Convolutional Neural Network Optimization for Disease Classification Tomato Plants Through Leaf Image

Muhammad Gibran
Department of Informatics
Diponegoro Universty
Semarang Indonesia
gibranm@students.undip.ac.id

Adi Wibowo
Department of Informatics
Diponegoro Universty
Semarang Indonesia
bowo.adi@live.undip.ac.id

2021 5th International Conference on Informatics and Computational Sciences (ICICoS) | 978-1-6654-3807-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICICOS53627.2021.9651893

***Abstract*— Pest and disease control is one way of cultivation to increase tomato production. One way to control pests and diseases is to identify early types and types of diseases in tomato plants. Diseases and pests that attack tomato plants can be observed through the leaves. The deep learning method can be used to classify the types of tomato plant diseases through the image of the leaves. Convolutional Neural Network (CNN) is a deep learning method that is often used to classify images. The CNN model in this study requires optimization in the form of dataset optimization and hyperparameter tuning optimization. Dataset optimization uses data augmentation to overcome imbalanced dataset problems, while hyperparamter tuning optimization uses learning rate, dropout, and number of convolution layers to overcome overfitting problems. The results of this study resulted in an increase in the accuracy value after using optimization adjustments. Previously, the accuracy of the model only reached 80%. After adjusting the dataset optimization using data augmentation and hyperparameter tuning adjustments with a dropout value of 0.9, a learning rate of 0.0001, and a convolution layer of 6, the accuracy increased to 94%.**

Keywords*— *Tomato Disease, Tomato Leaves, Deep Learning, Convolutional Neural Network

I. PRELIMINARY

Tomatoes are widely used for food and beverage processing and even the cosmetic industry [1]. To overcome this, it is necessary to increase tomato production. One of the efforts to increase tomato production is pest and disease control [2]. Diseases and pests that attack tomato plants can be observed through the leaves. The many types of diseases in tomato plants plus the similarity of symptoms between diseases that can be observed through the tomato leaves make it difficult for tomato growers or ordinary farmers to classify and provide early treatment for these tomato plant diseases [3]. By utilizing existing technological advances, these problems can be overcome easily.

Advances in artificial intelligence research now make it possible to make automatic detection of plant diseases from images. Research on the classification of diseases in tomato plants has been done previously by Sabrol et al [4]. They used a machine learning classification tree algorithm to classify 6 tomato plant diseases. Classification tree is one of the machine learning algorithms which can get good accuracy, but the algorithm still has poor generalization ability when the data set is large or unclear, thus affecting the accuracy value [5]. Currently, the method that is often used to detect and classify is deep learning. Deep Learning is an approach that can learn several feature layers hierarchically and form a feature representation of the input data so that there is no need to define the characteristics and segmentation process on the leaf [6]. The deep learning method has several architectures, one of which is Convolutional Neural Networks (CNN).

In this research the author will conduct research on optimization of convolutional neural network for the classification of tomato plant diseases through leaf imagery. The data obtained came from Kaggle which consists of 10 classes of disease labels. The data will be preprocessed using image augmentation and resizing before being entered into the CNN model for CNN training and testing. After that, an evaluation model will be carried out to measure the level of accuracy.

II. RESEARCH METHODS

In the following chapters, information about step by step in classifying diseases in tomato plants is explained through imagery of tomato leaves using CNN. The stages carried out begin with data collection and end with model evaluation.

