Vol. 6, No. 1, April 2025, hal. 9-20

Jurnal Computer Science and Information Technology (CoSciTech)

p-ISSN: 2723-567X
e-ISSN: 2723-5661

http://ejurnal.umri.ac.id/index.php/coscitech/index

Deteksi Dan Klasifikasi Penyakit Daun Tomat Menggunakan ResNet-50

Raynold[*1], Alva Hendi Muhammad[2]
Email: [1]raynold@students.amikom.ac.id, [2]alva@amikom.ac.id

[1]Magister Informatika, Universitas AMIKOM Yogyakarta
[2] Magister Informatika, Universitas AMIKOM Yogyakarta

Diterima: 06 Januari 2025 | Direvisi: 07 Februari 2025 | Disetujui: 30 April 2025

Abstrak

Tomat adalah makanan yang populer diseluruh dunia, terutama di indonesia. Banyak petani tomat yang mengalami gagal panen akibat kurangnya pemahaman dan keterlambatan dalam mengenali penyakit yang menyerang tanaman mereka. Tujuan dari penelitian ini adalah untuk mengidentifikasi dan menilai jenis penyakit pada daun tomat berdasarkan tren, sumber data, metodologi, dan karakteristik yang digunakan dalam mendeteksi penyakit pada daun tomat. Dataset yang digunakan bersumber dari kaggle terdiri dari 10 kelas dan berisikan total 11.000 gambar. Pembagian data yang digunakan terdiri data latih 90% dan data uji 10%. Dilakukan proses augmentasi dan fine-tuning bertujuan mengurangi over fitting. Penelitian ini menggunakan algoritma ResNet-50 untuk mendeteksi dan mengklasifikasikan penyakit pada daun tomat. ResNet akan membandingkan gambar daun untuk mengklasifikasikan nya dengan 10 kelas penyakit yang ada pada dataset. Dari metode resnet didapatkan nilai rata-rata akurasi sebesar 93%. Hal tersebut menunjukkan bahwa metode ResNet-50 untuk klasifikasi gambar dapat menghasilkan akurasi yang akurat dalam memecahkan masalah dunia nyata. Dengan adanya sistem deteksi penyakit daun tomat yang akurat menggunakan ResNet-50, petani dapat mengidentifikasi penyakit lebih cepat dan mengambil tindakan pencegahan atau pengobatan yang tepat, sehingga mengurangi risiko gagal panen dan meningkatkan produktivitas.

Kata kunci: *Deteksi penyakit, Daun Tomat, ResNet-50, Klasifikasi, 10 Kelas Penyakit*

Detection And Classification Tomato Leaf Disease Using ResNet-50

Abstract

Tomatoes are a popular food around the world, especially in Indonesia. Many tomato farmers experience crop failure due to lack of understanding and delays in recognizing diseases that attack their plants. The purpose of this study is to identify and assess the types of diseases on tomato leaves based on trends, data sources, methodologies, and characteristics used in detecting diseases on tomato leaves. The dataset used is sourced from kaggle consisting of 10 classes and contains a total of 11,000 images. The data division used consists of 90% training data and 10% test data. The augmentation and fine-tuning process is carried out to reduce over fitting. This research uses the ResNet-50 algorithm to detect and classify diseases on tomato leaves. ResNet will compare leaf images to classify them with 10 disease classes in the dataset. From the ResNet method, the average accuracy value is 93%. This shows that the ResNet-50 method for image classification can produce accurate accuracy in solving real-world problems. With an accurate tomato leaf disease detection system using ResNet-50, farmers can identify diseases faster and take appropriate preventive or treatment measures, thereby reducing the risk of crop failure and increasing productivity.

Keywords: *Disease detection, Tomato Leaf, ResNet-50, Classification, 10 Disease Classes*

1. PENDAHULUAN

Penyakit pada tumbuhan dapat mempengaruhi penurunan ekonomi suatu daerah secara signifikan. Penyakit yang menyerang tanaman tomat merupakan ancaman utama terhadap produktivitas pertanian dan ketahanan pangan. Untuk mengelola penyakit secara efektif dan meminimalkan potensi kerusakan tanaman, deteksi dini dan identifikasi penyakit secara akurat sangatlah penting[1], [2]. Di negara agraris, mayoritas penduduknya bergantung pada sektor pertanian.

Oleh karena itu, deteksi penyakit tanaman memainkan peran yang sangat diperlukan dalam meningkatkan perekonomian dengan meningkatkan produktivitas tanaman [3], [4]. Adanya cacat eksternal menurunkan harga pangan karena dua alasan. Pertama, cacat lahiriah menjadikan makanan tampak buruk. Kedua, hal ini merupakan indikator rendahnya nilai gizi atau bahkan makanan yang terkontaminasi [5], [6].

Identifikasi yang akurat adalah kunci pengendalian hama dan penyakit yang efektif [4], [7]. Sejumlah teknik deteksi penyakit telah diperkenalkan yang tidak hanya meminimalkan tingkat kesalahan tetapi juga digunakan secara komersial. Mengidentifikasi penyakit dengan mengambil gambar daun dinilai sebagai salah satu solusi menarik untuk mendeteksi penyakit. Gambar dianalisis menggunakan computer vision atau teknik image processing [4].

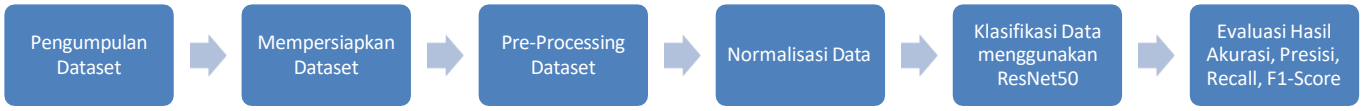
Penelitian ini menggunakan dataset yang terdiri dari 11.000 gambar dengan 10 kelas penyakit, yang mencakup variasi kondisi daun tomat. Hal ini meningkatkan generalisasi model dan memastikan keakuratan dalam skenario dunia nyata. Penelitian ini melakukan augmentasi data dan fine-tuning pada model ResNet-50 untuk mengurangi overfitting dan meningkatkan performa model. Pendekatan ini belum banyak diterapkan dalam konteks deteksi penyakit tanaman, terutama untuk tanaman tomat.

Gambar dianalisis menggunakan computer vision atau image processing [8]. Metode Deep Learning cukup menjanjikan dalam mendiagnosis penyakit tanaman [2]. Convolutional Neural Network (CNNs) merupakan salah satu metode yang sering digunakan dalam pengolahan gambar [8]. CNN bisa digunakan untuk mendeteksi dan mengenali objek pada sebuah gambar karena cara kerja CNN mirip dengan cara kerja visual pada manusia dan hewan[9]. CNN memiliki hasil yang sangat signifikan dalam pengenalan citra dan dengan menggunakan CNN, model tersebut dapat mengenali gambar penyakit pada daun secara akurat [10], [11]. CNN memiliki beberapa arsitektur didalam nya seperti LeNet5, AlexNet, GoogleNet (Inception), dan ResNet. ResNet merupakan salah satu arsitektur CNN yang cukup baik dalam hal akurasi karena menggunakan metode yang berbeda dengan varian CNN lainnya yaitu residual blocks dan skip connection [11]. ResNet telah dilatih untuk dapat mendeteksi kecacatan menggunakan ekstraksi dan kalibrasi.

