IDENTIFICATION OF BACTERIAL SPOT DISEASES ON PAPRIKA LEAVES USING CNN AND TRANSFER LEARNING

M. Ilhamsyah[1*)]; Ultach Enri[2]

Teknik Informatika[1,2]
Universitas Singaperbangsa Karawang[1,2]
https://unsika.ac.id
muhammad.ilhamsyah18009@student.unsika.ac.id[1*], ultach@staff.unsika.ac.id[2],

(*) Corresponding Author

Abstract— Paprika, often called bell peppers, is a plant with the Latin name Capsicum annuum var. gross. Paprika in Indonesia has a high selling value, so the opportunity for cultivating the paprika plant itself is enormous. However, the cultivation of this plant cannot be separated from the threat of disease that can affect the yield of paprika. Bacterial spot is one of them, and it is a disease that is very dangerous for paprika plants because the disease infects all parts of the plant. In this case, early detection is needed to carry out appropriate treatment to minimize the effects caused by bacterial spots. Detection of bacterial spots on paprika can be done by direct observation or conducting laboratory tests, but this requires people who have the appropriate knowledge and experience. Based on the above problems, the identification system can be an option in identifying bacterial spot disease in paprika. This research chose the Convolutional Neural Network (CNN) algorithm in the identification system. Because CNN is one of the algorithms that can receive input in the form of an image which is very suitable for the case of bacterial spots on peppers, this research dataset is divided into healthy leaves and leaves infected with bacterial spots. In this study, the implementation of CNN with transfer learning obtained results from a test accuracy of 90%, training accuracy 97% with a loss of 8.5%, validation accuracy of 97.5% with a loss of 6.9%.

Keywords: Deep Learning, Transfer learning, Classification, Paprika, Bell Peppers.

Abstract— *Tanaman paprika adalah tanaman dengan nama latin Capsium annuum var. grossum. Paprika di Indonesia memiliki nilai jual yang tinggi sehingga peluang budidaya tanaman paprika sendiri sangat besar. Namun dalam budidaya tanaman ini tidak lepas dari ancaman penyakit yang dapat mempengaruhi hasil panen paprika, bacterial spot merupakan salah satu penyakit yang sangat berbahaya bagi tanaman paprika sebab penyakit menginfeksi seluruh bagian tanaman. Dalam hal ini perlu deteksi dini untuk melakukan penanganan yang tepat sehingga efek yang ditimbulkan oleh bacterial spot dapat diminimalisir. Untuk deteksi bacterial spot pada paprika bisa dilakukan dengan pengamatan langsung ataupun dengan melakukan uji laboratorium, namun hal ini memerlukan orang yang memiliki kemampuan serta pengalaman yang baik. Berdasarkan permasalahan diatas sistem identifikasi dapat menjadi opsi dalam melakukan identifikasi penyakit bacterial spot pada paprika. Dalam sistem identifikasi, algoritma Convolutional Neural Network (CNN) dipilih sebab CNN adalah salah satu algoritma yang dapat menerima input berupa gambar yang mana sangat cocok pada kasus bacterial spot pada paprika, dalam dataset penelitian ini dibagi menjadi daun sehat dan daun terinfeksi bacterial spot. Pada penelitian ini, implementasi CNN dengan transfer learning mendapatkan hasil dari akurasi test sebesar 90%, akurasi training 97% dengan loss 8.5%, akurasi validasi 97,5% dengan loss 6,9%.*

Kata Kunci: *Deep Learning, Transfer learning, klasifikasi, Paprika, Bell Peppers.*

INTRODUCTION

Paprika is a tropical and subtropical plant. A plant with the Latin name Capsicum annuum var. gross can grow in areas with temperatures ranging from 21-27 C. In Indonesia alone this plant has been cultivated in various areas including: West Bandung, Cianjur, Bogor, Garut, Wonosobo, Batu City, Bali, West Nusa Tenggara, to Bantaeng in South Sulawesi (Reza et al., 2021).

