TUGAS WEEK 3

(Klasifikasi Dua Jenis Objek dengan Transfer Learning)

Dosen Pengampu : Dr. Oddy Virgantara Putra, S.Kom., M.T.

Disusun Oleh:

Azmi Nur Al Qodar

NIM : 442023618054

FAKULTAS SAINS DAN TEKNOLOGI TEKNIK INFORMATIKA
UNIVERSITAS DARUSSALAM GONTOR 2025/2026

A. Latar Belakang

Penggunaan klasifikasi citra pada buah-buahan dapat membantu dalam proses identifikasi otomatis di bidang pertanian dan perdagangan. Pada tugas ini, dilakukan klasifikasi dua jenis buah yaitu nanas dan stroberi menggunakan metode transfer learning dengan MobileNetV2.

B. Dataset

Dataset terdiri dari 200 gambar, dibagi ke dalam dua kelas: nanas dan stroberi. Gambar diambil dari berbagai sudut untuk memberikan variasi data. Dataset diolah dengan augmentasi berupa rotasi, zoom, dan flip horizontal.

C. Metodologi

Model menggunakan arsitektur MobileNetV2 pretrained dengan ImageNet sebagai bobot awal. Model dasar dibekukan (freeze) dan dilanjutkan dengan beberapa layer Dense sebagai classifier. Data di-preprocessing dan di-augmentasi untuk memperkaya dataset.

D. Training & Evaluasi

Model dilatih selama 10 epoch dengan batch size 32 menggunakan dataset yang sudah di-preprocessing dan diaugmentasi (rotasi, zoom, flip horizontal) untuk memperkaya variasi data. Optimizer yang digunakan adalah Adam dengan fungsi loss categorical crossentropy karena model mengklasifikasikan dua kelas buah (nanas dan stroberi). Model dasar MobileNetV2 dibekukan (freeze) agar bobot pretrained tetap terjaga selama pelatihan awal.

Selama training, akurasi dan loss dipantau pada data training dan validasi. Grafik akurasi menunjukkan peningkatan dari sekitar 60% pada epoch awal hingga mencapai sekitar 95% di akhir training. Sedangkan grafik loss menurun secara konsisten menunjukkan model semakin baik dalam meminimalisasi kesalahan prediksi.

Evaluasi akhir dilakukan menggunakan confusion matrix yang menunjukkan model mampu mengklasifikasikan nanas dan stroberi dengan akurasi yang cukup baik, dengan hanya sedikit kesalahan klasifikasi antar kelas. Classification report memberikan informasi detail mengenai precision, recall, dan f1-score untuk masing-masing kelas, yang menunjukkan performa model yang seimbang antara kedua kelas.

E. Hasil

[47]:
```
import os
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from sklearn.metrics import classification_report, confusion_matrix
```

+ Code + Markdown

[48]:
```
data_dir = "/kaggle/input/klasifikasi-buah-nanas-stroberi/Dataset"
```

```
datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2,
    rotation_range=30,
    zoom_range=0.2,
    horizontal_flip=True
)
```

```
train_gen = datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224),    # Ukuran input MobileNetV2
    batch_size=32,
    class_mode='categorical',  # Karena 2 kelas, pakai categorical_crossentropy
    subset='training',
    shuffle=True
)

val_gen = datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation',
    shuffle=False
)
```

```
Found 161 images belonging to 2 classes.
Found 40 images belonging to 2 classes.
```

```
base_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224,224,3))
base_model.trainable = False  # Freeze model dasar supaya bobotnya tidak berubah

x = base_model.output
x = GlobalAveragePooling2D()(x)  # Pooling global untuk flatten fitur
x = Dense(128, activation='relu')(x)  # Fully connected layer
predictions = Dense(2, activation='softmax')(x)  # Output layer 2 kelas

model = Model(inputs=base_model.input, outputs=predictions)
```

