Research Article

Plant Diseases Classification based Leaves Image using Convolutional Neural Network

Satrio Bagus Imanulloh [1], Ahmad Rofiqul Muslikh [2], and De Rosal Ignatius Moses Setiadi [1,*]

[1] Faculty of Computer Science, Dian Nuswantoro University, Semarang, Central Java 50131, Indonesia; e-mail : satriosatriobagus@gmail.com, moses@dsn.dinus.ac.id
[2] Faculty of Information Technology, University of Merdeka, Malang, East Java 65146, Indonesia; e-mail : rofickachmad@unmer.ac.id
* Corresponding Author: De Rosal Ignatius Moses Setiadi

Abstract: Plant disease is one of the problems in the world of agriculture. Early identification of plant diseases can reduce the risk of loss, so automation is needed to speed up identification. This study proposes a custom-designed convolutional neural network (CNN) model for plant disease recognition. The proposed CNN model is not complex and lightweight, so it can be implemented in model applications. The proposed CNN model consists of 12 CNN layers, which consist of eight layers for feature extraction and four layers as classifiers. Based on the experimental results of a plant disease dataset consisting of 38 classes with a total of 87,867 image records. The proposed model can get high performance and not overfitting, with 97%, 98%, 97% and 97%, respectively, for accuracy, precision, recall and f1-score. The performance of the proposed model is also better than some popular pre-trained models, such as InceptionV3 and MobileNetV2. The proposed model can also work well when implemented in mobile applications.

Keywords: Agricultural technology; Light convolutional neural network; Image Recognition; Plant Diseases Classification; Transfer learning recognition.

Received: July, 30th 2023
Revised: August, 18th 2023
Accepted: August, 27th 2023
Published: August, 30th 2023

1. Introduction

Plant disease is one of the challenges in agricultural production that is important to identify early on. Early identification can prevent bad things, such as decreased quality and quantity of agricultural products, leading to crop failure. To classify plant diseases requires the ability of experts, but this is certainly less effective if you want to get more and faster production. Technological developments have been utilized for agriculture and can even be implemented to automatically classify plant diseases[1], [2]. This technology involves several branches of science such as image processing and artificial intelligence. Machine learning and deep learning is a branch of artificial intelligence that is widely applied to complete this task[3], [4]. Several machine learning methods that are widely applied in classification tasks are k-nearest neighbor (KNN), support vector machine(SVM), decision tree(DT), random forest(RF), etc [5]–[8]. But the classification method cannot work alone, image features are needed as input for the training and testing process in the classification method. Some of these features can be color, texture, or shape. Color features such as red, green and blue (RGB), hue, saturation, value (HSV), etc[9], [10], texture features such as the gray level cooccurrence matrix (GLCM) and Histogram of an Oriented Gradient (HOG)[7], [11], while shape features such as metric and eecentricity [12]. These features need to be considered as input and must match the classifier to get the best classification performance, besides that it is necessary to determine several parameters such as the value of k in the KNN algorithm and the type of kernel in the SVM algorithm.

Currently, deep learning is more popular than machine learning, in which the convolutional neural network (CNN) is the most popular method in image processing, especially in this case recognition, classification or identification. CNN is capable of carrying out feature extraction tasks as well as classifying[13]. This phenomenon must be attributed to the powerful ability of CNN image processing, which can automatically extract features by stacking convolution layers. One of the hardest challenges in CNN is avoiding overfitting where the resulting model performs well with the training data, but does not generalize well to new, untrained data[14]. In addition, the training time on CNN can take longer and requires larger data to support its performance. The technique that can be used to reduce overfitting is to use a dropout layer where these dropouts are represented on the fully connected layer by randomly deactivating some unnecessary neurons[15], [16]. This may seem counter-intuitive, but it is an effective way to ensure that the learning model doesn't become too dependent on training images. Another technique that can be used to reduce overfitting is augmenting the image to be trained. Augmentation is generally done when the dataset is relatively small, in image enhancements that are usually carried out are rescale, horizontal flip, vertical flip, zoom range, rotation range, etc.

Currently there are quite a lot of CNN models proposed in various studies. Pre-trained CNN model, is a quite popular and practical CNN model, where we can use this model by using its functions with various libraries such as Keras.Some pre-trained models that are widely used are MobileNet, EfficientNet, VGG, RestNet, Inception, DenseNet, and Xception[17]–[19]. These models are generally designed in a complex way to get high performance in working on recognition cases in thousands of classes. But the higher number of parameters will consume a lot of hardware resources and time. MobileNetV2 and InceptionV3 are examples of CNN architectures that can achieve high levels of accuracy by keeping the number of parameters and computations as low as possible. [20]. The performance of these two models has been tested in several studies, such as [21]–[24]. But for some problems that are not too complicated or focus on certain objects in this case, the recognition of plant diseases, CNN with a simpler architecture should work well if designed properly. The results of the proposed model can compete with more complex CNN architectures and even surpass them[25].

