Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control
Journal homepage: http://kinetik.umm.ac.id
ISSN: 2503-2267
Vol. 7, No. 2, May, Pp. 187-196

Convolutional Neural Network (CNN) models for crop diseases classification

Deni Sutaji*[1,2], Harunur Rosyid[1]
Informatics Department, Universitas Muhammadiyah Gresik, Gresik, Indonesia[*1]
Computer Science, Informatics Institute, Gazi University, Ankara, Turkey[2]

Article Info
Keywords:
CNN Models, Crop Disease, Classification, Unfreeze Layers

Article history:
Received: April 29, 2022
Accepted: May 11, 2022
Published: May 31, 2022

Cite:
D. Sutaji and H. Rosyid, "Convolutional Neural Network (CNN) Models for Crop Diseases Classification", *KINETIK*, vol. 7, no. 2, May. 2022.
https://doi.org/10.22219/kinetik.v7i2.1443

*Corresponding author.
Deni Sutaji
E-mail address:
sutaji.deni@umg.ac.id

Abstract
Crop diseases have a significant impact on agricultural production. As a result, early diagnosis of crop diseases is critical. Deep learning approaches are now promising to improve disease detection. Convolutional Neural Network (CNN) models can detect crop disease using images with automatic feature extraction. This study proposes crop disease classification considering ten pre-trained CNN models. Fine-tuning for each model was conducted on the Plant Village dataset. The experimental results show that fine-tuning improves the model's performance with an average accuracy of 8.85%. The best CNN model was DenseNet121, with 94.48% and 98.97% accuracy for freezing all layers and unfreezing last block convolution layers. Moreover, fine-tuning produces less time-consuming with an average of 2.20 hours. VGG19 is the less time-consuming reduction by 8 hours. On the other hand, MobileNetV2 is the second-best performance model with less time-consuming than DenseNet121 and produces fewer parameters, which is affordable for embedding it to mobile devices.

1. Introduction

Crop diseases are one of the leading causes of the decline in agricultural production. In recent years, various crop diseases have increased and affected the agricultural, economic, and health sectors. The disease generally generates noticeable marks or lesions on the crop leaves. Therefore, it is crucial to detect the presence of the disease at an early stage. However, traditional approaches are commonly used by most farms and plantations to recognize the disease. Consequently, it is time-consuming and leads to high costs and misdetection [1].

Hence, researchers have designed the Convolutional Neural Network (CNN) models as a computer-aided disease diagnosis. CNN models can overcome the problem of object recognition, and classification has significantly improved in the last decade [2], [3]. In recent years, CNN models have emerged as the most extensively used model in crop disease detection [3]–[10]. The CNN model can be applied from scratch or by using pre-trained designs. Promisingly, pre-trained CNN models outperformed the scratch designs. AlexNet was the first proposed pre-trained CNN model by [11] on the 1.2 million high-resolution images within the 1.000 different classes in the ImageNet LSVRC-2010 contest. Furthermore, many CNN models were offered in the next contest ImageNet LSVRC.

Recently, several pre-trained CNN architectures have been suggested, for example, GoogleNet [12], VGGNet [13], ResNet [14], InceptionV3 [15], DenseNet [16], Xception [17], MobileNet [18], InceptionResNetV2 [19], NasNet [20], and EfficientNet [21]. In prior research, researchers used two methods in crop disease recognition using CNN models [22]. Some use transfer learning to fine-tune well-known designs. Others use well-known architectures, such as modifying the convolutional layers, changing the order of layers, adding attention layers before the classifier layer, and ensembling more than two pre-trained CNN models. They aim to improve the model performance.

The most considered dataset is the Plant Village dataset. In this dataset, [23] corresponded with the results of AlexNet and GoogLeNet, in detecting 26 crop diseases in laboratory leave images. They employed transfer learning to increase categorization performance and achieve an accuracy of 99.35%. CaffeNet was proposed by [24] on 15 class leaf images and achieved an accuracy of 96.0%. Cucumber leaves were considered by [25]–[27] using Deep Convolution Neural Network (Scratch designs), modified Deep Convolution Neural Network, and combined DeepLabV3+ and U-Net, respectively. Moreover, the detection of citrus diseases conducted by [22], [28], [29] used modified MobileNetV2 and an Ensemble AlexNet, VGG16, ResNet50, and InceptionResNetV2. Both cucumber and citrus crops achieved an accuracy of over 98%.

In addition to the entire categories of the Plant Village dataset, researchers considered multiclass classification with 38 classes proposed by [30]–[34]. Pre-trained MobileNet, EfficientNet, DensNet, and ensemble MobileNet with DenseNets were applied. All suggested model performances achieved an accuracy of over 92%, and 100% for the ensemble model was proposed by [34].

Other datasets such as iBean, Citrus, Rice, and Turk-Plants were also used to detect crop diseases. Several pre-trained models were used, such as modified and ensembled by researchers, such as [35]–[40]. The model's average performance is above 99%.

However, various datasets have been used in the literature, making it difficult to compare directly. Several detection systems work only on two classes, while the others detect crop diseases in over 38 classes. Considering this term, we suggested investigating the classification of crop disease by using ten pre-trained CNN algorithms to consider the entire Plant Village dataset. This dataset consists of 54.305 RGB leaf images with 256 x 256 pixels. The dataset subsists of 38 classes from 14 crop species: strawberries, apples, grapes, cherries, corn, oranges, blueberries, potatoes, soybeans, pumpkins, peppers, raspberries, peaches, and tomatoes.