Penelitian ini mengimplementasikan arsitektur ResNet-50, yang dikenal efektif dalam tugas klasifikasi gambar, untuk mendeteksi dan mengklasifikasikan penyakit pada daun tomat. Hal ini memberikan pendekatan baru yang lebih akurat dibandingkan metode tradisional atau model lain yang sebelumnya digunakan. Pada penelitian ini model ResNet-50 memprediksi klasifikasi penyakit pada daun tomat pada kasus multiclass. Berdasarkan kutipan tersebut dapat ditarik hipotesa jika penelitian ini dapat dilakukan dengan menggunakan metode deep learning yaitu ResNet-50, dengan parameter pengukuran yaitu gambar daun tomat dengan berbagai keadaan.

2. METODE PENELITIAN

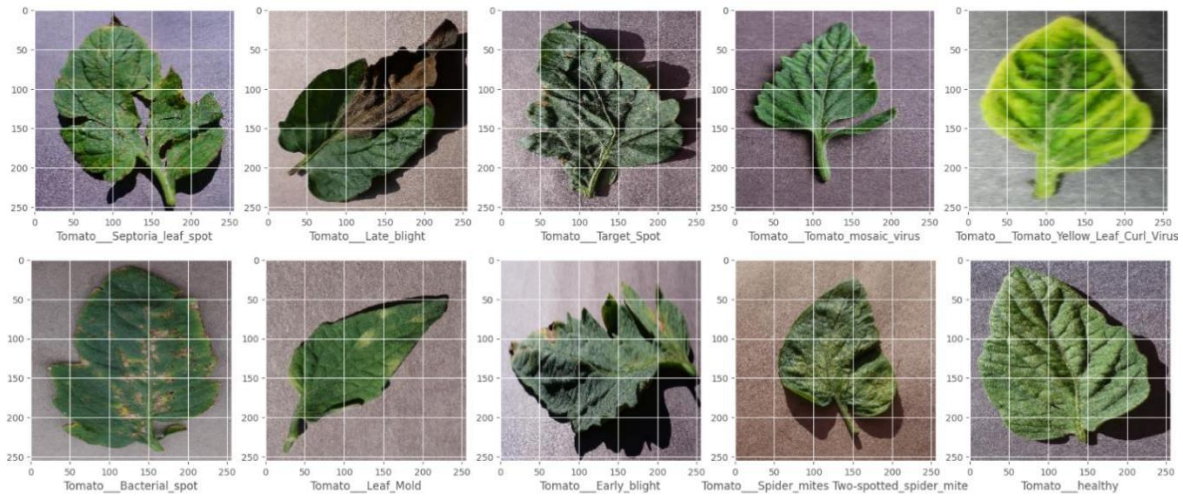
Tujuan utama dalam penelitian ini adalah menerapkan ResNet-50 dalam mengenali dan mengklasifikasikan penyakit pada daun tomat secara akurat dan efisien. Metodologi klasifikasi citra deteksi penyakit melibatkan serangkaian langkah sistematis yang digunakan untuk mengidentifikasi dan membedakan berbagai bentuk daun dalam citra. Proses klasifikasi citra penyakit pada daun tomat dapat digambarkan dalam kerangka penelitian pada gambar berikut.



Gambar 1. Kerangka Penelitian

2.1. Pengumpulan Data

Pada penelitian ini penulis menggunakan dataset yang bersifat publik yang terdapat pada situs Kaggle.com yang berjudul "Tomato Leaf Disease Detection". Pada dataset ini terdapat 11000 gambar yang akan di gunakan dalam penelitian ini. Dataset disimpan pada google drive untuk nantinya akan di proses menggunakan bahasa pemrograman python. Data yang terdapat didalam dataset telah dikelompokkan kedalam 10 *class(Bacterial Spot, Early Blight, Healthy, Late Blight, Leaf Mold, Septoria Leaf Spot, Spider Mites Two Spotted Spider Mite, Target Spot, Mosaic Virus, Yellow Leaf Curl Virus)* yang telah dimasukkan kedalam 2 folder yakni train dan test. Folder *train* digunakan untuk menyimpan data yang akan diproses selama tahap pembelajaran, sementara folder *test* digunakan untuk menguji dan memvalidasi data selama proses pelatihan. Ilustrasi *class* bisa dilihat pada gambar 2.



Gambar 2. *Class Disease*

2.2. Mempersiapkan Dataset

Proses yang dilakukan selanjutnya yaitu penetapan dataset. Penelitian ini menetapkan perbandingan data yaitu 90% : 10%, Dari 11000 gambar yang akan digunakan sebanyak 10.000 gambar untuk data latih (training) dan 1.000 gambar sebagai data uji (test). Masing-masing data train dan data test memiliki 10 kelas penyakit.

2.3. Pre-processing Data

Proses pre-processing merupakan teknik yang digunakan untuk mengubah data mentah dalam format yang berguna dan efisien. Sebelum memulai proses pelatihan data, langkah awalnya adalah mendaftarkan citra ke dalam dataset. Pembagian dataset dilakukan secara rata sehingga setiap kelas memiliki jumlah data pelatihan dan pengujian yang sama yakni masing-masing kelas pada data train memiliki 1000 gambar dan masing-masing kelas pada data test memiliki 100 gambar. Dalam tahapan ini, beberapa aspek dapat dipastikan beberapa hal yaitu, seperti akurasi data, kelengkapan, konsistensi, ketepatan waktu, terpercaya, dan kemampuan interpretasi yang baik.