Based on some of the literature above, this study aims to:

1. Designing a simple CNN model to carry out plant disease recognition
2. Comparing the performance of the proposed CNN model and compared to other popular CNN models, namely MobileNetV2 and InceptionV3.
3. Deploy the CNN model on the Android application to prove the model is implemented and can run smoothly in mobile applications.

The remainder of this paper is written in five sections. The first is related work, which explains prior art research, the second is the proposed method which explains the proposed CNN model and how to test the model, the third is results and discussion which explains the discussion of results and discussion of analysis, the fourth is the comparison which proves that the proposed model has contribution and is superior to the high-performance pre-trained model, and the last is a conclusion that contains a summary and future work of the research.

2. Related Work

Many research articles on plant disease classification have been published. Study [7] proposed classification of plant disease by RF method and HOG feature extraction. The hypothesis is because RF is a learning method that can build decision tree forests during the training process. This method is also able to overcome overfitting and is able to handle numeric and logical data. Whereas HOG is a texture feature extraction that produces Hu moment, Haralick texture and color histogram. The testing process uses a relatively small dataset on 160 images of papaya tree leaves. The accuracy of the proposed method reaches 70.14% and outperforms other ML methods such as SVM, KNN, and NB. RF is also implemented in [8], but in this research several different stages were carried out, namely image conversion from RGB to L*a*b* and image segmentation.

On research [26] CNN-based deep learning is proposed. The SNN model is designed based on the inception layer and residual connection, besides that depthwise separable convolution is used to reduce parameters. The model was tested on three types of plant disease datasets, namely plant village, rice disease and cassava data sets. Each dataset has a different number of classes. Plantvillage has 17 classes with records varying from 373 to 5357. Rice disease has four classes with records from 1308 to 1600, and the casava dataset has five classes with records 316 to 2658. The model is quite reliable with an accuracy above 99% for the Plantvillage dataset. and rice disease, but showed over-fitting results on the cassava dataset where the training and testing accuracy were 98.17% and 76.56%, respectively. In addition, this model is superior to various popular pre-trained or transfer learning methods such as VGG, ResNet, AlexNet and GoogleNet.

Research [27] also proposed CNN for plant disease recognition, CNN was built on the basis of VGGNet pre-trained on ImageNet and Inception module. The plantvillage dataset was also used in the study, and the accuracy of the proposed method was 93.78. This model is superior to other pre-trained methods such as DenseNet201, ResNet50, InceptionV3, and VGG19. Based on some of the literature above, it appears that the CNN model has much better performance and is open to development. This study proposes a custom CNN model with a simplified number of layers to recognize plant disease with a more varied dataset with a greater number of classes and records.

3. Proposed Method

This study has several research stages as shown in Fig. 1. In particular, this section discusses the design of the proposed CNN model in section 3.1 and the assessment model in section 3.2.

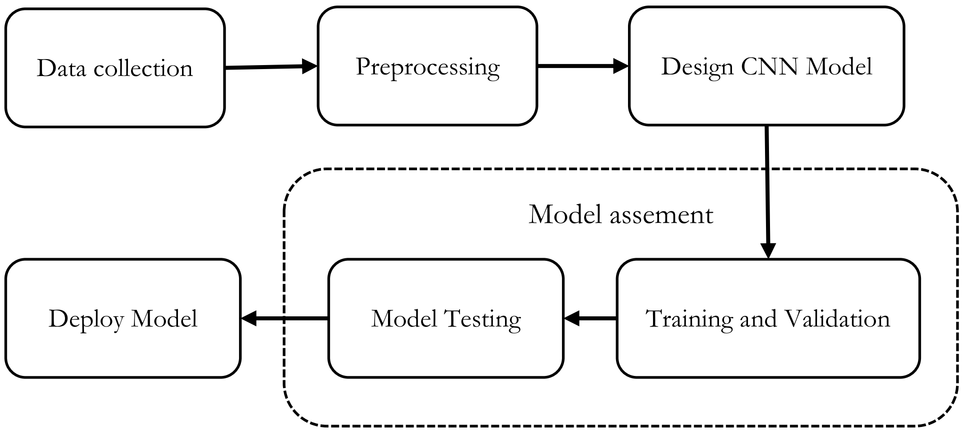


Figure 1. Research stages.

3.1. Proposed CNN Model

The proposed CNN model is specifically designed for the classification of plant diseases. We develop CNN layers based on theory and trial and error. The proposed model generally consists of convolutional, pooling, and fully connected layers. These layers have the main task of performing feature extraction and classification. After that, the model must be compiled with the optimizer, namely Adam, learning rate of 0.001 and batch size of 64. The proposed design of the CNN model is presented in Table 1.

Table 1. Proposed CNN Model.