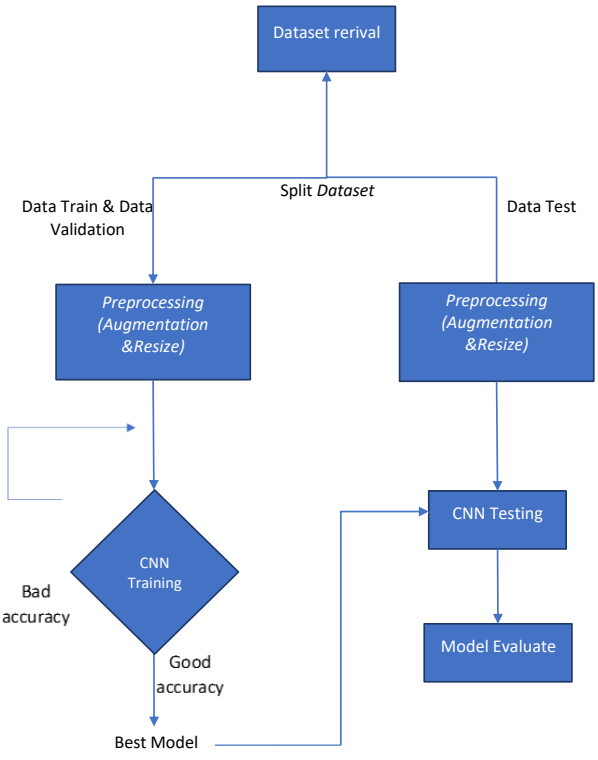

Fig 1. Research methods outline

Figure 1 describes the stages of the research method. The stage in this research begins with the data collection process. Then the data that has been obtained will be divided into 2 parts, namely data train & validation with test data with a size of 80% for training & validation data, and 20% for data testing, then both data will be preprocessed with data augmentation and image resizing. . The test & validation data will then be used for CNN model training, after the training, the results will be observed in the form of the accuracy of the model, if the accuracy is not good then hyperparameter adjustments will be made, and retraining will be carried out, if the accuracy is good then the model will be used as the best model and proceed to the CNN testing stage, and finally the model evaluation will be carried out.

A. Dataset Retrieval

The data used in this study was obtained through kaggle, *https://www.kaggle.com/kaustubhb999/tomatoleaf*. The data consists of 10 classes/labels, namely 1 healthy label and 9 disease labels with details of Bacterial spot, Black leaf mold, Gray leaf spot, Healthy, Late blight, and Powdery mildew. The number of original data amounted to 14,529 consisting of various images of tomato leaves which can be seen in table 1.

TABLE I. DATA DISTRIBUTION

No	Class/label	Data testing	Data training	
			training	validation
1	*Bacterial Spot*	341	1089	272
2	*Early Blight*	160	512	128
3	*Healthy*	255	815	203
4	*Late Blight*	306	977	244
5	*Leaf Mold*	284	907	226
6	*Septoria Leaf Spot*	284	858	214
7	*Spider Mites*	269	719	179
8	*Target Spot*	225	487	121
9	*Mosaic Virus*	60	192	47
10	*Yellow Leaf Curl*	858	2743	685

In table 1, From the 14,529 data, Of the 14,529 data, the data will be divided into 2 parts, namely the testing section and the training section for train data, with a ratio of 80% and 20%. the testing data section will be further divided into 2 parts, namely the testing data section and the validation data section, with a ratio of 20% and 80%.

Each dataset has a label as its class, which can be seen in Figure 2 The size of each data is still different. Therefore, a preprocessing technique is needed to be included in the classification model.

B. Preprocessing

Preprocessing is a form of dataset processing before entering into the classification model process. The datasets available for this study still require data alignment before entering the next stage. In this research, image preprocessing is done, namely image augmentation and image resizing. Because the amount of data distribution in each class is not balanced, a data augmentation process is needed. Next is the alignment of the size of each image that has been augmented to be included in the CNN model. The segmented image is resized to 128x128 pixels for all images. Thus, images that have gone through preprocessing can be included in the classification model for CNN model training.

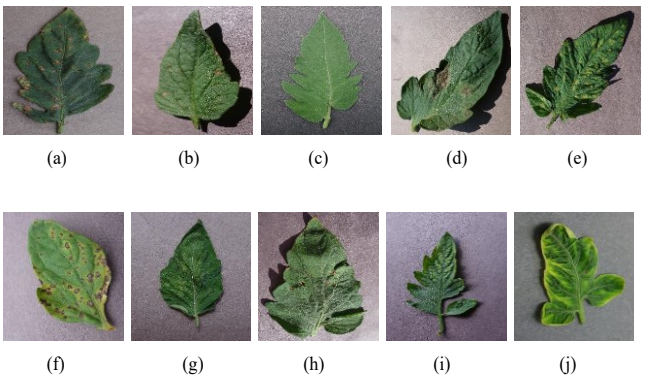

Fig 2. Tomato leaf image data (a) bacterial spot, (b) early blight, (c) healthy, (d) late blight, (e) lead mold, , (f) septoria lead mold, (g) spider miter, (h) target spot, (i) mosaic virus, (j) yellow leaf curl

- *Image Augmentation*
  Augmentation that is done is using a random 90° rotation technique, vertical flip, randomly adjusting brightness, and using the clahe method. library albumentations used in using the clahe method in this research. after using data augmentation it increased from 14,526 to 42,860/ The amount of data distribution after using the augmentation technique can be seen in table 2.
- *Resize Image Size*
  Image resizing or resizing is done after the data augmentation process. The size of the previous augmented image is still different and to align it in this study all augmented images are resized to 128x128 px (pixels). Figure 4 is an example of resize image size process.

