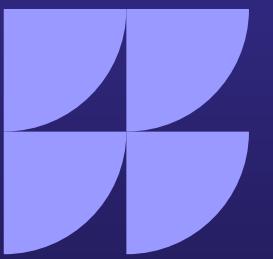


# PALM RECOGNITION

*Financial Technology*

AI



# Partisipan Kelompok



➤ **Defrizal Yahdiyan Risyad**

2206131



➤ **Rakha Dhifiargo Hariadi**

2209489

# Dataset

<https://www.kaggle.com/datasets/saqibshoaibdz/palmprint100people>

The screenshot shows a Kaggle dataset page for 'palmprint100people'. At the top, there's a profile icon for 'SAQIB SHOAIB - UPDATED 7 MONTHS AGO', a file count of '0', a 'Code' button, a 'Download' button, and a three-dot menu. The title 'palmprint100people' is displayed prominently. Below the title are tabs for 'Data Card', 'Code (0)', 'Discussion (0)', and 'Suggestions (0)'. The 'Data Card' tab is selected.

**About Dataset**

No description available

**Usability** 3.75

**License** MIT

**Update frequency** Unspecified

**Tags**

**Data Explorer**

Version 1 (20.74 MB)

Palmprint (4 directories)

**About this directory**

This directory does not have a description yet.

**Suggest Edits**

**Directories:**

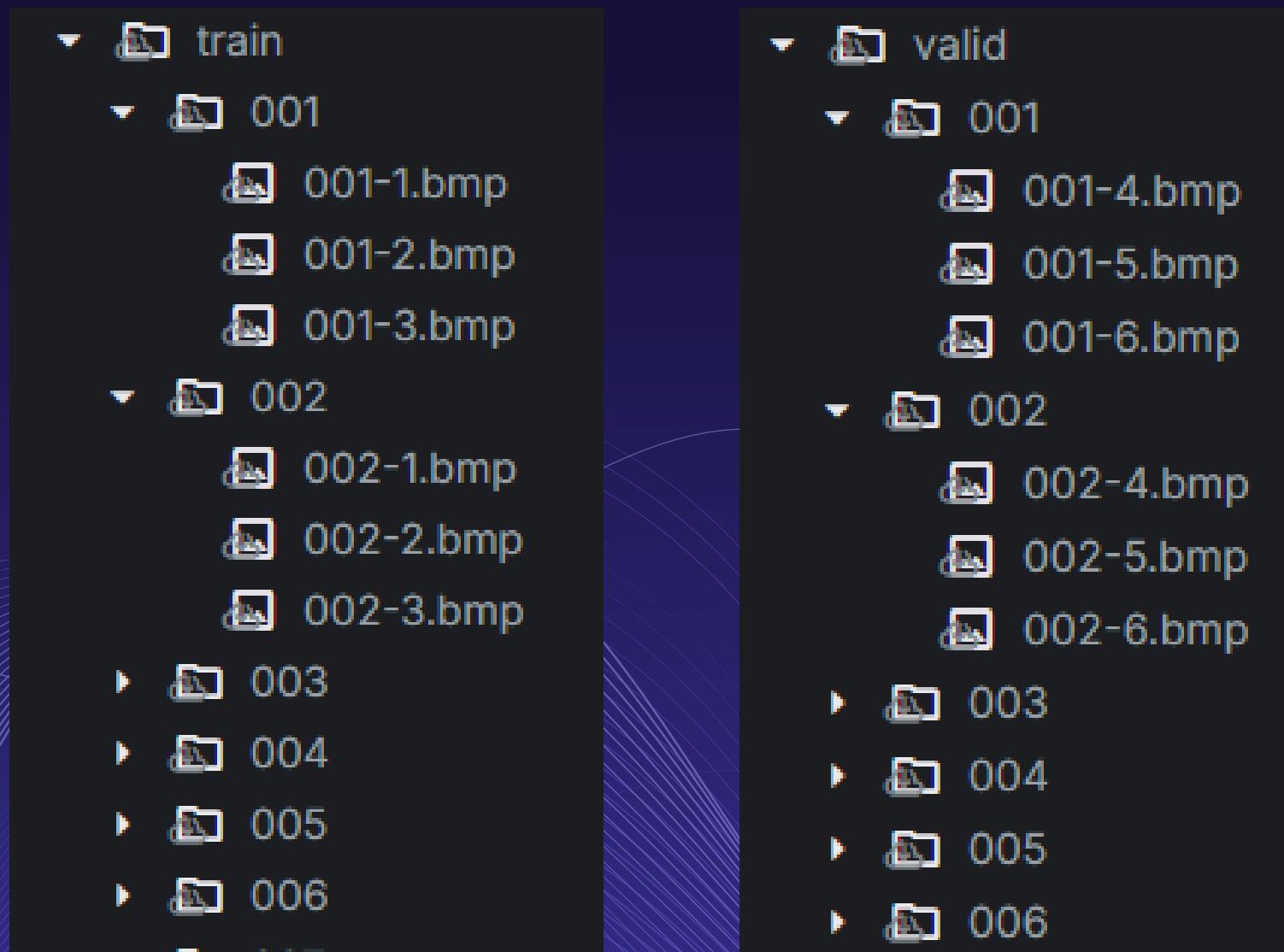
- testing (297 files)
- train (99 directories)
- training (297 files)
- valid (99 directories)

