Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Comparison of Accuracy in Detecting Tomato Leaf Disease with GoogleNet VS EfficientNetB3

Adi Dwifana Saputra[1)], Djarot Hindarto[2*)], Ben Rahman[3)], Handri Santoso[4)]
[1,4)] Program Studi Bigdata & IoT, Universitas Pradita, Indonesia
[2,3)] Program Studi Informatika, Fakultas Teknologi Komunikasi dan Informatika, Universitas Nasional,
[1)]adi.dwifana@student.pradita.ac.id, [2*)]djarot.hindarto@civitas.unas.ac.id, [3)] benrahman@civitas.unas.ac.id, [4)] handri.santoso@pradita.ac.id

Submitted : Feb 21, 2023 | **Accepted** : Feb 22, 2023 | **Published** : Apr 1, 2023

Abstract: Tomato diseases vary greatly, one of which is tomato leaf disease. Some variants of leaf diseases include late blight, septoria leaf, yellow leaf curl virus, bacteria, mosaic virus, leaf fungus, two-spotted spider mite, and powdery mildew. By knowing the disease on tomato leaves, you can find medicine for the disease. So that it can increase the production of tomatoes with good quality and a lot of quantity. The problem that often occurs is that farmers cannot determine the disease in plants, they try to find suitable herbal medicines for their plants. After being given the drug, many plants actually died due to the pesticides given to the tomato plants. This is detrimental to tomato farmers. This problem is caused by incorrect disease detection. Therefore, this study aims to solve the problem of disease detection in tomato plants, in a more specific case, namely tomato leaves. Detection in this study uses a deep learning algorithm that uses a Convolutional Neural Network, specifically GoogleNet and EfficientNetB3. The dataset used comes from kaggle and google image. Both data sets have been pre-processed to match the data set class. Image preprocessing is performed to produce appropriate image datasets and improve performance accuracy. The dataset is trained to get the model. The training using GoogleNet resulted in an accuracy of 98.10%, loss of 0.0602 and using EfficientNetB3 resulted in an accuracy of 99.94%, loss: 0.1966.

Keywords: Convolutional Neural Network; Dataset; EfficientNetB3; GoogleNet; Tomato leaf disease;

INTRODUCTION

Tomato plants come from the Americas, spread from Central America to South America. Tomato plants were first cultivated by the Aztecs and Incas in 700 BC. The spread of tomatoes in Indonesia started from the Philippines and other Asian countries in the 18th century (Shabira et al., 2020). Tomato plants generally live in the highlands, but the increase in people's need for consumption requires an expansion of the planting area, namely in the lowlands. Tomato plants that live in the lowlands still experience a decline in production due to the inability of these plants to the environment (Wijayani & Widodo, 2005). Efforts to develop tomatoes in Aceh have increased every year, in line with high market demand. To offset this demand, tomato cultivation needs to be continuously developed, both in terms of expanding the planting area and improving the genetic properties of plants. However, there are obstacles to the development of tomato cultivation in the lowlands, namely, the lack of superior varieties suitable for cultivation in the lowlands (Purwati & Khairunisa, 2007).

Tomato is a very important vegetable crop in the food industry (Sangeetha et al., 2023), (Ahmadbeyki et al., 2023). Without the presence of tomatoes, it feels less delicious in cooking. Indonesia has produced a lot of tomatoes, where the growth of tomatoes is very dependent on the health of the tomato plants. If the cultivation of tomato plants is disturbed, the production of tomatoes will be disrupted. Therefore, plant health needs to be maintained and maintained. Many tomatoes plant diseases attack such as leaves, stems, roots and fruit. Diseases of tomato plants need to be done with uses such as drugs for diseases or the use of pesticides. Pesticides will work optimally if used appropriately for certain diseases. One type of pesticide is not able to cure all diseases in tomato plants. This is what makes tomato farmers lose money, if farmers do not understand the types of diseases in plants. Most farmers do not have knowledge of diseases, so by buying certain products, they may not be able to cure plant

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

diseases. Sometimes there are sales of pesticide products providing information that may not necessarily cure diseases in plants.

Of the several diseases in tomato plants, this research focuses on tomato leaf diseases. Various types of tomato leaf diseases in fig 1, are examples of healthy tomato leaves. This can be a consideration for the comparison process with tomato leaves affected by the disease.