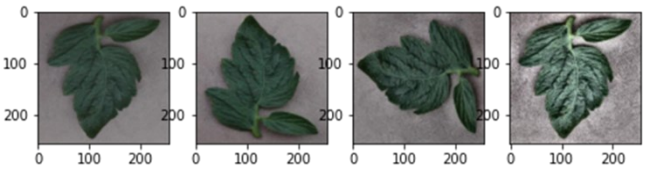

Fig 3. Result of vertical flip, random brightness, rotate 90, CLAHE

TABLE II. DATASET AFTER AUGMENTED

No	Class/label	Testing Data	Training Data	
			training	validation
1	*Bacterial Spot*	858	2743	685
2	*Early Blight*	858	2743	685
3	*Healthy*	858	2743	685
4	*Late Blight*	858	2743	685
5	*Leaf Mold*	858	2743	685
6	*Septoria Leaf Spot*	858	2743	685
7	*Spider Mites*	858	2743	685
8	*Target Spot*	858	2743	685
9	*Mosaic Virus*	858	2743	685
10	*Yellow Leaf Curl*	858	2743	685

Fig 4. Example of resize image size process

C. CNN Training

At this stage, model training will be carried out on the CNN model that has been created. The model consists of convolution, max pooling, batch normalization, dropout, flatten and dense layers, which will be explained as follows:

- Convolution

This layer is the core of the CNN architecture. This layer serves to study and present the features of the input image. In the convolution layer there is a kernel or can also be called a filter in the form of a matrix which will be processed with the input image represented by a matrix so that a feature is obtained, this is called the operation on the convolution layer [8].

- Max Pooling

The main idea of pooling is to reduce complexity at the next layer. The pooling process can also be considered similar to resolution reduction. The pooling operation will shrink the feature map, while maintaining the most dominant feature. Max-pooling is one of the most commonly used types of pooling methods [8].

- Batch normalization

Batch Normalization serves to solve the problem of covariate shift, namely changes in the distribution of inputs into the inner layer of the network by improving the mean and variance of the inputs [8].

- Dropout

Dropout is a technique that is often used on CNN which aims to prevent overfitting. The Dropout process randomly disables some neurons from the neural network layer. By disabling the units (neurons) the neural network model will try to distribute the features evenly to all active neurons and force the model to learn several independent features [9].

- Dense

Dense or neural network layer has a hidden layer, output and input. This layer has the same working principle as multilayer perceptron where each unit is connected to other units in the layer.

- Softmax

softmax is an activation function that converts the value in the output layer into a probability value to represent the magnitude of the probability distribution for a number of classes [9].

- Relu

ReLu is an activation function which in practice can solve the problem of saturation and missing gradients [7].

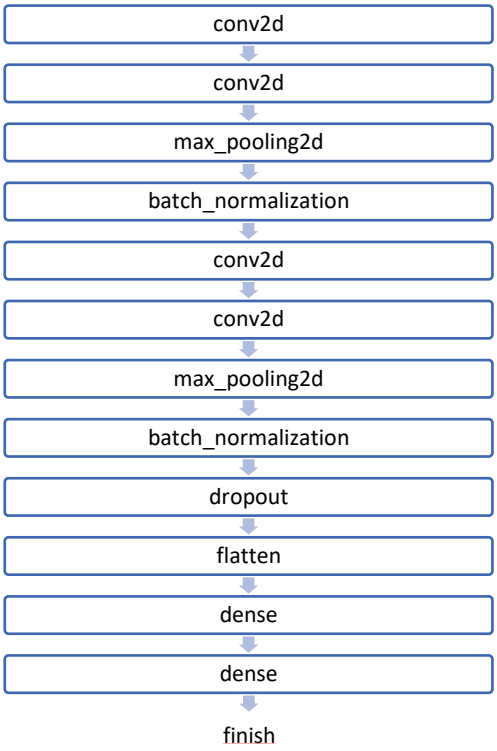

Fig 5. CNN model architecture

The flowchart in Figure 5 represents illustrates the process of the CNN model that has been created. Before starting the training, the weight and bias initialization process will be carried out.