**Summary**

1188 files

# Dataset Task 1

<https://www.kaggle.com/datasets/saqibshoaibdz/palmprint100people>



## Informasi Data Train

Jumlah gambar train: 297

Total Subjek/Label Unik: 99

Shape dari array gambar train: (297, 128, 128, 3)

## Informasi Data Validasi

Jumlah gambar validasi: 198

Total Subjek/Label Unik: 99

## Informasi Data Test

Jumlah gambar uji: 99

Total Subjek/Label Unik: 99

# Dataset Task 1

Sampel Data Train untuk Label [1, 2]

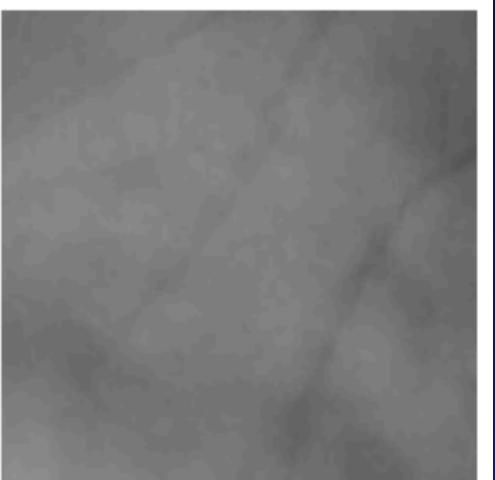
Label: 1



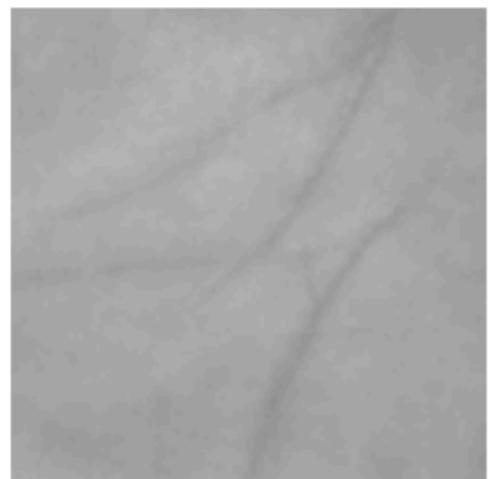
Label: 1



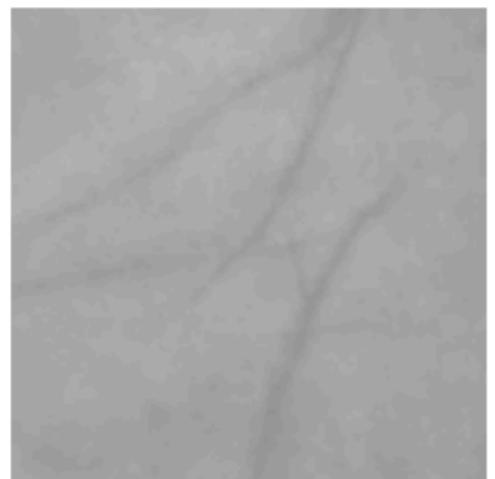
Label: 1



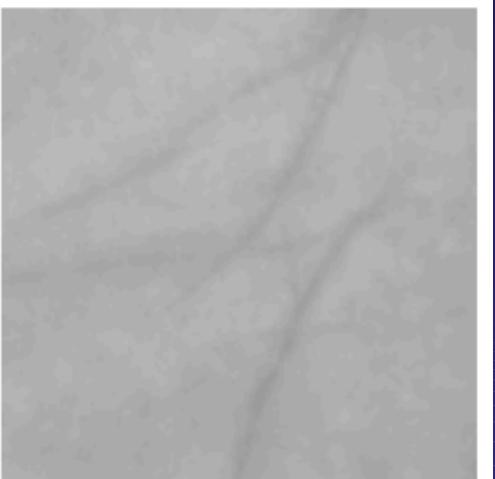
Label: 2



Label: 2



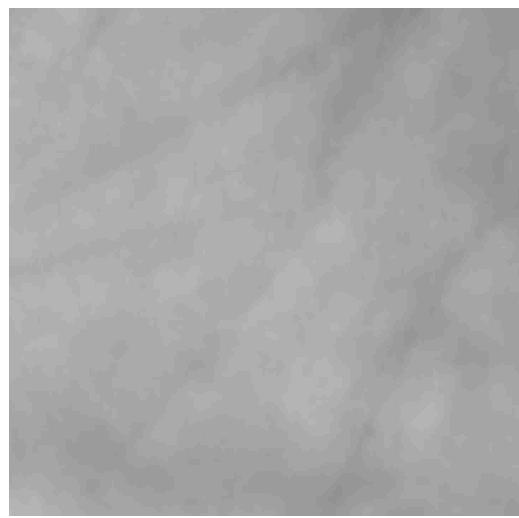
Label: 2



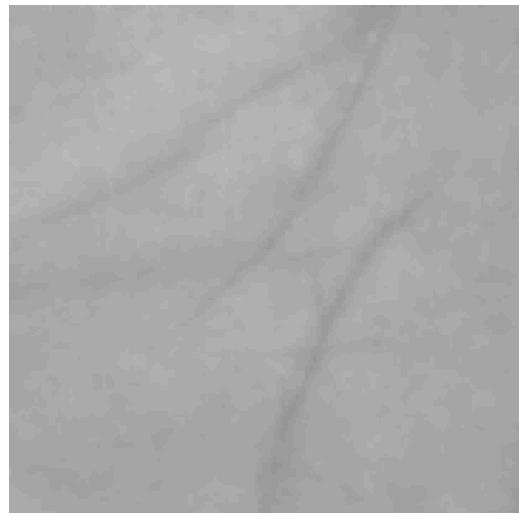
# Dataset Task 1

Sampel Data Validation untuk Label [1, 2]