2.4. Normalisasi Data

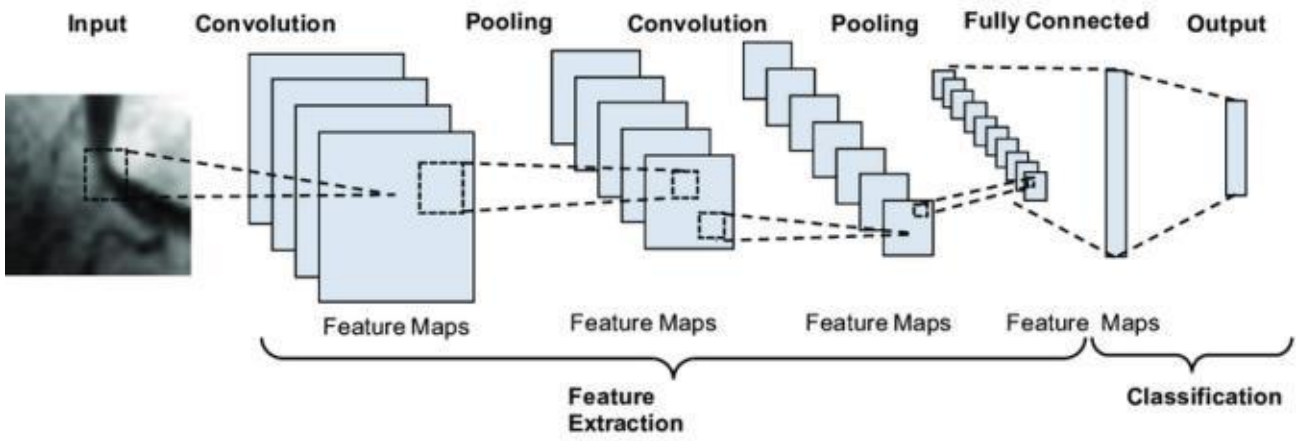
Langkah berikutnya dalam proses ini adalah melakukan normalisasi data menggunakan ImageDataGenerator, yang merupakan kelas dalam modul TensorFlow. Tujuan dari normalisasi ini adalah memastikan bahwa semua nilai piksel berada dalam rentang yang dinormalisasi. Normalisasi dilakukan dengan mengubah skala nilai piksel gambar ke rentang 0-1, dan ini dilakukan dengan mengatur parameter menjadi 1/255. Dengan melakukan penskalaan ulang gambar, model diharapkan tidak terlalu fokus pada fitur tertentu hanya karena fitur tersebut memiliki nilai piksel yang lebih tinggi. Normalisasi ini dapat membantu model belajar secara lebih efektif dan konvergen lebih cepat selama proses pelatihan.

2.5. Klasifikasi Data menggunakan ResNet50

Tomat bisa terjangkit penyakit melalui bakteri, virus, kelainan genetic, dan factor nutrisi [6]. Identifikasi penyakit terhadap daun tomat dapat dilakukan menggunakan gambar. Pada [8] kamera digunakan untuk mengambil data tumbuhan dan kemudian data tersebut di preprocessing menggunakan algoritma regresi dan algoritma-algoritma lainnya.

2.5.1 Convolutional Neural Network

CNN adalah sebuah arsitektur neural network yang terdiri dari layer layer konvolusi dan pooling yang digunakan untuk memproses data berupa gambar atau citra [12]. Konsep dari CNN adalah mengambil fitur fitur penting dari sebuah gambar atau citra dan menghasilkan output yang diharapkan. CNN tidak jauh berbeda dengan neural network lainnya, terdiri dari neuron yang memiliki weight, bias, dan activation function. CNN digunakan untuk menganalisis gambar visual, mendeteksi dan mengenali objek pada gambar, yang merupakan vektor berdimensi tinggi yang akan melibatkan banyak parameter untuk mencirikan jaringan [13]. CNN terdiri dari neuron yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi. Contoh seperti pada gambar 3.

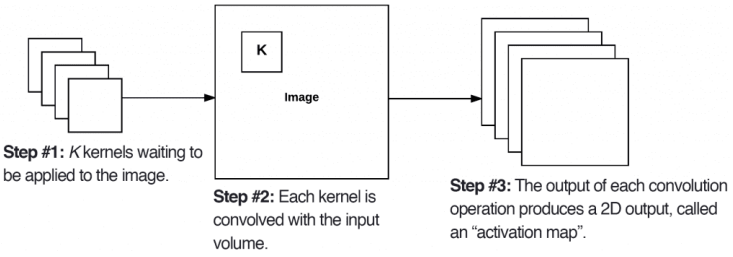


Gambar 3. Arsitektur CNN

Fitur *Extraction Layer* (gambar 5) merupakan proses *encoding* dari sebuah *image* menjadi *features* berupa angka-angka yang merepresentasikan *image* tersebut *(Feature Extraction). Feature extraction* ini terdiri dari dua bagian, yaitu *convolutional layer* dan *pooling layer*.

2.5.1.1 Layer Convolutional

Convolutional layer terdiri dari neuron yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi (pixels). Convolutional layer adalah blok bangunan inti dari jaringan syaraf tiruan. Parameter lapisan convolutional layer terdiri dari sekumpulan K filter yang dapat dipelajari, dimana setiap filter memiliki lebar dan tinggi, dan hampil selalu berbentuk persegi. Filter-filter ini berukuran kecil tetapi meluas ke seluruh kedalaman volume. Untuk input ke CNN, kedalaman adalah jumlah saluran dalam gambar (yaitu, kedalaman tiga saat bekerja dengan gambar RGB, satu untuk setiap saluran). Untuk volume yang lebih dalam di sebuah jaringan, kedalamannya adalah jumlah filter yang diterapkan pada lapisan sebelumnya. Contoh gambar 4.



Gambar 4. Feature Extraction Layer

Kiri: Pada setiap lapisan konvolusi dalam CNN, terdapat K kernel yang diterapkan ke volume input. Tengah: Masing-masing dari K kernel dikonvolusi dengan volume input. Kanan: Setiap kernel menghasilkan output 2D, yang disebut peta aktivasi.

2.5.1.2 Padding

Padding berfungsi untuk mempertahankan ukuran gambar aslinya. Ketika melakukan proses konvolusi, setiap kali setelah selesai operasi konvolusi, ukuran gambar asli akan menyusut, dalam tugas klasifikasi gambar ada beberapa lapisan konvolusi sehingga setelah operasi konvolusi berulang kali, gambar asli akan menjadi kecil. Ketika kernel bergerak diatas gambar asli, kernel menyentuh tepi gambar lebih sedikit dan lebih sering menyentuh bagian tengah dan juga tumpang tindih dibagian tengah, fitur sudut dari gambar apapun atau pada bagian tepi tidak banyak digunakan dalam output.

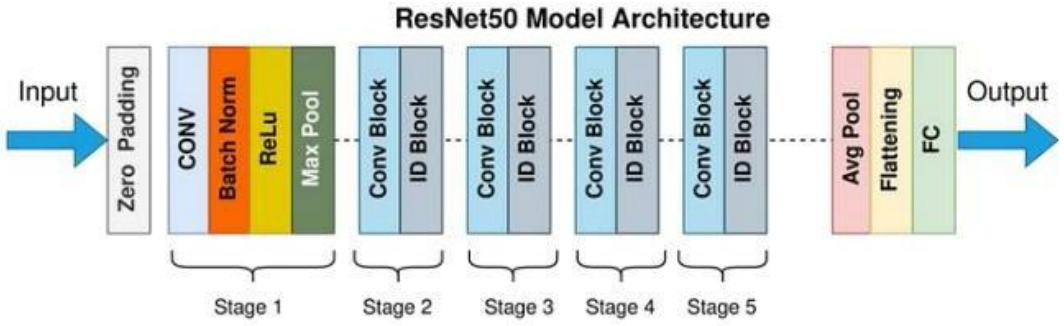
2.5.1.3 TensorFlow

TensorFlow adalah open source library untuk machine learning yang di release oleh Google yang mendukung beberapa bahasa pemrograman (Pallavi et al., 2020). TensorFlow digunakan untuk membuat objek untuk membangun model Machine Learning. Dalam proses Transfer Learning, Tensorflow berperan untuk memproses Inception-v3 Model untuk di training ulang menggunakan data yang baru dan kemudian menghasilkan classifier dengan komputasi yang cepat dan akurasi yang baik. Tensorflow dapat digunakan pada semua sistem operasi.