Another problematic aspect of comparing which CNN models can perform better is the various configurations and modifications of the pre-trained CNN models to detect crop disease. As a simple investigation to compare, we proposed regular and unfroze the final convolutional blocks aforementioned pre-trained CNN models. It aims to generate high-dimensional extracted features to provide a more helpful characteristic of an object. Our main contributions to this research are:

1. Comparing the performance of 10 pre-trained CNN models to detect crop disease.
2. Improving the performance of 10 pre-trained CNN models by unfroze the final convolution blocks.

This research is organized as follows: Section 1 introduces crop diseases, state-of-the-art methods for classifying crop diseases, explains the problem of statements, and gives solutions as a contribution to the research. Section 2 discusses research methods. Section 3 presents the experimental results and performance comparison. Eventually, Section 4 explains the conclusion of this study.

2. Research Method

2.1 Method

This study considered 10 pre-trained CNN architectures, such as AlexNet, GoogleNet, VGG16, VGG19, ResNet50V2, InceptionV3, Xception, MobileNetV2, DenseNet121, and InceptionResnetV2 to classify crop diseases. These CNN architectures were developed with a transfer learning approach and implemented fine-tuning by unfreezing the last block of convolutional layers. It aims to enhance the high-dimensional extracted features. Updating the weight parameter impacts capture more details of high-dimensional extracted features. Figure 1 illustrates our research method, and Figure 2 describes our suggested transfer learning and fine-tuning design.

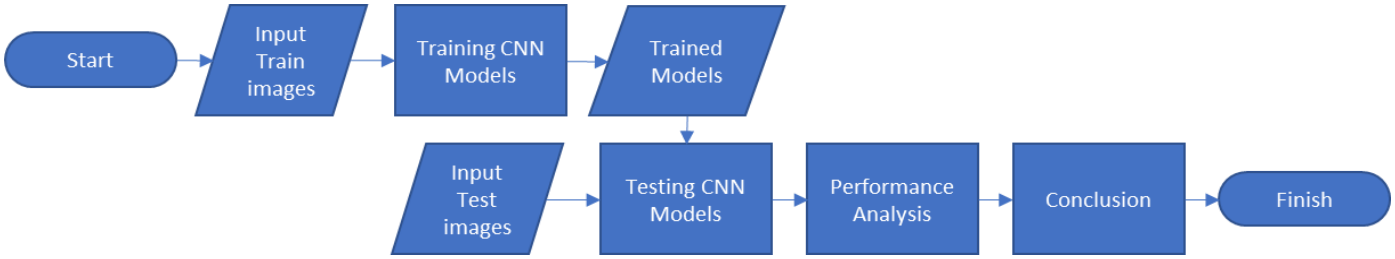

Figure 1. Our Research Method

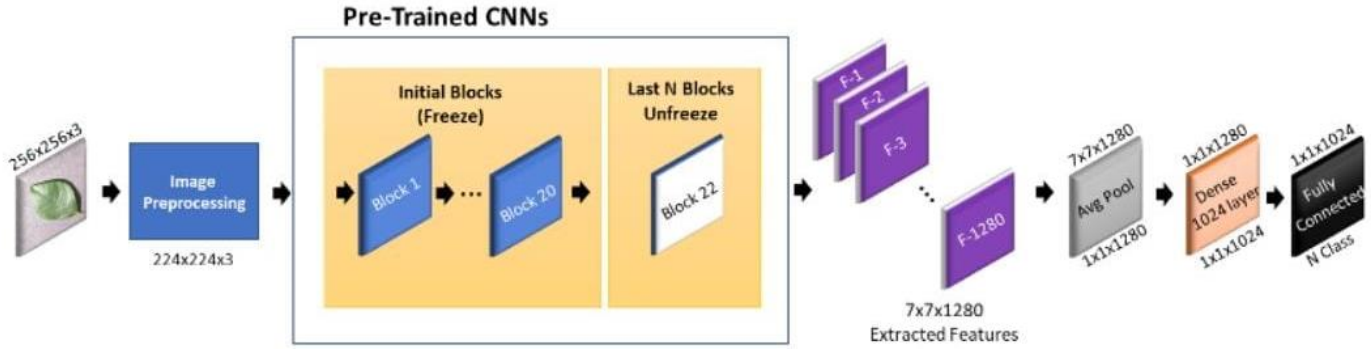

Figure 2. Our Proposed Transfer Learning and Fine-Tuning CNNs Design

2.2 Dataset

Plant Village dataset consists of 38 classes and includes 54,305 images with 256 x 256 size in the RGB channel. The dataset consists of 14 plant species: apples, blueberries, cherries, corn, grapes, oranges, peppers, raspberries, potatoes, pumpkins, peaches, raspberries, soybeans, strawberries, and tomatoes. The diseases from the 14 plant

species are distributed as 17 fungal diseases, 4 bacterial diseases, 2 fungal diseases, 2 viral diseases, and the last disease caused by mites has 1 class. Meanwhile, there are 12 plant species in the healthy category of plant leaves [23]. The considered dataset was collected from https://www.kaggle.com/abdallahalidev/plantvillage-dataset. The example of each class image is shown in Figure 3.

We pre-processed the collected dataset in four steps. The first step is resizing all images into 224 x 224 pixels. The second step is to split and distribute the dataset into a training set, a validation set, and a test set with a composition of 70%, 10%, and 20%, respectively. The detail of distribution is described in Figure 3. The third step is enforced image normalization, dividing the intensity value of each pixel by 255. The reason is to reduce computational costs because the intensity value of each pixel is between 0 and 1 [41].