Label: 1

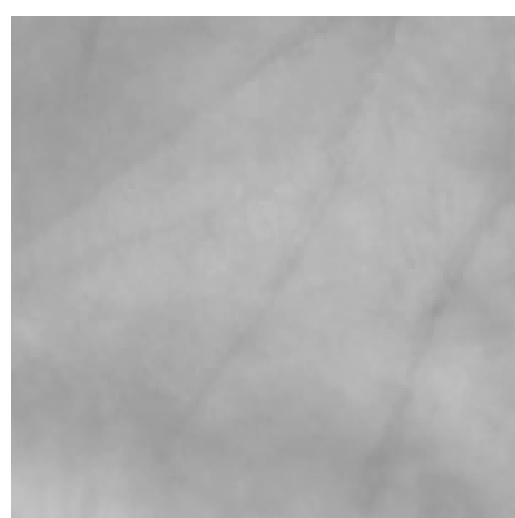


Label: 2

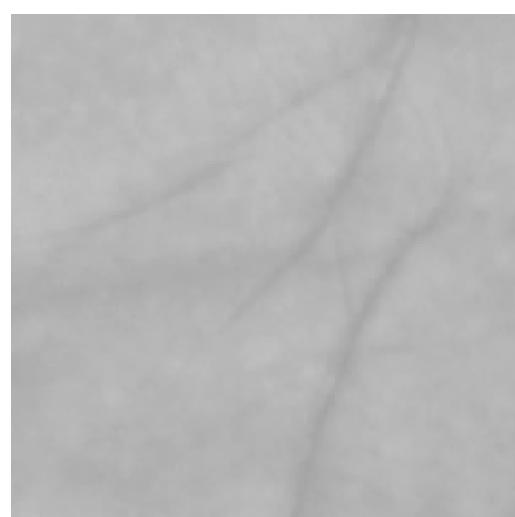


Sampel Data Test untuk Label [1, 2]

Label: 1



Label: 2



# Eksperimen 1 Task 1

## Data Preparation

```
# Normalisasi
X_train_norm = X_train.astype('float32') / 255.0
X_val_norm = X_val.astype('float32') / 255.0
X_test_new_norm = X_test_new.astype('float32') / 255.0

# One-Hot Encoding
num_classes = np.max(y_train) + 1
y_train_cat = to_categorical(y_train, num_classes=num_classes)
y_val_cat = to_categorical(y_val, num_classes=num_classes)
y_test_new_cat = to_categorical(y_test_new, num_classes=num_classes)
```

# Eksperimen 1 Task 1

## Model Preparation

```
input_shape = X_train_norm.shape[1:]  
  
model = Sequential([  
    Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),  
    MaxPooling2D((2, 2)),  
  
    Conv2D(64, (3, 3), activation='relu'),  
    MaxPooling2D((2, 2)),  
  
    Flatten(),  
  
    Dense(128, activation='relu'),  
    # Dropout(0.2),  
  
    Dense(num_classes, activation='softmax')  
])  
  
model.summary()
```

| Layer (type)                   | Output Shape         | Param #   |
|--------------------------------|----------------------|-----------|
| conv2d (Conv2D)                | (None, 126, 126, 32) | 896       |
| max_pooling2d (MaxPooling2D)   | (None, 63, 63, 32)   | 0         |
| conv2d_1 (Conv2D)              | (None, 61, 61, 64)   | 18,496    |
| max_pooling2d_1 (MaxPooling2D) | (None, 30, 30, 64)   | 0         |
| flatten (Flatten)              | (None, 57600)        | 0         |
| dense (Dense)                  | (None, 128)          | 7,372,928 |
| dense_1 (Dense)                | (None, 100)          | 12,900    |

Total params: 7,405,220 (28.25 MB)  
Trainable params: 7,405,220 (28.25 MB)  
Non-trainable params: 0 (0.00 B)

# Eksperimen 1 Task 1

## Eksekusi

```
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=[ 'accuracy'])

history = model.fit(
    X_train_norm,
    y_train_cat,
    epochs=50,
    # batch_size=32,
    validation_data=(X_val_norm, y_val_cat),
)
```