- Initialization of Weights & Bias

The process of initializing weights and biases is carried out as parameters that can be trained in CNN training. The weights will be initialized to the convolution layer which is commonly referred to as the filler, then to the fully connected layer. In this training the weight initialization process will use the Glorot uniform initialization technique, or what is commonly called the Xavier uniform initializer, while the initialization process can use zero initialization.

- Feedforward phase

In the feedforward phase, several processes will be carried out starting from convolution as feature extraction to the neural network process to determine the classification of an image input in accordance with the CNN model architecture that has been determined in Figure 5.

- Backpropagation phase

In the backpropagation phase, an update on the weights and biases will be carried out to determine the minimum error. However, to facilitate the calculation, this test is carried out with a maximum epoch of 1, so this test will only perform one update on the weights and biases. Before updating the weights at the fully connected layer, first to calculate the error at each neural network layer, then only then can you update the weight value for each neural network layer.

D. CNN Testing

Step 1: Input the image as input in the form of the matrix to be tested, as follows:

Fig 6. Example of input image

If it is transformed into a matrix form it will be like this:

$$Image = \begin{bmatrix} 108 & 103 & 118 & \cdots & 111 \\ 124 & 119 & 134 & \cdots & 127 \\ 107 & 102 & 117 & \cdots & 124 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 150 & 146 & 158 & \cdots & 159 \end{bmatrix} \quad (1)$$

Step 2: Perform feedforward and backpropagation on the model with input data. Until the step of calculating softmax activation. The trained model is used for the following tests. All parameters have been updated. The output of this testing process is as follows:

$$[1 \quad 0 \quad ... \quad 0]$$

The class value in the bacterial spot class is 1, because the result of the softmax activation function is more than 0.5. while in the other classes the value is 0, because the result of the softmax activation function is less than 0.5. With the highest value being in class 1, namely bacterial spot, the model classifies the image into the bacterial spot class. Tests are carried out on all data in the test dataset and then the accuracy is calculated.

E. Model Evaluation

To calculate the accuracy of a CNN model, it is necessary to calculate the accuracy in each class, in the sense that it is necessary to calculate the accuracy from class 1 to 10, to calculate the accuracy in class and then calculates the accuracy value in class 2 and so on until class 10, then to calculate the accuracy of the CNN model is by calculating the average accuracy value of each class, resulting in confusion matrix table 3.3 obtained the CNN model accuracy value is 80%.

III. Result & Discussion

The test and the results of this research consist of several test scenarios consists of several test scenarios. Scenario 1 aims to compare the accuracy values obtained by the CNN model using pre-processing augmentation techniques and not using pre-processing techniques. Scenario 2 aims to find the best architecture by finding the best dropout value and learning rate. Scenario 3 aims to find the number of convolution layers to find the best CNN architecture.

A. Scenario 1

Scenario 1 aims to compare the accuracy values obtained by the CNN model using augmentation pre-processing techniques and not using pre-processing techniques. This scenario uses a dropout value of 0.5, a learning rate value of 0.01, and 4 convolution layers. The testing process is carried out using the best epoch with a max epoch value of 100 and the variable patience 10. The CNN model architecture used in scenario 1 is shown in table 3.

TABLE III. CNN Model Architecture

Layer	Type	Output Shape	Parameter
Layer1	*Conv1*	(126, 126, 32)	896
Layer2	*Conv2*	(124, 124, 32)	9248
Layer3	*Pool*	(62, 62, 32)	0
Layer4	*Batch_norm*	(62, 62, 32)	128
Layer5	*Conv3*	(60, 60, 32)	9248
Layer6	*Conv4*	(58, 58, 32)	9248
Layer7	*Pool*	(29, 29, 32)	0
Layer8	*Batch_norm*	(29, 29, 32)	128
Layer9	*Dropout*	(29, 29, 32)	0
Layer10	*Flatten*	26912	0
Layer11	*Dense*	64	1722432
Layer12	*Dense*	10	390

B. Discussion of scenario 1

Scenario 1 aims to compare the accuracy values obtained by the CNN model using augmentation preprocessing techniques and not using preprocessing techniques. Some of the parameter values used are the dropout value of 0.5, the learning rate value of 0.01, and 4 convolution layers. In this scenario, the best epoch is used with a max epoch value of 200 and a patience variable of 10. The CNN model that does not use preprocessing techniques gets an accuracy value of testing data of 80%, while the CNN model that uses preprocessing techniques gets an accuracy value of 87%. The graphs of the accuracy model and the loss model can be seen in Figures 7 and 8. The results of scenario 1 are shown by the confusion matrix in table 4 and table 5.