No	Layer	Parameter
1	Convolutional layer	Filters=32, kernel_size=(3,3), activation="relu", and input_shape=(128, 128, 3)
2	Max pooling layer	Pool_size=(2, 2)
3	Convolutional layer	Filters=64, kernel_size=(3,3), and activation="relu"
4	Max pooling layer	Pool_size=(2, 2)
5	Convolutional layer	Filters=128, kernel_size=(3,3), and activation="relu"
6	Max pooling layer	Pool_size=(2, 2)
7	Convolutional layer	Filters=256, kernel_size=(3,3), and activation="relu"
8	Max pooling layer	Pool_size=(2, 2)
9	Flatten layer	none
10	Dense layer	Units=1024 and activation="relu"
11	Dropout Layer	Rate=0,2
12	Dense layer	Units=38 and activation="softmax"

Based on the layers proposed in Table 1, each function and its parts are explained as follows:

3.1.1. Feature Extraction

The initial eight layers arranged in Table 1 serve as feature extractors, consisting of two types of layers: the convolution layer and pooling. The convolutional layer aims to detect features found within the local region of the input images that are common across the dataset and map their appearance onto a feature map. Several Conv2D parameters are used, including: a) Filter is the number of output filters in the convolution, usually in the form of an integer, b) Kernel size is to determine the height and width of the 2D convolution window. Usually filled in in the form of a tuple or list of 2 integers, c) Activation is filled with the activation function to be used, d) Input shape is adjusted to the target image size that has been determined/written. The pooling layer is used to reduce the image size or matrix size. In this study, using the MaxPooling2D function. The parameter used is the pool size. Pool size takes the maximum value from the specified pool window. The value of this parameter is a tuple containing two integers.

3.1.2. Classifier

The CNN model has a classifier that can be integrated, the last four layers in Table 1 serve as classifiers which consist of three types of layers, namely flatten, dense and dropout layers. Flatten is used to make inputs that have more than one dimension into one dimension, usually used before fully connected. In this study, the flattened layer does not contain parameters. The dense layer is the architectural model layer which contains many neurons. The parameters used include units and activation. Units are the values of the dimensions of the input space, usually integer values. Activation is filled with the activation function to be used, according to the function of the layer. The dropout layer is used to reduce the connection of neurons which is expected to prevent overfitting in the model. In this study, the dropout layer value of the parameter rate is filled with a predetermined value. The value is between 0 to 1.

3.2. Model Assessment

Based on Fig. 1 two stages have an assessment, the first is training and validation, the second is testing. In training and validation, accuracy is calculated with a function in the Keras library, namely the accuracy_score function from the sklearn.metrics module. Meanwhile, the loss function is used to calculate the difference between the predictions of the CNN model and the actual labels in the training data. In this research, the sparse_categorical_crossentropy parameter is used because it is used for multi-class classification.

At the testing assessment stage, accuracy, recall, precision and f1-score are calculated based on the confusion matrix. Formulas (1), (2), (3), and (4) are used to calculate accuracy, precision, recall and f1-score, respectively.

$$acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$prec = \frac{TP}{TP + FP} \tag{2}$$

$$rec = \frac{TP}{TP + FN} \tag{3}$$

$$f1 = \frac{2(prec * rec)}{prec + rec} \tag{4}$$

Where Positive Predictions = True Positive (TP) + False Positive (FP), Negative Predictions = False Negative (FN) + True Negative (TN), TP is the number of correctly predicted positive data, FP is the number of negative data incorrectly predicted as positive, TN is the number of negative data that is correctly predicted, FN is the number of positive data incorrectly predicted as negative.

4. Results and Discussion

This research was implemented on a public dataset downloaded from the Kaggle website, namely the New Plant Diseases Dataset [28]. This dataset comprises 87867 images of healthy and diseased leaves categorized into 38 different classes. Which all images are used in this study. Table 2 presents more detailed information regarding the name and number of each class.

Table 2. Dataset Details (Records and Class Name).

Class Name	Training	Validation	Testing
Apple scab	2016	499	5
Apple black rot	1987	493	4
Apple cedar rust	1760	436	4
Apple healthy	2008	497	5
Blueberry healthy	1816	450	4
Cherry powdery mildew	1683	417	4
Cherry healthy	1826	452	4
Corn maize cercospora leaf spot gray leaf spot	1642	406	4
Corn maize common rust	1907	473	4
Corn maize northern leaf blight	1908	473	4
Corn maize helathy	1859	461	4
Grape black rot	1888	468	4
Grape esca black measles	1920	476	4
Grape leaf blight isariopsis leaf spot	1722	426	4
Grape healthy	1692	419	4
Orange haunglongbin citrus greening	2010	498	5
Peach bacterial spot	1838	455	4
Peach healthy	1728	428	4
Pepper bell bacterial spot	1913	474	4
Pepper bell healthy	1988	493	4
Potato early blight	1939	481	4
Potato late blight	1939	481	4
Potato healthy	1824	452	4
Raspberry healthy	1781	441	4
Soybean healthy	2022	500	5
Squash powdery mildew	1736	430	4
Strawberry leaf scorch	1774	440	4
Strawberry healthy	1824	452	4
Tomato bacterial spot	1702	421	4
Tomato early blight	1920	476	4
Tomato late blight	1851	459	4
Tomato leaf mold	1882	466	4
Tomato septoria leaf spot	1745	432	4
Tomato spider mites two spotted spider mite	1741	431	4
Tomato target spot	1827	453	4
Tomato yellow leaf curl virus	1961	486	4
Tomato mosaic virus	1790	444	4
Tomato healthy	1926	477	4
Total	70295	17416	156

