who adapted MobileNet and reported a training accuracy of 97.22% after ten epochs on a small dataset. Putri *et al.* [6] also evaluated MobileNet for chili leaf disease classification.

Transfer learning from large pre-trained networks has produced the highest reported figures. Mashuri *et al.* [9] fine-tuned VGG16 on five classes (healthy, yellowish, whitefly, leaf curl, and leaf spot) and reported an average accuracy of about 99.6%. Jajila *et al.* [8] fine-tuned MobileNetV2 on six classes and reported an accuracy of 99% on a test set of 1,500 images. Agustin *et al.* [11] compared a CNN with an SVM on 2,000 images in five classes (bacterial, fungal, viral, healthy, and non-leaf) and found accuracies of 85% and 78%, respectively. Table 1 summarises these studies.

Table 1. Summary of related studies on chili leaf disease classification

Study	Method	Images / classes	Reported result
Siswipraptini et al. [3]	LVQ (HSV + Canny edges)	30 / 3	Acc. 80%; F1 81%
Nurmadinah et al. [4]	GLCM + HSV + SVM (RBF)	1,150 / 5	Acc. 91.30%
Arafa et al. [5]	ANN	1,600 / 4	Acc. 93%
Firdaus and Litanianda [10]	CNN	– / 4	Val. acc. 82%
Munir et al. [7]	MobileNet	100 / 3	Train. acc. 97.22%
Agustin et al. [11]	CNN (SVM: 78%)	2,000 / 5	Acc. 85%
Mashuri et al. [9]	VGG16 (transfer learning)	– / 5	Acc. ≈ 99.6%
Jajila et al. [8]	MobileNetV2 (transfer learning)	– / 6	Acc. 99% (1,500 test images)
This study	**Fine-tuned MobileNet**	**4,688 / 7**	**Acc. 95%; F1 0.95 (weighted)**

Note: results are those reported in the original studies, which used different datasets and evaluation protocols and are therefore not directly comparable. Acc. = accuracy; Val. = validation; Train. = training; LVQ = learning vector quantization; – = not stated.

Beyond chili, plant disease recognition has largely been benchmarked on PlantVillage [15], from which Mohanty *et al.* [1] trained deep networks on 54,306 laboratory images of 38 classes. The same study reported that accuracy fell to 31.4% on images collected under conditions different from the training data, which shows that high accuracy on curated images does not guarantee performance in the field. PlantDoc [16] was created to address this gap with 2,598 images of 13 plant species taken in non-laboratory conditions. Lightweight architectures are attractive for such applications: MobileNet [12] and its successors MobileNetV2 [17] and MobileNetV3 [18] were designed for resource-constrained devices, and EfficientNet [19] offers a scaling family that trades accuracy against model size. These models have not yet been compared with MobileNet on chili leaf data in the studies reviewed here.

Table 1 should be read with caution. The studies differ in the number and definition of classes (some mix diseases, pests, and symptoms, or add non-leaf classes), in the source and size of the images, in the split between training and evaluation data, and in whether the reported figure is training, validation, or test accuracy. Like most chili studies, the present work uses curated images only. Its contribution is therefore a seven-class, openly reported baseline and a deployment demonstration rather than a state-of-the-art accuracy.

III. MATERIALS AND METHODS

A. Dataset

The images come from the public Chilli Plant Diseases Dataset on Kaggle [13], which contains 6,755 images in eight classes. The anthracnose class was excluded because anthracnose mainly affects the fruit, whereas this study targets leaf diseases. The remaining seven classes contain 4,688 images (Table 2). Labels are given by the folder name of each image. Before training, the images were inspected, and corrupted, blurred, duplicated, or mislabeled images were removed.

The data were divided into training, validation, and test subsets. The 80/10/10 ratio shown in Fig. 1 is the nominal design of the workflow; the partition actually used (Table 2) is 4,488 training, 100 validation, and 100 test images, that is, about 95.7%, 2.1%, and 2.1% of the 4,688 images. Ten validation and ten test images were set aside for each disease class and 40 of each for the healthy class, and all remaining images were used for training. The evaluation subsets are therefore small, and the class distribution is imbalanced: veinal mottle virus accounts for 43.1% of all images and damping off for only 4.0%.