Paprika is a type of chili. To be more precise, it is a sweet chili originating from America. Besides that, paprika is also a valuable commodity, and it can be seen from the production of paprika which has increased in the last five years. One of the places where it is cultivated is Pasirlangu Village, a paprika-producing area (Suminar et al., 2019). Based on data from Badan Pusat Statistik in 2016, the production of paprika in Indonesia reached 5,256 tons and increased in 2020 to 17,822 tons (Badan Pusat

PILAR Nusa Mandiri: Journal of Computing and Information System
Vol. 18 No. 1 March 2022 | DOI: 10.33480/pilar.v18i1.2755

Statistik Indonesia, 2020). This figure is likely to continue to grow in line with population growth and the growth of the culinary industry in Indonesia.

Globally, according to data from the Food and Agriculture Organization of the United Nations (FAO) in 2020, Indonesia ranks 3rd for chili commodities, including bell peppers, after China, Mexico, and Turkey (Food and Agriculture Organization of the United Nations Statistics, 2020). From 2000 until 2018, world consumption of paprika continued to increase. In 2017 as much as 70% of paprika was produced in Asia (Biswas et al., 2018).

Because it is still in the same genus as chili, paprika risks the same diseases as chili. One of them is a Bacterial Spot. This disease is destructive and attacks chili and tomato plants, including bell peppers, especially in warm and humid conditions (Bogatzevska et al., 2021).

The bacterium Xanthomonas campestris pv. vesicatoria causes bacterial spots. This bacterial infection can cause damage to plants and the resulting fruit. The Symptoms that appear if infected in spots can be found on almost all plant parts, including leaves and fruit (Rahman & Miah, 2020) .

Recognizing this disease can be seen directly or through laboratory testing. But of course, if you see directly, you must have sufficient knowledge about this disease to recognize it. However, with increasingly rapid technological developments, accompanied by the emergence of innovations that can help humans, one of which is digital image processing. The utilization of Image Preprocessing can help agricultural business actors to identify abnormal plants. Image Preprocessing was used to identify Bacterial Spot disease on Paprika Leaves in this study.

This research aims to create a machine learning model to identify Bacterial Spot disease on bell peppers through leaf imagery. The dataset used is the "New Plant Diseases Dataset" dataset from Kaggle. The paprika data is divided into two parts, namely healthy and infected with bacterial spots. This research will use the Transfer Learning method on the Convolutional Neural Network (CNN), one part of deep learning.

Previously there have been several studies using CNN, one of which is Disease Detection on Potato Leaves Using Image Processing with the Convolutional Neural Network Method by Abdul Jalil Rozaqi (Rozaqi et al., 2021).

In his research in January 2021, Abdul Jalil Rozaqi used the Convolutional Neural Network (CNN) to identify diseases in potato leaves. The dataset used is divided into normal, late blight, and early blight. And the total data used in this study was 1152 image data, then divided into data training 80%and testing 20%. The layers used are the Convolutional, pooling, flatten, and dense layers. With the input in the form of a shape with dimensions of 150 x 150. The result of this study is a model with accuracy training of 95% and validation accuracy of 94% (Rozaqi et al., 2021).

Sarirotul Divineyah, in his research in 2018 titled "Implementation of Deep Learning in Identification of Plant Types Based on Leaf Image Using Convolutional Neural Network." He has created an Artificial Neural Network using architecture Convolutional Neural Network (CNN) to identify plant species through leaf imagery (Ilahiyah & Nilogiri, 2018).

The dataset used has 2000 total leaf images classified into 20 classes. The layers used are pooling, ReLu, and softmax, using the alexnet architecture that Krizhevsky introduced in 2012. The input shape for this architecture is 227 x 227. and the results of this study, the model has a training accuracy of 85% and a testing accuracy of 90% (Ilahiyah & Nilogiri, 2018).

Both studies used artificial neural networks for image identification, but the architecture has different layers because the input shape is required. Based on these two studies, this study will create an artificial neural network using the Convolutional Neural Network (CNN) architecture. The architecture is used to build a model to identify paprika leaves infected by Bacterial Spot disease and the healthy ones. As a form of further development, this research will also apply the transfer learning model in the CNN architecture to help optimize the model's accuracy.

MATERIALS AND METHODS

This research method uses Convolutional Neural Network (CNN) to apply deep learning using a dataset already available from the Kaggle platform called "*New Plant Diseases Dataset*." CNN is Chosen because it is a method that is quite often used in classification cases. The workings of CNN imitate the workings of the human brain, starting input to output in one direction (Umri & Delica, 2021). CNN is divided into convolution, pooling, and fully connected to the output.