Based on Table 2, we divided the leaf images into 70295 training images (≈80%), 17416 validation images (≈19.8%), and 156 testing images (≈0.02%). In addition, validation split 0.01 is used. The sample image used is presented in Fig. 2.

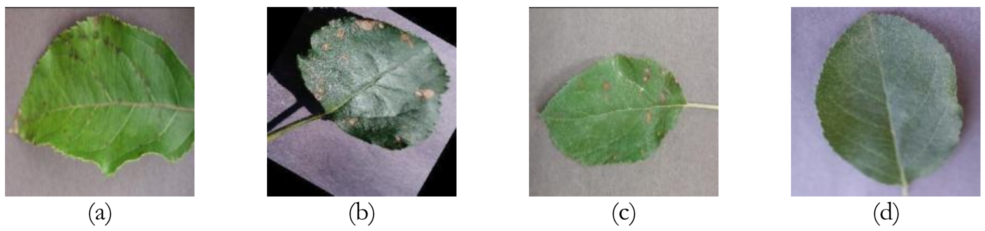

(a) (b) (c) (d)

Figure 2. Example of a leaf image on an apple plant{(a) Apple scab; (b) Apple black rot; (c) Apple cedar rust; (d) Apple Healty}

In the preprocessing stage, the normalization process is carried out with rescale. This is used to scale the training, validation, and evaluation datasets to a range of 0-1. The goal is to speed up the CNN model when doing calculations. Next, adjust the target_size of the image to fit the input layer. In this study the target size is used (128, 128). Next, the CNN model training process is carried out on Google Colab with the training dataset (training_set), and the validation dataset (valid_set), which have been separated. The number of epochs used in this study was 20. The results of the training and validation plots are presented in Fig. 3.

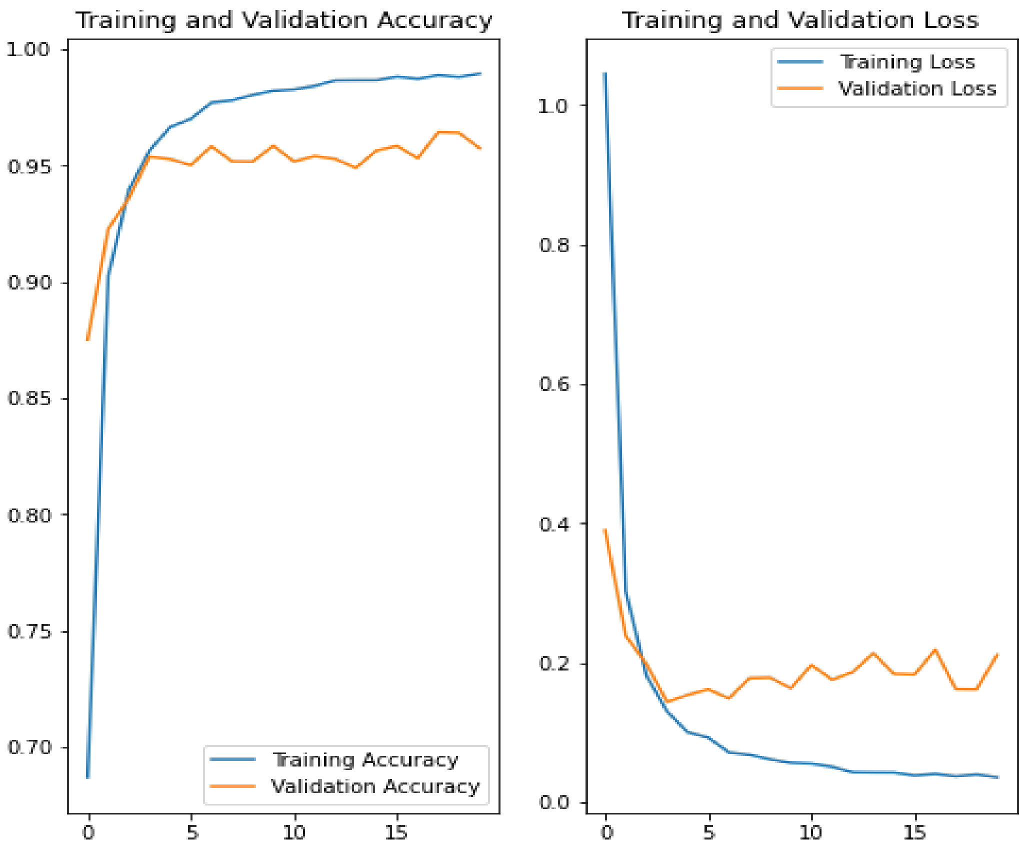


Figure 3. Accuracy and Loss Plot Results of Proposed Model.