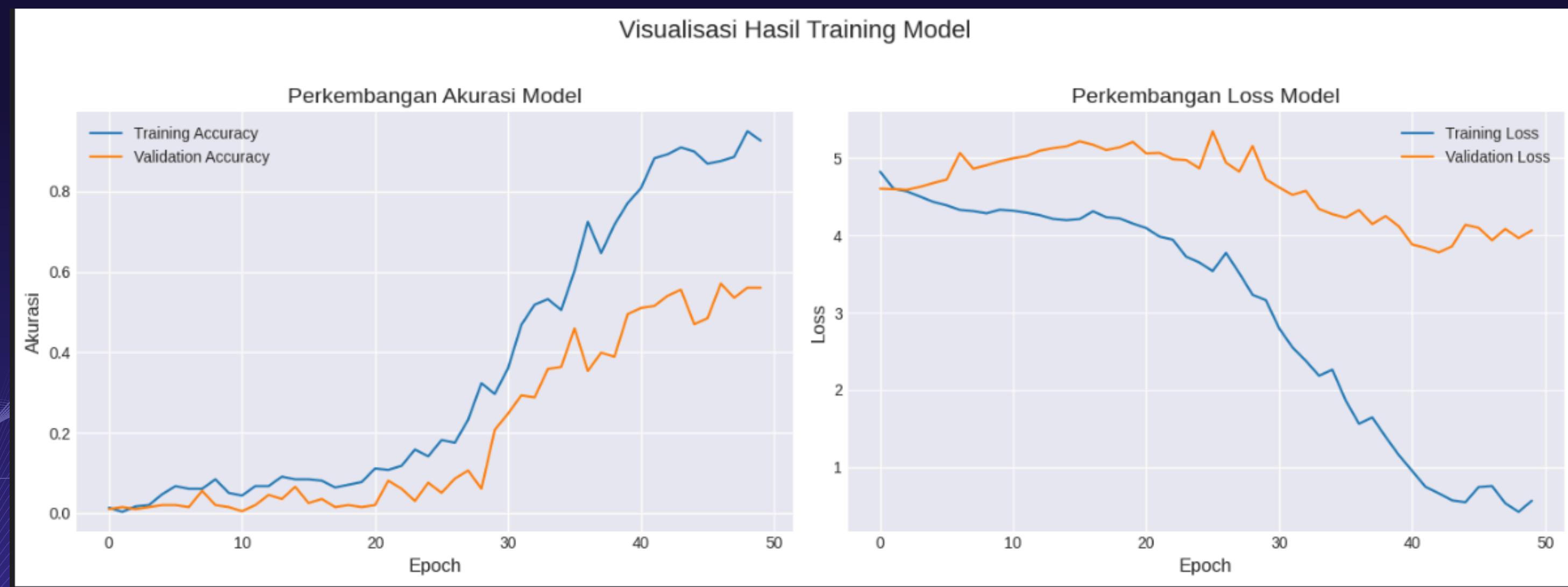
# Eksperimen 1 Task 1

## Eksekusi

```
Epoch 44/50
10/10 ----- 0s 24ms/step - accuracy: 0.9218 - loss: 0.5484 - val_accuracy: 0.5556 - val_loss: 3.8615
Epoch 45/50
10/10 ----- 0s 25ms/step - accuracy: 0.8977 - loss: 0.5309 - val_accuracy: 0.4697 - val_loss: 4.1398
Epoch 46/50
10/10 ----- 0s 24ms/step - accuracy: 0.8753 - loss: 0.7158 - val_accuracy: 0.4848 - val_loss: 4.0992
Epoch 47/50
10/10 ----- 0s 24ms/step - accuracy: 0.8939 - loss: 0.6902 - val_accuracy: 0.5707 - val_loss: 3.9387
Epoch 48/50
10/10 ----- 0s 25ms/step - accuracy: 0.9105 - loss: 0.4505 - val_accuracy: 0.5354 - val_loss: 4.0850
Epoch 49/50
10/10 ----- 0s 24ms/step - accuracy: 0.9654 - loss: 0.3933 - val_accuracy: 0.5606 - val_loss: 3.9675
Epoch 50/50
10/10 ----- 0s 25ms/step - accuracy: 0.9404 - loss: 0.4680 - val_accuracy: 0.5606 - val_loss: 4.0682
```

# Eksperimen 1 Task 1

## Eksekusi



# Eksperimen 1 Task 1

## Evaluasi

|              | 0.00 | 0.00 | 0.00 | 1  |
|--------------|------|------|------|----|
| Subject 98   | 0.00 | 0.00 | 0.00 | 1  |
| Subject 99   | 1.00 | 1.00 | 1.00 | 1  |
| accuracy     |      |      | 0.43 | 99 |
| macro avg    | 0.32 | 0.43 | 0.36 | 99 |
| weighted avg | 0.32 | 0.43 | 0.36 | 99 |

# Eksperimen 2 Task 1

## Eksekusi

```
model_2.compile(optimizer='adam',
                 loss='categorical_crossentropy',
                 metrics=[ 'accuracy'])

history_2 = model_2.fit(
    X_train_norm,
    y_train_cat,
    epochs=100,
    # batch_size=32,
    validation_data=(X_val_norm, y_val_cat),
)
```