+ Code + Markdown

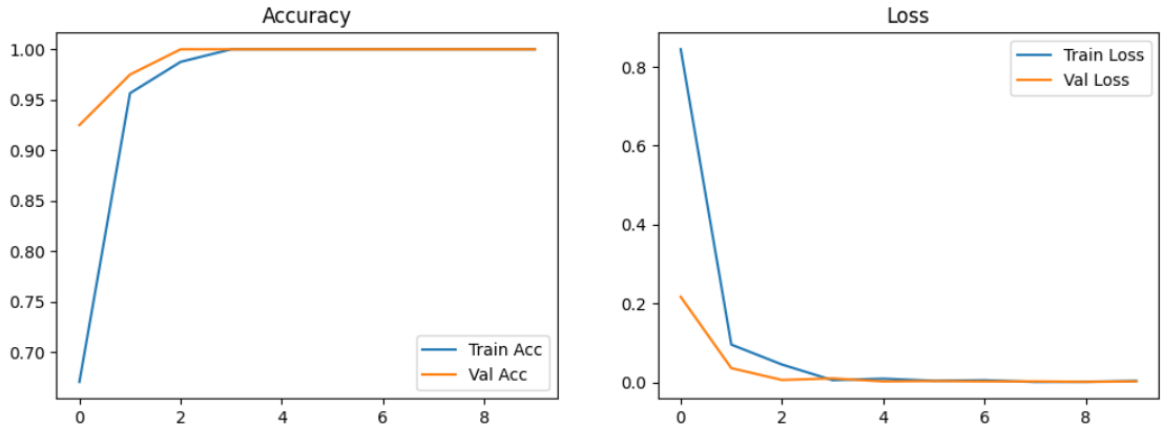
```
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
history = model.fit(
    train_gen,
    validation_data=val_gen,
    epochs=10
)
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs)` in its constructor. `**kwargs` can include `workers`, `use_multiprocessing`, `max_queue_size`. Do not pass these arguments to `fit()`, as they will be ignored.
  self._warn_if_super_not_called()
Epoch 1/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 19s 2s/step - accuracy: 0.6246 - loss: 1.1018 - val_accuracy: 0.9250 - val_loss: 0.2166
Epoch 2/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 8s 1s/step - accuracy: 0.9501 - loss: 0.1239 - val_accuracy: 0.9750 - val_loss: 0.0356
Epoch 3/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 8s 1s/step - accuracy: 0.9781 - loss: 0.0730 - val_accuracy: 1.0000 - val_loss: 0.0057
Epoch 4/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 1.0000 - loss: 0.0059 - val accuracy: 1.0000 - val loss:
```

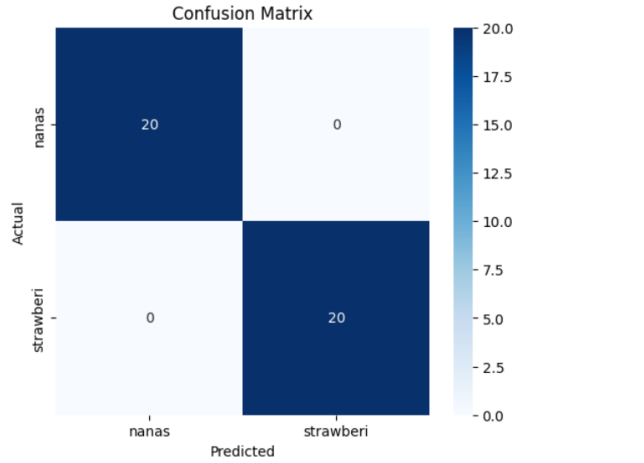
```
6/6 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 1.0000 - loss: 0.0059 - val_accuracy: 1.0000 - val_loss:
0.0097
Epoch 5/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 1.0000 - loss: 0.0102 - val_accuracy: 1.0000 - val_loss:
0.0020
Epoch 6/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 8s 1s/step - accuracy: 1.0000 - loss: 0.0024 - val_accuracy: 1.0000 - val_loss:
0.0030
Epoch 7/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 1.0000 - loss: 0.0083 - val_accuracy: 1.0000 - val_loss:
0.0020
Epoch 8/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 1.0000 - loss: 5.7630e-04 - val_accuracy: 1.0000 - val_los
s: 0.0020
Epoch 9/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 1.0000 - loss: 8.1500e-04 - val_accuracy: 1.0000 - val_los
s: 8.7922e-04
Epoch 10/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 1.0000 - loss: 0.0018 - val_accuracy: 1.0000 - val_loss:
0.0022
```

```python
plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label='Train Acc')
plt.plot(history.history['val_accuracy'], label='Val Acc')
plt.legend()
plt.title('Accuracy')

plt.subplot(1,2,2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Val Loss')
plt.legend()
plt.title('Loss')
plt.show()
```




```python
val_gen.reset()  # Reset generator validasi
pred = model.predict(val_gen)
pred_classes = np.argmax(pred, axis=1)
true_classes = val_gen.classes
labels = list(val_gen.class_indices.keys())

cm = confusion_matrix(true_classes, pred_classes)
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=labels, yticklabels=labels)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

```
2/2 ━━━━━━━━━━━━━━━━━━━━ 5s 2s/step
```



[56]:
```
print(classification_report(true_classes, pred_classes, target_names=labels))
```

```
              precision    recall  f1-score   support

      nanas       1.00      1.00      1.00        20
   strawberi      1.00      1.00      1.00        20

    accuracy                           1.00        40
   macro avg       1.00      1.00      1.00        40
weighted avg       1.00      1.00      1.00        40
```

+ Code + Markdown

[57]:
```
base_model.trainable = True
# Tentukan layer berapa ke atas yang bisa di-train ulang
for layer in base_model.layers[:100]:
    layer.trainable = False
```

F. Kesimpulan

Model MobileNetV2 dengan transfer learning efektif untuk klasifikasi nanas dan stroberi. Tantangan utama adalah memastikan dataset cukup representatif dan augmentasi yang tepat. Pengalaman ini meningkatkan pemahaman saya mengenai transfer learning dan pemrosesan citra.