Fig. 1 Healthy tomato plants.
Source: Google Image

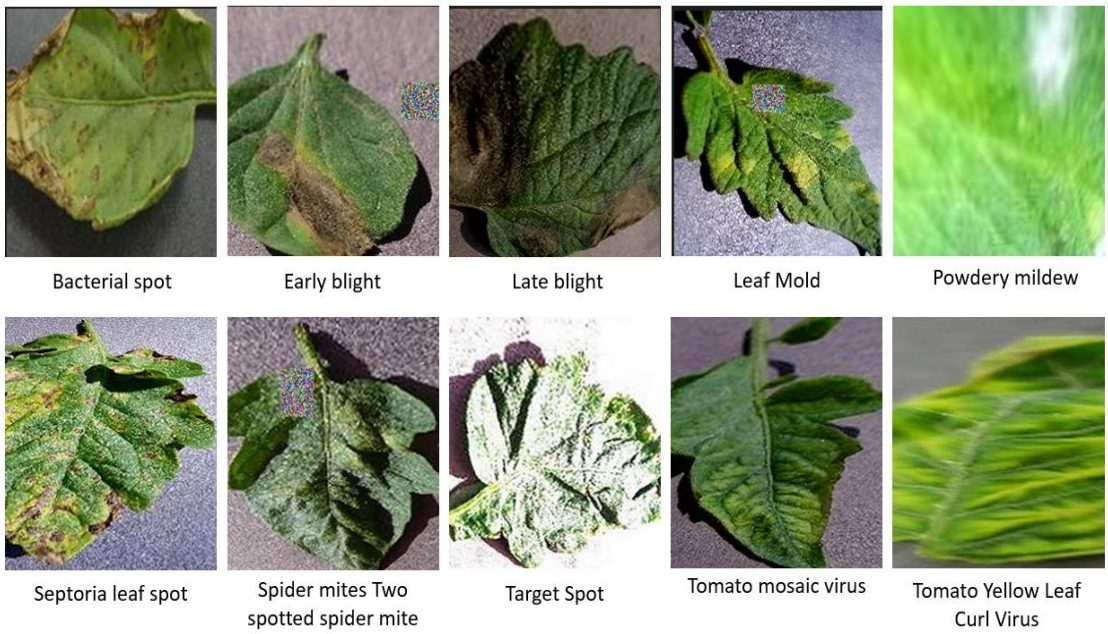

Fig. 2 Tomato leaf disease
Source: Kaggle

Fig. 2 is a dataset containing diseases on tomato leaves. The dataset in this research contains images of tomato leaves. This research processes disease images on tomato leaves. Later the image will be extracted for further

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

processing using the Deep Learning algorithm for the field of computer vision. The purpose of this research is to detect diseases on tomato leaves using the GoogleNet and EfficientNet B3 algorithms. Next, do a comparison between GoogleNet and EfficeintNet B3.

LITERATURE REVIEW

Previous research has discussed a lot about tomato leaf diseases using machine learning and deep learning. In table 1, it describes the research that has discussed diseases on tomato leaves, using machine learning methods and deep learning methods.

Table 1. The discussion of tomato leaf disease research uses machine learning and deep learning

Reference	Title	Advantages	Weakness
(Alviansyah et al., 2017)	Identification of Diseases in Tomato Plants Based on Leaf Color and Shape Using the Web-Based Naive Bayes Classifier Method	Detecting disease on tomato leaves using the sobel operator edge method and the naïve bayes classifier method. In this study the accuracy reached 82.98%.	In the study "Identification of Diseases in Tomato Plants Based on Leaf Color and Shape Using the Web-Based Naive Bayes Classifier Method" using a small dataset, namely an image dataset of around 47 images.
(Putri, 2021)	Implementation of Backpropagation Artificial Neural Network (ANN) for Classifying Types of Diseases on Tomato Plant Leaves	In this research detects diseases on tomato leaves using the Artificial Neural Network (ANN) method with the Backpropagation method. Accuracy in this research reached 78%.	The research "Implementation of Backpropagation Artificial Neural Network (ANN) for Classifying Types of Diseases on Tomato Plant Leaves" uses a dataset of 50 images of diseased tomato leaves and a dataset of 50 images of healthy leaves. The image dataset is too small for the use of the Artificial Neural Network (ANN) method.
(Felix et al., 2019)	Implementation of CNN and SVM for Identification of Tomato Diseases via Leaves	Research by detecting tomato leaf diseases using the Support Vector Machine (SVM) algorithm and Deep Learning algorithms using the Convolutional Neural Network (CNN). Extraction feature, Gaussian/Radial Basis Function (RBF) kernel and training using the SMO algorithm with a maximum violating pair for training using the Support Vector Machine. Convolutional Neural Network uses Texture feature extraction from the GLCM (Gray Level Co-occurrence Matrix) process previously produced 2 (two) texture feature data (energy and entropy) in the form of matrices. The matrices are then converted to images. Image is used as input in the convolution and ReLU (Rectified Linear Unit)	The weakness of the research "Implementation of CNN and SVM for Identification of Tomato Diseases via Leaves", the use of image datasets is around 200 image datasets. It would be better to use a dataset of around 2000 images. This can produce more models in detecting diseases in tomato plant leaves.

