Kelompok 12

PROYEK AKHIR MACHINE LEARNING

KLASIFIKASI PENYAKIT PADA DAUN KOPI

Kelompok 12

OUR TEAM

YEN RYLIN HUTASOIT
11422055

MELVAYANA MANIK
11422006

NEHEMIA SITORUS
11422028

JOICE SHARON
11422056

MEGA MALAU
11422027

DAFTAR ISI

LATAR BELAKANG

Kopi adalah komoditas bernilai tinggi, namun penyakit tanaman dapat menurunkan produktivitas.

Identifikasi manual yang lambat dan rawan kesalahan kini dapat digantikan oleh machine learning, seperti CNN dengan arsitektur ResNet-50, untuk deteksi penyakit daun kopi secara cepat, akurat, dan efisien. Teknologi ini mendukung petani meningkatkan hasil panen dan keberlanjutan industri kopi.

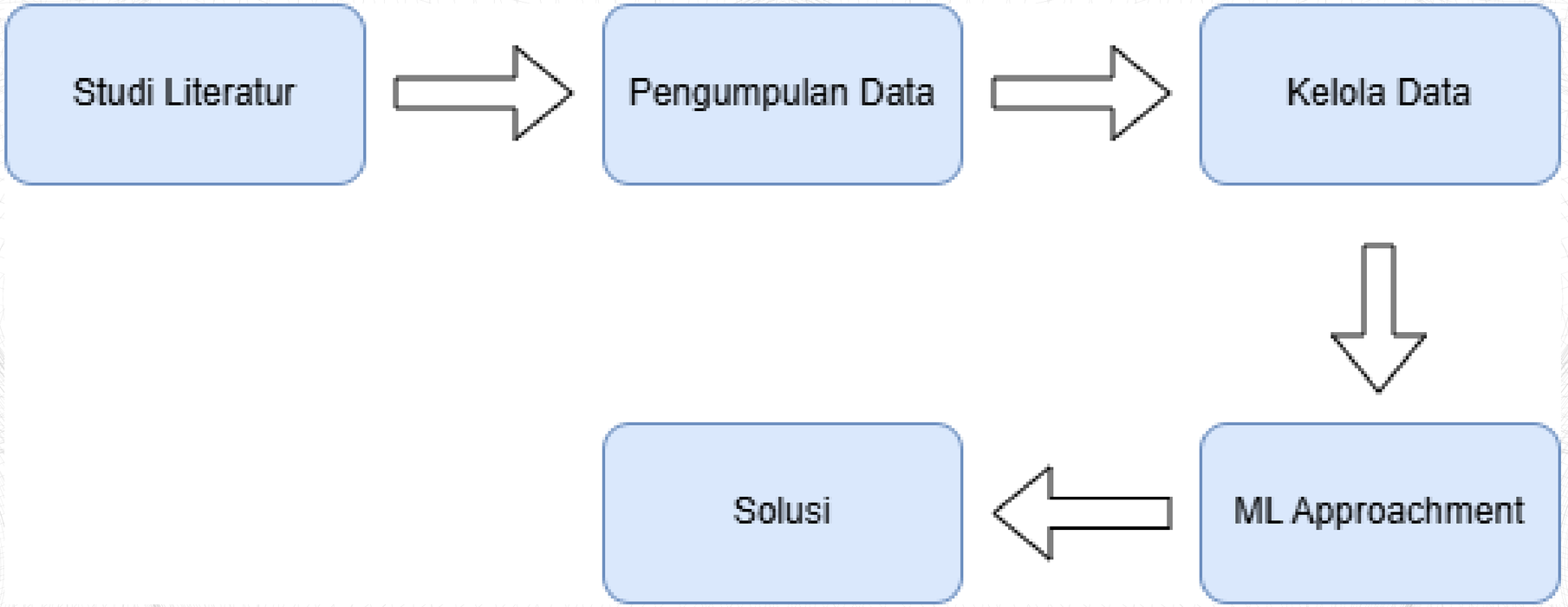
TUJUAN PENELITIAN

Penelitian ini bertujuan untuk meningkatkan efisiensi dalam identifikasi penyakit daun kopi dengan memanfaatkan teknologi machine learning.