Furthermore, the last step is image augmentation, completing the image in the data series more varied since the acquired image is subject to different conditions (angle, illumination, and background). Image augmentation is only performed on the training data because this model can accept different conditions of the leaf image under test. This step's configuration includes image rotation with a rotation angle of 30 degrees, zooming image with a percentage of 0.3, cropping with width_shift_range, height_shift_range, shear_range, and zoom_range settings with values of 0.3. The last configuration is horizontal_flip with the setting True.

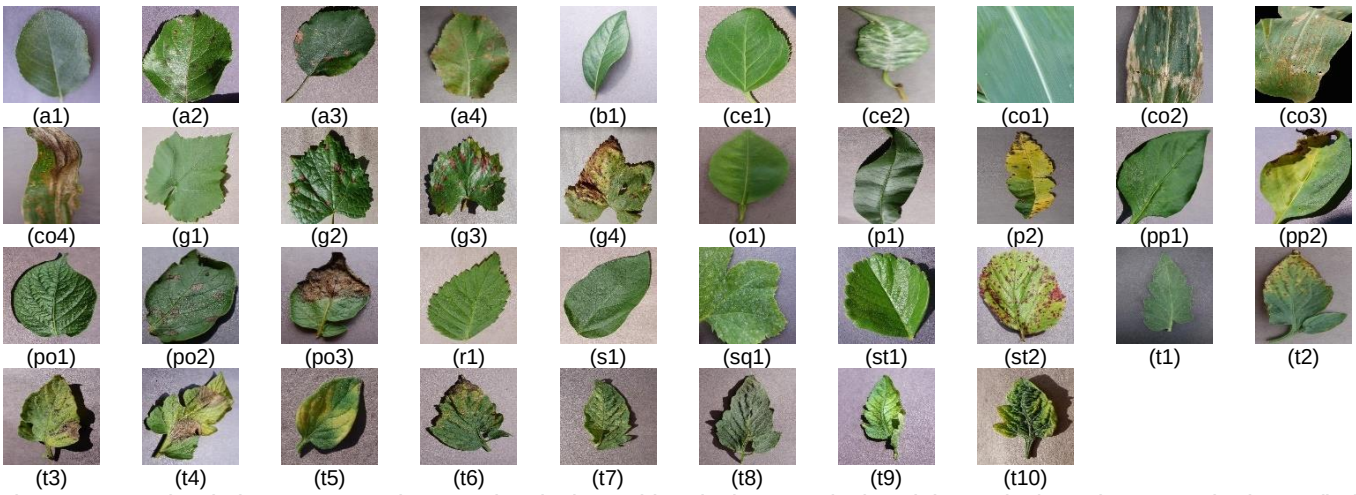

Figure 3. Randomly image examples: Apples (a1) Healthy, (a2) Scap, (a3) Balck rot, (a4) Cedar rust; Blueberry (b1) Healthy; Cherry (ce1) Healthy, (ce2) Powdery mildew; Corn, (co1) Healthy, (co2) Cercospora, (co3) Rust, (co4) Northern blight; Grape (g1) Healthy, (g2) Blak rot, (g3) Black measles; Orange (o1) Healthy; Peach (p1) Healthy, (p2) Bacterial spot; Pepper (pp1) Healthy, (pp2) Bacterial spot; Potato (po1) Healthy, (po2) Early blight; Raspberry (r1) Healthy; Soybean (s1) Healthy; Squash (sq1) Powdery mildew; Strawberry (st1) Scorch; Tomato (t1) Healthy, (t2) Bacterial spot, (t3) Early blight, (t4) Late blight, (t5) Mold, (t6) Septoria spot, (t7) Spider mites, (t8) Target spot, (t9) Mosaic virus, (t10) Yellow curl virus

2.3 Software Tools for Experimental

Our experimental research was conducted on the Google Colab Pro with the following specifications: Intel(R) Xeon(R) 2 CPU @ 2.20GHz Processor, 13GB RAM, and GPU Tesla P100-PCIE-16GB. This platform runs Python and Jupyter notebooks with Tensorflow and Keras libraries as a backend for VGG16, VGG19, ResNet50, InceptionV3, Xception, MobileNetV2, and InceptionResnetV2 models. Moreover, we used the PyTorch framework to train and test pre-trained AlexNet and GoogleNet models.

2.4 Test Scenarios

In this research, we considered two test scenarios. We initially considered regular transfer learning for ten pre-trained CNN architectures without fine-tuning. Then, transfer learning and fine-tuning as a second scenario. We were unfreezing the final convolutional block layers to enhance the high-dimensional extracted features.

This research considers Accuracy (Acc), Sensitivity (Sen), Specificity (Spe), and Precision (Prec) metrics to calculate the performance of tested CNN models. The correctly classified positive ratio of all True Positives is distinguished as Sensitivity. The correctly classified negative ratio of all False Positives is marked as Specificity. Then, Accuracy directs to the ratio of correctly classified samples to all samples. Precision guides to the correctly classified positive ratio of all accurate recognitions.

3. Results and Discussion

Determining the hyperparameters for model training is critical work. We accept Adam as the optimizer, including a learning rate of 0.0001 as the best value. We initially tried different learning rate values starting from 0.1 to 0.0001 in 0.1 additions. We also tested other optimizers, such as SGD and RMSprop. Therefore, Adam optimizer produces excellent training, validation, and testing performances. Another parameter is an epoch, which we set at 100 and consider to have an early stop value of 10. Training can be more efficient, and the process does not waste time in 100 epochs. We also employed cross-entropy to track each epoch's decreasing error rate and accuracy. Finally, a batch size of 25, 18, and 18 for training, validation, and testing data were presented in Table 1.