Next, evaluation testing is carried out using the confusion_matrix function, the results of which are presented in Fig. 4. Furthermore, from the confusion matrix table, accuracy, precision, recall and f1-score are calculated, and the results are 97%, 98%, 97% and 97% respectively. These results indicate that the performance of the proposed model is very reliable and stable both in the training, validation and testing processes.

The final stage is deploying the model into a mobile application with Android Studio. The created Android Studio project will import model files in .tflite format. The contents of the tflite file are then copied to the classifyImage() function to classify leaf images. Things that are done include:

1. Instantiate the modelCustom class with the model name.
2. Create and manage input.
3. Normalize or rescale.
4. Run model inference and get results.
5. Search and display the class index with the largest value.

The appearance of the application that is made is quite simple, consisting of 1 imageview to display uploaded leaf images, 2 textviews to display the words "*Hasil Klasifikasi*" and their classification class, and 2 buttons to upload images from the gallery or take photos from the camera. Display applications made in this study as in Fig. 5.

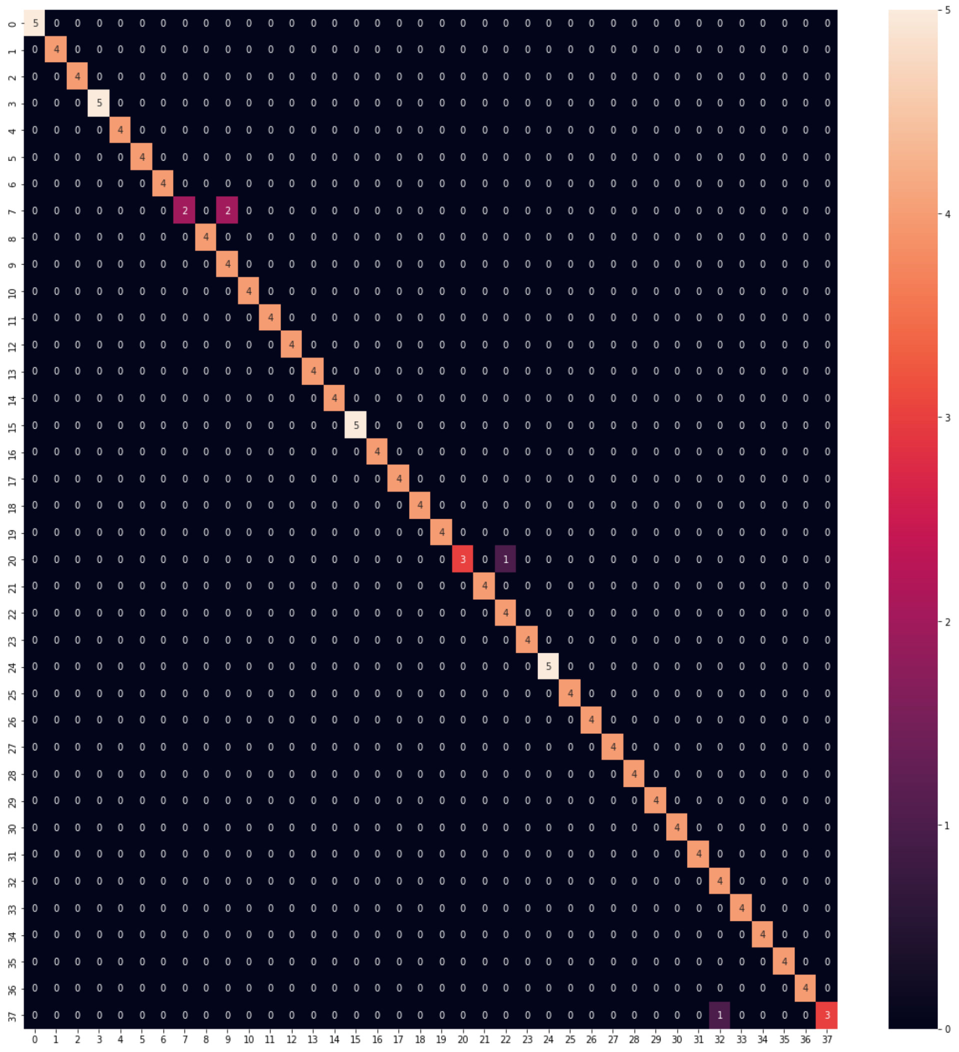
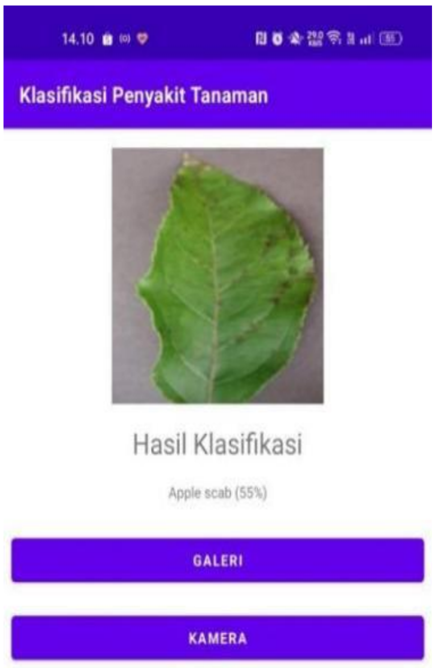