*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X

		processes, max-pooling (down sampling) and flatten. Getting SVM accuracy around 95% and CNN accuracy around 97,5%.	
(Rozaqi et al., 2021)	Disease Detection on Potato Leaves Using Image Processing with the Convolutional Neural Network Method	Research for diseases on potato leaves using the Convolutional Neural Network algorithm. And get an accuracy of about 95% and a loss value of around 0.0711.	The research "Disease Detection on Potato Leaves Using Image Processing with the Convolutional Neural Network Method" uses a dataset of 1200 images. The use of datasets for this research is still relatively small.
(Muchtar et al., 2021)	Septoria Detection in Tomato Plants with Raspberry Pi-based Deep Learning Method	This research describes the detection of tomato leaves using the Deep Learning method. In carrying out its implementation, the training result model is embedded into the Raspberry Pi equipped with the Intel Movidius Neural Computing Stick. Intel Movidius Neural Computing is a Vision Processing Unit (VPU) or processing accelerator. The accuracy of this research is 95.89%.	Does not display how many image datasets were trained. So that the article readers don't know how to do training using image datasets.
(Awalia, 2022)	Identification of Leaf Mold Disease on Tomato Leaves Using the Densenet121 Model Based on Transfer Learning	This research detects disease on tomato leaves using the Convolutional Neural Network (CNN) algorithm with the DenseNet121 method and transfer learning. from the training results get an accuracy of 92.6%.	The research "Identification of Leaf Mold Disease on Tomato Leaves Using the Densenet121 Model Based on Transfer Learning", has not made comparisons with other methods or algorithms such as VGG19, Inception or GoogleNet.

From the research in table 1, there are several advantages and disadvantages. This research does not look for weaknesses in tomato leaf research. But it complements the research that has been done. The image data used is still small if it is processed with the Deep Learning algorithm. There are no limitations for datasets using Deep Learning. The research gap between this research and previous research is the use of datasets for Deep Learning. The state-of-the-art is that this research uses a dataset of 7000 images, accuracy with Google-Net: 98.10%, loss of 0.0602 and EfficientNetB3 resulted in an accuracy of 99.94%, loss: 0.1966.

METHOD

Dataset, totaling more than 20,000 images of tomato leaves with 10 diseases and 1 healthy class. Images collected from the lab and from the wild. The aim is to develop a mild model that can predict tomato leaf disease, only for training model classification. The dataset consists of 1 class of healthy leaf dataset, 10 class of diseased leaf image dataset: Late blight, Early rot disease, Septoria leaf, Tomato Yellow Leaf Curl Virus, Bacteria, Target Place, Tomato virus, Leaf Print, Spidermites Two-spotted spider mite, Powdery mildew.

Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a type of deep learning neural network that is commonly used in image and video processing tasks (Uparkar et al., 2023), (Pereira et al., 2023). CNNs are designed to process data

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

that has a grid-like topology, such as an image, where each element of the grid (pixel) is connected to its local neighbors.

A CNN typically consists of several layers, with the first layer being the input layer, and the last layer being the output layer. The input layer receives the image data, and the output layer produces the final prediction or classification. In between the input and output layers are multiple hidden layers, which are used to extract features from the input data.

The most important layers of a CNN are the convolutional layers and the pooling layers. The convolutional layers are used to extract features from the input data by applying a set of filters to the input data. Each filter is designed to detect a specific feature in the input data. The pooling layers are used to reduce the spatial resolution of the input data by combining the values of nearby pixels. This helps to reduce the computational cost of the network and also helps to prevent overfitting.

After the feature extraction, the features are passed through fully connected layers, which are used to classify the input image into one of several predefined classes. A CNN can also be used for image segmentation and object detection tasks.

CNNs have been known to perform well on image and video recognition tasks, and are used in a variety of applications such as self-driving cars, medical imaging, and facial recognition. The specific mathematical formula used in a convolutional neural network (CNN) will vary depending on the specific architecture of the network and the problem it is trying to solve. However, a general formula for the computation of a convolutional layer in a CNN can be represented as:

$$\text{Output}(i, j, k) = \sum m \sum n \sum c \ \text{Input}(i + m, j + n, c) \quad (1)$$

* Filter (m, n, c, k)

In this formula, the output of the convolutional layer is represented by the 3D tensor Output, which has dimensions (i, j, k). The input to the convolutional layer is represented by the 3D tensor Input, which has dimensions (i, j, c). The filter used in the convolutional operation is represented by the 4D tensor Filter, which has dimensions (m, n, c, k).