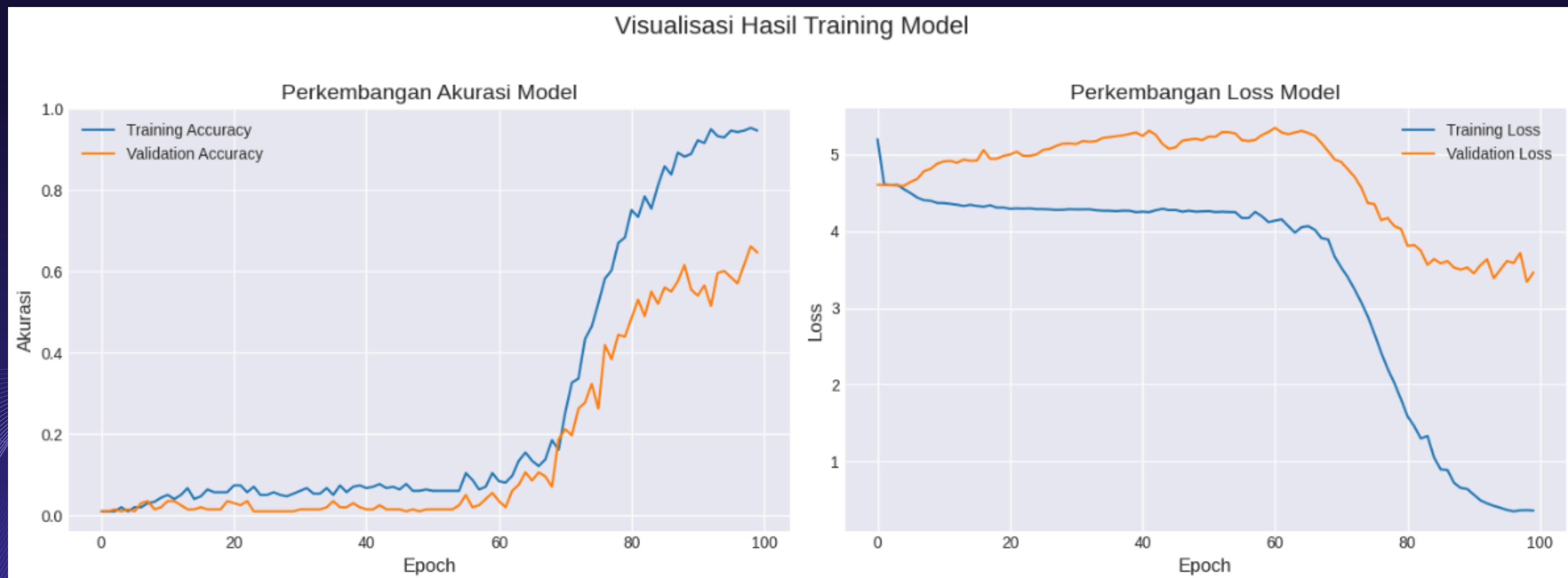
# Eksperimen 2 Task 1

## Eksekusi

```
Epoch 95/100
10/10 ━━━━━━━━ 0s 24ms/step - accuracy: 0.9284 - loss: 0.3730 - val_accuracy: 0.6010 - val_loss: 3.4990
Epoch 96/100
10/10 ━━━━━━━━ 0s 24ms/step - accuracy: 0.9524 - loss: 0.3438 - val_accuracy: 0.5859 - val_loss: 3.6121
Epoch 97/100
10/10 ━━━━━━━━ 0s 24ms/step - accuracy: 0.9623 - loss: 0.3273 - val_accuracy: 0.5707 - val_loss: 3.5856
Epoch 98/100
10/10 ━━━━━━━━ 0s 24ms/step - accuracy: 0.9411 - loss: 0.3678 - val_accuracy: 0.6162 - val_loss: 3.7153
Epoch 99/100
10/10 ━━━━━━━━ 0s 25ms/step - accuracy: 0.9580 - loss: 0.3102 - val_accuracy: 0.6616 - val_loss: 3.3396
Epoch 100/100
10/10 ━━━━━━━━ 0s 24ms/step - accuracy: 0.9510 - loss: 0.3250 - val_accuracy: 0.6465 - val_loss: 3.4634
```