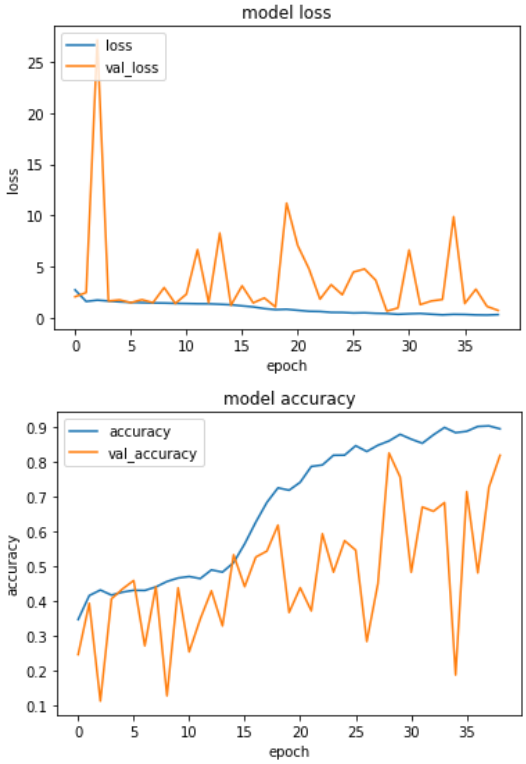

Fig 7. Model accuracy and loss without using augmentation techniques

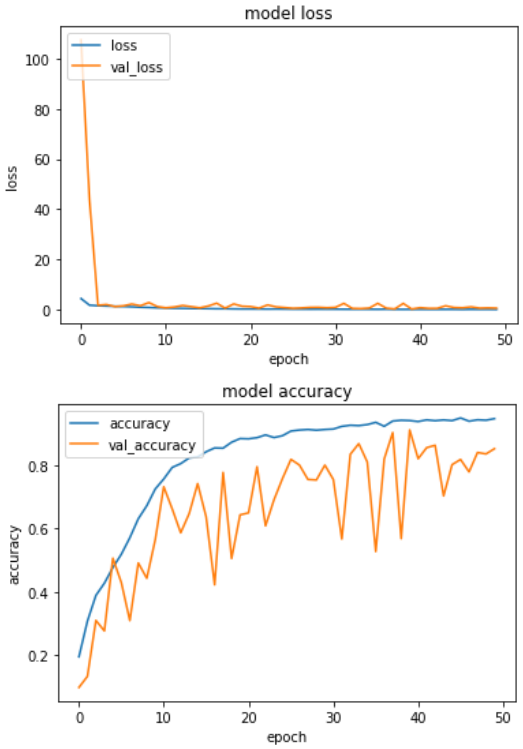

Fig 8. Accuracy and loss models using augmentation techniques

The confusion matrix in table 4 displays the results of the confusion matrix from the CNN model without using pre-processing augmentation techniques on data testing. From these results obtained an accuracy value of 80%. The graph of model accuracy and model loss can be seen in Figure 7. From the graph, it can be seen that the model is still experiencing overfitting. Hence the problem by using the augmentation technique.

The confusion matrix in table 5 displays the results of the confusion matrix from the CNN model without using pre-processing augmentation techniques on data testing. From these results obtained an accuracy value of 87%. The graph of model accuracy and model loss can be seen in Figure 8. From the graph, it can be seen that the model is still experiencing overfitting. Then, this problem will be solved in scenario 2.

C. *Scenario 2*

Scenario 2 is carried out to find the dropout value and learning rate that produces the best CNN architecture model performance and overcomes the problem of overfitting. The test is carried out using several dropout values and learning rates. The accuracy value is used as a benchmark for model performance.

