



Performance Comparison of Convolutional Neural Network and MobileNetV2 for Chili Diseases Classification

Achmad Naila Muna Ramadhani¹, Galuh Wilujeng Saraswati², Rama Tri Agung³, Heru Agus Santoso⁴

^{1,2,4}Informatics Engineering, Faculty of Computer Science, Dian Nuswantoro University, Semarang, Indonesia

³Informatics Engineering, Faculty of Computer Science, Pembangunan Nasional Veteran Yogyakarta University, Yogyakarta, Indonesia

¹111201912206@mhs.dinus.ac.id, ²galuhwilujengs@dsn.dinus.ac.id, ³123180053@student.upnyk.ac.id,

⁴heru.agus.santoso@dsn.dinus.ac.id

Abstract

Chili is an important agricultural commodity in Indonesia and plays a significant role in the nation's economic growth. Its demand by households and industries reaches up to 61%. However, this high demand also means that monitoring efforts need to be intensified, particularly for chili plant diseases that can greatly impact yields. If these diseases are not promptly addressed, they can lead to a decrease in production levels, which can negatively affect the economy. With technological advancements, automatic monitoring using image processing is now highly feasible, making monitoring more efficient and effective. Common chili plant diseases include Chili leaf yellowing disease, Chili leaf curling disease, and cercospora leaf spots and Magnesium Deficiency with symptoms that can be observed through the shape and color of the leaves. This research aims to classify chili plant diseases by comparing the CNN algorithm and the pre-trained MobileNetV2 based model performance using Confusion Matrix. The study shows that the MobileNetV2 model, trained with a learning rate of 0.001, produces a more optimal model with an accuracy of 90% and based on the calculation of the confusion matrix, the average percentage values for recall, precision, and F1 score are 92%. These findings highlight the potential of image processing and pre-trained models to support efforts to monitor plant diseases and improve chili production.

Keywords: chili; comparison; CNN; mobilenetV2

1. Introduction

Indonesia is known as an agrarian country, so its people still rely on the agricultural sector. Chili is one of the largest agricultural commodities in Indonesia. Based on data from the Badan Pusat Statistik (2022), chili production in Indonesia over the past three years has produced a total of 7.09 tons. The annual breakdown shows that in 2019, chili commodities produced around 1.37 tons, in 2020 produced 1.5 tons, and in the last year, 2021, chili commodities were able to produce 1.39 tons. In this data, the amount of chili production increases every year, such as in 2020, which reached 1.51 million tons. This amount increased by 9.76% compared to the previous year. Chili plants also get the title as a determinant of national economic growth in Indonesia because the demand for chili in households and industries reaches as much as 61%. The high demand for chili makes the price of chili very fluctuating and is one of the reasons for high inflation rates in Indonesia [1]. Various factors that can cause an increase in chili prices due to a decrease in chili

production levels include poor seed quality, worsening soil fertility levels, inadequate chili plant cultivation techniques, and common problems that occur in plants such as pests and diseases [2].

One of the causes of the decline in chili production rates is due to pests and diseases that can cause losses in both the quantity and quality of chili plants. Some diseases that often affect chili production rates in Indonesia are yellowing and curling diseases [2]. If chili harvest production rates decrease in both quality and quantity due to pests and diseases, it will have an impact on the economy. Therefore, early detection of plant diseases is essential in plant maintenance and care. Diseases that are not detected and allowed to develop will result in damage to the plants. This is what can cause a decrease in the quality and quantity of chili harvests, which can affect the country's economy [3].

Based on the problem, a solution is needed to improve the maximum level of chili production. As science advances, innovations have been made to detect plant

diseases automatically using computer technology. Disease detection using a computer is considered highly recommended due to its fairly accurate results, and one of the commonly used methods is Deep Learning [3].

In the study conducted by Athallah Tsany Rakha Dzaky from Telkom University entitled "Detection of Chili Plant Diseases Using Convolutional Neural Network Method". In this study, the researcher used Image Processing method using Convolutional Neural Network assisted by background removal process and Image Augmentation by rotating the image with an interval of 30 degrees for each existing image. This study resulted in a model accuracy of 89.7% [3].