2.5.2 RESNET-50

ResNet-50 adalah jaringan saraf convolutional yang terdiri dari 50 lapisan. ResNet, kependekan dari Residual Networks adalah jaringan saraf klasik yang digunakan sebagai tulang punggung untuk banyak tugas visi 16 komputer. Terobosan mendasar dengan

ResNet adalah memungkinkan kami untuk melatih jaringan saraf yang sangat dalam dengan 150+ lapisan. Contoh seperti gambar 5.

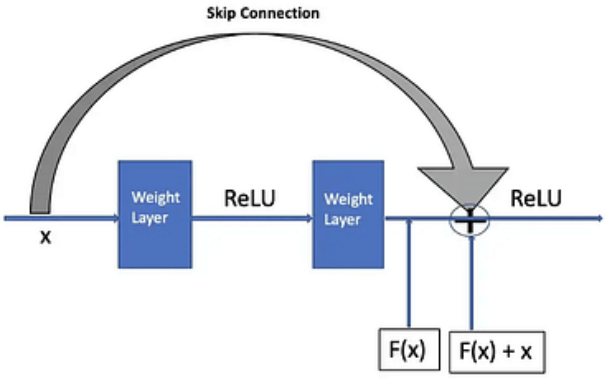


Gambar 5. ResNet-50 Model *Architecture*

Arsitektur ResNet-50 (gambar 7) dibagi menjadi empat bagian utama: lapisan konvolusi, blok identitas, blok konvolusi, dan lapisan yang terhubung sepenuhnya. Lapisan konvolusional bertanggung jawab untuk mengekstraksi fitur dari input gambar, blok identitas dan blok konvolusional memproses dan mengubah fitur-fitur, dan lapisan yang terhubung sepenuhnya membuat klasifikasi akhir.

Meliputi analisis masalah dan desain yang digunakan untuk memecahkan permasalahan. Analisis menggambarkan masalah yang ada dan akan diselesaikan dalam penelitian. Desain menggambarkan bagaimana menyelesaikan permasalahan dan disajikan dalam bentuk diagram dengan penjelasan lengkap. Untuk metode baru harus dijelaskan secara rinci agar peneliti lain dapat mereproduksi percobaan. Sedangkan metode yang sudah mapan bisa dijelaskan dengan memetik rujukan[4-6].

ResNet-50 telah dilatih pada dataset ImageNet yang besar, mencapai tingkat kesalahan yang setara dengan kinerja manusia, menjadikan model yang kuat untuk berbagai tugas klasifikasi gambar seperti deteksi objek, pengenalan wajah, dan analisis gambar medis. Selain itu ResNet-50 juga digunakan untuk ekstraksi fitur-fitur deteksi objek dan segmentasi semantic. ResNet-50 juga memiliki masalah seperti gradien yang hilang, dan Resnet-50 menggunakan skip connection untuk mengatasi masalah tersebut. Contoh seperti gambar 6.



Gambar 6. Skip Connections

Skip connection (Gambar 8) juga dikenal sebagai residual connection yang merupakan fitur utama dari arsitektur ResNet50, koneksi ini digunakan untuk memungkinkan jaringan mempelajari arsitektur yang lebih dalam tanpa mengalami hilangnya gradien. Gradien yang hilang adalah masalah yang terjadi ketika melatih deep neural network, dimana gradien parameter dilapisan yang lebih kecil sehingga menyulitkan lapisan tersebut untuk belajar dan berkembang. Pada ResNet50, skip connections digunakan dalam identity block dan convolutional block. Identity block memberikan input melalui serangkaian lapisan konvosional dan menambahkan input kembali kepada output, sedangkan convolutional block menggunakan lapisan konvolusional 1x1 untuk mengurangi jumlah filter sebelum lapisan konvolusional 3x3 dan kemudian menambahkan input kembali kepada output.

2.7. Hasil Klasifikasi

Klasifikasi dilakukan melalui evaluasi nilai akurasi, yang merupakan suatu matrik untuk mengevaluasi hasil dari model klasifikasi. Akurasi diperoleh dengan membagi jumlah prediksi model yang benar dengan total prediksi yang dilakukan. Proses

klasifikasi dapat juga melibatkan penggunaan Confusion Matrix, yang memberikan informasi tentang kinerja model pada data uji.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (1)$$

Dimana:

TP : True Positive(Data positif yang diklasifikasikan dengan benar)
TN : True Negative(Data negatif yang diklasifikasikan dengan benar)
FP : False Positive(Data negatifyang diklasifikasikan sebagai data positif)
FN : False Negative(Data negatifyang diklasifikasikan sebagai data negatif)

3. HASIL DAN PEMBAHASAN

Bagian ini akan menguraikan langkah-langkah pelaksanaan yang mencakup pengumpulan data, mempersiapkan dataset, pre-processing data, normalisasi data, klasifikasi data menggunakan ResNet50, dan hasil.

3.1 Pengumpulan Data

Dataset yang telah terkumpul di masukkan kedalam direktori folder train yang di dalamnya diberi nama sesuai dengan jenis penyakit nya yaitu *(Bacterial Spot, Early Blight, Healthy, Late Blight, Leaf Mold, Septoria Leaf Spot, Spider Mites Two Spotted Spider Mite, Target Spot, Mosaic Virus, Yellow Leaf Curl Virus)* dengan total data sebanyak 11.000 gambar dengan pixel dan ukuran gambar yang acak sebelum dilakukan proses augmentasi data.

3.2. MempersiapkanDataset

Pada tahap ini dilakukan proses pembagian dataset, perbandingannya ditetapkan 90% : 10%. Dari perbandingan tersebut 11.000 gambar dataset yang ada digunakan sebanyak 10.000 gambar untuk data latih (training) dan 1.000 gambar sebagai data uji (test).Berikut merupakan tabel ringkasan data latih dan data uji.