Table 1. Dataset Split Distribution (70:10:20) of Training, Validation, and Testing

Crop Species	Total	Train	Val	Test	Crop Species	Total	Train	Val	Test
Apple Scab	630	441	63	126	Pepper Healthy	1478	1035	148	295
Apple Black root	621	435	62	124	Potato Ealy Blight	1000	700	100	200
Apple Cedar	275	192	28	55	Potato Late Blight	1000	700	100	200
Apple Healthy	1645	1151	165	329	Potato Healthy	152	106	15	31
Blueberry	1502	1051	150	301	Raspberry	371	260	37	74
Cherry Powdery	1052	736	105	211	Soybean	5090	3563	509	1018
Cherry Healthy	854	598	85	171	Squash Powdery Mildew	1835	1285	183	367
Corn Cercospora	513	359	51	103	Strawberry scorch	1109	776	111	222
Corn rust	1192	834	119	239	Strawberry Healthy	456	319	46	91
Corn Northern	985	690	98	197	Tomato Bacterial spot	2127	1489	213	425
Corn Healthy	1162	813	116	233	Tomato Early Blight	1000	700	100	200
Grape Black rot	1180	826	118	236	Tomato Late Blight	1909	1336	191	382
Grape Esca	1383	968	138	277	Tomato Leaf Mold	952	666	95	191
Grape Blight	1076	753	108	215	Tomato Septoria spot	1771	1240	177	354
Grape Healthy	423	296	42	85	Tomato Spider mites	1676	1173	168	335
Orange	5507	3855	551	110	Tomato Target Spot	1404	983	140	281
Peach Bacterial	2297	1608	230	459	Tomato Yellow curs virus	5357	3750	536	1071
Peach Healthy	360	252	36	72	Tomato mosaic virus	373	261	37	75
Pepper Bacterial	997	698	100	199	Tomato Healthy	1591	1114	159	318
					Total	54305	38012	5430	10863

3.1 Experimental Results Scenario 1

The CNN model weight parameters were unchanged in this scenario because layers were configured as trainable equal false. We added one layer before the classifier layer, GlobalAveragePooling2D. It aims to provide features in one channel, as the classifier layer needs. The experimental results show that GoogleNet, DenseNet121, ResNet50, and MobileNetV2 achieved better accuracy than the seven pre-trained CNN models. It is similar to previous studies done by [23], [38], [42], and [29], respectively, that model performances were outperformed.

On the other hand, the less time-consuming was provided by the AlexNet model. Moreover, MobileNetV2, both model performance and time-consuming outperformed. Table 2 presents the model performances in the first scenario design.

Table 2. Model Performance for the First Scenario

No	Model	Acc			Prec	Rec	F1	Computation Time	Epoch Stop
		Train	Val	Test	Test data				
1	AlexNet	95.63	89.47	89.47	91.42	89.47	89.54	3:10:55.722126	26
2	GoogleNet	95.53	96.32	96.22	96.55	96.22	96.23	10:07:40.969602	22
3	VGG16	81.04	86.00	84.18	84.91	84.18	83.95	8:27:10.904154	85
4	VGG19	78.42	84.33	81.80	81.81	81.80	81.25	12:49:57.617564	97
5	InceptionV3	88.39	89.28	88.14	88.51	88.14	87.96	4:36:43.910856	37
6	ResNet50V2	92.67	94.62	93.26	93.56	93.26	93.21	4:07:58.418349	33
7	InceptionResnetV2	89.32	91.58	90.64	90.85	90.64	90.46	5:51:53.510285	39

8	DenseNet121	93.40	94.99	94.48	94.62	94.48	94.45	4:55:42.577147	39
9	MobileNetV2	93.08	93.17	93.11	93.35	93.11	93.11	4:03:49.903792	44
10	Xception	91.81	92.30	84.57	86.45	84.57	84.65	7:44:02.629347	47

3.2 Experimental Results Scenario 2

As mentioned in the previous subsection, this scenario considered unfreezing the last convolution blocks of the CNN models. The model performances significantly improved. Each architecture achieved accuracy above 97%. Moreover, the time-consuming of each model decreasing about 50%. Table 3 describes the model performances of the second scenario design.

Table 3. Model Performance for the Second Scenario

No	Model	Acc			Prec	Rec	F1	Computation Time	Epoch Stop
		Train	Val	Test	Test data				
1	AlexNet	99.29	98.78	98.12	98.30	98.25	98.25	1:29:02.212469	31
2	GoogleNet	99.28	99.02	98.34	98.34	98.42	98.42	1:32:59.543974	27
3	VGG16	98.11	98.51	98.27	98.28	98.27	98.26	2:15:17.964204	16
4	VGG19	96.64	97.29	97.19	97.31	97.19	97.19	4:13:36.753806	27
5	InceptionV3	97.73	98.38	97.30	97.47	97.30	97.31	2:01:08.610317	21
6	ResNet50V2	98.22	98.91	98.43	98.45	98.43	98.43	2:19:37.840994	18
7	InceptionResnetV2	98.70	99.04	98.76	98.78	98.76	98.75	3:13:28.997381	22
8	DenseNet121	99.13	99.15	98.97	99.00	98.97	98.97	3:38:19.336488	27
9	MobileNetV2	98.74	99.10	98.95	98.96	98.95	98.94	2:07:51.701822	22
10	Xception	99.53	99.45	97.58	97.68	97.58	97.58	6:12:13.524522	25

3.3 Research Analysis and Limitations

Implementing fine-tuning by unfreezing the last convolution block layers significantly affects the system's performance. As shown in Table 4, the accuracy value for each CNN model has improved with 4.49, 15.39, and 8.85 values of lower, higher, and average, respectively. The model can perform better when the weights of high-dimensional features are updated.