D. *Discussion of Scenario 2*

Scenario 2 testing is done by changing the dropout and learning rate values in the model. In scenario 2, several dropout values and learning rates were tested. The values used for dropout are 0.01, 0.1, 0.5, 0.8, 0.9, and 0.95. Meanwhile, the learning rate values are 0.01, 0.001, and 0.001. The result of testing this scenario is shown in table 4. The results of scenario 2 in table 6 can be seen that the dropout value is 0.9. in this scenario, it proves that adding the dropout value will add the accuracy value to the maximum point, which is 92%. However, at the dropout value of 0.95, the accuracy value decreased to 91%.

TABLE IV. RESULT OF SCENARIO 2

No	*Dropout*	*Learning Rate*	*Accuracy*
1	0.01	0.01	77%
2	0.01	0.001	83%
3	0.01	0.0001	85%
4	0.1	0.01	79%
5	0.1	0.001	87%
6	0.1	0.0001	86%
7	0.5	0.01	87%
8	0.5	0.001	87%
9	0.5	0.0001	88%
10	0.8	0.01	85%
11	0.8	0.001	88%
12	0.8	0.0001	91%
13	0.9	0.01	83%
14	0.9	0.001	88%
15	0.9	0.0001	92%
16	0.95	0.01	91%
17	0.95	0.001	91%
18	0.95	0.0001	91%

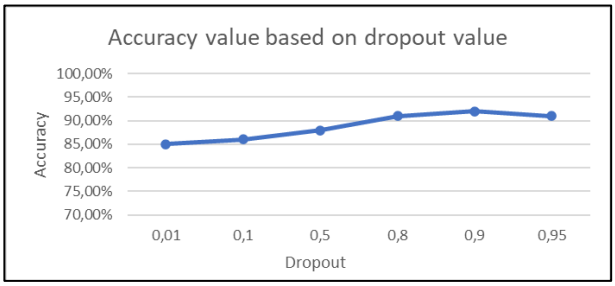

Fig 9. Effect of dropout value on accuracy

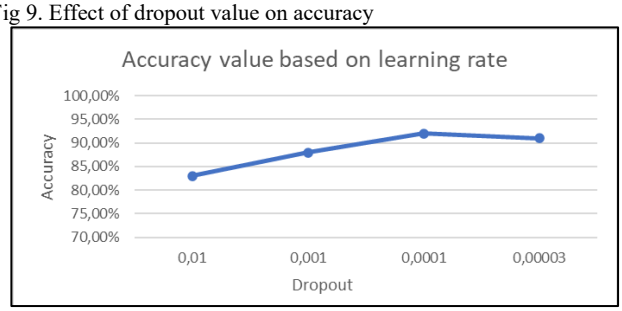

Fig 10. Effect of learning rate value on accuracy

Figure 11 shows the graph of the accuracy model and the loss model from the results of scenario 2, from the graph it looks better than the previous scenario and does not look overfitting. Next, it will be continued in scenario 3 to find the best number of convolutions to be able to increase the accuracy value of this CNN model architecture.

E. *Scenario 3*

Scenario 3 aims to find the number of convolution layers that produce the best model performance and overcome the problem of overfitting. The test is carried out by adding a convolution layer.

F. *Discussion of Scenario 3*

In scenario 3, models with several convolution layers are tested. The learning rate value of 0.9 and the dropout value of 0.0001 were used in this scenario testing. The number of convolution layers tested in this scenario are 4, 6 and 8 layers. The accuracy results from scenario 3 can be seen in table 7. The results of scenario 3 from table 10 can be seen that the large number of convolution layers does not always increase the accuracy value, because the shape of the data can fade