This study also uses an approach methodology of transfer learning to help improve the model performance. The architecture is CNN modified using a pre-trained model in the first layer.

The pre-trained model used for the first layer is the VGG16 model. Karen Simonyan and Andrew Zisserman introduced VGG16 in their article published at Oxford University in 2014 entitled "Very Deep Convolutional Networks for Large-Scale Image Recognition." There is explained that VGG16

was the winner in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2014, which was developed to improvise the previous model, Alexnet (Simonyan & Zisserman, 2018).

VGG16 architecture is composed of 16 layers with an input shape of 224x224x3. And in general, VGG16 consists of 4 layers: Convolution + ReLu, Max Pooling, Fully Connected + ReLu, Softmax (Rezende et al., 2018).

Next is the architectural schema of the implemented model :

Table 1. Implemented CNN + VGG16 Model Architecture

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14714688
conv2d (Conv2D)	(None, 5, 5, 128)	589952
max_pooling2d (MaxPooling2D)	(None, 2, 2, 128)	0
flatten (Flatten)	(None, 512)	0
dense (Dense)	(None, 256)	131328
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 2)	514

Total params:	15,436,482
Trainable params:	721,794
Non-trainable params:	14,714,688

Table 1 shows the Convolutional Neural Network (CNN) architecture of the deep learning method used in this study. It can be seen in the architecture using the pre-trained VGG16 model as the base model on the first layer. Then proceed with 1 CNN block, which consists of the conv2d layer, max pooling, fully connected, and the last layer is the output layer.

Loss Function

The loss function is used to measure the error made by a machine learning model (Hakim & Rainarli, 2019). In this study, the loss function used is Categorical Cross-Entropy. This loss function is prevalent in training deep structures in the case of binary Classification (Rusiecki, 2019). Categorical Cross-Entropy calculates the difference between the given probability distributions of the CNNs. in the case of using the softmax function as in the current study, a probability distribution is needed (Gowdra et al., 2020).

Stages of research

Several primary stages can be done in the case of machine learning and deep learning. These stages are :

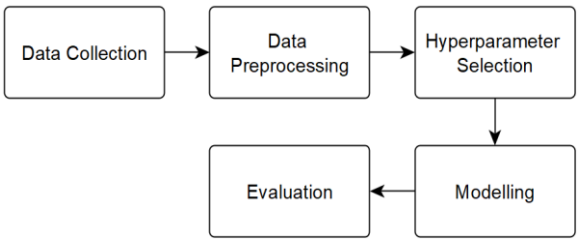

Figure 1. Stages Of Research Flow

In Figure 1, it is explained that the research stages consist of five stages: Data Collection, Data Preprocessing, Hyperparameter Selection, Modeling, and Evaluation.

Data Collection

This research used data from Samir Bhattarai that was uploaded on Kaggle, called "*New Plant Diseases Dataset*. " The data consists of 38 classes. However, only two classes will be used in this study, and it consists of 3901 training data and 970 validation data.

The following is a sample of the data used:

Figure 2. Sample Image Paprika Leaves That Infected with Bacterial Spots.

Figure 3. Sample Image Healthy Paprika Leaves.

Figures 2 and 3 visualize samples from each class, Pepper bell Bacterial spot and Pepper bell healthy.

Preprocessing Data

PILAR Nusa Mandiri: Journal of Computing and Information System
Vol. 18 No. 1 March 2022 | DOI: 10.33480/pilar.v18i1.2755

Data augmentation techniques can be developed to increase the variety of data to have more quantity without looking for new data. At this stage, image augmentation is performed for each data. In addition to augmentation, each data is also resized to fit the input shape required by the model.

After the augmentation, the image is normalized to be treated the same regardless of image resolution. Normalization of data is carried out from the image [0,255] so that it becomes [0,1] by using the "rescale" parameter.

Hyperparameter Selection

In this study, no hyperparameter tuning was carried out. However, the parameter selection was determined directly using the "adam" optimizer, then used a dropout layer of 0.5 and a dense layer of 256 with ReLu and Softmax activation.

Modeling

The modeling used ten epochs, and each epoch consisted of 100 steps for training and 16 steps for validation. That is based on the number of available datasets. Besides that, modeling also uses a function callback to stop the training process when the model has reached the desired condition. In this research, the condition is adjusted when the loss is<15%, and the accuracy is>96%. That is to avoid the model being overfitting.