# Eksperimen 2 Task 1

## Eksekusi



# Eksperimen 2 Task 1

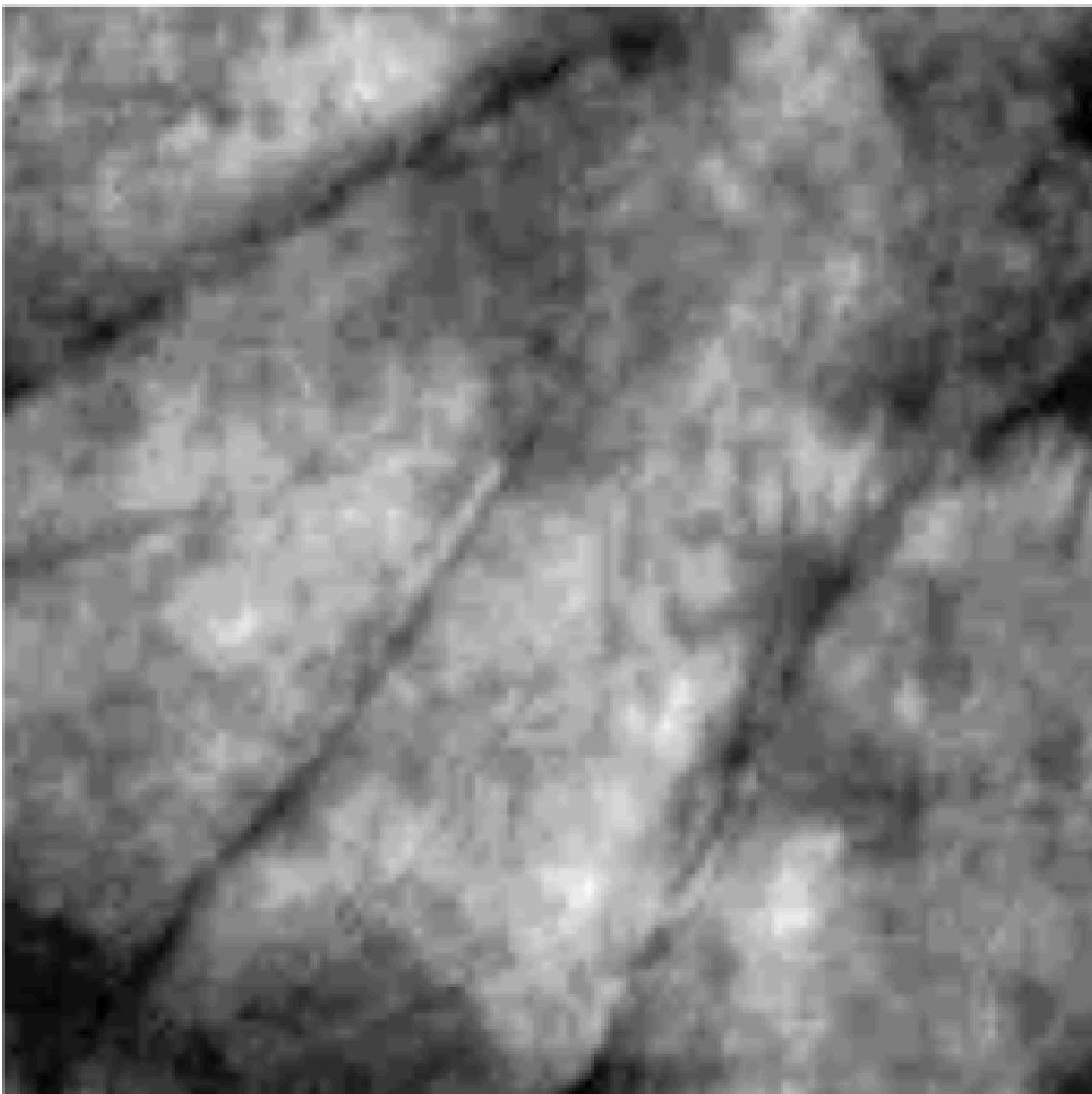
## Evaluasi

|              |      |      |      |    |
|--------------|------|------|------|----|
| Subject 97   | 0.00 | 0.00 | 0.00 | 1  |
| Subject 98   | 0.00 | 0.00 | 0.00 | 1  |
| Subject 99   | 1.00 | 1.00 | 1.00 | 1  |
| accuracy     |      |      | 0.52 | 99 |
| macro avg    | 0.41 | 0.52 | 0.44 | 99 |
| weighted avg | 0.41 | 0.52 | 0.44 | 99 |