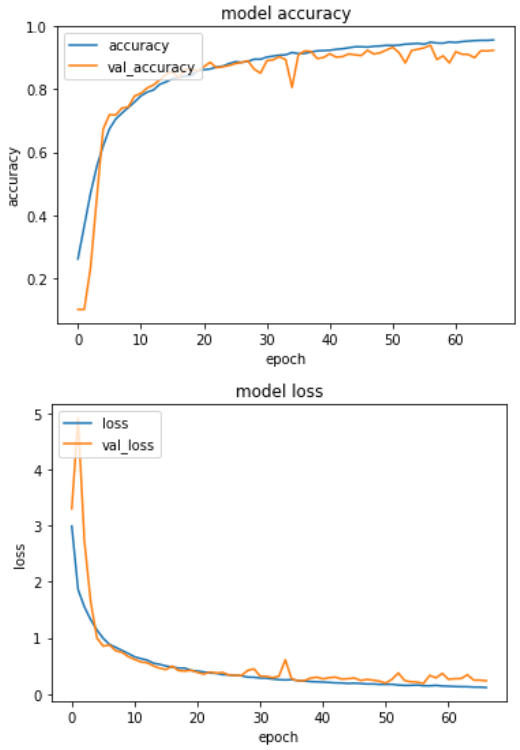

Fig 11. CNN model accuracy and loss with a learning rate of 0.0001 and a dropout of 0.9

TABLE I. RESULT OF SCENARIO 3

No	Number of Convolution Layers	Accuracy
1	4	92%
2	6	94%
2	8	88%

G. Scenario 3

Scenario 3 aims to find the number of convolution layers that produce the best model performance and overcome the problem of overfitting. The test is carried out by adding a convolution layer.

H. Discussion of Scenario 3

In scenario 3, models with several convolution layers are tested. The learning rate value of 0.9 and the dropout value of 0.0001 were used in this scenario testing. The number of convolution layers tested in this scenario are 4, 6 and 8 layers. The accuracy results from scenario 3 can be seen in table 7. The results of scenario 3 from table 1 can be seen that the large number of convolution layers does not always increase the accuracy value, because the shape of the data can fade. The shape of the faded data makes it difficult for the model to distinguish between disease classes, resulting in misclassification of diseases in tomato plants.

IV. CONCLUSION

Based on the experiments that have been carried out, the conclusion that can be drawn is that the CNN model using data augmentation has a better accuracy value, which is 87%, compared to without using data augmentation which achieves an accuracy of only 80%.

The dropout value of 0.9 and the learning rate of 0.0001 can increase the accuracy of the CNN model with an accuracy value of 92%, and can overcome overfitting. Increasing the dropout value to 0.95 with a learning rate of 0.0001 reduces the accuracy to 91%. Reducing the learning rate value to 0.00003 with a dropout value of 0.9 reduces the accuracy to 91%. Adding the convolution layer to 6 layers can increase the model accuracy to 94%, and can overcome overfitting. Increasing the convolution layer to 8 layers reduces the accuracy to 88%.

REFERENCES

[1] M. Wijaya, "Medium Tanijoy," 30 November 2020. [Online]. Available: https://medium.com/tanijoy/sederet-fakta-komoditas-tomat-di-indonesia-a121ee4b703a.

[2] Fadel, R. Yusuf and Syakur, "Pertumbuhan dan Hasil Tanaman Tomat (Lycopersicum esculentum Mill.) Pada Pemberian Berbagai Jenis Mulsa," *Agrotekbis,* no. Universitas Tadulako, Palu, pp. 152-160, 2017.

[3] J. Shijie, J. Peiyi, Siping, Hui and L. Haibo, "Automatic Detection of Tomato Diseases and Pests," *IEEE,* no. 17, pp. 3507-3510, 2017.

[4] H. Sabrol and K. Satish, "Tomato Plant Disease Classification in Digital," in *International Conference on Communication and Signal Processing*, Chandigarh, 2016.

[5] D. Jiang, D. Li, Y. Yang and S. Yu, "A Tomato Leaf Diseases Classification Method Based on Deep Learning," in *2020 Chinese Control And Decision Conference*, Hefei, 2020.

[6] K. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture ,* pp. 311-318, 2018.

[7] Kouretas and V. Paliouras, "Simplified Hardware Implementation of the Softmax Activation Function.," in *International Conference on Modern Circuits and Systems Technologies (MOCAST).*, Thessaloniki.

[8] A. Ghosh, F. S. Sultana and A. Chakrabarti, "Fundemantal Concepts of Convolutional Neural Network.," *Recent Trends and Advances in Artificial Intelligence and Internet of Things,* pp. 519-567, 2020.

[9] S. Ioffe and C. Szegedy, "Batch Normaluzation: Accelerating Deep Network Training by Reducing Internal Covariate Shift.," 2 March 2015. [Online]. Available: https://arxiv.org/abs/1502.03167. [Accessed 13 December 2020].