Tabel 1. Detail Data Train dan Data Test

Kelas	Total Gambar	Data Train (90%)	Data Test (10%)
Bacterial Spot	1.100	1.000	100
Early Blight	1.100	1.000	100
Healthy	1.100	1.000	100
Late Blight	1.100	1.000	100
Leaf Mold	1.100	1.000	100
Septoria Leaf Spot	1.100	1.000	100
Spider Mites Two Spotted Spider Mite	1.100	1.000	100
Target Spot	1.100	1.000	100
Mosaic Virus	1.100	1.000	100
Yellow Leaf Curl Virus	1.100	1.000	100

3.3. Preprocessing Data

Proses augmentasi data merupakan metode umum dalam pelatihan model machine learning, khususnya pada tugas-tugas seperti pengenalan pola atau pengolahan citra dan digunakan untuk melipat gandakan gambar tanpa harus menghilangkan informasi penting dari gambar tersebut. Augmentasi data melibatkan pembuatan variasi tambahan dari dataset pelatihan yang sudah ada dengan menerapkan perubahan kecil, seperti translation, rotations, change in scale, shearing dan horizontal (and some cases, vertical) flips, sehingga menghasilkan dataset yang lebih luas dan lebih beragam. Tujuan utama dari augmentasi data adalah untuk meningkatkan kemampuan generalisasi model dan mengurangi kemungkinan overfitting pada dataset pelatihan yang terbatas. Setelah dilakukan proses augmentasi, gambar datasetnya menjadi akan memiliki ukuran pixel yang sama yaitu 224x224.

```
# Image augmentation
aug = ImageDataGenerator(rotation_range=25, width_shift_range=0.1, height_shift_range=0.1, shear_range=0.2,
                         zoom_range=0.2, horizontal_flip=True, fill_mode="nearest")
```

Preprocessing Fine Tuning

```
[ ] for layer in base_model.layers:
        layer.trainable = False
```

```
[ ] model.compile(optimizer=keras.optimizers.RMSprop(learning_rate=0.01),
                  loss='categorical_crossentropy', metrics=['acc'])
```

```
[ ] datagen = ImageDataGenerator(samplewise_center=False,
                                 samplewise_std_normalization=False,
                                 horizontal_flip=True,
                                 vertical_flip=False,
                                 height_shift_range=0.15,
                                 width_shift_range=0.15,
                                 rotation_range=5,
                                 shear_range=0.01,
                                 fill_mode='nearest',
                                 zoom_range=0.10)

    train_it = datagen.flow_from_directory(train_data_dir,
                                           target_size=(224,224),
                                           color_mode='rgb',
                                           batch_size=32,
                                           class_mode="categorical")

    valid_it = datagen.flow_from_directory(validation_data_dir,
                                           target_size=(224,224),
                                           color_mode='rgb',
                                           batch_size=32,
                                           class_mode="categorical")
```

```
for layer in base_model.layers[-10:]:  # misalnya membuka 10 lapisan terakhir
    layer.trainable = True

# Kompilasi ulang model dengan learning rate yang lebih rendah
model.compile(keras.optimizers.RMSprop(learning_rate=1e-5), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
history = model.fit(
    train_it,
    steps_per_epoch=train_it.samples // train_it.batch_size,
    validation_data=valid_it,
    validation_steps=valid_it.samples // valid_it.batch_size,
    epochs=10  # bisa ditambah tergantung kebutuhan
)
```

Gambar 7. Preprocessing Data (augmentasi dan fine-tuning)

Augmentasi :

Rotation_range=25	: Memutar gambar hingga 25 derajat secara acak
width_shift_range=0.1	: Menggeser gambar ke kanan/kiri hingga 10% dari lebar gambar
height_shift_range=0.1	: Menggeser gambar ke atas/bawah hingga 10% dari tinggi gambar
shear_range=0.2	: Menerapkan transformasi shearing (memiringkan gambar)
zoom_range=0.2	: Memperbesar atau memperkecil gambar hingga 20%
horizontal_flip=True	: Membalik gambar secara horizontal (mirroring)
fill_mode="nearest"	: Mengisi area kosong setelah transformasi dengan metode "nearest" (menggunakan nilai piksel terdekat)

Tujuan penerapan augmentasi yaitu :

a. Meningkatkan jumlah data pelatihan (Mengatasi overfitting)
b. Meningkatkan generalisasi model
c. Mensimulasikan berbagai kondisi dunia nyata

Fine Tuning :

a. Membuat objek ImageDataGenerator

samplewise_center=False	: Tidak melakukan normalisasi dengan mengurangi nilai rata-rata tiap sampel
samplewise_std_normalization=False	: Tidak melakukan normalisasi dengan standar deviasi tiap sampel
horizontal_flip=True	: Membalik gambar secara horizontal (mirroring)
vertical_flip=False	: Tidak membalik gambar secara vertikal
height_shift_range=0.15	: Menggeser gambar secara vertikal hingga 15% dari tinggi gambar
width_shift_range=0.15	: Menggeser gambar secara horizontal hingga 15% dari lebar gambar
rotation_range=5	: Memutar gambar hingga 5 derajat secara acak
shear_range=0.01	: Menerapkan sedikit distorsi shearing (0.01)
fill_mode='nearest'	: Area kosong setelah transformasi diisi dengan nilai piksel terdekat
zoom_range=0.10	: Memperbesar atau memperkecil gambar hingga 10%

b. Membaca Data Pelatihan dengan Augmentasi

train_data_dir	: Path ke direktori dataset pelatihan
target_size=(224,224)	: Ukuran gambar yang digunakan dalam model
color_mode='rgb'	: Gambar berwarna (RGB)
class_mode="categorical"	: Untuk klasifikasi multi-kelas (one-hot encoding)

Membaca gambar dari train_data_dir, menerapkan augmentasi, dan menyiapkan batch data untuk model.

c. Membaca Data Validasi

validation_data_dir	: Path ke direktori dataset validasi
target_size=(224,224)	: Ukuran gambar yang digunakan dalam model
color_mode='rgb'	: Gambar berwarna (RGB)
class_mode="categorical"	: Klasifikasi multi-kelas

Membaca gambar dari validation_data_dir, tetapi biasanya **augmentasi tidak diterapkan pada data validasi** karena validasi digunakan untuk mengevaluasi kinerja model tanpa modifikasi gambar.

3.4 Normalisasi Data

Tujuan dari normalisasi ini adalah memastikan bahwa semua nilai piksel berada dalam rentang yang dinormalisasi. Normalisasi dilakukan dengan mengubah skala nilai piksel gambar ke rentang 0-1, dan ini dilakukan dengan mengatur parameter menjadi 1/255. Dengan melakukan penskalaan ulang gambar, model diharapkan tidak terlalu fokus pada fitur tertentu hanya karena fitur tersebut memiliki nilai piksel yang lebih tinggi. Gambar dibawah merupakan hasil setelah dilakukan proses *rescale data*.

```
from tensorflow import keras
base_model = keras.applications.ResNet50(
    weights='imagenet',
    input_shape=(224,224, 3),
    include_top=False)

Downloading data from https://storage.googleapis.com/te
94765736/94765736 ━━━━━━━━━━━━━━━━━━━━ 1s 0us/step
```

Gambar 7. Rescale Dataset

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(samplewise_center=False,
                             samplewise_std_normalization=False,
                             horizontal_flip=True,
                             vertical_flip=False,
                             height_shift_range=0.15,
                             width_shift_range=0.15,
                             rotation_range=5,
                             shear_range=0.01,
                             fill_mode='nearest',
                             zoom_range=0.10)

train_it = datagen.flow_from_directory(train_data_dir,
                                       target_size=(224,224),
                                       color_mode='rgb',
                                       class_mode="categorical")

valid_it = datagen.flow_from_directory(validation_data_dir,
                                       target_size=(224,224),
                                       color_mode='rgb',
                                       class_mode="categorical")
```