Table 2. Number of images per class and data subset

Class	Training	Validation	Test	Total
Healthy	320	40	40	400
Whitefly	500	10	10	520
Yellowish	500	10	10	520
Leaf curl virus	500	10	10	520
Leaf spot	500	10	10	520
Damping off	168	10	10	188
Veinal mottle virus	2,000	10	10	2,020
Total	**4,488**	**100**	**100**	**4,688**

B. Proposed Method

The research workflow, summarised in Fig. 1, has nine stages: dataset collection, dataset splitting, preprocessing, MobileNet model construction, training, evaluation, implementation in a Streamlit web application, system testing, and analysis of the results.

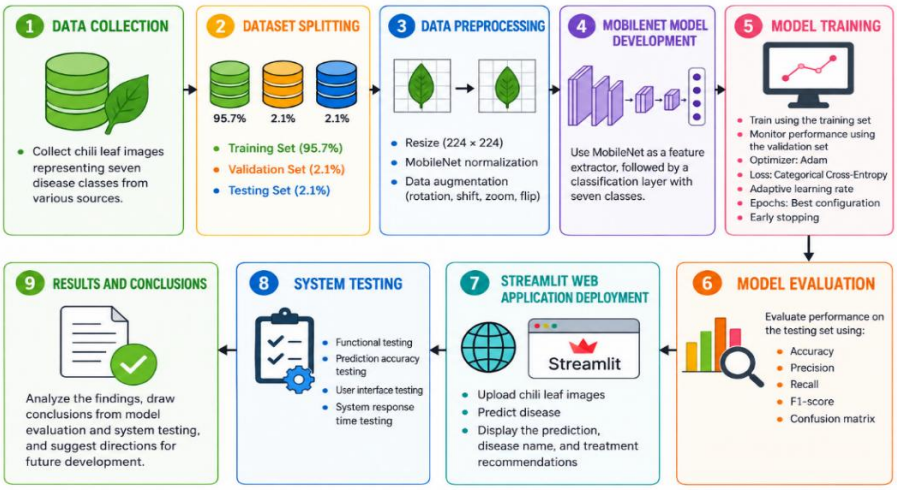

Fig. 1. Stages of the proposed system research.

1) Preprocessing and Augmentation

All images are resized to 224 × 224 pixels, the input size of MobileNet, and scaled with the MobileNet preprocessing function, which maps pixel values from the range [0, 255] to [−1, 1]. Augmentation is applied to the training images only and consists of random horizontal flips, rotations, zooming, and shifts, which increase the variation in leaf position and orientation and reduce the risk of overfitting. The validation images are only resized and preprocessed.

2) Model Architecture

MobileNet [12] replaces standard convolutions with depthwise convolutions, which filter each channel separately, followed by 1 × 1 pointwise convolutions, which combine the channels; this reduces the number of parameters and the computation. In this study, a MobileNet backbone with pre-trained weights serves as the feature extractor and is fine-

tuned, following the transfer-learning principle that features learned on a large dataset transfer to related tasks [20]. All layers except the last 30 are frozen, so only the deepest, most task-specific layers are adapted to chili leaf images. The backbone output passes through global average pooling, a dropout layer with a rate of 0.4 [21], and a dense layer with seven units and softmax activation, which converts the output scores into a probability for each of the seven classes. The network is trained by minimizing the categorical cross-entropy loss between these predicted probabilities and the one-hot encoded class labels. The resulting model has 3,236,039 parameters, of which 2,402,823 are trainable and 833,216 are frozen.

3) Web Application

The saved model is loaded once by the Streamlit application [14] and cached, so that it is not reloaded at every interaction. The user uploads a leaf image in JPG, JPEG, or PNG format; the application converts it to RGB, resizes it to 224 × 224 pixels, applies the same preprocessing function as in training, and passes it to the model. The application then displays the class with the highest probability, its confidence, the three most probable classes, and a list of treatment recommendations for the predicted disease (Figs. 2, 3, 4). Functional testing covered launching the application, loading and caching the model, uploading an image, and displaying the prediction.