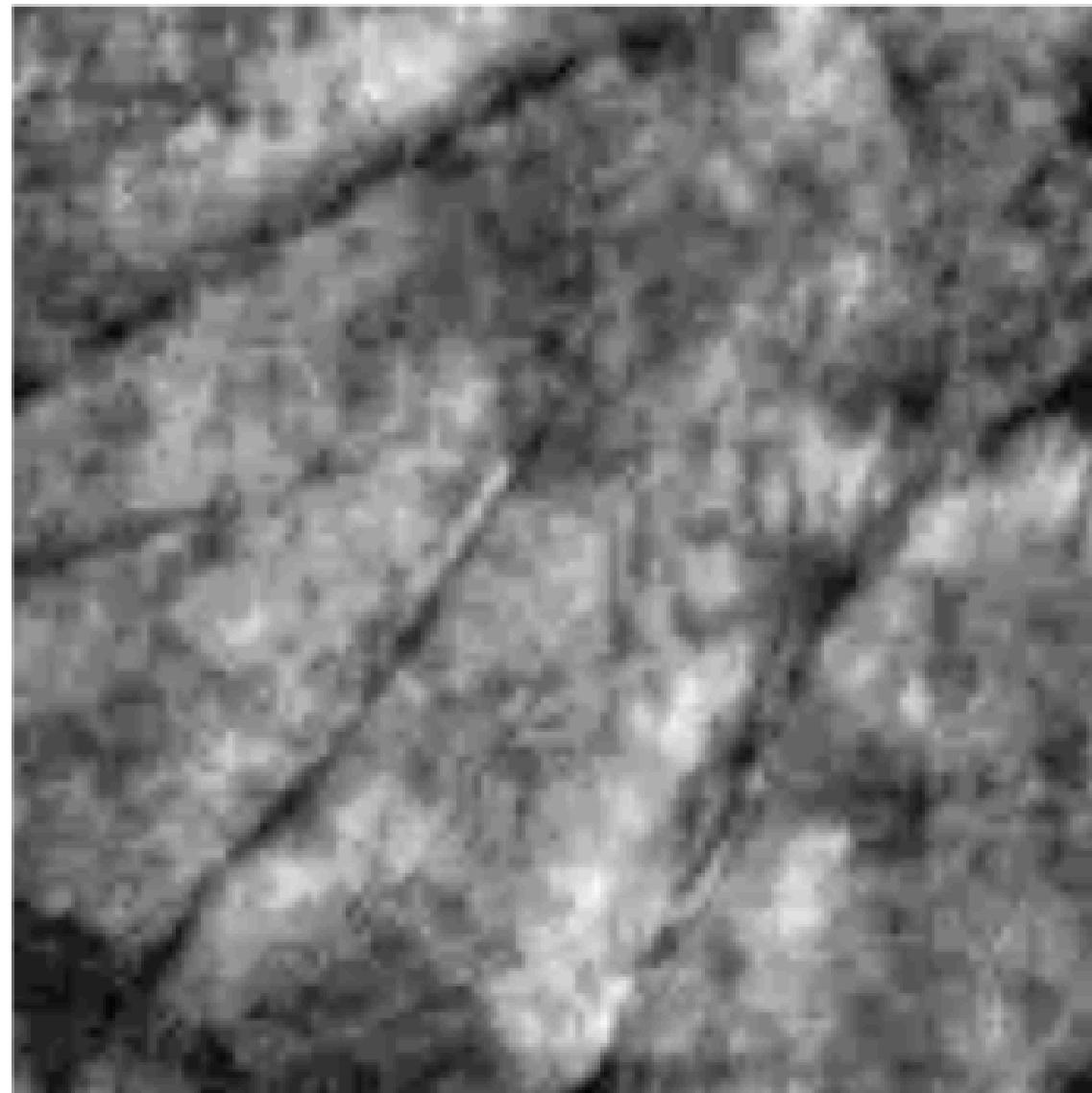
# Eksperimen 3 Task 1

Praproses

Grayscale



Setelah CLAHE



# Eksperimen 3 Task 1

## Data preparation

```
X_train_norm = X_train_clahe.astype('float32') / 255.0
X_val_norm = X_val_clahe.astype('float32') / 255.0
X_test_new_norm = X_test_new_clahe.astype('float32') / 255.0

X_train_reshaped = np.expand_dims(X_train_norm, axis=-1)
X_val_reshaped = np.expand_dims(X_val_norm, axis=-1)
X_test_new_reshaped = np.expand_dims(X_test_new_norm, axis=-1)

num_classes = np.max(y_train) + 1
y_train_cat = to_categorical(y_train, num_classes=num_classes)
y_val_cat = to_categorical(y_val, num_classes=num_classes)
y_test_new_cat = to_categorical(y_test_new, num_classes=num_classes)
```

# Eksperimen 3 Task 1

## Model preparation

```
model_3 = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    # Dropout(0.5),
    Dense(num_classes, activation='softmax')
])

model_3.compile(optimizer='adam',
                 loss='categorical_crossentropy',
                 metrics=['accuracy'])
```

# Eksperimen 3 Task 1

## Eksekusi

```
history_3 = model_3.fit(  
    X_train_reshaped,  
    y_train_cat,  
    epochs=50,  
    # batch_size=32,  
    validation_data=(X_val_reshaped, y_val_cat),  
)
```

# Eksperimen 3 Task 1

## Eksekusi

```
Epoch 45/50
10/10 ----- 0s 21ms/step - accuracy: 1.0000 - loss: 0.0173 - val_accuracy: 0.6111 - val_loss: 3.7349
Epoch 46/50
10/10 ----- 0s 22ms/step - accuracy: 1.0000 - loss: 0.0117 - val_accuracy: 0.6111 - val_loss: 3.7386
Epoch 47/50
10/10 ----- 0s 22ms/step - accuracy: 1.0000 - loss: 0.0120 - val_accuracy: 0.6061 - val_loss: 3.7346
Epoch 48/50
10/10 ----- 0s 22ms/step - accuracy: 1.0000 - loss: 0.0112 - val_accuracy: 0.5909 - val_loss: 3.7534
Epoch 49/50
10/10 ----- 0s 22ms/step - accuracy: 1.0000 - loss: 0.0092 - val_accuracy: 0.6061 - val_loss: 3.7529
Epoch 50/50
10/10 ----- 0s 22ms/step - accuracy: 1.0000 - loss: 0.0098 - val_accuracy: 0.6061 - val_loss: 3.7672
```

# Eksperimen 3 Task 1

## Eksekusi



# Eksperimen 3 Task 1

## Evaluasi

|              | 0.00 | 0.00 | 0.00 | 1  |
|--------------|------|------|------|----|
| Subject 97   | 0.00 | 0.00 | 0.00 | 1  |
| Subject 98   | 0.00 | 0.00 | 0.00 | 1  |
| Subject 99   | 0.00 | 0.00 | 0.00 | 1  |
| accuracy     |      |      | 0.45 | 99 |
| macro avg    | 0.33 | 0.45 | 0.37 | 99 |
| weighted avg | 0.33 | 0.45 | 0.37 | 99 |

# Hasil sementara

Hasil kurang maksimal.