Sistem yang dikembangkan bertujuan untuk membantu mendeteksi penyakit yang terdapat pada daun kopi sehingga produktivitas dan hasil panen dapat ditingkatkan.

Penelitian ini juga mendukung keberlanjutan industri kopi melalui penerapan solusi berbasis teknologi yang modern dan andal.

METODE PENELITIAN



DATASET

Terdiri dari 4 Kelas

- Hawar Daun
- Karat Daun
- Bercak Daun
- Pengorok Daun

Augmentasi

- Flip: Horizontal, vertikal
- Rotate 90 derajat
- Rotation between: -45 derajat dan 45 derajat

MODEL

MobileNetV2

Layer (type)	Output Shape	Param #
mobilenetv2_1.00_224 (Functional)	(None, 7, 7, 1280)	2,257,984
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
batch_normalization (BatchNormalization)	(None, 1280)	5,120
dense (Dense)	(None, 64)	81,984
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 4)	260

Total params: 6,851,726 (26.14 MB)
Trainable params: 2,253,188 (8.60 MB)
Non-trainable params: 92,160 (360.00 KB)
Optimizer params: 4,506,378 (17.19 MB)

```python
# Load MobileNetV2 base model
base_model = tf.keras.applications.MobileNetV2(
    weights='imagenet',
    include_top=False,
    input_shape=(IMAGE_SIZE, IMAGE_SIZE, CHANNELS)
)

# Freeze base model
base_model.trainable = False

# Build model
model = Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.BatchNormalization(),
    layers.Dense(64, activation='relu'),  # Tingkatkan jumlah unit
    layers.Dropout(0.5),  # Dropout lebih besar untuk regularisasi
    layers.Dense(N_CLASSES, activation='softmax')
])
```

ResNet50

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 7, 7, 2048)	23,587,712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 4)	8,196

Total params: 23,595,908 (90.01 MB)
Trainable params: 8,196 (32.02 KB)
Non-trainable params: 23,587,712 (89.98 MB)

```python
# Muat model ResNet50 dengan pre-trained weights dan exclude top layer (untuk fine-tuning)
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=input_shape)

# Tambahkan layer tambahan setelah base model
model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(4, activation='softmax')  # 4 kelas output
])
```

PARAMETER TUNNING

MobileNetV2

Learning Rate (initial)	0.0005
Learning Rate (Fine tuning)	0.00001
Learning Rate Scheduler	Mengurangi learning rate setelah epoch ke-3 menggunakan fungsi eksponensial.
ReduceLROnPlateau	Faktor: 0.2, Patience: 1, Min: 1e-6
Early Stopping	Patience: 5, Restore Best Weights
Epochs	8

Batch Size	32
Image Size	(224, 224)
Dropout Rate	0.5
Dense Layer Units	64
Optimizer	Adam
Loss Function	Sparse Categorical Crossentropy
Activation(Output Layer)	Softmax

PARAMETER TUNNING

ResNet50

Epochs	25
batch_size	32
target_size	(224, 224)

PARAMETER TUNNING

MobileNetV2

```
# Set parameters
BATCH_SIZE = 32
IMAGE_SIZE = 224
EPOCHS = 8  # Tambah epochs untuk train lebih lama
CHANNELS = 3
N_CLASSES = 4
```

```
# Fungsi scheduler untuk mengurangi learning rate berdasarkan epoch
def scheduler(epoch, lr):
    if epoch < 3:
        return lr  # Tetap sama di awal
    else:
        return float(lr * tf.math.exp(-0.1))  # Ubah hasil ke tipe float
```

```
# Compile model
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.0005),
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)
```

```
# Re-compile model
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.00001),
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)
```

```
# Callbacks
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=1, min_lr=1e-6)
stop_early = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
lr_scheduler = LearningRateScheduler(scheduler, verbose=1)
```