Table 4. Model Performances Comparison

No	Models	Test Accuracy %			Computational Time		
		Scenario 1	Scenario 2	Improve	Scenario 1	Scenario 2	Gap (hour)
1	AlexNet	89.18	98.12	8.94	01:03:50	01:29:02	0.00
2	GoogleNet	94.00	98.34	4.34	02:02:11	01:33:00	0.00
3	VGG16	84.18	98.27	14.09	08:27:11	02:15:18	6.00
4	VGG19	81.80	97.19	15.39	12:49:58	04:13:37	8.00
5	InceptionV3	88.14	97.30	9.16	04:36:44	02:01:09	2.00
6	ResNet50V2	93.26	98.43	5.17	04:07:58	02:19:38	1.00
7	InceptionResnetV2	90.64	98.76	8.12	05:51:54	03:13:29	2.00
8	DenseNet121	94.48	98.97	4.49	04:55:43	03:38:19	1.00
9	MobileNetV2	93.11	98.95	5.84	04:03:50	02:07:52	1.00
10	Xception	84.57	97.58	13.01	07:44:03	06:12:14	1.00
	Average	89.34	98.19	8.85	-	-	2.20

The highest accuracy obtained by the DenseNet121 model of 98.97% is quite promising for crop disease detection. It will significantly assist farmers or agricultural experts in detecting crop diseases.

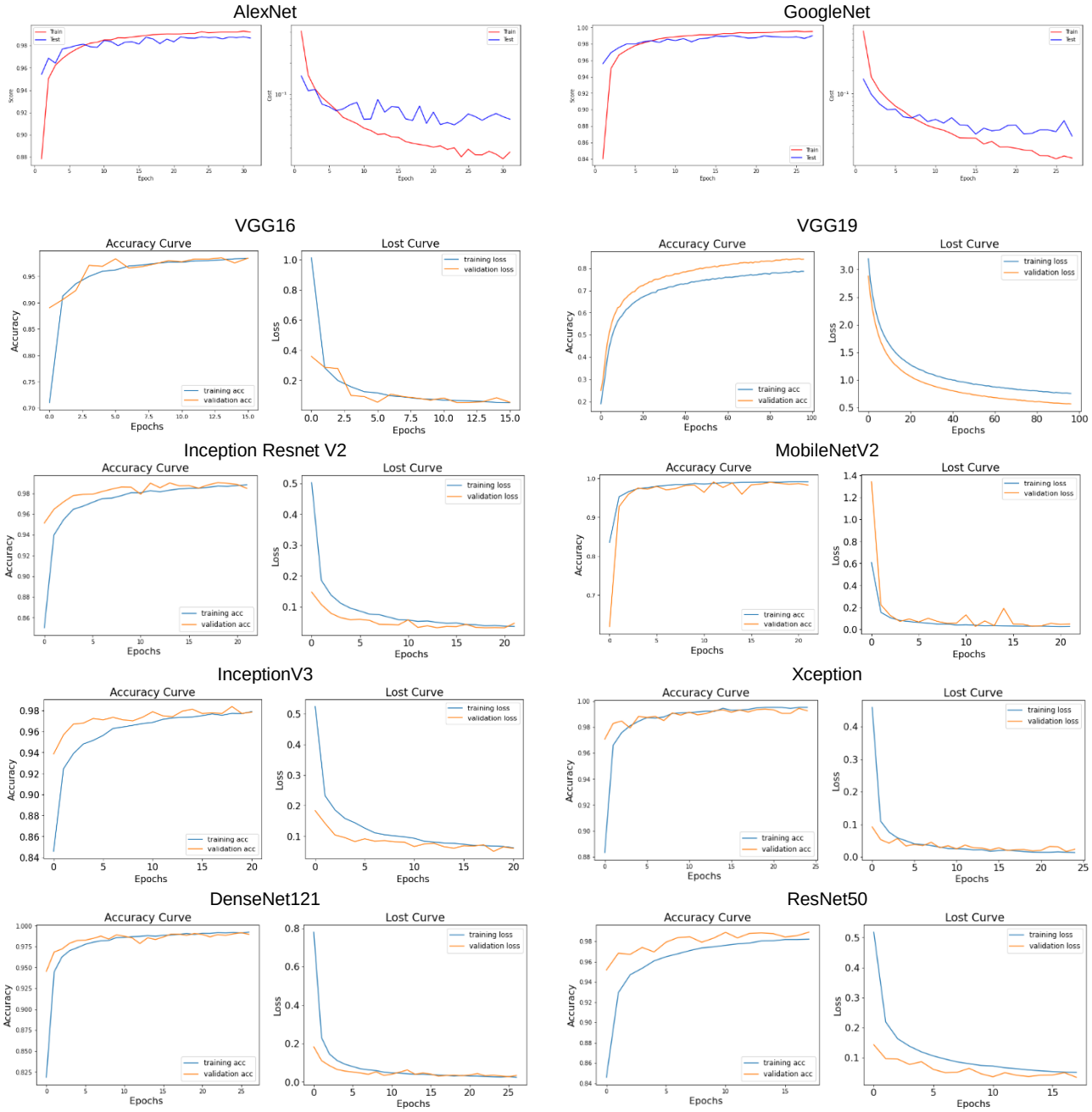

Figure 4. Training Accuracy and Loss Curve of the Second Scenario Design

In addition to improving accuracy, unfreezing the final block of the convolution layer speeds up the training process. The average reduction in time consumption in each model is almost 50%, or 2.20 hours, with the most significant reduction being 8 hours.