Evaluation

In conducting the evaluation, the researcher used the available test data plus the paprika leaf image data taken from google image to determine the performance of the model that had been made. Then to measure the performance of the model, several calculations will be used, including(Andika et al., 2019):

1. Accuracy is the calculation of the correct number of proportions from the classification results of the model
2. Recall, Intuitive withdrawal is the model's ability to find all positive samples
3. Precision is the model's ability not to give positive labels as negative samples.

RESULTS AND DISCUSSION

Dataset size

The used dataset consists of 2 classes, 'Pepper bell Bacterial spot' and 'Pepper bell healthy.' Each class is loaded via google drive using google colab. The following are the results of data collection from the dataset.

```
[44] def dataset_len(path, classes):
         size = []
         for i in classes:
             size.append(len(os.listdir(path+"/train/"+ i))+
                         len(os.listdir(path+"/valid/"+ i)))

         df = pd.DataFrame(columns = ['Class', 'Number_Of_Images'])
         df['Class'] = classes
         df['Number_Of_Images'] = size

         return df

[45] dataset_len(base_path, classes)
```

	Class	Number_Of_Images
0	Pepper_bell___Bacterial_spot	2388
1	Pepper_bell___healthy	2483

Figure 4. Dataset Size Snippet Code

Figure 4 is a code snippet used to calculate the amount of data in each class to determine the size of the dataset. And it can be seen that there are 2388 data for the 'Pepper bell Bacterial spot' class and 2483 for the 'Pepper bell healthy' class. From the amount of data in each class, it can be concluded that the dataset is balanced.

Preprocessing

Preprocessing applies image augmentation, which is used for each data resized to 224x224 pixels, following the input conditions required by the model. In addition, researchers also set the image rotation distance of 20 degrees with the "rotation_range" parameter, then activate the "horizontal_flip" option, then use the "shear_range" parameter so that the model can see the image from various points of view. Then the "fill_mode" parameter is set to "nearest" so that it can fill in the blanks using the nearest pixel value if the image is stretched. Here is one of the images after augmentation:

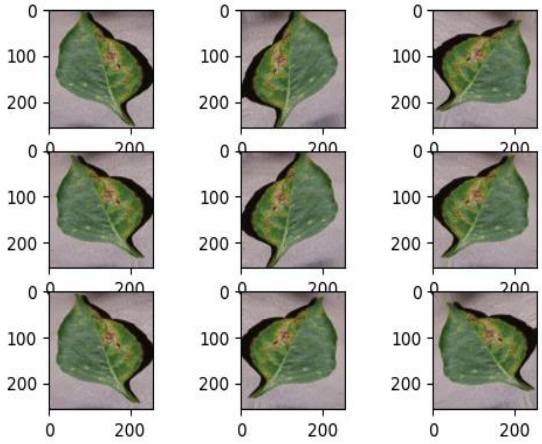

Figure 5. Sample of Augmented Image

Figure 5 is a sample image that has been augmented, and it can be seen that the image that has been augmented has changed to be more varied.

After each image is augmented, it is normalized to get an image that is easier to process regardless of image resolution. Here is the image array before and after normalization:

```
[[[161.66324  145.66324  148.15303 ]
  [159.67148  143.67148  145.92694 ]
  [157.67972  141.67972  143.70085 ]
  ...
  [150.97528  135.97528  138.97528 ]
  [116.947334 101.947334 104.947334]
  [119.41749  104.41749  107.41749 ]]

 [[166.7377   150.7377   153.7377  ]
  [166.62054  150.62054  153.62054 ]
  [166.50339  150.50339  153.50339 ]
  ...
  [152.41162  137.41162  140.41162 ]
  [110.96258   95.96258   98.96258 ]
  [121.93108  106.93108  109.93108 ]]

 [[167.73051  151.73051  154.73051 ]
  [167.61336  151.61336  154.61336 ]
  [167.49619  151.49619  154.49619 ]
  ...
  [153.84796  138.84796  141.84796 ]
  [104.97783   89.97783   92.97783 ]
  [124.44468  109.44468  112.44468 ]]

 ...
```