C. Experimental Setup

Training and evaluation were run in Google Colab with Python, TensorFlow [22], and Keras; scikit-learn [23] was used to compute the confusion matrix and the classification report. The random seeds of Python, NumPy, and TensorFlow were fixed to 42. The model was optimised with Adam [24]; Table 3 lists the full training configuration. Ten epochs were used because training and validation accuracy were still increasing, and the losses were still decreasing steadily, up to the tenth epoch. A single training run was performed, and the model after the tenth epoch was saved and used in the application. The script imported the Keras callbacks EarlyStopping, ReduceLROnPlateau, and ModelCheckpoint, but the reported run is described by a fixed number of epochs and the final-epoch weights, and the report does not document that any callback changed training; the stopping criterion was therefore the preset epoch count. The hardware accelerator and the library versions were not recorded, and the exact magnitudes of the augmentation transforms are not given in the source report, so these items should be added to the repository when the code is released.

Table 3. Training configuration

Parameter	Setting
Backbone	MobileNet with pre-trained weights; last 30 layers trainable
Input size	224 × 224 × 3
Classification head	Global average pooling, dropout (0.4), dense layer (7 units, softmax)
Loss function	Categorical cross-entropy
Optimizer	Adam, learning rate 0.00001
Batch size / epochs	32 / 10
Random seed	42
Software	Google Colab; TensorFlow and Keras; scikit-learn

D. Evaluation Metrics

Performance is evaluated on the 100-image evaluation set (Table 2) with accuracy, precision, recall, and F1-score computed from the confusion matrix. For class k, TP_k, FP_k, and FN_k are the numbers of true positives, false positives, and false negatives:

$$\text{Accuracy} = \frac{\sum_{k=1}^{K} TP_k}{N} \tag{3}$$

$$\text{Precision}_k = \frac{TP_k}{TP_k + FP_k} \tag{4}$$

$$\text{Recall}_k = \frac{TP_k}{TP_k + FN_k} \tag{5}$$

$$F1_k = \frac{2 \times \text{Precision}_k \times \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k} \tag{6}$$

where N is the number of evaluation images and $K = 7$. Overall precision, recall, and F1-score are reported as support-weighted averages over the classes, as in the scikit-learn classification report, and macro averages are given for reference.

IV. RESULTS

A. Training Behaviour

Over the ten epochs, training accuracy increased from 41.84% to 90.44% and training loss decreased from 1.7071 to 0.2658. On the validation images, accuracy increased from 58.00% to 95.00% and loss decreased from 1.0902 to 0.1483.

B. Classification Performance

On the 100 evaluation images the model classified 95 correctly, an overall accuracy of 95% (95% Wilson interval 88.8–97.8%). The weighted precision, recall, and F1-score are 0.96, 0.95, and 0.95, and the macro averages are 0.94, 0.93, and 0.93. Table 4 gives the per-class results and Table 5 the confusion matrix. Damping off, healthy, and veinal mottle leaves were recognised perfectly (precision, recall, and F1-score all 1.00). One whitefly image was missed (recall 0.90) and predicted as leaf spot. Yellowish leaves had the lowest recall (0.60): three were predicted as leaf curl and one as leaf spot. These errors lowered the precision of leaf curl (0.77) and leaf spot (0.83), whereas the recall of both classes was 1.00.

Table 4. Per-class precision, recall, and F1-score on the evaluation set (N = 100)

Class	Precision	Recall	F1-score	Support
Damping off	1.00	1.00	1.00	10
Healthy	1.00	1.00	1.00	40
Leaf curl virus	0.77	1.00	0.87	10
Leaf spot	0.83	1.00	0.91	10
Veinal mottle virus	1.00	1.00	1.00	10
Whitefly	1.00	0.90	0.95	10
Yellowish	1.00	0.60	0.75	10
Macro average	0.94	0.93	0.93	100
Weighted average	**0.96**	**0.95**	**0.95**	**100**

Note: overall accuracy is 0.95. Per-class values follow the confusion matrix in Table 5.