```
Found 10000 images belonging to 10 classes.
Found 1000 images belonging to 10 classes.
```

Gambar 8. Hasil Rescale Dataset

3.5 Klasifikasi Data menggunakan ResNet-50

Suatu tampilan yang disebut Ringkasan Model (model summary) berisi informasi terkait arsitektur model ResNet-50, termasuk jumlah parameter, ukuran setiap layer, dan jumlah output yang dihasilkan pada setiap layer. Ringkasan arsitektur model umumnya digunakan untuk melakukan pemeriksaan guna memastikan bahwa model ResNet-50 telah dibangun dengan benar dan untuk mengidentifikasi masalah yang mungkin perlu diperbaiki sebelum memulai proses pelatihan. Selain itu, Ringkasan Model juga berguna dalam memahami aliran data melalui setiap layer dari model dan kontribusi masing- masing layer dalam melakukan prediksi akhir. Gambar 9 merupakan model dari rancangan klasifikasi ResNet-50.

Model: "functional"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 224, 224, 3)	0	-
conv1_pad (ZeroPadding2D)	(None, 230, 230, 3)	0	input_layer[0][0]
conv1_conv (Conv2D)	(None, 112, 112, 64)	9,472	conv1_pad[0][0]
conv1_bn (BatchNormalization)	(None, 112, 112, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 112, 112, 64)	0	conv1_bn[0][0]
pool1_pad (ZeroPadding2D)	(None, 114, 114, 64)	0	conv1_relu[0][0]
pool1_pool (MaxPooling2D)	(None, 56, 56, 64)	0	pool1_pad[0][0]
conv2_block1_1_conv (Conv2D)	(None, 56, 56, 64)	4,160	pool1_pool[0][0]
conv5_block3_2_relu (Activation)	(None, 7, 7, 512)	0	conv5_block3_2_bn[0][…
conv5_block3_3_conv (Conv2D)	(None, 7, 7, 2048)	1,050,624	conv5_block3_2_relu[0…
conv5_block3_3_bn (BatchNormalization)	(None, 7, 7, 2048)	8,192	conv5_block3_3_conv[0…
conv5_block3_add (Add)	(None, 7, 7, 2048)	0	conv5_block2_out[0][0… conv5_block3_3_bn[0][…
conv5_block3_out (Activation)	(None, 7, 7, 2048)	0	conv5_block3_add[0][0]
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0	conv5_block3_out[0][0]
dropout (Dropout)	(None, 2048)	0	global_average_poolin…
dense (Dense)	(None, 512)	1,049,088	dropout[0][0]
dropout_1 (Dropout)	(None, 512)	0	dense[0][0]
dense_1 (Dense)	(None, 10)	5,130	dropout_1[0][0]

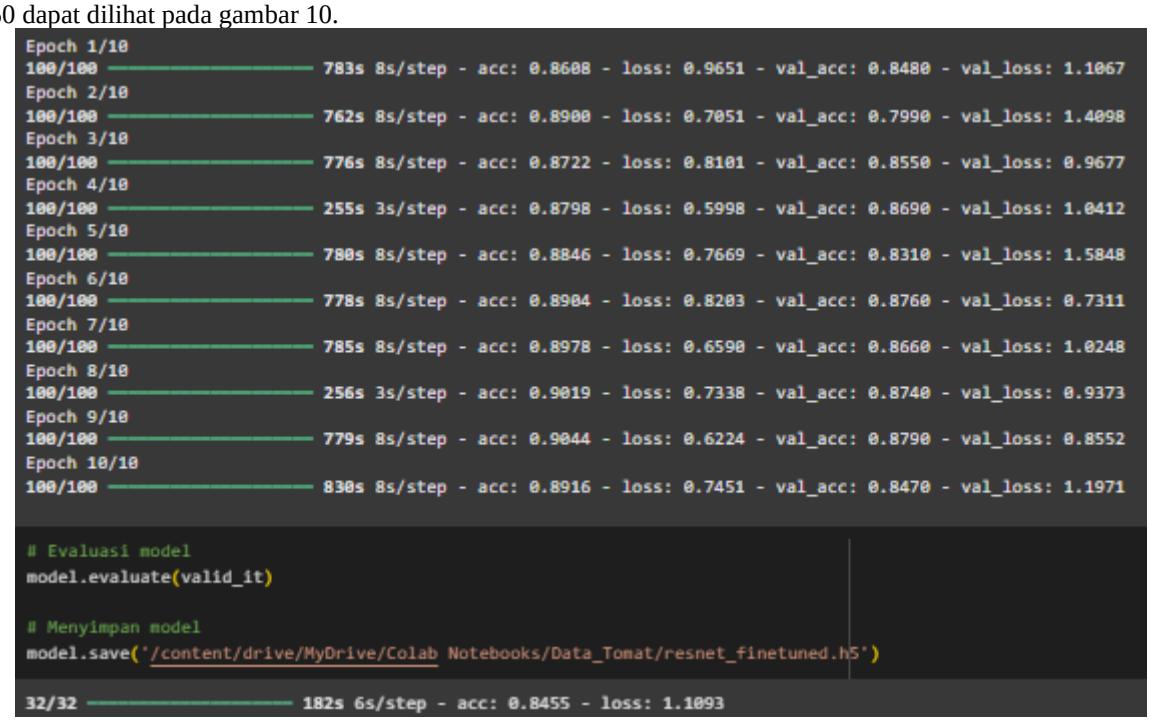
Total params: 24,641,930 (94.00 MB)
Trainable params: 1,054,218 (4.02 MB)
Non-trainable params: 23,587,712 (89.98 MB)

Gambar 9. Model Rancangan Klasifikasi ResNet-50

Keseluruhan, model ini memiliki struktur yang terdiri dari beberapa layer convolutional, global average polling 2D, dan lapisan output Dense untuk klasifikasi dengan 10 kelas. Ukuran total model yang cukup besar (94.00 MB), tetapi jumlah parameter yang dapat dilatih relatif lebih kecil (4.02 MB) dibandingkan dengan total parameter. Ini menunjukkan bahwa sebagian besar kompleksitas model berasal dari convolutional layer, yang diharapkan telah mempelajari fitur-fitur umum dari data yang besar sebelumnya.

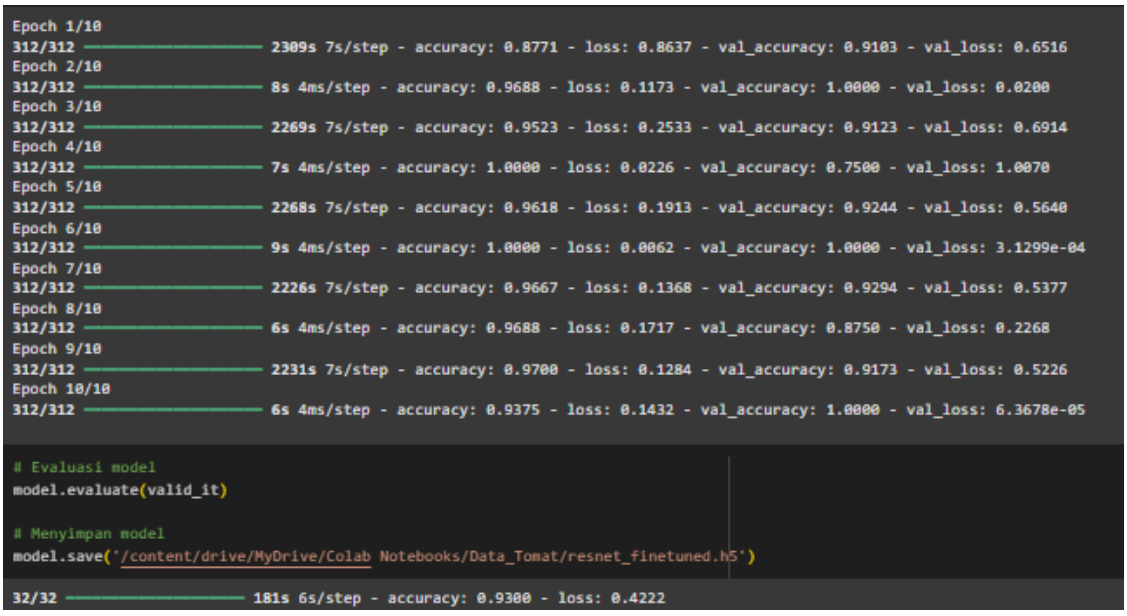
3.6 Proses Pelatihan ResNet-50

Pada proses ini menggunakan data latih dan data uji dengan jumlah iterasi (epoch) yang digunakan sebanyak 10 kali, namun hasil akurasi tertinggi didapatkan pada epoch ke 9. Dalam setiap iterasinya, seluruh data latih dan data uji akan diolah, di mana data latih berasal dari folder pelatihan dan data validasi dari folder pengujian. Setelah proses pelatihan model selesai, output training akan disimpan dalam variabel bernama resnet_finetuned.h5. Variabel ini berisi berbagai informasi, termasuk nilai loss dan akurasi pada setiap iterasi selama pelatihan dan validasi. Data ini berguna untuk mengevaluasi kinerja model dan melakukan penyesuaian yang diperlukan jika diperlukan. Gambar dari proses pelatihan untuk klasifikasi penyakit daun tomat menggunakan ResNet-50 dapat dilihat pada gambar 10.