Figure 6. Sample of Image Array Before Normalization

```
[[[0.6555713  0.5928262  0.59674776]
  [0.6673598  0.60461473 0.6085363 ]
  [0.6791483  0.6164032  0.6203248 ]
  ...
  [0.47511125 0.4162877  0.42805243]
  [0.47757006 0.41874653 0.43051124]
  [0.4852095  0.42638597 0.43815067]]

 [[0.654902   0.5921569  0.59607846]
  [0.654902   0.5921569  0.59607846]
  [0.654902   0.5921569  0.59607846]
  ...
  [0.4656706  0.40684706 0.41861176]
  [0.49055094 0.4317274  0.4434921 ]
  [0.47222862 0.4134051  0.4251698 ]]

 [[0.689813   0.6270679  0.6309895 ]
  [0.67802453 0.61527944 0.619201  ]
  [0.666236   0.6034909  0.60741246]
  ...
  [0.46238977 0.40356624 0.41533095]
  [0.5003898  0.44156626 0.45333096]
  [0.4592477  0.40042418 0.4121889 ]]

 ...
```

Figure 7. Sample of Image Array After Normalization

In Figures 6 and 7, it can be seen that the image array before normalization is still based on [0,255], and after normalization, the image array is based on [0,1].

Keras's "ImageDataGenerator" library is used to perform augmentation in this research. Here is implementation of "ImageDataGenerator" for the preprocessing stage:

```
train_datagen = ImageDataGenerator(
                    rescale=1./255,
                    rotation_range=20,
                    horizontal_flip=True,
                    shear_range = 0.2,
                    fill_mode = 'nearest')

validation_datagen = ImageDataGenerator(
                    rescale=1./255,
                    rotation_range=20,
                    horizontal_flip=True,
                    shear_range = 0.2,
                    fill_mode = 'nearest')

train_generator = train_datagen.flow_from_directory(
                    train_path,
                    target_size=(224,224),
                    batch_size=32,
                    class_mode='categorical',
                    )

validation_generator = validation_datagen.flow_from_directory(
                    validation_path,
                    target_size=(224,224),
                    batch_size=32,
                    class_mode='categorical',
                    )
```

Figure 8. Implementation ImageDataGenerator on Python

Figure 8 shows the train data and validation data provided and then augmented using ImageDataGenerator using predetermined parameters.

Training and Validation

Model training was conducted using 3901 training data and 970 validation data. Other than that, it consists of ten trials or ten epochs. However, a callback function is utilized to stop the testing process when it reaches the desired condition in this research.

After execution, the training process lasts for five epochs, and in the fifth epoch, the callback function is called because it has met the desired conditions. The following is a table of the results of the training model:

Table 2. Model Compile Result

Epoch	accuracy	loss	val_accuracy	val_loss
1	0.8323	0.3698	0.9238	0.1746
2	0.9237	0.1893	0.9023	0.2154
3	0.9406	0.1444	0.9668	0.0855
4	0.9521	0.1256	0.9570	0.1233
5	0.9700	0.0845	0.9746	0.0695

Table 2 shows the compiled results of the model. The model is set to iterate for ten epochs but by implementing a callback function to stop when the model has reached the desired result. It can be seen that the model stops iterating at the fifth epoch.

The first epoch obtained an accuracy of 83.2%, and a loss of 37.0% accuracy validation is 92.3%, and loss is validation17%. In the first test, it

PILAR Nusa Mandiri: Journal of Computing and Information System
Vol. 18 No. 1 March 2022 | DOI: 10.33480/pilar.v18i1.2755

can be seen that the loss or error made by the model is still relatively high; it is 37%.

In the second epoch, accuracy is 92.4%, the loss is 18.9%, the validation accuracy is 90.2%, and the loss is validation 21.5%. In the second test, it can be seen that the loss validation or errors made by the model during validation increased compared to the first epoch.

In the third epoch, the accuracy is 94.1%, the loss is 14.4%, the validation accuracy is 96.7%, and the loss is validation 8.5%. In the third test, the model began to show good performance, but the accuracy had not reached the desired 96%.

The fourth epoch model is much better than the initial conditions but has not reached the desired condition. In the fourth epoch, accuracy is 95.2%, the loss is 12.6%, and the validation accuracy is 95.7%. The loss is validation12.3%.

In the fifth epoch, accuracy is 97.0%, the loss is 8.5%, the validation accuracy is 97.5%, and the loss is validation 6.9%.