The research conducted by Abwabul Jinan and B. Herawan from Potensi Utama University titled "Classification of Rice Plant Diseases Using Convolutional Neural Network Method through Leaf Images (Multilayer Perceptron)". In this study, the dataset comparison was 90:10 for training and testing data using RGB images as training and testing data. It resulted in an accuracy of 91.7% [4].

The study conducted by Nasha Hikmatia A.E. and Muhammad Ihsan Zul entitled "Indonesian Sign Language to Voice Translator Android Application using Tensorflow". This study focuses on sign language translation using transfer learning method, specifically MobileNetV2, which is assisted by data normalization and Image Augmentation to increase the number of required training data. This study resulted in an accuracy of 54.8% [5].

The research conducted by Faisal Dharma Adhinata, Nia Annisa Ferani Tanjung, Widi Widayat, Gracia Rizka Pasfica, and Fadlan Raka Satura, titled "Comparative Study of VGG16 and MobileNetV2 for Masked Face Recognition". This study focuses on face classification between masked and unmasked faces using a comparison between VGG16 and MobileNetV2 transfer learning. In the preprocessing stage, the researchers rescaled the data, rotated the image by 45 degrees, performed horizontal and vertical flipping, and shifting to increase the amount of data to be trained. In this study, MobileNetV2 achieved an accuracy of 95% [6].

Previous studies have shown that deep learning algorithms, namely CNN and MobileNetV2 transfer learning, are often used to address various issues related to image detection systems, resulting in relatively high and optimal levels of accuracy. Therefore, this research was conducted to compare the classification of different types of diseases in chili plants using CNN and MobileNetV2 methods. The implementation of these methods in the classification of chili plant diseases is expected to assist users in diagnosing chili plant diseases, thereby helping to maintain the quality and quantity of chili production.

2. Research Methods

This research aims to determine the classification of diseases in chili plants using Deep Learning method with Convolutional Neural Network algorithm. The research flow that will be carried out is as follows in Figure 1.

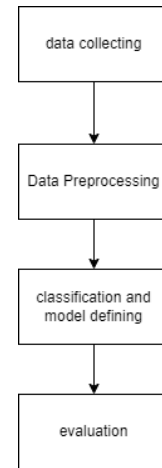


Figure 1. Research Method Flow


2.1. Data Collecting

The process of taking chili plant disease image data was carried out from around 11:00 am to 1:00 pm in Sengon village, Getasan district, Salatiga. The images were taken using a 13-megapixel Redmi 7A smartphone camera. In addition to privately collected image data, there is also image data accessed through a public data platform, namely Github. The collected chili plant disease images are color images with varying pixel sizes. The images have a garden background and are focused on a single infected leaf.

The types of diseases that can be classified include Chili Yellowish Leaf, this is a disease that attacks the leaves of chili plants caused by the *geminivirus*. Chili plants infected with *geminivirus* have several variations of symptoms such as green leaf veins with yellow tissue, yellow leaves with thick leaf veins, yellow leaf veins but the tissue remains green, and leaf edges that curl upwards [7]. Cercospora Leaf Spot This is a disease caused by the fungus *Cercospora capsici*. This type of fungus attacks the host on the leaves. Leaf spot disease in chili is one of the most common diseases affecting chili plants in Indonesia. This disease can cause damage from seedling to fruiting stages [7]. Chili Leaf Curl This disease is often caused by the Gemini virus. The Gemini virus is most commonly transmitted by the *Bemisia Tabaci whitefly*. Symptoms that often occur in plants infected with this disease include chlorosis in the leaf veins and small or stunted leaf size [8]. Magnesium is a substance that helps plants in forming chlorophyll, fats, and carbohydrates. Magnesium deficiency in chili plants will result in reduced seed growth and cause

leaves to dry up and die quickly [9]. The division of public and private data is represented in Table 1.

Table 1. Division of public and private data

Label Name	Public Data	Private Data	Photo
Leaf Curl	300	375	
Yellowish	412	232	
Magnesium Deficiency	0	232	
Cercospora Leaf Spot	0	258	
Healthy	430	255	
Amount Total	1142	1352	

On Table 1, the collected image data consists of 2494 images and has 5 different classes, with 4 classes consisting of 644 yellowing disease leaves, 675 curly leaves, 232 magnesium deficiency leaves, and 258 spot anthracnose leaves. The remaining 685 image data is for the class to identify healthy chili leaves.