However, our research did not ultimately outperform the earlier research models. We accomplished just simple fine-tuning by unfreezing the final convolution block layers. For example, the CNN AlexNet and GoogleNet models proposed by [23] get an accuracy value of 99.35%, while our proposed model is only 98.34%. If we refer to the model performance graph in Figure 4, our proposed model is overfitting, and the VGG19 model followed this condition. While the DenseNet121 model, which is our best model, does not outperform the model proposed by [31]. Our DenseNet121 model only gets 98.97% accuracy and 99.75% for [31] model.

On the other hand, our model outperforms the model proposed by [43], which was only implemented on citrus crops. Their models get 97%, 96.2%, and 97% accuracy with the VGG16, VGG19, and InceptionV3 models. Meanwhile, our proposed model achieved an accuracy of 98.27%, 97.19%, and 97.30%. This positive result also outperforms the

Table 5. Performance Comparison with Prior Studies

Authors	Datasets	Model Proposed	Accuracy
Mohanty [23]	Entire Plant Village	AlexNet, GoogleNet	99.35%
Sladojevic [24]	15 class custom leaf images	CaffeNet	96.30%
Gandhi [30]	Entire Plant Village	MobileNetV1 and GAN	92%
Ma [25]	Cucumber Plant Village	Deep Convolution Neural Network (Scratch)	93.40%
Too [31]	Entire Plant Village	DenseNets	99.75%
Geetharamani [32]	Entire Plant Village	Deep Convolution Neural Network (Scratch)	96.46%
Barman [29]	Citrus	Modified MobileNetV2 SSCNN	92% MobileNetV2, 99% SSCNN
Atila [44]	Entire Plant Village	EfficientNet (B0 – B7)	99.91% on B5, 99.97% on B4.
I. Ahmad [35]	Custom Tomato Leaf (lab & field)	Inception V3	99.60% on lab data 93.70% on field data
Chen [36]	Apple, Grape, Potato (Plant Village), Custom Maize and Rice	MobileNet-Beta	99.94% on plant village 99.85% on custom
Chen [37]	Plant Village Maize, Custom rice and maize	INC-VGG	92% on plant village 80.38% custom
Tiwari [45]	User-defined (3 specieses of Plant village), iBean , Citrus, and Rice	DenseNet with 5-fold Cross-Validation	99.19%
Gajjar [39]	Custom Plant Village (Apple, Corn Potato, Tomato) and field dataset	CNN adoption from Inception Design (Scratch)	96.88%
Sujatha [43]	Private Citrus leaf	VGG16, VGG19, Inception V3	97%, 96.2%, 97%
Chen [40]	Custom Plant Village and local dataset	Modified MobileNetV2 + Soft Attention	99.71% on Plant Village 99.13% on local
Khanramaki [22]	Private Citrus leaf	Ensemble AlexNet, VGG16, ResNet50, and InceptionResNetV2	99.04% on augmented 98.64%
Wang [27]	Cucumber leaf	DeeplabV3+MobileNet	92.85%
Turkoglu [1]	Turk-Plants	Ensemble six CNN models (feature extractor) and SVM (classifier).	97.56% MV Model, 96,83% EF
Vallabhajosyula [34]	Entire Plant Village	Ensemble weighted MobileNet (0.3451), Densenet121 (0.6541), DenseNet201 (0.0009)	100%
Our Proposed	Entire Plant Village	DenseNet121 (scenario 2)	98.97%
		MobileNetV2 (scenario 2)	98.95%

Model proposed by [37], which obtained an accuracy of 92% on rice plants. Likewise, the InceptionResNetV2 model we propose has an accuracy of 98.76%, higher than the model proposed by [22] of 98.64%. This positive result is followed by the Xception model we propose to get almost the same results as the Xception model proposed by [46], which are 97.58% and 97.60%, respectively. Unfortunately, our MobileNetV2 model did not follow these positive results. Our MobileNetV2 model is inferior to some of the previously proposed models, which applied modification to an ensemble of MobileNet proposed by [29], [30], [36] and obtains more than 99% accuracy. Their models extract features better than our model. Moreover, their models added several convolutional layers before the classification layer. Our

MobileNetV2 model provides 98.95% of accuracy in the second scenario. Table 5 describes the performance comparison of our model with earlier studies.

In completing the discussion of this section, the fine-tuning model that we propose is relatively simple compared to the model proposed by previous studies. Some of them applied architectural modifications, ensembles, and other advance works. Fortunately, our model is not entirely inferior to the previously proposed model. Some models like VGG16, VGG19, InceptionV3, and InceptionResNetV2 outperform models by [43]. The best models we proposeDenseNet121 and MobileNetV2, promise to be improved in terms of accuracy and time consumption. Moreover, MobileNetV2 has fewer parameters than DenseNet121, which means more affordable to embed in the mobile device for live detection.

Although the proposed models in this article have achieved good results for the entire Plant Village dataset, there are several works to improve the performance:

1. Adding several convolution layers before the classifier layer.
2. Model modifications by unfreezing the initial convolution layers to update low-dimensional weight or ensembling models.