Jumlah data terbatas, 99 label, tapi per label hanya maksimal 6 data.

Transfer learning menjadi solusi.

# Eksperimen 4 Task 1

## Data preparation (MobileNetV3)

```
X_train_rgb = np.stack([X_train_clahe]*3, axis=-1)
X_val_rgb = np.stack([X_val_clahe]*3, axis=-1)
X_test_new_rgb = np.stack([X_test_new_clahe]*3, axis=-1)

X_train_processed = tf.keras.applications.mobilenet_v3.preprocess_input(X_train_rgb)
X_val_processed = tf.keras.applications.mobilenet_v3.preprocess_input(X_val_rgb)
X_test_new_processed = tf.keras.applications.mobilenet_v3.preprocess_input(X_test_new_rgb)

y_train_cat = to_categorical(y_train, num_classes=num_classes)
y_val_cat = to_categorical(y_val, num_classes=num_classes)
y_test_new_cat = to_categorical(y_test_new, num_classes=num_classes)
```

# Eksperimen 4 Task 1

Model preparation (MobileNetV3)

```
base_model = tf.keras.applications.MobileNetV3Large(input_shape=(128, 128, 3),
                                                     include_top=False,
                                                     weights='imagenet')

base_model.trainable = False

x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(128, activation='relu')(x)
# x = Dropout(0.5)(x)
predictions = Dense(num_classes, activation='softmax')(x)

model_4 = Model(inputs=base_model.input, outputs=predictions)

model_4.compile(optimizer='adam',
                 loss='categorical_crossentropy',
                 metrics=['accuracy'])
```

# Eksperimen 4 Task 1

Model preparation (MobileNetV3)

| Layer   | Description | Shape   | Weights               |
|---|-------------|---------|-----------------------|
| global_average_poo...<br>(GlobalAveragePool...) | (None, 960) |         | 0 activation_19[0]... |
| dense_8 (Dense)                                 | (None, 128) | 123,008 | global_average_p...   |
| dense_9 (Dense)                                 | (None, 100) | 12,900  | dense_8[0][0]         |

**Total params:** 3,132,260 (11.95 MB)  
**Trainable params:** 135,908 (530.89 KB)  
**Non-trainable params:** 2,996,352 (11.43 MB)

# Eksperimen 4 Task 1

Model preparation (MobileNetV3)

| Layer   | Description | Shape   | Weights               |
|---|-------------|---------|-----------------------|
| global_average_poo...<br>(GlobalAveragePool...) | (None, 960) |         | 0 activation_19[0]... |
| dense_8 (Dense)                                 | (None, 128) | 123,008 | global_average_p...   |
| dense_9 (Dense)                                 | (None, 100) | 12,900  | dense_8[0][0]         |

**Total params:** 3,132,260 (11.95 MB)  
**Trainable params:** 135,908 (530.89 KB)  
**Non-trainable params:** 2,996,352 (11.43 MB)

# Eksperimen 4 Task 1

Eksekusi

```
history_4 = model_4.fit(  
    X_train_processed,  
    y_train_cat,  
    epochs=25,  
    # batch_size=32,  
    validation_data=(X_val_processed, y_val_cat),  
)
```

# Eksperimen 4 Task 1

## Eksekusi

```
Epoch 22/25
10/10 —————— 0s 24ms/step - accuracy: 1.0000 - loss: 0.1348 - val_accuracy: 0.8131 - val_loss: 1.1283
Epoch 23/25
10/10 —————— 0s 23ms/step - accuracy: 1.0000 - loss: 0.1233 - val_accuracy: 0.7980 - val_loss: 1.1078
Epoch 24/25
10/10 —————— 0s 24ms/step - accuracy: 1.0000 - loss: 0.1152 - val_accuracy: 0.7727 - val_loss: 1.0942
Epoch 25/25
10/10 —————— 0s 22ms/step - accuracy: 1.0000 - loss: 0.0960 - val_accuracy: 0.7778 - val_loss: 1.0932
```

# Eksperimen 4 Task 1

## Eksekusi



# Eksperimen 4 Task 1

## Evaluasi

|              |             |             |             |    |
|--------------|-------------|-------------|-------------|----|
| Subject 98   | <b>1.00</b> | <b>1.00</b> | <b>1.00</b> | 1  |
| Subject 99   | <b>1.00</b> | <b>1.00</b> | <b>1.00</b> | 1  |
| accuracy     |             |             | <b>0.72</b> | 99 |
| macro avg    | <b>0.62</b> | <b>0.72</b> | <b>0.65</b> | 99 |
| weighted avg | <b>0.62</b> | <b>0.72</b> | <b>0.65</b> | 99 |

# Kesimpulan

Pada kasus dataset ini, data terlalu sedikit untuk dilakukan training dengan metode biasa, sehingga harus memanfaatkan pre-trained model..

Penggunaan MobileNetV3 menjadi pilihan dikarenakan asumsi jika nanti inferensi menggunakan perangkat mobile.

Dan jika dibandingkan melalui efisiensi, jelas metode transfer learning merupakan pilihan terbaik mengingat waktu training hampir sama dengan metode konvensional dengan akurasi yang sangat jauh.

**Thank You  
for Your Attention**