PARAMETER TUNNING

ResNet50

```
history = model.fit(
    train_images,
    train_labels,
    epochs=25,
    verbose=1,
    validation_data=(val_images, val_labels)
)
```

```
    target_size=(224, 224),
    batch_size=32,
```

EVALUASI PERFORMA MODEL

Oleh : Kelompok 12

MOBILENETV2

Classification Report:

	precision	recall	f1-score	support
bercak-daun	0.73	0.80	0.76	40
hawar-daun	0.54	0.76	0.63	17
karat-daun	0.69	0.48	0.56	23
pengorok-daun	1.00	0.67	0.80	12
accuracy			0.70	92
macro avg	0.74	0.68	0.69	92
weighted avg	0.72	0.70	0.69	92

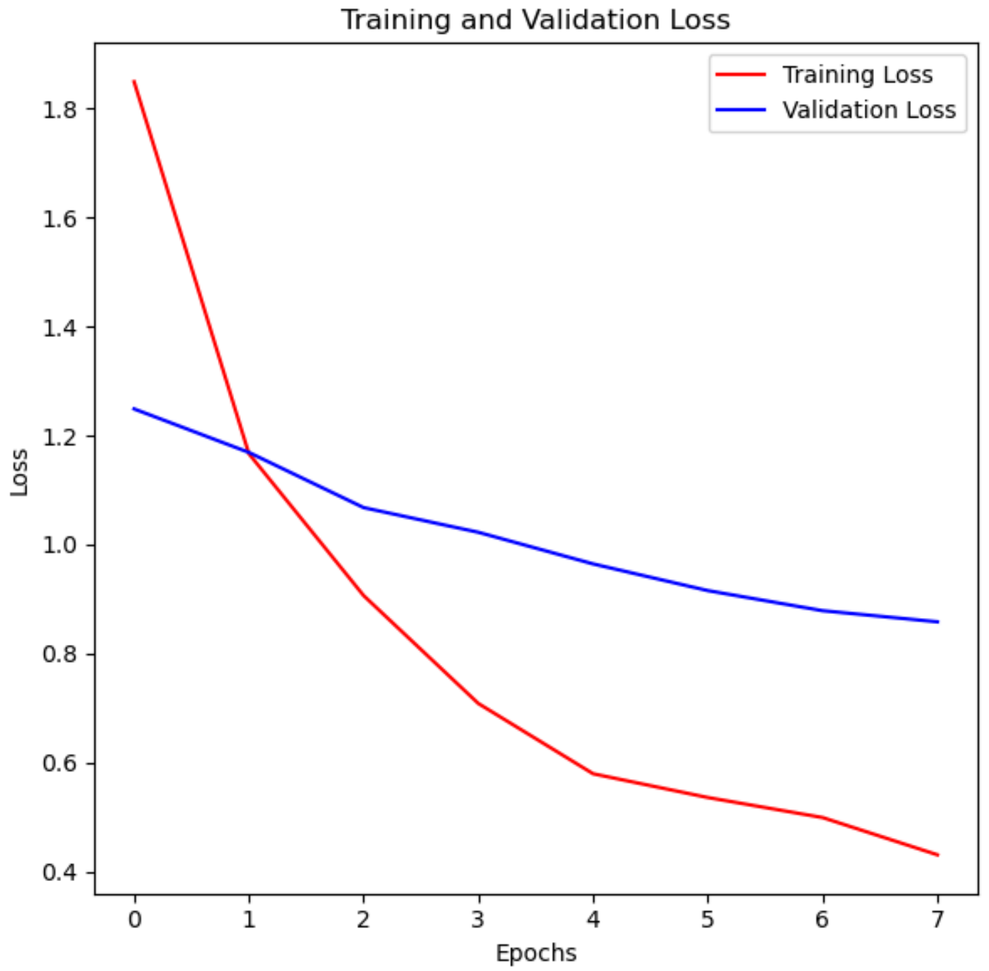
RESNET50

	precision	recall	f1-score	support
0	0.85	0.85	0.85	40
1	0.77	1.00	0.87	17
2	0.90	0.78	0.84	23
3	0.80	0.67	0.73	12
accuracy			0.84	92
macro avg	0.83	0.82	0.82	92
weighted avg	0.84	0.84	0.83	92