4. Conclusion

Fine-tuning by unfreezing the final convolution layer blocks can significantly improve the performance of pre-trained 10 CNN models. From the results of our experiments, the best performance was obtained by DenseNet121 and MobileNetV2 models of 98.97% and 98.95%, respectively. This result increases the average performance of CNN models by 8.85%. Meanwhile, there was a drastic decrease of 50% for each model in time consumption, or 2.20 hours on average. Further research can be developed with architectural modifications by adding several convolution layers before the classifier layer, unfreezing the initial convolution layers to update low-dimensional weight, and ensembling models.

Acknowledgment

Thank you to Lembaga Penelitian dan Pengabdian Masyarakat. (LPPM) University of Muhammadiyah Gresik (UMG). This reseach article supported by LPM UMG Nomor: 066/MoU.In/II.3.UMG/DPPM/F/2021 for Internal Research Funded.

References

[1] M. Turkoglu, B. Yanikoğlu, and D. Hanbay, “PlantDiseaseNet: convolutional neural network ensemble for plant disease and pest detection,” *Signal, Image and Video Processing*, 2021. https://doi.org/10.1007/s11760-021-01909-2
[2] N. Ahmad, S. Asghar, and S. A. Gillani, “Transfer learning-assisted multi-resolution breast cancer histopathological images classification,” *Visual Computer*, 2021. https://doi.org/10.1007/s00371-021-02153-y
[3] S. Kaur, S. Pandey, and S. Goel, “Plants Disease Identification and Classification Through Leaf Images: A Survey,” *Archives of Computational Methods in Engineering*, vol. 26, no. 2, pp. 507–530, Apr. 2019. https://doi.org/10.1007/s11831-018-9255-6
[4] S. Sachar and A. Kumar, “Survey of feature extraction and classification techniques to identify plant through leaves,” *Expert Systems with Applications*, vol. 167. Elsevier Ltd, Apr. 01, 2021. https://doi.org/10.1016/j.eswa.2020.114181
[5] V. Singh, N. Sharma, and S. Singh, “A review of imaging techniques for plant disease detection,” *Artificial Intelligence in Agriculture*, vol. 4, pp. 229–242, 2020. https://doi.org/10.1016/j.aiia.2020.10.002
[6] J. Lu, L. Tan, and H. Jiang, “Review on convolutional neural network (CNN) applied to plant leaf disease classification,” *Agriculture (Switzerland)*, vol. 11, no. 8. MDPI AG, Aug. 01, 2021. https://doi.org/10.3390/agriculture11080707
[7] A. Abade, P. A. Ferreira, and F. de Barros Vidal, “Plant diseases recognition on images using convolutional neural networks: A systematic review,” *Computers and Electronics in Agriculture*, vol. 185. Elsevier B.V., Jun. 01, 2021. https://doi.org/10.1016/j.compag.2021.106125
[8] R. Manavalan, “Automatic identification of diseases in grains crops through computational approaches: A review,” *Computers and Electronics in Agriculture*, vol. 178. Elsevier B.V., Nov. 01, 2020. https://doi.org/10.1016/j.compag.2020.105802
[9] Z. Iqbal, M. A. Khan, M. Sharif, J. H. Shah, M. H. ur Rehman, and K. Javed, “An automated detection and classification of citrus plant diseases using image processing techniques: A review,” *Computers and Electronics in Agriculture*, vol. 153. Elsevier B.V., pp. 12–32, Oct. 01, 2018. https://doi.org/10.1016/j.compag.2018.07.032
[10] L. C. Ngugi, M. Abelwahab, and M. Abo-Zahhad, “Recent advances in image processing techniques for automated leaf pest and disease recognition – A review,” *Information Processing in Agriculture*, vol. 8, no. 1. China Agricultural University, pp. 27–51, Mar. 01, 2021. https://doi.org/10.1016/j.inpa.2020.04.004
[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in Advances in Neural Information Processing Systems 25, 2012, pp. 1–9.
[12] C. Szegedy et al., “Going Deeper with Convolutions,” Sep. 2014.
[13] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” Sep. 2014.
[14] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 2015.
[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” Dec. 2015.
[16] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” Aug. 2016.
[17] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” 2017.
[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” Jan. 2018.
[19] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” Feb. 2016.
[20] B. Zoph, V. Vasudevan, J. Shlens, and Q. v. Le, “Learning Transferable Architectures for Scalable Image Recognition,” Jul. 2017.
[21] M. Tan and Q. v Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” 2019.