Figure 4. Confusion matrix of testing assessment of proposed CNN model.



Klasifikasi Penyakit Tanaman

Hasil Klasifikasi

Apple scab (55%)

Figure 5. Deployment of proposed CNN model in Android Operating System.

5. Comparison

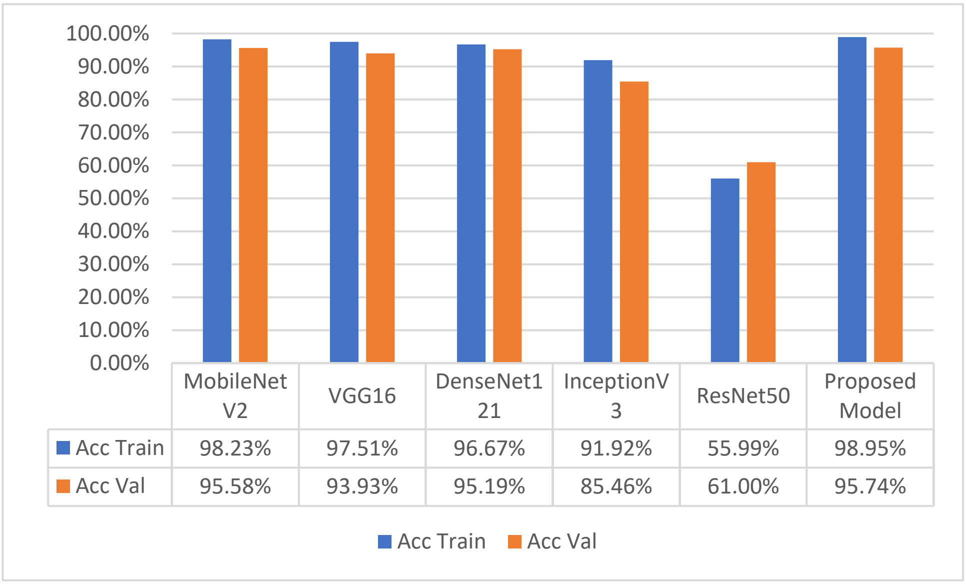
In this study, the proposed model is compared with pre-trained models that are popular and have high accuracy, such as MobileNetV2, VGG16, DenseNet121, RestNet50, and InceptionV3. Because by default the pre-trained model has a number of classes that are different from the dataset, several fine tuning settings are made, such as:

- layers.Flatten()(x)
- x = layers.Dense(1024, activation="relu")(x)
- x = layers.Dropout(0.2)(x)
- predictions = layers.Dense(n_classes, activation="softmax")(x)

In addition, freeze training is carried out on the layers so that the training process runs more efficiently. In more detail Table 3 presents data about the parameters used, while Figure 6 presents a comparison of training and validation accuracy.

Table 3. Parameter Comparison with Other CNN Models.

CNN Model	Trainable params	Non trainable params	Total Params
MobileNetV2	21011494	2257984	23269478
VGG16	8428582	14714688	23143270
DenseNet121	16817190	7037504	23854694
InceptionV3	8428582	21802784	30231366
ResNet50	33594406	23587712	57182118
Proposed Model	9865574	0	9865574



	MobileNet V2	VGG16	DenseNet121	InceptionV3	ResNet50	Proposed Model
Acc Train	98.23%	97.51%	96.67%	91.92%	55.99%	98.95%
Acc Val	95.58%	93.93%	95.19%	85.46%	61.00%	95.74%

Figure 6. Performance Comparison with Other CNN Models.

Based on the results presented in Table 3, it can be seen that the performance of the proposed CNN model is superior in terms of accuracy. The total parameters used are also the fewest, so the computational complexity is of course relatively lower. However, the DenseNet121 model has the least overfitting because the difference in accuracy is minimal. Nonetheless, the proposed CNN model has provided good performance with high accuracy, precision, recall, and f1-score levels. Such performance as being able to remember/identify positive examples and predict classification correctly. This shows that the proposed CNN model is effective in the task of recognizing plant diseases.