2.2. Pre-Processing

the preprocessing stage is carried out by normalizing the pixels in the image and resizing the original image dimensions to 300 x 300 pixels. The resized image data is then augmented using the ImageDataGenerator by rotating the image by 25 degrees and applying vertical and horizontal flips to increase the amount of data that will be used as training data. Then, the augmented data is generated into 1000 image data for each label due to the imbalance in the previously taken data, which only had around 230 image data for magnesium deficiency and also cercospora spot diseases. After the raw data has been normalized, resized, and generated to prevent imbalanced datasets during training, the next step is to perform data augmentation again to adjust the image size to match the input size used in the model architecture. In both models, which are the CNN architecture model and the pre-trained model using

MobileNetV2, 224 x 224 image data are used as input data for training. The result amount of datasets after splitting is represented in Table 2.

Table 2. Amount of Data After Splitting

Data Group	Amount
Training	3500
Validation	1500
Amount	5000

On Table 2 the dataset will be divided into two parts, which are training data and validation data. The training data is used to train and create a model that is adjusted to the target_size on the model's input layer. Validation data is used as validation or testing data that is fed into the model [10]. The percentage used for the division between training and validation data is 70% for training data and the remaining 30% is used as validation data.

2.3. Classification

The classification process was carried out using the Deep CNN method with a self-made model architecture and using transfer learning method using MobileNetV2. CNN is the most well-known and commonly used algorithm method in image processing. The advantage of CNN compared to its predecessors is its ability to automatically identify relevant features. The image data used as input data consists of three dimensions, namely length x width x color channel. Length and width refer to the size of the image in pixels, while the color channel indicates what color the image has. For example, an RGB image will have a channel size of 3, where each channel represents a basic color unit of red, green, and blue [11]. CNN has weights, biases, and activations that enable the neural network to output results according to the training data that has been given [3]. The architecture of convolutional neural network is represented in Figure 2.

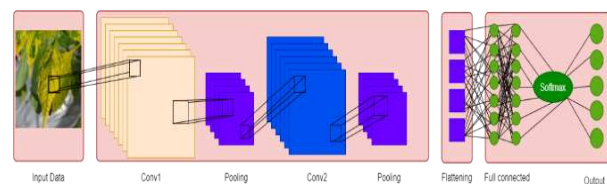


Figure 2.CNN Architecture

Based on Figure 2 in the CNN architecture, there are several layers, the first of which is the convolutional layer that will perform convolution on the image with a defined filter without damaging the initial image structure. This layer functions to extract features that will be used to train the model [3]. Next, there is a pooling layer that functions to reduce the dimensionality of the data used. There are several types of pooling, and each pooling has different characteristics. In this study, MaxPooling was used, which will keep the largest value from a group of data. This pooling will discard values that are lower than the maximum value [3].

The third layer is the fully connected layer which is the initial layer of the Neural Network that transforms all data into one dimension. This transformation process is called flattening. In this layer, there is also a node that is connected and has weights and an activation function. Then, the softmax classification is used to determine the probability values of the prediction results if the input belongs to one of the categories [12]. The softmax activation can be used for classification in multiple classes. Softmax is used in the last layer or as the output layer, which functions to normalize the calculation by converting the final output layer into probability values between 0 and 1 [13].

MobileNetV2 is one of the algorithms of a mobile-based convolutional neural network that can be used to address excessive computer resource requirements. MobileNetV2 is an improvement over the MobileNet architecture. There is a difference between the Deep CNN architecture and the MobileNet architecture in the use of convolution layers. In MobileNetV2, the filter thickness is adapted to the thickness of the input image. MobileNetV2 uses depthwise convolution, pointwise convolution, linear bottleneck, and shortcut connections between bottlenecks [5]. The architecture of MobileNetV2 is represented in Table 3.