MODEL EVALUATION

Oleh : Kelompok 12

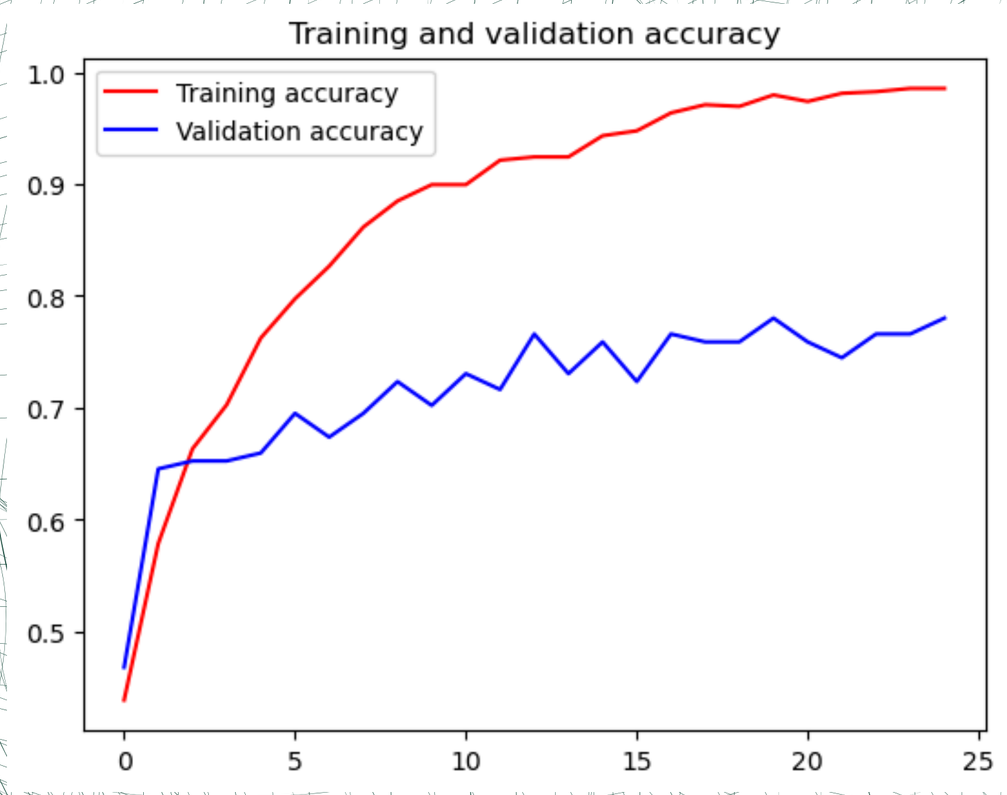
EVALUASI AKURASI MOBILENETV2

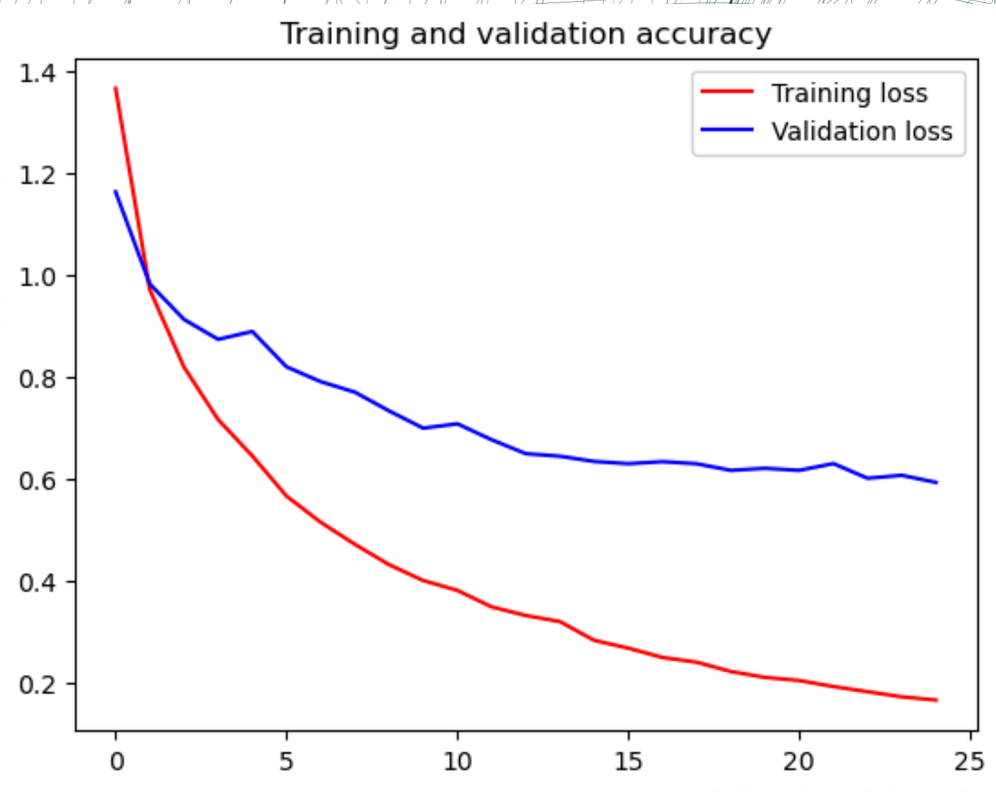


MODEL EVALUATION

Oleh : Kelompok 12

EVALUASI AKURASI RESNET50

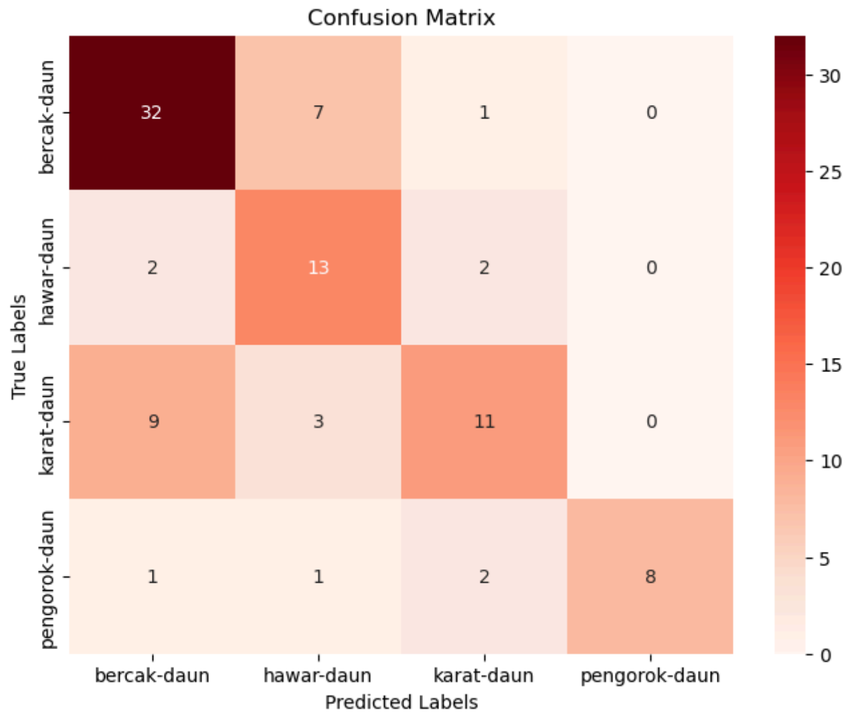