The symbols i, j, m, and n represent the spatial dimensions of the output and input tensors, respectively. The symbols c and k represent the number of channels in the input and output tensors, respectively. The convolution operation is performed by taking the dot product of the input tensor and the filter tensor, and then summing over the m and n dimensions.

A common operation after a convolutional layer is to apply a non-linear activation function, such as ReLU (Rectified Linear Unit) function, which is typically applied element-wise to the output of the convolutional layer. This activation function can be applied with a formula:

$$\text{Output}(i,j,k) = \max(0, \text{Input}(i,j,k)) \quad (2)$$

This is a basic formula for the computation of a convolutional layer in a CNN, but the actual implementation of CNNs can be quite complex and may include additional operations such as pooling, batch normalization and dropout.

EfficientNet B3

Efficient-Net B3 (de Zarzà et al., 2022) is a family of convolutional neural networks (CNNs) (Lee et al., 2022) that were developed by Google AI in 2019. The goal of EfficientNet is to improve the accuracy of CNNs while also reducing their computational cost. This is achieved by using a combination of several techniques such as scaling up the model's architecture, using deeper and wider layers, and applying a compound scaling method to balance the trade-off between accuracy and efficiency. The EfficientNet model architecture is based on the concept of scaling up the model's architecture while also scaling up the input image size. This is achieved by increasing the resolution of the input image, and also increasing the depth and width of the network layers. Additionally, the model also uses depth-wise separable convolutions which are more efficient than standard convolutions.

The compound scaling method is used to balance the trade-off between accuracy and efficiency. This method adjusts the model's architecture, input image resolution, and model's parameters to optimize the model's performance. The result is a family of models, each with a different level of accuracy and efficiency. EfficientNet has been shown to achieve state-of-the-art accuracy on several image classification benchmarks, while also being more computationally efficient than other models. It is used in various applications such as object detection, image segmentation, and more. The specific formula used in the EfficientNet method depends on the specific architecture

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

of the network and the problem it is trying to solve. However, the general method for building an EfficientNet model is based on the compound scaling method which consists of three main elements:

1. Scaling the model's architecture
2. Scaling the input image resolution
3. Scaling the model's parameters

The formula used to scale the model's architecture is:

$$\text{Width}(i) = \alpha * \text{Width}(i-1) \quad (3)$$

$$\text{Depth}(i) = \beta * \text{Depth}(i-1) \quad (4)$$

$$\text{Resolution}(i) = \gamma * \text{Resolution}(i-1) \quad (5)$$

Where i is the index of the model in the EfficientNet family, Width(i) is the width of the model at index i, Depth(i) is the depth of the model at index i, Resolution(i) is the input image resolution of the model at index i, α, β, and γ are scaling coefficients that are chosen to optimize the model's performance.

The formula used to scale the model's parameters is:

$$\text{Learning Rate} = \eta * \min(1, (\text{batch size} * \text{steps}) / (256 * \text{steps})) \quad (6)$$

$$\text{Weight Decay} = \lambda * \eta \quad (7)$$

Where η is the initial learning rate, batch size is the number of examples per batch and steps is the total number of training steps. λ is a coefficient that is chosen to optimize the model's performance.

The EfficientNet method also uses a set of techniques to improve the model's efficiency such as depth-wise separable convolutions, swish activation function, and efficient head architecture. The specific implementation of the EfficientNet method may include additional techniques to improve the model's performance, such as data augmentation, and transfer learning.

GoogleNet

Google-Net (Teerakawanich et al., 2020) is a convolutional neural network architecture developed by Google for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2014. The architecture uses the Inception module, which allows the network to extract features at multiple scales and dimensions, resulting in improved performance over traditional CNN architectures. GoogleNet was able to achieve state-of-the-art performance on the ImageNet dataset and was a key factor in the development of deep learning techniques.

The mathematical formulation of GoogleNet can be quite complex, as it involves a large number of convolutional layers, pooling layers, and fully connected layers. However, the core component of the architecture is the Inception module, which can be mathematically represented as follows:

1. The input of the Inception module is passed through multiple convolutional layers, each with different filter sizes (e.g., 1x1, 3x3, 5x5) and strides. These convolutional layers extract features at different scales and dimensions.
2. The output of these convolutional layers is then passed through a pooling layer, which is used to reduce the spatial dimensions of the input.
3. The output of the pooling layer is concatenated with the output of the 1x1 convolutional layer, which serves as a "bottleneck" layer that helps to reduce the number of parameters in the network.
4. The concatenated output is then passed through additional fully connected layers before producing the final output.