Table 3. MobileNetV2 Architecture

Input	Operator	T	c	n	S
$224^2 \times 3$	Conv2d	-	32	1	2
$112^2 \times 32$	Bottleneck	1	16	1	1
$112^2 \times 16$	Bottleneck	6	24	2	2
$56^2 \times 24$	Bottleneck	6	32	3	2
$28^2 \times 32$	Bottleneck	6	64	4	2
$14^2 \times 64$	Bottleneck	6	96	3	1
$14^2 \times 96$	Bottleneck	6	160	3	2
$7^2 \times 160$	Bottleneck	6	320	1	1
$7^2 \times 320$	Conv2 1x1	-	1280	1	1
$7^2 \times 1280$	Avgpool 7x7	-	-	1	-
$56^2 \times 24$	Conv2d 1x1	-	K	-	-

In the MobileNetV2 architecture like on Table 3, the topmost classification layer is not very useful. Instead, the last layer before the flattening operation can be used as a common practice. This layer can be referred to as the bottleneck layer. As a comparison to the topmost layer, the bottleneck layer is generally kept more often. In the MobileNetV2 model, pre-trained model weights on ImageNet are loaded using the argument "include_top=False" to remove the topmost classification layer [14]. The development and training process was carried out using Google Colab. The layer arrangement for the Deep CNN used for training can be seen on Table 4. In Table. 4, it can be seen that there are 2 Conv2D layers, which are convolutional layers. Then, there is also a pooling layer after the first and second convolutional layers. The pooling layer used is the MaxPooling layer, which will perform selection or pooling and then perform calculations on each pool, and

then select the largest value to maintain the features in the image data [15].

Table 4. Deep CNN Architecture

Layer (type)	Output Shape	Param #
conv2d_8 (Conv2D)	(None, 222, 222, 64)	1792
max_pooling2d_9 (MaxPooling2D)	(None, 111, 111, 64)	0
conv2d_9 (Conv2D)	(None, 109, 109, 64)	36928
max_pooling2d_9 (MaxPooling2D)	(None, 54, 54, 64)	0
flatten_4 (Flatten)	(None, 186624)	0
dense_9 (Dense)	(None, 256)	47776000
dense_10 (Dense)	(None, 256)	65792
dense_11 (Dense)	(None, 5)	1285

The last layer is a Full Connected Layer that starts at the Flatten layer, which functions to transform the dimension of the data into one dimension [16]. The ReLU activation is used in all convolutional layers, as well as in the first two Fully Connected Layers. The Softmax activation is used in the last Fully Connected Layer. Softmax will function to transform the value of each node that has been calculated into a probability level value that indicates the classification result of the input image data [3]. Meanwhile, the layer arrangement in the transfer learning method using MobileNetV2 can be seen on Table 5.

Table 5. Architecture MobileNetV2

Layer (type)	Output Shape	Param #
xInput_2(InputLayer)	[(None, 222, 222, 3)]	0
MobileNetV2_1.00_224(Fun	(None, 7, 7, 1280)	2257984
ctional)		
Gaussian_noise(Gaussian	(None, 7, 7, 1280)	0
Noise)		
Dropout_10(Dropout)	(None, 7, 7, 1280)	0
Global_average_pooling2d	(None, 1280)	0
(GlobalAveragePooling2D)		
dense_20(Dense)	(None, 5)	6405

Based on Table 5 creating a model using transfer learning, the first thing to do is to create a base model. In this classification, the model used is MobileNetV2. The layers in the MobileNetV2 model that will be used for feature extraction are the last layer before the flatten operation (the bottleneck layer). The MobileNetV2 model will be loaded using 3 input channels with ImageNet weights and Include_top=False will be used because the layer to be used for feature extraction is only the last layer before the bottleneck layer. Therefore, all layers in the MobileNetV2 model will be frozen to prevent certain layer weights from being updated during training [5]. Next, there is a gaussian noise layer and dropout layer that are very useful for reducing overfitting by adding noise and randomly reducing the number of neurons during training.

Then there is the GlobalAveragePooling2 layer, which is used to convert the features into a vector with a size of 1280 for an image. The last layer is a Dense layer,

which is used to make a single prediction for each image [14]. In this study, several parameters were used to train the model, such as the loss parameter, which for this study is Categorical Crossentropy used to find the error value during model training. Categorical Crossentropy is used because the dataset used for classification is considered multi-class, with 5 different disease labels. This category also affects the activation function of the last Fully Connected layer, which uses Softmax. The next parameter is the learning rate, which affects how quickly the model adapts.