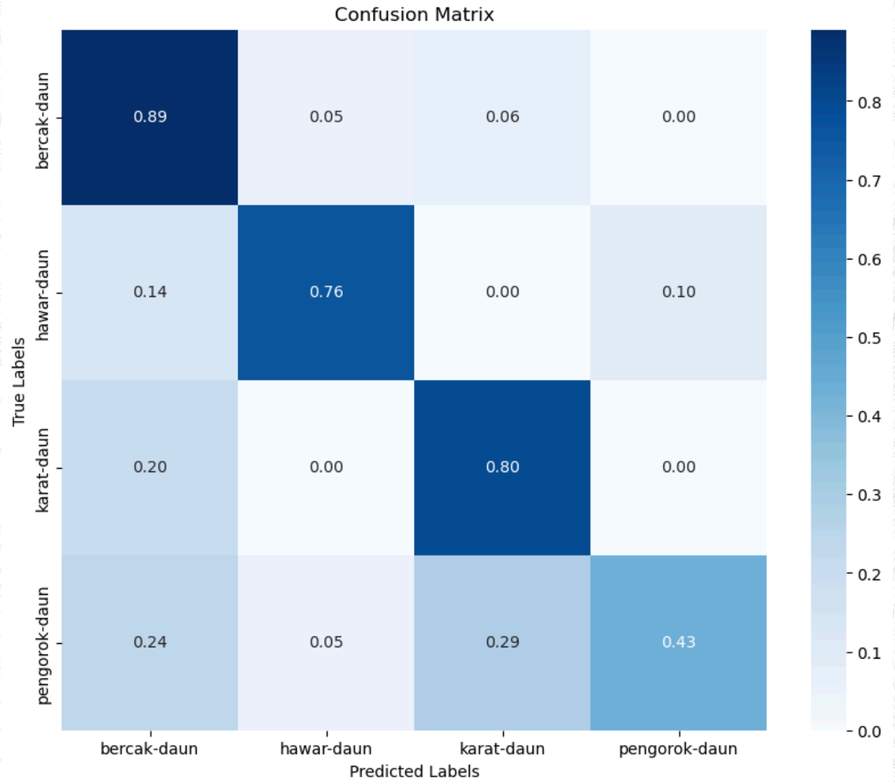
MODEL EVALUATION

CONFUSION MATRIX

MobileNetV2

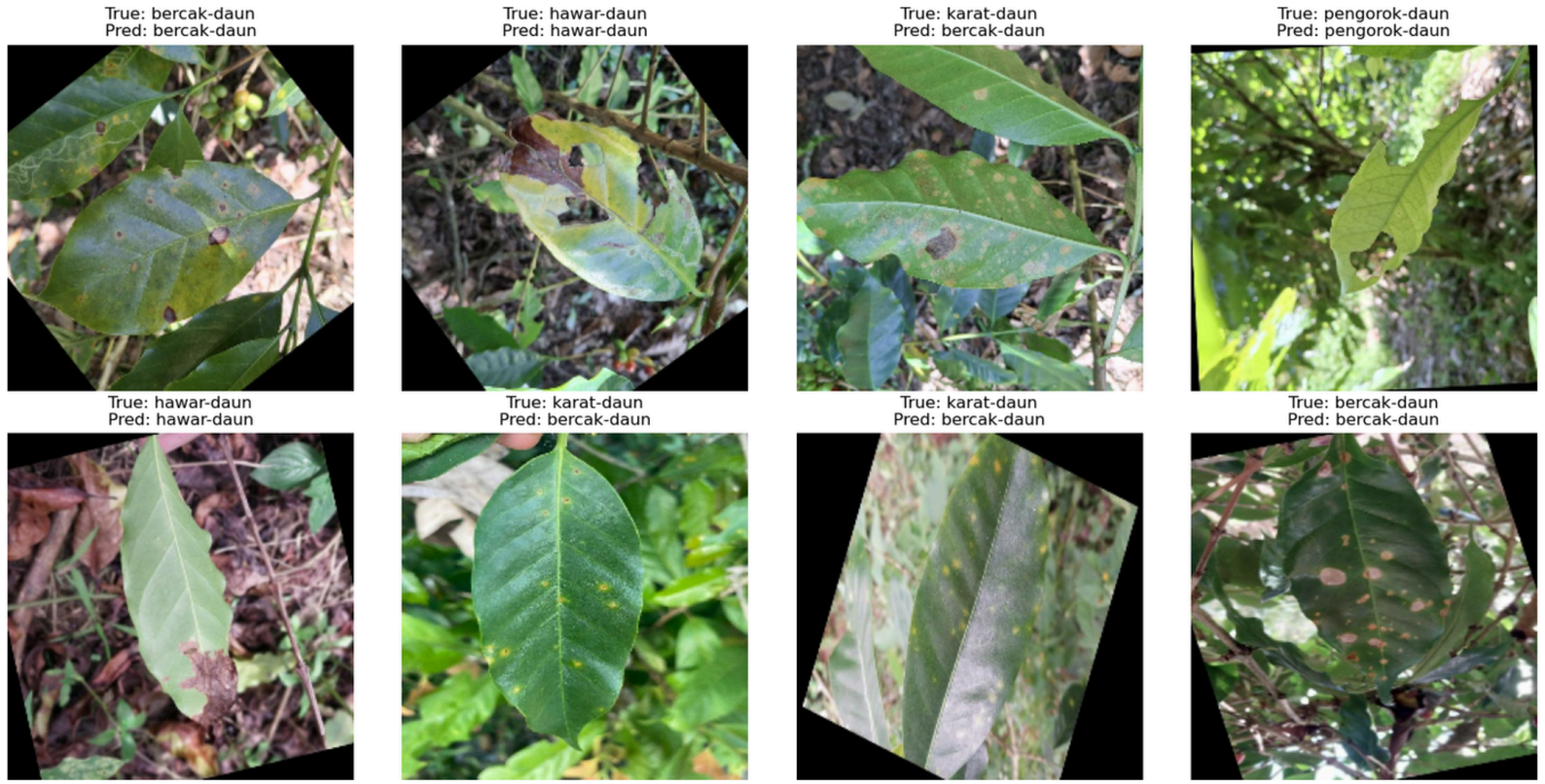


ResNet50

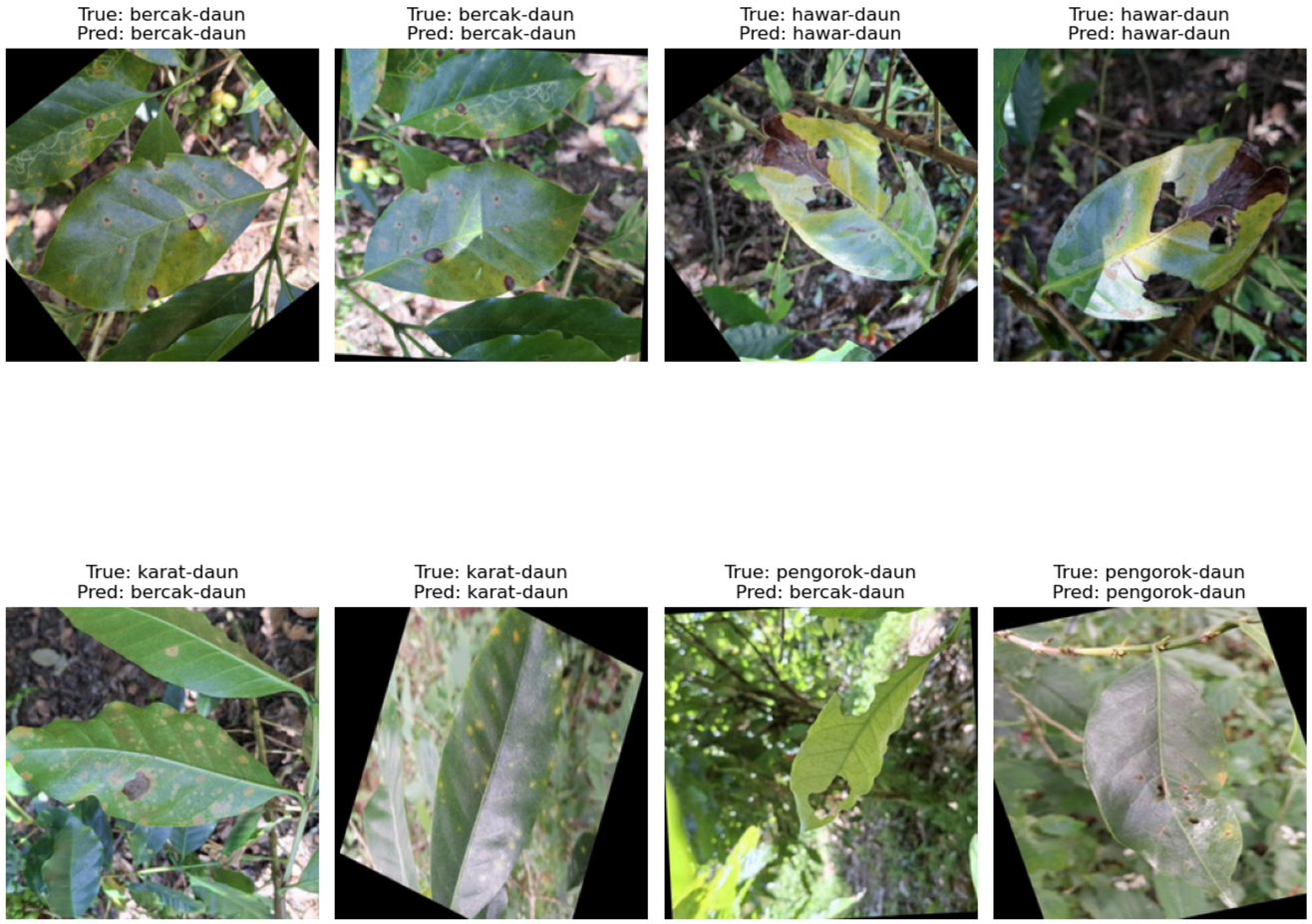


HASIL

MobileNetV2



ResNet50


True: bercak-daun
Pred: bercak-daun
True: bercak-daun
Pred: bercak-daun
True: hawar-daun
Pred: hawar-daun
True: hawar-daun
Pred: hawar-daun
True: karat-daun
Pred: bercak-daun
True: karat-daun
Pred: karat-daun
True: pengorok-daun
Pred: bercak-daun
True: pengorok-daun
Pred: pengorok-daun

- Dalam hal efisiensi, MobileNetV2 lebih unggul untuk dataset kecil atau sederhana karena waktu pelatihannya lebih cepat dan memerlukan sumber daya yang lebih rendah.
- Namun, ResNet-50 menawarkan akurasi yang lebih tinggi pada dataset besar atau kompleks karena arsitekturnya yang lebih canggih.
- Oleh karena itu, MobileNetV2 lebih cocok untuk aplikasi yang membutuhkan pelatihan cepat dengan keterbatasan, sementara ResNet-50 lebih ideal untuk kasus yang memprioritaskan akurasi tinggi dan generalisasi pada data yang kompleks.

Oleh : Kelompok 12

THANK YOU