Table 5. Confusion matrix on the evaluation set (rows: true class; columns: predicted class)

True class	DO	HE	LC	LS	VM	WF	YE
Damping off (DO)	10	0	0	0	0	0	0
Healthy (HE)	0	40	0	0	0	0	0
Leaf curl virus (LC)	0	0	10	0	0	0	0
Leaf spot (LS)	0	0	0	10	0	0	0
Veinal mottle virus (VM)	0	0	0	0	10	0	0
Whitefly (WF)	0	0	0	1	0	9	0
Yellowish (YE)	0	0	3	1	0	0	6

Note: off-diagonal entries are misclassifications; the total number of images is 100.

C. Web Application Interface

Figs. 2–4 show the deployed application (interface text in Indonesian). After the application is launched with Streamlit, it is served at a local and a network address and opened in a web browser. Fig. 2 shows the upload step. Once a file is chosen, the page lists its file name and size, which confirms that the image was loaded, and displays a preview so that the user can check that the intended leaf was selected before analysis. The button labelled “Analisis Gambar” (analyse image) below the preview starts the classification.

Fig. 2. Image upload page of the web application, showing the file name and size, the image preview, and the analysis button. The example image is a leaf-spot sample.

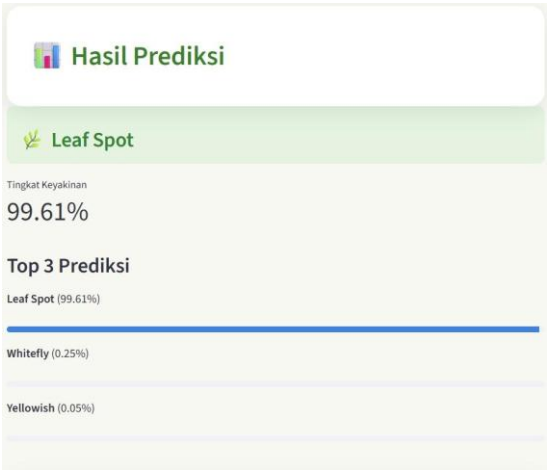

Fig. 3. Prediction page of the web application, showing the predicted disease (Leaf Spot), the confidence (99.61%), and the top-3 class probabilities.

The prediction page (Fig. 3) presents the disease predicted by the MobileNet model and the model's confidence. In the example, the image was classified as leaf spot with a confidence of 99.61%. Below the main result, the three classes with the highest probabilities are listed: leaf spot (99.61%), whitefly (0.25%), and yellowish (0.05%). This ranking shows how strongly the model prefers its first choice over the alternatives; here almost all of the probability is assigned to a single class. The example illustrates the interface and is not part of the quantitative evaluation.

Finally, the treatment page (Fig. 4) gives recommendations matched to the predicted disease. For leaf spot, the application advises removing leaves that have spots, avoiding watering directly onto the leaves, spacing the plants to improve air circulation, cleaning up diseased leaves that have fallen around the plants, and using a fungicide as recommended if the infestation is severe. These messages are generic cultural-control suggestions written for the application; they are not derived from a cited agronomic source and have not been validated by plant pathologists. The use of a fungicide in particular depends on the pathogen and on the product label, so it should follow the advice of an agricultural extension officer.

Fig. 4. Treatment recommendations displayed for the predicted disease (Leaf Spot).

V. DISCUSSION

The model separates most classes well, but the perfect scores for damping off, healthy, and veinal mottle leaves should be read with care. Each disease class has only ten evaluation images, so a single error changes its recall by ten percentage points, and the 95% interval for the overall accuracy (88.8–97.8%) is wide.

The errors are concentrated in the yellowish class. Yellowing is a general symptom that accompanies several conditions, including virus infections that also cause curling, so overlapping appearance is a plausible explanation for the confusion with leaf curl and leaf spot; this explanation was not tested here. Scarcity of training data does not explain it, because yellowish had 500 training images, as many as whitefly, leaf curl, and leaf spot. Label ambiguity or genuinely overlapping symptoms are therefore more likely causes, and a visual review of the misclassified images would be a useful next step.