In this classification, Adam optimizers and a learning rate of 0.001 and 0.1 is used. The epoch value used is 30 epochs. The epoch value indicates the number of iterations performed by the model during training. Callbacks such as early stopping and checkpoint callback are also used to prevent overfitting and underfitting, and to save the best accuracy weights during model training.

2.4 Evaluation

In the evaluation process, a Confusion Matrix is used to obtain the calculation of recall, precision, f1-score, and also accuracy values, which are used to measure how well the model performs classification on the dataset used. Then, from the calculated values of the Confusion Matrix, a comparison will be made between the two highest values to determine which model performs the best in classification.

3. Results and Discussions

3.1. Result

In data science, testing and evaluating models is very important. The most commonly used metrics for evaluating model performance in classification are confusion matrix, accuracy, precision, and recall [17]. The result of training model performance using 0.001 learning rate is represented in Table 6.

Table 6. Result Training Model Using 0.001 Learning Rate

Model	Learning Rate=0.001			
	Acc	Loss	Val Acc	Val Loss
Deep CNN	0.8223	0.4225	0.7607	0.6225
MobileNetV2	0.9151	0.2382	0.9193	0.2647

Based on Table 6 the first training process using a learning rate of 0.001 and a Deep CNN model architecture, an accuracy of 83% and a validation accuracy of 76% were achieved. Whereas using the pre-trained MobileNetV2 model was able to produce an accuracy of 91% and a validation accuracy of 91%. For the loss results using a learning rate of 0.001 in the Deep CNN model architecture, a loss value of 42% and a validation loss of 62% were obtained. While the MobileNetV2 model was able to produce a smaller loss value of 23% and a validation. The result of training

model performance using 0.001 learning rate is represented in Table 7.

Table 7. Result Training Model Using 0.0001 Learning Rate

Model	Learning Rate=0.0001			
	Acc	Loss	Val Acc	Val Loss
Deep CNN	0.8466	0.3830	0.8598	0.3707
MobileNetV2	0.8123	0.5954	0.8211	0.5933

Based on Table 7, when using a learning rate of 0.0001, the Deep CNN model architecture was able to achieve an accuracy of 84% and a validation accuracy of 85%. Meanwhile, the MobileNetV2 pre-trained model trained with a learning rate of 0.0001 was able to produce an accuracy value of 81% and a validation accuracy of 82%. Then, for the loss value, the Deep CNN model architecture trained with a learning rate of 0.0001 was able to produce a loss value of 38% and a validation loss of 37%. The MobileNetV2 model was able to produce a loss and validation loss value of 59%.

3.2 Discussion

Based on Table 5 and 6, it can be seen that the model using the pre-trained MobileNetV2 with a learning rate of 0.001 is the most optimal model. This is because the MobileNetV2 model trained with a learning rate of 0.001 can achieve the highest accuracy of 90% and validation accuracy of 89.5%, as well as producing the lowest loss value of 26.21% and a validation loss of 27.02%.

Next, testing and performance comparison of the models will be conducted using Confusion Matrix. The following is the Confusion Matrix obtained from the training results of the most optimal model, which is the model that uses MobileNetV2 with a learning rate of 0.001 and Deep CNN architecture with a learning rate of 0.001 using the available dataset. By using the Confusion Matrix, we can calculate the values to determine how well the model works. F1 Score is also a Confusion Matrix. If the value of F1 Score is high, it means that both recall and precision are also high. In other words, the model created is able to recognize the types of diseases on chili plant leaves well. To calculate the value of F1 Score, Recall, and Precision, we can use Equation 1 until 4 [18].

Precision as in Equation 1 can tell us how many true positive predictions the model made. The value of this metric can determine whether the model can be used or not.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

Recall as mention in Equation 2 shows the number of true positive cases that can be correctly predicted by the model. F1 Score gives us a combination of data between precision and recall, which means that if we try to

increase the value of precision, the recall will decrease, and vice versa [19].

$$F - Measure = 2 * \frac{precision * recall}{precision + recall} \quad (3)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

Where True Positive (TP) is the number of positive data records correctly classified as positive values, False Negative (FN) is the number of positive data records classified as negative values, False Positive (FP) is the number of negative data records classified as having positive values, True Negative (TN) is the number of negative data records correctly classified as negative values [20].