[22] M. Khanramaki, E. Askari Asli-Ardeh, and E. Kozegar, "Citrus pests classification using an ensemble of deep learning models," *Computers and Electronics in Agriculture*, vol. 186, Jul. 202. https://doi.org/10.1016/j.compag.2021.106192
[23] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, no. September, Sep. 2016. https://doi.org/10.3389/fpls.2016.01419
[24] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification," *Computational Intelligence and Neuroscience*, vol. 2016, 2016. https://doi.org/10.1155/2016/3289801
[25] J. Ma, K. Du, F. Zheng, L. Zhang, Z. Gong, and Z. Sun, "A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network," *Computers and Electronics in Agriculture*, vol. 154, pp. 18–24, Nov. 2018. https://doi.org/10.1016/j.compag.2018.08.048
[26] S. Zhang, S. Zhang, C. Zhang, X. Wang, and Y. Shi, "Cucumber leaf disease identification with global pooling dilated convolutional neural network," *Computers and Electronics in Agriculture*, vol. 162, pp. 422–430, Jul. 2019. https://doi.org/10.1016/j.compag.2019.03.012
[27] C. Wang, P. Du, H. Wu, J. Li, C. Zhao, and H. Zhu, "A cucumber leaf disease severity classification method based on the fusion of DeepLabV3+ and U-Net," *Computers and Electronics in Agriculture*, vol. 189, Oct. 2021. https://doi.org/10.1016/j.compag.2021.106373
[28] H. T. Rauf, B. A. Saleem, M. I. U. Lali, M. A. Khan, M. Sharif, and S. A. C. Bukhari, "A citrus fruits and leaves dataset for detection and classification of citrus diseases through machine learning," *Data in Brief*, vol. 26, Oct. 2019. https://doi.org/10.1016/j.dib.2019.104340
[29] U. Barman, R. D. Choudhury, D. Sahu, and G. G. Barman, "Comparison of convolution neural networks for smartphone image based real time classification of citrus leaf disease," *Computers and Electronics in Agriculture*, vol. 177, Oct. 2020. https://doi.org/10.1016/j.compag.2020.105661
[30] R. Gandhi, S. Nimbalkar, N. Yelamanchili, and S. Ponkshe, "Plant Disease Detection Using CNNs and GANs as an Augmenting Approach," May 2018.
[31] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," *Computers and Electronics in Agriculture*, vol. 161, pp. 272–279, Jun. 2019. https://doi.org/10.1016/j.compag.2018.03.032
[32] G. Geetharamani, P. J. Arun, M. Agarwalc, and S. K. Gupta, "Identification of plant leaf diseases using a nine-layer deep convolutional neural network," *Computers and Electrical Engineering*, vol. 76, pp. 323–338, Jun. 2019. https://doi.org/10.1016/j.compeleceng.2019.04.011
[33] Ü. Atila, M. Uçar, K. Akyol, and E. Uçar, "Plant leaf disease classification using EfficientNet deep learning model," *Ecological Informatics*, vol. 61, Mar. 2021. https://doi.org/10.1016/j.ecoinf.2020.101182
[34] S. Vallabhajosyula, V. Sistla, and V. K. K. Kolli, "Transfer learning-based deep ensemble neural network for plant leaf disease detection," *Journal of Plant Diseases and Protection*, 2021. https://doi.org/10.1007/s41348-021-00465-8
[35] I. Ahmad, M. Hamid, S. Yousaf, S. T. Shah, and M. O. Ahmad, "Optimizing pretrained convolutional neural networks for tomato leaf disease detection," *Complexity*, vol. 2020, 2020. https://doi.org/10.1155/2020/8812019
[36] J. Chen, D. Zhang, and Y. A. Nanehkaran, "Identifying plant diseases using deep transfer learning and enhanced lightweight network," *Multimedia Tools and Applications*, vol. 79, no. 41–42, pp. 31497–31515, Nov. 2020. https://doi.org/10.1007/s11042-020-09669-w
[37] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, "Using deep transfer learning for image-based plant disease identification," *Computers and Electronics in Agriculture*, vol. 173, Jun. 2020. https://doi.org/10.1016/j.compag.2020.105393
[38] V. Tiwari, R. C. Joshi, and M. K. Dutta, "Dense convolutional neural networks based multiclass plant disease detection and classification using leaf images," *Ecological Informatics*, vol. 63, Jul. 2021. https://doi.org/10.1016/j.ecoinf.2021.101289
[39] R. Gajjar, N. Gajjar, V. J. Thakor, N. P. Patel, and S. Ruparelia, "Real-time detection and identification of plant leaf diseases using convolutional neural networks on an embedded platform," *Visual Computer*, 2021. https://doi.org/10.1007/s00371-021-02164-9
[40] J. Chen, D. Zhang, M. Suzauddola, and A. Zeb, "Identifying crop diseases using attention embedded MobileNet-V2 model," *Applied Soft Computing*, vol. 113, Dec. 2021. https://doi.org/10.1016/j.asoc.2021.107901
[41] E. Prasetyo, N. Suciati, and C. Fatichah, "Multi-level residual network VGGNet for fish species classification," *Journal of King Saud University - Computer and Information Sciences*, 2021. https://doi.org/10.1016/j.jksuci.2021.05.015
[42] V. Kumar, H. Arora, Harsh, and J. Sisodia, "ResNet-based approach for detection classification of plant leaf diseases," in *Proceedings of the International Conference on Electronics and Sustainable Communication Systems (ICESC 2020)*, 2020, pp. 495–502. https://doi.org/10.1109/ICESC48915.2020.9155585
[43] R. Sujatha, J. M. Chatterjee, N. Z. Jhanjhi, and S. N. Brohi, "Performance of deep learning vs machine learning in plant leaf disease detection," *Microprocessors and Microsystems*, vol. 80, Feb. 2021. https://doi.org/10.1016/j.micpro.2020.103615
[44] Ü. Atila, M. Uçar, K. Akyol, and E. Uçar, "Plant leaf disease classification using EfficientNet deep learning model," *Ecological Informatics*, vol. 61, Mar. 2021. https://doi.org/10.1016/j.ecoinf.2020.101182
[45] V. Tiwari, R. C. Joshi, and M. K. Dutta, "Dense convolutional neural networks based multiclass plant disease detection and classification using leaf images," *Ecological Informatics*, vol. 63, Jul. 2021. https://doi.org/10.1016/j.ecoinf.2021.101289
[46] M. A. Moid and M. A. Chaurasia, "Transfer Learning-based Plant Disease Detection and Diagnosis System using Xception," in *Proceedings of the 5th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), I-SMAC 2021*, 2021, pp. 451–455. https://doi.org/10.1109/I-SMAC52330.2021.9640694