Overall, GoogleNet architecture (Jahandad et al., 2019)was designed to be deep, with multiple layers and a large number of filters, which allows it to learn complex features and improve performance on image recognition tasks.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

RESULT

In training the tomato dataset using the GoogleNet algorithm, it can be obtained as follows:

1. *Architecture GoogleNet*

```
In [19]:  1 model.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_1 (InputLayer)           [(None, 224, 224, 3  0           []
                                )]

 conv_1_7x7/2 (Conv2D)          (None, 112, 112, 64  9472        ['input_1[0][0]']
                                )

 max_pool_1_3x3/2 (MaxPooling2D  (None, 56, 56, 64)  0           ['conv_1_7x7/2[0][0]']
 )

 batch_normalization (BatchNorm  (None, 56, 56, 64)  256         ['max_pool_1_3x3/2[0][0]']
 alization)

 conv2d (Conv2D)                (None, 56, 56, 64)   4160        ['batch_normalization[0][0]']

 conv2d_1 (Conv2D)              (None, 56, 56, 192)  110784      ['conv2d[0][0]']

 batch_normalization_1 (BatchNo  (None, 56, 56, 192)  768        ['conv2d_1[0][0]']
 rmalization)

 max_pooling2d (MaxPooling2D)   (None, 28, 28, 192)  0           ['batch_normalization_1[0][0]']

 conv2d_3 (Conv2D)              (None, 28, 28, 96)   18528       ['max_pooling2d[0][0]']

 conv2d_5 (Conv2D)              (None, 28, 28, 16)   3088        ['max_pooling2d[0][0]']

 max_pooling2d_1 (MaxPooling2D)  (None, 28, 28, 192)  0          ['max_pooling2d[0][0]']

 conv2d_2 (Conv2D)              (None, 28, 28, 64)   12352       ['max_pooling2d[0][0]']

 conv2d_51 (Conv2D)             (None, 7, 7, 192)    159936      ['inception_5a[0][0]']

 conv2d_53 (Conv2D)             (None, 7, 7, 48)     39984       ['inception_5a[0][0]']

 max_pooling2d_10 (MaxPooling2D  (None, 7, 7, 832)   0           ['inception_5a[0][0]']
 )

 conv2d_50 (Conv2D)             (None, 7, 7, 384)    319872      ['inception_5a[0][0]']

 conv2d_52 (Conv2D)             (None, 7, 7, 384)    663936      ['conv2d_51[0][0]']

 conv2d_54 (Conv2D)             (None, 7, 7, 128)    153728      ['conv2d_53[0][0]']

 conv2d_55 (Conv2D)             (None, 7, 7, 128)    106624      ['max_pooling2d_10[0][0]']

 inception_5b (Concatenate)     (None, 7, 7, 1024)   0           ['conv2d_50[0][0]',
                                                                  'conv2d_52[0][0]',
                                                                  'conv2d_54[0][0]',
                                                                  'conv2d_55[0][0]']

 average_pooling2d (AveragePool  (None, 1, 1, 1024)  0           ['inception_5b[0][0]']
 ing2D)

 flatten (Flatten)              (None, 1024)         0           ['average_pooling2d[0][0]']

 dropout (Dropout)              (None, 1024)         0           ['flatten[0][0]']

 output (Dense)                 (None, 11)           11275       ['dropout[0][0]']

==================================================================================================
Total params: 5,985,851
Trainable params: 5,985,339
Non-trainable params: 512
__________________________________________________________________________________________________
```