Using Equation 1 until 4 and based on the values obtained from the Confusion Matrix of the most optimal model training results, the values for each metric are as Table 8.

Table 8. Comparison of classification report based on confusion matrix

Model	Precision	Recall	F1 Score
Deep CNN(Lr=0.0001)	84%	81%	80%
MobileNetV2(Lr=0.001)	92%	92%	92%

Based on the comparison table of the Classification Report between the Deep CNN architecture trained with a learning rate of 0.0001 and MobileNetV2 with a learning rate of 0.001 on Table 7, it can be seen that the models are able to classify diseases on chili leaves well. In this comparison, it can be seen that MobileNetV2 trained with a learning rate of 0.001 has a superior value compared to the Deep CNN architecture. This is based on the values obtained when testing with Confusion Matrix, where MobileNetV2 trained with a learning rate of 0.001 is able to produce an average score of 92%, while the Deep CNN model can only produce an average score of 80%.

4. Conclusion

Deep learning method using Deep CNN (Convolutional Neural Network) algorithm and MobileNetV2 pre-trained model can classify five types of diseases on chili plant leaves. However, in this case, the model using MobileNetV2 trained with a learning rate of 0.001 obtained more optimal results. This model can achieve a fairly good accuracy of 91%. However, there are still misclassifications caused by the condition of the sample data used for training, which may be damaged or biased between one class and another. The model using CNN architecture obtained a lower result, which is 84% compared to MobileNetV2

The model using Deep CNN architecture obtained a lower result, which is 84% compared to MobileNetV2, which produced an accuracy of 92%. This may be due to the less appropriate hyperparameter selection, while MobileNetV2 can generate higher accuracy because it

uses filter thickness according to the input image used. The CNN architecture can also be improved using the Data Augmentation method, which is used to increase the number of data and also provide appropriate Hyperparameters and Learning Rates so that the model can quickly adapt to the training data and produce a fairly good accuracy.

Overall, this research can be used as a reference to know the performance comparison between deep Convolutional Neural Network and MobileNetV2 using Confusion Matrix for chili diseases classification. For the future work, it can be tested with larger number of datasets and disease type labels than before, so that this research is not only focused on leaf diseases but on the entire plant parts. Additionally, different models and methods can be used as a comparison to conduct classification and prediction in order to obtain better accuracy result.

References

- [1] P. Nuvasiyah, F. Nhita, and D. Saepudin, "Price Prediction of Chili Commodities in Bandung Regency Using Bayesian Network," *Int. J. Inf. Commun. Technol.*, vol. 4, no. 2, p. 19, 2019, doi: 10.21108/ijocit.2018.42.204
- [2] H. D. A. Hamid and Nurul Hidayat, "Diagnosis Penyakit Tanaman Cabai Menggunakan Metode Modified K- Nearest Neighbor (MKNN)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 3, no. 3, pp. 2881–2886, 2019.
- [3] A. Tsany and R. Dzaky, "Deteksi Penyakit Tanaman Cabai Menggunakan Metode Convolutional Neural Network," *e-Proceeding Eng.*, vol. 8, no. 2, pp. 3039–3055, 2021.
- [4] A. Jinan, B. H. Hayadi, and U. P. Utama, "Journal of Computer and Engineering Science Volume 1, Nomor 2, April 2022 Klasifikasi Penyakit Tanaman Padi Menggunakan Metode Convolutional Neural Network Melalui Citra Daun (Multilayer Perceptron)," vol. 1, no. April, pp. 37–44, 2022.
- [5] N. Hikmatia A.E and M. Ihsan Zul, "Aplikasi Penerjemah Bahasa Isyarat Indonesia menjadi Suara berbasis Android menggunakan Tensorflow," *J. Komput. Terap.*, vol. 7, no. Vol. 7 No. 1 (2021), pp. 74–83, 2021, doi: 10.35143/jkt.v7i1.4629.
- [6] F. D. Adhinata, N. A. F. Tanjung, W. Widayat, G. R. Pasfica, and F. R. Satura, "Comparative Study of VGG16 and MobileNetV2 for Masked Face Recognition," *J. Ilm. Tek. Elektro Komput. dan Inform.*, vol. 7, no. 2, p. 230, 2021, doi: 10.26555/jiteki.v7i2.20758.
- [7] W. Aceh and A. Dwi, "Inventarisasi Penyakit Pada Tanaman Cabai Merah (Capsicum annum L.) di Kebun Warga Gampong Suak Raya Kecamatan Johan Pahlawan Kabupaten Aceh Barat Inventory," vol. 8, no. 1, pp. 70–75, 2022, doi: https://doi.org/10.35308/jal.v8i1.5293.
- [8] Z. Ulinnuha and R. N. K. Syarifa, "Insidensi penyakit daun keriting kuning beberapa varietas cabai pada berbagai tingkat toleransi terhadap intensitas cahaya rendah," *AGROSCRIPT J. Appl. Agric. Sci.*, vol. 3, no. 2, pp. 78–89, 2021.
- [9] D. Tania, "Mengenal Gejala Kurangnya Unsur Hara pada Cabai," 2021. https://www.neurafarm.com/blog/InfoTania/Budidaya Tanaman/mengenal-gejala-kurangnya-unsur-hara-pada-cabai (accessed Oct. 08, 2022).
- [10] K. Wisaeng and W. Sa-Ngiamvibool, "Detection of plant leaf diseases using K-mean++ intermeans thresholding algorithm," *J. Comput. Sci.*, vol. 16, no. 9, pp. 1237–1249, 2020, doi: 10.3844/jcssp.2020.1237.1249.
- [11] L. Alzubaidi et al., *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*, vol.