At the tenth epoch the validation accuracy (95.00%) was higher than the training accuracy (90.44%). This can occur when augmentation and dropout are active only during training, and it suggests that the model had not overfitted after ten epochs. Training was stopped while the validation loss was still decreasing, so longer training with monitoring might improve the results further.

Table 1 places the result in context. The accuracy of 95% is higher than those reported for the pipelines based on hand-crafted features [3], [4] for the artificial neural network [5], and for the plain CNN of Agustin *et al.* [11]. It is lower than the roughly 99% reported for fine-tuned VGG16 [9] and MobileNetV2 [8], which used different class sets (five and six classes, respectively), whose dataset sizes were not stated in the sources, and whose evaluation protocols may differ, so the figures cannot be ranked directly. It is also lower than the 97.22% training accuracy reported for a MobileNet model [7], but that figure was obtained on training data from a very small dataset and is not comparable with a held-out evaluation. The present task also covers more classes (seven instead of three to five) and, in several cases, more images, which makes it harder in principle. A direct comparison would require the same data and the same evaluation protocol.

In practical terms, the backbone has about 3.2 million parameters, which should be light enough to be served from a modest machine and to be considered later for mobile deployment [12], [17], although model size, memory use, and latency were not measured here, so efficiency is inferred from the parameter count only. Delivering the model through Streamlit could let farmers or extension officers obtain an indicative classification and general first-aid messages from a photograph without installing software, although this has not been tested with users. From a methodological point of view, the results support partial fine-tuning of a pre-trained network as a workable strategy for small agricultural datasets.

Several limitations apply. First, the evaluation set is small (100 images) and class-imbalanced, and only a single training run was made, so no variability across runs can be reported. Second, the confusion matrix was computed with the data loader that was also used for validation monitoring, so the metrics should be confirmed on the separate test subset with repeated runs. Third, the training set is imbalanced (2,000 veinal mottle images against 168 for damping off), and the images come from a public dataset, so performance on field photographs with different lighting and backgrounds is unknown. Fourth, the application always outputs one of the seven classes and cannot recognise images that are not chili leaves or that show unseen diseases. Finally, MobileNet was not compared with other architectures, and the treatment recommendations have not yet been validated with agronomists.

VI. CONCLUSION

This study built and evaluated a web-based classifier for seven chili leaf conditions using a MobileNet fine-tuned on 4,688 images. On the 100-image evaluation set the model reached an accuracy of 95% and weighted precision, recall, and F1-score of 0.96, 0.95, and 0.95. Damping off, healthy, and veinal mottle leaves were classified without error, whereas yellowish leaves were the weakest class (recall 0.60) and were mostly confused with leaf curl. The model was deployed in a Streamlit application that returns the predicted disease, its confidence, the three most probable classes, and generic treatment recommendations. This demonstrates that a classification model can be made available through a web browser, but the results are preliminary: they come from curated public images and a small evaluation set, and they should not be taken as evidence of field performance.

Future work should confirm the results on the independent test subset with repeated runs, add field images taken under varied lighting, angles, and backgrounds, and rebalance the training data. The application should also learn to reject images that are not chili leaves or that show unseen diseases, and its recommendations should be validated by agronomists. Further steps include comparing MobileNet with other lightweight architectures, measuring response time, and developing a mobile version of the application.

AUTHOR CONTRIBUTIONS

Ulfa Ma'idatul Hasanah: Conceptualization, Methodology, Software, Investigation, Writing – original draft. Mochammad Faid and M. Noer Fadli Hidayat: Supervision, Validation, Writing – review & editing.

DATA AVAILABILITY

The dataset used in this study is publicly available on Kaggle [13]. The trained model and the application source code are available from the corresponding author on reasonable request.

DECLARATION OF COMPETING INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

[1] 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.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[3] P. C. Siswipraptini, A. Haris, and W. N. Sari, "Klasifikasi citra penyakit daun cabai menggunakan algoritma learning vector quantization [Image classification of chili leaf diseases using the learning vector quantization algorithm]," *Faktor Exacta*, vol. 16, no. 2, pp. 119–125, 2023, doi: 10.30998/faktorexacta.v16i2.15900.