Fig. 3 Summary Architecture Google-Net
Source: Property Researcher

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

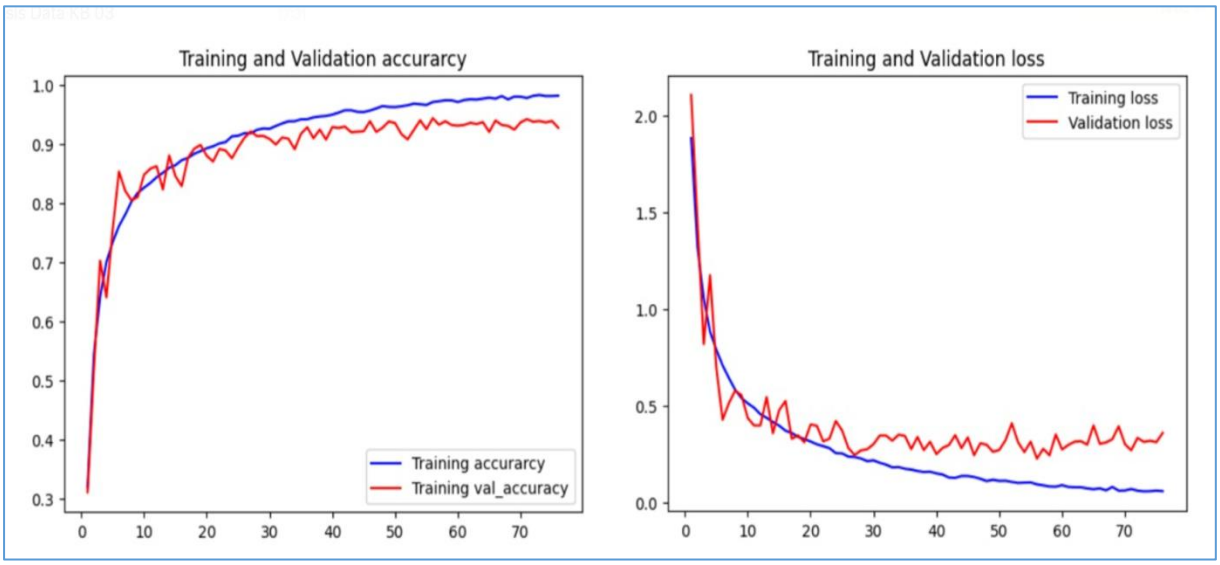

Fig. 4 Training and validation accuracy & loss Google-Net
Source: Property Researcher

The use of the GoogleNet Algorithm in detecting diseases in tomato leaves achieves an accuracy of 98.10% and a loss of around 0.0602.

2. EfficientNet B3

```
def make_model(img_size, lr, mod_num=3):
    img_shape=(img_size[0], img_size[1], 3)
    if mod_num == 0:
        base_model=tf.keras.applications.efficientnet.EfficientNetB0(include_top=False, weights="imagenet",input_shape=img_s
        msg='Created EfficientNet B0 model'
    elif mod_num == 3:
        base_model=tf.keras.applications.efficientnet.EfficientNetB3(include_top=False, weights="imagenet",input_shape=img_s
        msg='Created EfficientNet B3 model'
    elif mod_num == 5:
        base_model=tf.keras.applications.efficientnet.EfficientNetB5(include_top=False, weights="imagenet",input_shape=img_s
        msg='Created EfficientNet B5 model'

    else:
        base_model=tf.keras.applications.efficientnet.EfficientNetB7(include_top=False, weights="imagenet",input_shape=img_s
        msg='Created EfficientNet B7 model'

    base_model.trainable=True
    x=base_model.output
    x=BatchNormalization(axis=-1, momentum=0.99, epsilon=0.001 )(x)
    x = Dense(256, kernel_regularizer = regularizers.l2(l = 0.016),activity_regularizer=regularizers.l1(0.006),
                    bias_regularizer=regularizers.l1(0.006) ,activation='relu')(x)
    x=Dropout(rate=.4, seed=123)(x)
    output=Dense(class_count, activation='softmax')(x)
    model=Model(inputs=base_model.input, outputs=output)
    model.compile(Adamax(learning_rate=lr), loss='categorical_crossentropy', metrics=['accuracy'])
    msg=msg + f' with initial learning rate set to {lr}'
    print_in_color(msg)
    return model

lr=.001
model=make_model(img_size, lr) # using B3 model by default
```

Fig. 5 EfficientNet-B3 model
Source: Property Researcher

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 2, April 2023
DOI : https://doi.org/10.33395/sinkron.v8i2.12218
e-ISSN : 2541-2019
p-ISSN : 2541-044X