6. Conclusions

This study has successfully proposed a CNN model that performs satisfactorily for classifying plant diseases. Classification results in the training, validation and testing processes show stable results and relatively mild overfitting. In addition, you can get a shorter training time by designing a simple CNN model. Overall, the proposed model is also superior compared to other popular pre-trained models such as MobileNetV2 and InceptionV3. In addition, the simple CNN model can be implemented well in mobile applications. For future research, this can be minimized by reducing the difference in training accuracy and validation with other techniques such as data augmentation, adding or removing layers.

Author Contributions: Conceptualization: S.B. Imanulloh and D.R.I.M. Setiadi; methodology: S.B. Imanulloh and D.R.I.M. Setiadi software: S.B. Imanulloh.; validation: S.B. Imanulloh, D.R.I.M. Setiadi and A.R. Muslikh.; formal analysis: D.R.I.M. Setiadi.; investigation: D.R.I.M. Setiadi, A.R. Muslikh; resources, S.B. Imanulloh; data curation, S.B. Imanulloh.; writing—original draft preparation: S.B. Imanulloh.; writing—review and editing: D.R.I.M. Setiadi, A.R. Muslikh; visualization: S.B. Imanulloh.; supervision, D.R.I.M. Setiadi.; project administration: A.R. Muslikh.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] M. S. P. Ngongoma, M. Kabeya, and K. Moloi, "A Review of Plant Disease Detection Systems for Farming Applications," *Appl. Sci.*, vol. 13, no. 10, p. 5982, May 2023, doi: 10.3390/app13105982.

[2] M. Jung *et al.*, "Construction of deep learning-based disease detection model in plants," *Sci. Rep.*, vol. 13, no. 1, p. 7331, May 2023, doi: 10.1038/s41598-023-34549-2.

[3] I. Ahmed and P. K. Yadav, "A systematic analysis of machine learning and deep learning based approaches for identifying and diagnosing plant diseases," *Sustain. Oper. Comput.*, vol. 4, no. February, pp. 96–104, 2023, doi: 10.1016/j.susoc.2023.03.001.

[4] C. Jackulin and S. Murugavalli, "A comprehensive review on detection of plant disease using machine learning and deep learning approaches," *Meas. Sensors*, vol. 24, no. July, p. 100441, 2022, doi: 10.1016/j.measen.2022.100441.

[5] S. S. Harakannanavar, J. M. Rudagi, V. I. Puranikmath, A. Siddiqua, and R. Pramodhini, "Plant leaf disease detection using computer vision and machine learning algorithms," *Glob. Transitions Proc.*, vol. 3, no. 1, pp. 305–310, 2022, doi: 10.1016/j.gltp.2022.03.016.

[6] T. S. Xian and R. Ngadiran, "Plant Diseases Classification using Machine Learning," *J. Phys. Conf. Ser.*, vol. 1962, no. 1, 2021, doi: 10.1088/1742-6596/1962/1/012024.

[7] S. Ramesh *et al.*, "Plant Disease Detection Using Machine Learning," in *2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C)*, Apr. 2018, pp. 41–45. doi: 10.1109/ICDI3C.2018.00017.

[8] A. Devaraj, K. Rathan, S. Jaahnavi, and K. Indira, "Identification of Plant Disease using Image Processing Technique," in *2019 International Conference on Communication and Signal Processing (ICCSP)*, Apr. 2019, pp. 0749–0753. doi: 10.1109/ICCSP.2019.8698056.

[9] A. Susanto, Z. H. Dewantoro, C. A. Sari, D. R. I. M. Setiadi, E. H. Rachmawanto, and I. U. W. Mulyono, "Shallot Quality Classification using HSV Color Models and Size Identification based on Naive Bayes Classifier," *J. Phys. Conf. Ser.*, vol. 1577, no. 1, 2020, doi: 10.1088/1742-6596/1577/1/012020.

[10] O. R. Indriani, E. J. Kusuma, C. A. Sari, E. H. Rachmawanto, and D. R. I. M. Setiadi, "Tomatoes classification using K-NN based on GLCM and HSV color space," in *Proceedings - 2017 International Conference on Innovative and Creative Information Technology: Computational Intelligence and IoT, ICITech 2017*, Nov. 2018, vol. 2018-Janua, pp. 1–6. doi: 10.1109/INNOCIT.2017.8319133.

[11] D. Fajri Riesaputri, C. Atika Sari, I. M. S. De Rosal, and E. Hari Rachmawanto, "Classification of Breast Cancer using PNN Classifier based on GLCM Feature Extraction and GMM Segmentation," in *2020 International Seminar on Application for Technology of Information and Communication (iSemantic)*, Sep. 2020, pp. 83–87. doi: 10.1109/iSemantic50169.2020.9234207.