[4] N. Nurmadinah, F. Wajidi, and N. Arifin, "Leaf disease detection in chili plants using GLCM and HSV combination with SVM classification," *Insect (Informatics and Security): Jurnal Teknik Informatika*, vol. 11, no. 2, pp. 178–189, 2025, doi: 10.33506/insect.v11i2.4820.

[5] N. P. Arafa, S. R. Basri, R. Ratnasari, and R. A. Saputra, "Klasifikasi penyakit pada daun tanaman cabai dengan pendekatan artificial neural network (ANN) [Classification of diseases on chili plant leaves using an artificial neural network approach]," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 6, pp. 12865–12871, 2024, doi: 10.36040/jati.v8i6.12140.

[6] L. P. Putri, M. R. Muttaqin, and Y. R. Ramadhan, "Classification of chilli leaf disease using MobileNet model," *TeknoIS: Jurnal Ilmiah Teknologi Informasi dan Sains*, vol. 15, no. 1, pp. 74–86, 2025, doi: 10.36350/jbs.v15i1.297.

[7] M. M. Munir, P. Kasih, and A. Sanjaya, "Penerapan arsitektur MobileNet dalam CNN pada klasifikasi penyakit daun cabai [Application of the MobileNet architecture in a CNN for chili leaf disease classification]," in *Prosiding SEMNAS INOTEK (Seminar Nasional Inovasi Teknologi)*, vol. 8, no. 3, 2024, pp. 1626–1632, doi: 10.29407/5m496763.

[8] R. Jajila, N. A. Ramdhan, P. Wahyuningsih, and B. Irawan, "Optimasi MobileNetV2 dengan transfer learning untuk klasifikasi penyakit daun cabai [MobileNetV2 optimization with transfer learning for chili leaf disease classification]," *Infotek: Jurnal Informatika dan Teknologi*, vol. 9, no. 1, pp. 296–307, 2026, doi: 10.29408/jit.v9i1.33812.

[9] A. S. Mashuri, A. Sunyoto, and Kusnawi, "Klasifikasi penyakit pada daun cabai menggunakan arsitektur VGG16 [Classification of chili leaf diseases using the VGG16 architecture]," *Journal of Electrical Engineering and Computer (JEECOM)*, vol. 6, no. 2, pp. 305–313, 2024, doi: 10.33650/jeecom.v6i2.9116.

[10] U. P. Firdaus and Y. Litanianda, "Deteksi penyakit pada daun cabai menggunakan metode convolutional neural network [Chili leaf disease detection using a convolutional neural network]," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 11, no. 4, 2024, doi: 10.35957/jatisi.v11i4.8310.

[11] A. Agustin, M. A. Bianto, and H. Ardiansyah, "Perbandingan klasifikasi daun cabai dengan metode CNN dan SVM untuk deteksi penyakit [Comparison of chili leaf classification using CNN and SVM methods for disease detection]," *Jurnal Pengembangan Teknologi Informasi dan Komunikasi (JUPTIK)*, vol. 4, no. 1, pp. 187–196, 2026, doi: 10.52060/juptik.v4i1.4403.

[12] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[13] R. Bandara. "Chilli plant diseases dataset." Kaggle. Accessed: Oct. 21, 2026. [Online]. Available: https://www.kaggle.com/datasets/ravindubandara3002/chilli-plant-diseases-dataset

[14] Streamlit. "Streamlit documentation." Snowflake Inc. Accessed: Oct. 21, 2026. [Online]. Available: https://docs.streamlit.io

[15] 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*.

[16] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, "PlantDoc: A dataset for visual plant disease detection," in *Proc. 7th ACM IKDD CoDS and 25th COMAD*, Hyderabad, India, 2020, *arXiv:1911.10317*.

[17] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, 2018, pp. 4510–4520, doi: 10.1109/CVPR.2018.00474.

[18] A. Howard *et al.*, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Seoul, South Korea, 2019, pp. 1314–1324, doi: 10.1109/ICCV.2019.00140.

[19] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, vol. 97, 2019, pp. 6105–6114.

[20] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 27, 2014, pp. 3320–3328.

[21] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.

[22] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*.

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

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, 2015, *arXiv:1412.6980*.