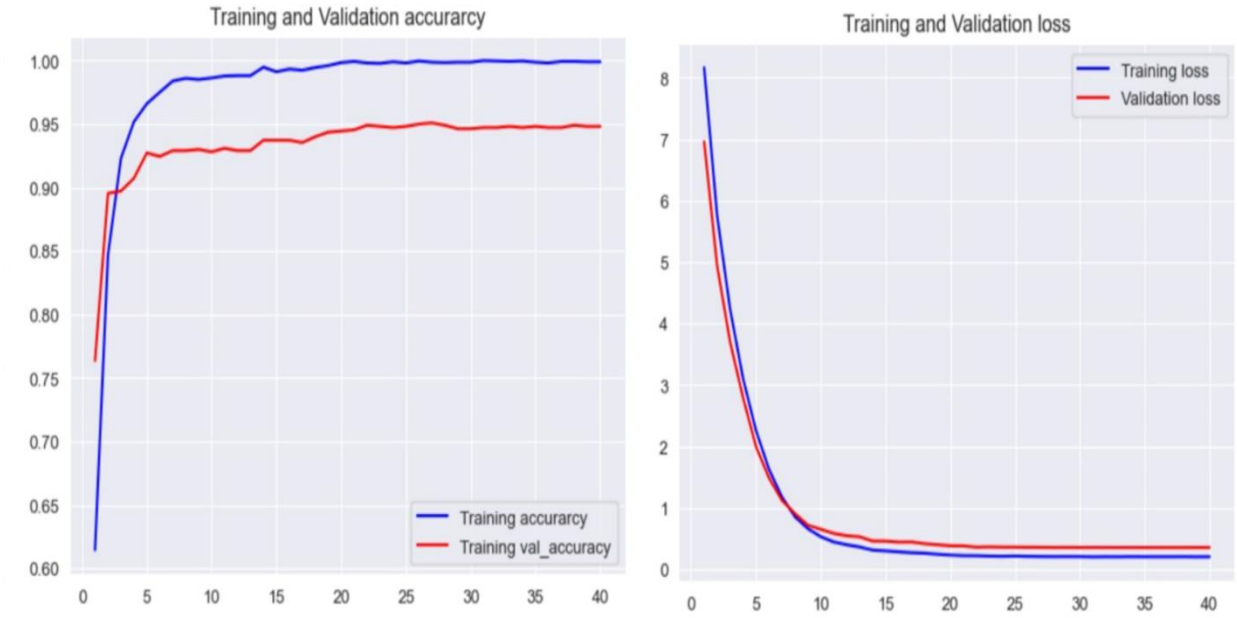

Fig. 6 Training and validation accuracy & loss EfficientNet-B3
Source: Property Researcher

The use of the GoogleNet Algorithm in detecting diseases in tomato leaves achieves an accuracy of 99.94% and a loss of around 0.1966

DISCUSSIONS

EfficientNet is a neural convolution algorithm (CNN) that is used to classify images. In detecting diseases on tomato leaves, EfficientNet can be used to identify the types of diseases that appear on tomato leaves by classifying the images of tomato leaves taken. One of the advantages of using EfficientNet in disease detection on tomato leaves is its ability to capture features better than other algorithms used for disease detection. This is because EfficientNet has an architecture that is more efficient in capturing important features and can save on the number of parameters used. In addition, EfficientNet also has the ability to transfer learning, which is to utilize knowledge obtained from other datasets to be used in new datasets. This can improve accuracy in disease detection on tomato leaves. In the research conducted, EfficientNet B3 was able to achieve a fairly high accuracy in detecting diseases on tomato leaves, namely 99.94%. However, keep in mind that these results only refer to the conditions and datasets used in the study.

GoogleNet is a neural convolution algorithm (CNN) developed by Google. In detecting diseases on tomato leaves, GoogleNet can be used to identify the types of diseases that appear on tomato leaves by classifying the images of tomato leaves taken. GoogleNet has an architecture known as Inception, which allows the algorithm to capture different features in an image using multiple filters on a single layer. This can improve accuracy in disease detection on tomato leaves. Apart from that, GoogleNet also has the ability to transfer learning, which is to use knowledge obtained from other datasets to be used in new datasets. This can improve accuracy in disease detection on tomato leaves. In the research conducted, GoogleNet was able to achieve an accuracy of 98.10% in disease detection on tomato leaves. However, keep in mind that these results only refer to the conditions and datasets used in the study. GoogleNet, although it has lower accuracy than EfficientNet B3 in this study, GoogleNet can still be used as an alternative in detecting diseases on tomato leaves. However, keep in mind that the selection of the algorithm used must be in accordance with the conditions and available datasets.

CONCLUSION

From the results given, it can be seen that EfficientNet B3 has higher accuracy than GoogleNet. This significant difference in accuracy may indicate that EfficientNet B3 may be better at handling a given problem compared to GoogleNet. However, keep in mind that these results are only valid for the given problem and cannot be generalized to other problems. EfficientNet B3 has higher accuracy compared to GoogleNet, with values of 99.94% and 98.10% respectively. This shows that EfficientNet B3 is more effective in classifying images than GoogleNet.

*name of corresponding author

REFERENCES

Ahmadbeyki, A., Ghahderijani, M., Borghaee, A., & Bakhoda, H. (2023). Energy use and environmental impacts analysis of greenhouse crops production using life cycle assessment approach: A case study of cucumber and tomato from Tehran province, Iran. *Energy Reports*, *9*, 988–999. https://doi.org/10.1016/j.egyr.2022.11.205

Alviansyah, F., Ruslianto, I., & Diponegoro, M. (2017). Identifikasi Penyakit Pada Tanaman Tomat Berdasarkan Warna Dan Bentuk Daun Dengan Metode Naive Bayes Classifier Berbasis Web. *Jurnal Coding Sistem Komputer Untan*, *5*(1), 23–32.