- 8, no. 1. Springer International Publishing, 2021. doi: 10.1186/s40537-021-00444-8.
- [12] W. Gong, H. Chen, Z. Zhang, M. Zhang, and H. Gao, "A Data-Driven-Based Fault Diagnosis Approach for Electrical Power DC-DC Inverter by Using Modified Convolutional Neural Network with Global Average Pooling and 2-D Feature Image," *IEEE Access*, vol. 8, pp. 73677–73697, 2020, doi: 10.1109/ACCESS.2020.2988323.
- [13] A. Kholik, "KLASIFIKASI MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN) PADA TANGKAPAN LAYAR HALAMAN INSTAGRAM," vol. 2, no. 2, pp. 10–20, 2021.
- [14] N. Gupta, "A Pre-Trained Vs Fine-Tuning Methodology in Transfer Learning," *J. Phys. Conf. Ser.*, vol. 1947, no. 1, 2021, doi: 10.1088/1742-6596/1947/1/012028.
- [15] R. Munarto and A. Darma, "Klasifikasi Gender dan Usia Berdasarkan Citra Wajah Manusia Menggunakan Convolutional Neural Network," *Setrum Sist. Kendali-Tenaga-elektronika-telekomunikasi-komputer*, vol. 10, no. 2, pp. 30–43, 2021, doi: 10.36055/setrum.v10i2.12991.
- [16] O. Stephen, M. Sain, U. J. Maduh, and D. U. Jeong, "An Efficient Deep Learning Approach to Pneumonia Classification in Healthcare," *J. Healthc. Eng.*, vol. 2019, 2019, doi: 10.1155/2019/4180949.
- [17] G. Sambasivam and G. D. Opiyo, "A predictive machine learning application in agriculture: Cassava disease detection and classification with imbalanced dataset using convolutional neural networks," *Egypt. Informatics J.*, vol. 22, no. 1, pp. 27–34, 2021, doi: 10.1016/j.eij.2020.02.007.
- [18] J. Browniee, "How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification," 2020. <https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/#:~:text=Wecan calculate the recall,Recall %3D 90 %2F 100> (accessed Oct. 10, 2022).
- [19] A. Tasnim, M. Saiduzzaman, M. A. Rahman, J. Akhter, and A. S. M. M. Rahaman, "Performance Evaluation of Multiple Classifiers for Predicting Fake News," *J. Comput. Commun.*, vol. 10, no. 09, pp. 1–21, 2022, doi: 10.4236/jcc.2022.109001.
- [20] A. Kulkarni, D. Chong, and F. A. Batarseh, *Foundations of data imbalance and solutions for a data democracy*. Elsevier Inc., 2020. doi: 10.1016/B978-0-12-818366-3.00005-8.