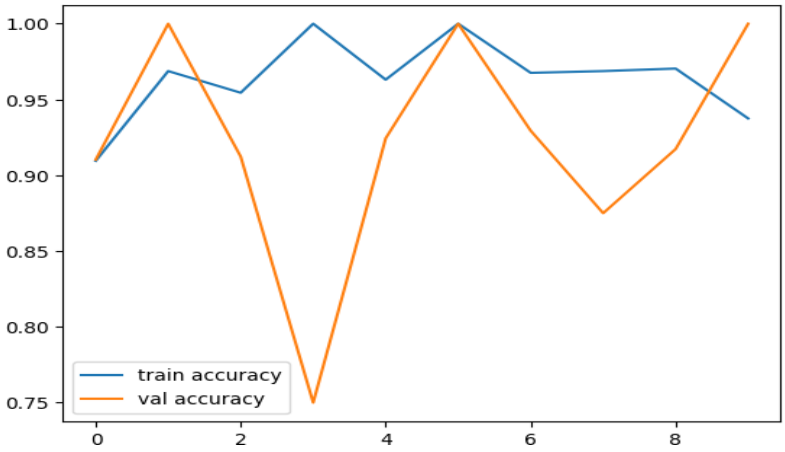
Gambar 10. Hasil Pelatihan Klasifikasi Penyakit Daun Tomat sebelum di Augmentasi

Pada gambar diatas telah dilakukan training dan evaluasi pada data pelatihan penyakit daun tomat dan didapatkan hasil rata-rata akurasi sebesar 84% dengan nilai loss sebesar 1.1093. Pelatihan data ini dilakukan dengan menggunakan 10 epoch.



Gambar 11. Hasil Pelatihan Klasifikasi Penyakit Daun Tomat setelah di Augmentasi

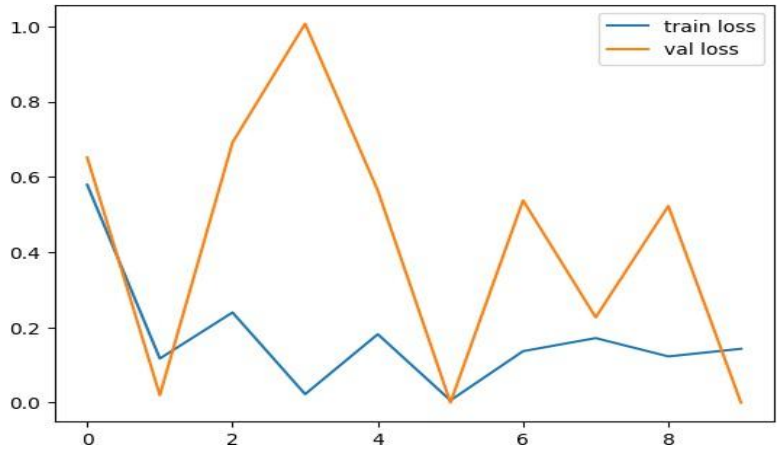
Pada gambar diatas telah dilakukan training pada data pelatihan penyakit daun tomat dan didapatkan hasil rata-rata akurasi sebesar 93% dengan nilai loss sebesar 0.4222. Pelatihan data ini dilakukan dengan menggunakan 10 epoch.



Gambar 12. Grafik Train Akurasi

Grafik Akurasi (Train & Validation Accuracy)

a. Akurasi pelatihan (train accuracy) tinggi dan stabil, mendekati 1.0.
b. Akurasi validasi (val accuracy) berfluktuasi signifikan, menunjukkan ketidakstabilan model terhadap data validasi.
c. Fluktuasi ini bisa menjadi indikasi bahwa model mengalami overfitting, yaitu model terlalu belajar dari data pelatihan tetapi kurang mampu menggeneralisasi ke data validasi.



Gambar 13. Grafik Train Loss

Grafik Loss (Train & Validation Loss)

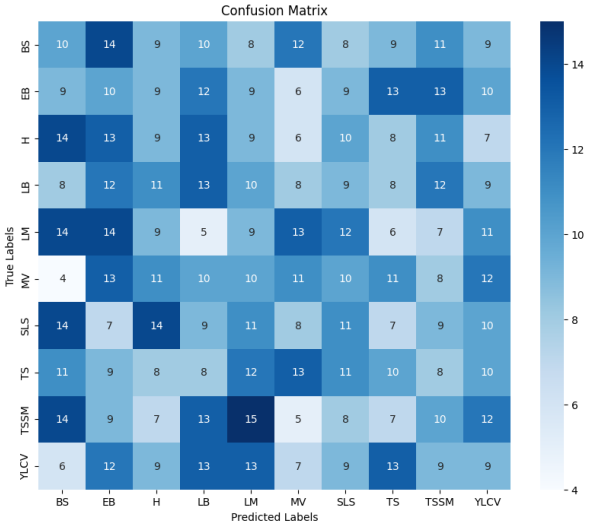
a. Train loss stabil dan rendah, yang menunjukkan bahwa model berhasil mempelajari pola pada data pelatihan.
b. Validation loss sangat berfluktuasi, dengan beberapa titik puncak yang sangat tinggi.
c. Ini mengindikasikan bahwa model kadang-kadang gagal memprediksi data validasi dengan baik, yang bisa berarti model tidak cukup generalizable.

Kesimpulan Konvergensi Model

a. Model sudah berhasil belajar dari data pelatihan, terlihat dari train loss yang rendah dan train accuracy yang tinggi.
b. Model kemungkinan mengalami overfitting, karena validation accuracy dan validation loss berfluktuasi.
c. Perbedaan signifikan antara train dan val loss menunjukkan bahwa model kurang stabil terhadap data baru.

3.7 Confussion Matrix

Alat yang digunakan dalam evaluasi model klasifikasi untuk memvisualisasikan kinerja model dengan membandingkan prediksi model terhadap label sebenarnya (true labels).



Gambar 14. Confussion Matrix

Struktur Confussion Matrix

a. True Labels (Label Sebenarnya): Label yang sebenarnya dari data, biasanya berada di baris.
b. Predicted Labels (Label Prediksi): Label yang diprediksi oleh model, biasanya berada di kolom.
c. Setiap Sel dalam Matriks: Menunjukkan jumlah instance yang diprediksi sebagai suatu kelas tertentu dibandingkan dengan kelas sebenarnya.

4. KESIMPULAN