Awalia, N. (2022). Identifikasi Penyakit Leaf Mold Pada Daun Tomat Menggunakan Model Densenet121 Berbasis Transfer Learning. *Jurnal Ilmiah Ilmu Komputer*, *8*(1), 49–52. https://doi.org/10.35329/jiik.v8i1.212

de Zarzà, I., de Curtò, J., & Calafate, C. T. (2022). Detection of glaucoma using three-stage training with EfficientNet. *Intelligent Systems with Applications*, *16*(September), 1–10. https://doi.org/10.1016/j.iswa.2022.200140

Felix, F., Faisal, S., Butarbutar, T. F. M., & Sirait, P. (2019). Implementasi CNN dan SVM untuk Identifikasi Penyakit Tomat via Daun. *Jurnal SIFO Mikroskil*, *20*(2), 117–134. https://doi.org/10.55601/jsm.v20i2.670

Jahandad, Sam, S. M., Kamardin, K., Amir Sjarif, N. N., & Mohamed, N. (2019). Offline signature verification using deep learning convolutional Neural network (CNN) architectures GoogLeNet inception-v1 and inception-v3. *Procedia Computer Science*, *161*, 475–483. https://doi.org/10.1016/j.procs.2019.11.147

Lee, A. L. S., To, C. C. K., Lee, A. L. H., Li, J. J. X., & Chan, R. C. K. (2022). Model architecture and tile size selection for convolutional neural network training for non-small cell lung cancer detection on whole slide images. *Informatics in Medicine Unlocked*, *28*, 100850. https://doi.org/10.1016/j.imu.2022.100850

Muchtar, K., Chairuman, Yudha Nurdin, & Afdhal Afdhal. (2021). Pendeteksian Septoria pada Tanaman Tomat dengan Metode Deep Learning berbasis Raspberry Pi. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *5*(1), 107–113. https://doi.org/10.29207/resti.v5i1.2831

Pereira, R., Rocha, E., Pinho, D., & Santos, J. P. (2023). Convolutional neural networks for identification of moving combustion chambers entering a brazing process. *Procedia Computer Science*, *217*(2022), 1106–1116. https://doi.org/10.1016/j.procs.2022.12.309

Purwati, E., & Khairunisa. (2007). *Budidaya Tomat Dataran Rendah: Dengan Varietas Unggul Serta Tahan Hama & Penyakit*. Penebar Swadaya. http://opacperpus.jogjakota.go.id/index.php/home/detail_koleksi?kd_buku=014012&id=1&kd_jns_buku=SR

Putri, A. W. (2021). Implementasi Artificial Neural Network (ANN) Backpropagation Untuk Klasifikasi Jenis Penyakit Pada Daun Tanaman Tomat. *MATHunesa: Jurnal Ilmiah Matematika*, *9*(2), 344–350. https://doi.org/10.26740/mathunesa.v9n2.p344-350

Rozaqi, A. J., Sunyoto, A., & Arief, M. rudyanto. (2021). Deteksi Penyakit Pada Daun Kentang Menggunakan Pengolahan Citra dengan Metode Convolutional Neural Network. *Creative Information Technology Journal*, *8*(1), 22. https://doi.org/10.24076/citec.2021v8i1.263

Sangeetha, K., Ramyaa, R. B., Mousavi, A., & Radhakrishnan, M. (2023). Extraction , characterization , and application of tomato seed oil in the food industry : An updated review. *Journal of Agriculture and Food Research*, *11*(February), 100529. https://doi.org/10.1016/j.jafr.2023.100529

Shabira, S. P., Hereri, A. I., & Kesumawati, E. (2020). Identifikasi Karakteristik Morfologi dan Hasil Beberapa Jenis Tanaman Tomat (Lycopersicum esculentum) di Dataran Rendah. *Jurnal Ilmiah Mahasiswa Pertanian*, *4*(2), 51–60. https://doi.org/10.17969/jimfp.v4i2.11042

Teerakawanich, N., Leelaruji, T., & Pichetjamroen, A. (2020). Short term prediction of sun coverage using optical flow with GoogLeNet. *Energy Reports*, *6*, 526–531. https://doi.org/10.1016/j.egyr.2019.11.114

Uparkar, O., Bharti, J., Model, R., Pateriya, K., Ashutosh, X., Uparkar, O., Bharti, J., Pateriya, R. K., Gupta, R. K., & Sharma, A. (2023). ScienceDirect ScienceDirect Vision Transformer Outperforms Deep Convolutional Neural Network-based Model in Classifying X-ray Images Vision Transformer Outperforms Deep Convolutional Neural Network-based in Classifying. *Procedia Computer Science*, *218*(2022), 2338–2349. https://doi.org/10.1016/j.procs.2023.01.209

Wijayani, A., & Widodo, W. (2005). Usaha Meningkatkan Kualitas Beberapa Varietas Tomat Dengan Sistem Budidaya Hidroponik. *Ilmu Pertanian*, *12*, 77–83.

*name of corresponding author