The five models are much better in accuracy and loss obtained from all epochs. In this fifth test, the model has met the desired conditions so that the test ends by calling the callback function.

The following is the trend of the results of the tests that have been carried out:

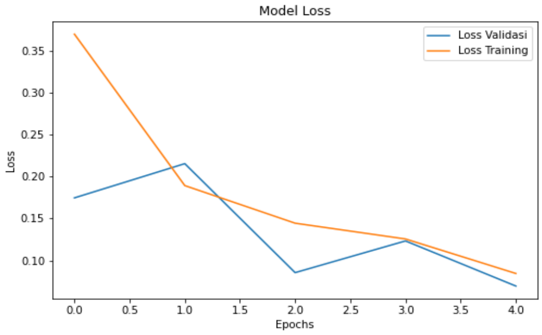

Figure 9. Graph of Loss Training and Validation

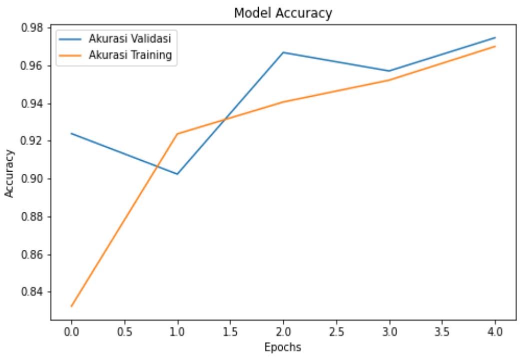

Figure 10. Graph of Accuracy Training and Validation

It can be seen in Figure 9 that the trend of the loss model in both the train data and data validation has decreased very well in line with the trend of accuracy, also constantly increasing in general in Figure 10.

The results in this study have an accuracy of 97% and validation accuracy of 97.5%, with a loss of 8.5% and a validation loss of 6.9%. these results have shown a better improvement compared to the research conducted by Abdul Jalil Rozaqi in the application of CNN for Disease Detection in Potato Leaves, which has an accuracy of 95.3% with a validation accuracy of 94.1%.

In addition to testing the model performance on new data, This study also tested 20 data tests that the model had never seen before. The following are the results of testing with test data:

Tabel 3. Testing Result

Image Data	Actual Value	Predicted Value
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot
	Bacterial_spot	Bacterial_spot

Image Data	Actual Value	Predicted Value
	Healthy	Healthy
	Healthy	Healthy
	Healthy	Bacterial_spot
	Healthy	Healthy
	Healthy	Bacterial_spot
	Healthy	Healthy
	Healthy	Healthy
	Healthy	Healthy
	Healthy	Healthy
	Healthy	Healthy

Table 3 shows results from testing the model using test data that the model has never seen. From 20 test data tested, the model managed to predict 18 of them correctly.

From the test results using data test, the obtained confusion matrix is as follows:

Tabel 4. Confusion Matrix Data Test

n=20	Actual Positive (+)	Actual Negative (-)
Predicted Positive (+)	TP : 8	FP : 0
Predicted Negative (-)	FN : 2	TN : 10

In table 4, the confusion matrix shows that the model has 8 True Positive results, no False Positives, 2 False Negatives, and 10 True Negatives. From this confusion matrix, we can calculate accuracy, recall, and precision as follows:

$$\text{accuracy} = \frac{(TP + TN)}{TP + FP + TN + FN} = \frac{18}{20} \times 100\% = 90\%$$

$$\text{recall} = \frac{TP}{TP + FN} = \frac{8}{10} \times 100\% = 80\%$$

$$\text{precision} = \frac{TP}{TP + FP} = \frac{8}{8} \times 100\% = 100\%$$

The accuracy results were obtained from test data that the model has never seen. It shows that the model has a reasonably good accuracy performance of 90%. The model's ability to not give a positive label to the data test negative is very high, reaching 100% on the 20 data test. However, the model's ability to identify positive data (recall) is relatively low compared to other performances, 80% of the 20 test data.