Berdasarkan hasil dari penelitian yang telah dilakukan, dapat disimpulkan bahwa penerapan metode ResNet-50 dalam klasifikasi penyakit daun tomat dapat menjadi metode yang efektif dengan mendapatkan hasil akurasi sebesar 93%. Hasil ini didapatkan dari pengujian terhadap 10 kelas penyakit yang terdapat didalam dataset. Hasil penelitian ini dapat diterapkan dilapangan dengan memanfaatkan teknologi IOT agar bisa membantu petani dalam mendapatkan data hasil pengujian secara realtime. Untuk mendapatkan akurasi yang lebih baik, pada penelitian selanjutnya dapat dilakukan dengan penerapan metode preprocessing lain agar mendapatkan hasil akurasi yang lebih baik lagi.Dapat juga dengan menerapkan Optimizer yang berbeda untuk membantu menemukan solusi yang lebih stabil.

DAFTAR PUSTAKA

[1] Y. Li, H. Wang, L. M. Dang, A. Sadeghi-Niaraki, and H. Moon, "Crop pest recognition in natural scenes using convolutional neural networks," *Comput. Electron. Agric.*, vol. 169, no. December 2019, p. 105174, 2020, doi: 10.1016/j.compag.2019.105174.

[2] C. Vengaiah and S. R. Konda, "Improving Tomato Leaf Disease Detection with DenseNet-121 Architecture," *Int. J. Intell. Syst. Appl. Eng.*, vol. 11, no. 3, pp. 442–448, 2023.

[3] J. A. Wani, S. Sharma, M. Muzamil, S. Ahmed, S. Sharma, and S. Singh, *Machine Learning and Deep Learning Based Computational Techniques in Automatic Agricultural Diseases Detection: Methodologies, Applications, and Challenges*, vol. 29, no. 1. Springer Netherlands, 2022. doi: 10.1007/s11831-021-09588-5.

[4] H. Kibriya, R. Rafique, W. Ahmad, and S. M. Adnan, "Tomato Leaf Disease Detection Using Convolution Neural Network," in *Proceedings of 18th International Bhurban Conference on Applied Sciences and Technologies, IBCAST 2021*, 2021, pp. 346–351. doi: 10.1109/IBCAST51254.2021.9393311.

[5] J. A. da Costa, Arthur Z.; Figueroa, Hugo E.H.; Fracarolli, "Computer vision based detection of external defects on tomatoes using deep learning," *Biosyst. Eng. 234*, vol. 190, pp. 131–144, 2020.

[6] A. Z. da Costa, H. E. H. Figueroa, and J. A. Fracarolli, "Computer vision based detection of external defects on tomatoes using deep learning," *Biosyst. Eng.*, vol. 190, pp. 131–144, 2020, doi: 10.1016/j.biosystemseng.2019.12.003.

[7] J. Shijie, J. Peiyi, H. Siping, and Sl. Haibo, "Automatic detection of tomato diseases and pests based on leaf images," *Proc. - 2017 Chinese Autom. Congr. CAC 2017*, vol. 2017-Janua, pp. 3507–3510, 2017, doi: 10.1109/CAC.2017.8243388.

[8] C. Vengaiah and S. R. Konda, "INTELLIGENT SYSTEMS AND APPLICATIONS IN Improving Tomato Leaf Disease Detection with DenseNet-121 Architecture," *Int. J. Intell. Syst. Appl. Eng.*, vol. 11, no. 3, pp. 442–448, 2023.

[9] I. Cholissodin and A. A. Soebroto, "AI , MACHINE LEARNING & DEEP LEARNING (Teori & Implementasi)," no. July 2019, 2021.

[10] N. Awalia and A. Primajaya, "Identifikasi Penyakit Leaf Mold Daun Tomat Menggunakan Model DenseNet-121," *J. Ilm. Ilmu Komput.*, vol. 8, no. 1, pp. 49–54, 2022, [Online]. Available: http://ejournal.fikom-unasman.ac.id

[11] S. Suprihanto, I. Awaludin, M. Fadhil, and M. A. Z. Zulfikor, "Analisis Kinerja ResNet-50 dalam Klasifikasi Penyakit pada Daun Kopi Robusta," *J. Inform.*, vol. 9, no. 2, pp. 116–122, 2022, doi: 10.31294/inf.v9i1.13049.

[12] Y. Li, H. Wang, L. M. Dang, A. Sadeghi-Niaraki, and H. Moon, "Crop pest recognition in natural scenes using convolutional neural networks," *Comput. Electron. Agric.*, vol. 169, 2020, doi: 10.1016/j.compag.2019.105174.

[13] P. A. Nugroho, I. Fenriana, and R. Arijanto, "Implementasi Deep Learning Menggunakan Convolutional Neural Network (Cnn) Pada Ekspresi Manusia," *Algor*, vol. 2, no. 1, pp. 12–21, 2020.