[12] E. Hari Rachmawanto, G. Rambu Anarqi, D. R. I. Moses Setiadi, and C. Atika Sari, "Handwriting Recognition Using Eccentricity and Metric Feature Extraction Based on K-Nearest Neighbors," in *Proceedings - 2018 International Seminar on Application for Technology of Information and Communication: Creative Technology for Human Life, iSemantic 2018*, Nov. 2018, pp. 411–416. doi: 10.1109/ISEMANTIC.2018.8549804.

[13] J. Heaton, "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning," *Genet. Program. Evolvable Mach.*, vol. 19, no. 1–2, pp. 305–307, Jun. 2018, doi: 10.1007/s10710-017-9314-z.

[14] A. Susanto, I. U. Wahyu Mulyono, C. Atika Sari, E. Hari Rachmawanto, D. R. I. M. Setiadi, and M. K. Sarker, "Handwritten Javanese script recognition method based 12-layers deep convolutional neural network and data augmentation," *IAES Int. J. Artif. Intell.*, vol. 12, no. 3, p. 1448, Sep. 2023, doi: 10.11591/ijai.v12.i3.pp1448-1458.

[15] 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, pp. 1929–1958, 2014.

[16] T. Liu, P. Zheng, and J. Bao, "Deep learning-based welding image recognition: A comprehensive review," *J. Manuf. Syst.*, vol. 68, no. March, pp. 601–625, Jun. 2023, doi: 10.1016/j.jmsy.2023.05.026.

[17] V. Singh, A. Chug, and A. P. Singh, "Classification of Beans Leaf Diseases using Fine Tuned CNN Model," *Procedia Comput. Sci.*, vol. 218, no. 2022, pp. 348–356, 2023, doi: 10.1016/j.procs.2023.01.017.

[18] S. Verma, P. Kumar, and J. P. Singh, "A meta-learning framework for recommending CNN models for plant disease identification tasks," *Comput. Electron. Agric.*, vol. 207, no. June 2022, p. 107708, 2023, doi: 10.1016/j.compag.2023.107708.

[19] M. T. Ahad, Y. Li, B. Song, and T. Bhuiyan, "Comparison of CNN-based deep learning architectures for rice diseases classification," *Artif. Intell. Agric.*, Jul. 2023, doi: 10.1016/j.aiia.2023.07.001.

[20] S. M. Hassan, A. K. Maji, M. Jasiński, Z. Leonowicz, and E. Jasińska, "Identification of Plant-Leaf Diseases Using CNN and Transfer-Learning Approach," *Electronics*, vol. 10, no. 12, p. 1388, Jun. 2021, doi: 10.3390/electronics10121388.

[21] K. Dong, C. Zhou, Y. Ruan, and Y. Li, "MobileNetV2 Model for Image Classification," *Proc. - 2020 2nd Int. Conf. Inf. Technol. Comput. Appl. ITCA 2020*, pp. 476–480, Dec. 2020, doi: 10.1109/ITCA52113.2020.00106.

[22] R. Indraswari, R. Rokhana, and W. Herulambang, "Melanoma image classification based on MobileNetV2 network," *Procedia Comput. Sci.*, vol. 197, pp. 198–207, Jan. 2022, doi: 10.1016/J.PROCS.2021.12.132.

[23] C. Wang *et al.*, "Pulmonary image classification based on inception-v3 transfer learning model," *IEEE Access*, vol. 7, pp. 146533–146541, 2019, doi: 10.1109/ACCESS.2019.2946000.

[24] K. Joshi, V. Tripathi, C. Bose, and C. Bhardwaj, "Robust Sports Image Classification Using InceptionV3 and Neural Networks," *Procedia Comput. Sci.*, vol. 167, pp. 2374–2381, Jan. 2020, doi: 10.1016/J.PROCS.2020.03.290.

[25] L. Fu, S. Li, Y. Sun, Y. Mu, T. Hu, and H. Gong, "Lightweight-Convolutional Neural Network for Apple Leaf Disease Identification," *Front. Plant Sci.*, vol. 13, no. May, pp. 1–10, 2022, doi: 10.3389/fpls.2022.831219.

[26] S. M. Hassan and A. K. Maji, "Plant Disease Identification Using a Novel Convolutional Neural Network," *IEEE Access*, vol. 10, pp. 5390–5401, 2022, doi: 10.1109/ACCESS.2022.3141371.

[27] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, "Using deep transfer learning for image-based plant disease identification," *Comput. Electron. Agric.*, vol. 173, no. March, p. 105393, Jun. 2020, doi: 10.1016/j.compag.2020.105393.

[28] S. Bhattarai, "New Plant Diseases Dataset," *Kaggle*, 2018. https://www.kaggle.com/datasets/vipoooool/new-plant-diseases-dataset