CONCLUSIONS

From the research that has been done, it can be concluded that the implementation of a Convolutional Neural Network (CNN) based model with an approach transfer learning on paprika leaf image data has a reasonably good performance. The dataset distribution is 3901 training data, 970 validation data, and the required input is 224x224, resulting in a model with an accuracy of 97% with a loss of 8.5% and a validation accuracy of 97.5% with a loss validation of 6.9%. The test data model could predict 18 of the 20 test data that had never been seen before. Moreover, the test stage delivers 90% accuracy, 80% recall, and 100% precision. As a form of suggestion for further development, the model can be further developed to have a good performance, significantly to increase the ratio of true positive predictions (recall). Besides that, it can also be developed so that the model can still have good performance even though the size and quality of the image are different from being implemented on various types of images.

REFERENCES

Andika, L. A., Pratiwi, H., & Handajani, S. S. (2019). Klasifikasi penyakit pneumonia menggunakan metode convolutional neural network dengan optimasi adaptive momentum. *Indonesian Journal of Statistics and Its Applications*, *3*(3), 331–340.

adan Pusat Statistik Indonesia. (2020). *Produksi Tanaman Sayuran 2020*. https://www.bps.go.id/indicator/55/61/1/produksi-tanaman-sayuran.html

PILAR Nusa Mandiri: Journal of Computing and Information System
Vol. 18 No. 1 March 2022 | DOI: 10.33480/pilar.v18i1.2755

Biswas, T., Guan, Z., & Wu, F. (2018). An overview of the US bell pepper industry. *EDIS*, *2018*(2).

Bogatzevska, N., Vancheva-Ebben, T., Vasileva, K., Kizheva, Y., & Moncheva, P. (2021). An overview of the diversity of pathogens causing bacterial spot on tomato and pepper in Bulgaria. *Bulgarian Journal of Agricultural Science*, *27*(1), 137–146.

Food and Agriculture Organization of the United Nations Statistics. (2020). *Countries by commodity*. https://www.fao.org/faostat

Gowdra, N., Sinha, R., MacDonell, S., & Yan, W. (2020). *Maximum Categorical Cross Entropy (MCCE): A noise-robust alternative loss function to mitigate racial bias in Convolutional Neural Networks (CNNs) by reducing overfitting*.

Hakim, D. M., & Rainarli, E. (2019). Convolutional Neural Network untuk Pengenalan Citra Notasi Musik. *Techno. Com*, *18*(3), 214–226.

Ilahiyah, S., & Nilogiri, A. (2018). Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *JUSTINDO (Jurnal Sistem Dan Teknologi Informasi Indonesia)*, *3*(2), 49–56.

Rahman, M., & Miah, N. A. (2020). *Bacterial Leaf Spot of Pepper*. WVU Extension. https://extension.wvu.edu/

Reza, P. M. A., Syuhriatin, S., & Rahayu, S. M. (2021). Analisis Pertumbuhan Tanaman Paprika (Capsicum annuum var. grossum) Berdasarkan Pola Tanam. *LOMBOK JOURNAL OF SCIENCE*, *3*(1), 23–32.

Rezende, E., Ruppert, G., Carvalho, T., Theophilo, A., Ramos, F., & de Geus, P. (2018). Malicious software classification using VGG16 deep neural network's bottleneck features. In *Information Technology-New Generations* (pp. 51–59). Springer.

Rozaqi, A. J., Sunyoto, A., & rudyanto Arief, M. (2021). Deteksi Penyakit Pada Daun Kentang Menggunakan Pengolahan Citra dengan Metode Convolutional Neural Network. *Creative Information Technology Journal*, *8*(1), 22–31.

Rusiecki, A. (2019). Trimmed categorical cross-entropy for deep learning with label noise. *Electronics Letters*, *55*(6), 319–320.

Simonyan, K., & Zisserman, A. (2018). Very deep convolutional networks for large-scale image recognition. arXiv. *ArXiv Preprint ArXiv:1409.1556*.

Suminar, J. R., Karolina, C. M., & Ratnasari, E. (2019). Lumbung Paprika Indonesia: Desa Pasirlangu Studi Kasus Komunikasi Pertanian di Desa Pasirlangu Kabupaten Bandung Barat sebagai Lumbung Pertanian Paprika di Indonesia. *Studia Komunika: Jurnal Ilmu Komunikasi*, *2*(2), 33–42.

Umri, B. K., & Delica, V. (2021). Penerapan transfer learning pada convolutional neural networks dalam deteksi covid-19. *JNANALOKA*, 53–61.