In [1]:

```python
import os
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.preprocessing import image_dataset_from_directory

# Paths
train_dir = "/kaggle/input/tomato-leaf-disease/Tomato Leaf Disease/train"
test_dir  = "/kaggle/input/tomato-leaf-disease/Tomato Leaf Disease/test"
```

```
2025-09-17 10:11:26.468656: E external/local_xla/xla/stream_executor/cuda/cuda_fft.c
c:477] Unable to register cuFFT factory: Attempting to register factory for plugin c
uFFT when one has already been registered
WARNING: All log messages before absl::InitializeLog() is called are written to STDE
RR
E0000 00:00:1758103886.645434      36 cuda_dnn.cc:8310] Unable to register cuDNN fac
tory: Attempting to register factory for plugin cuDNN when one has already been regi
stered
E0000 00:00:1758103886.695561      36 cuda_blas.cc:1418] Unable to register cuBLAS f
actory: Attempting to register factory for plugin cuBLAS when one has already been r
egistered
```

In [2]:

```python
# Data augmentation
data_augmentation = keras.Sequential([
    layers.RandomFlip("horizontal"),
    layers.RandomRotation(0.2),
    layers.RandomZoom(0.2),
    layers.RandomContrast(0.2),
])

# Load training & test sets
img_size = (224, 224)
batch_size = 32

train_data = image_dataset_from_directory(
    train_dir,
    image_size=img_size,
    batch_size=batch_size,
    shuffle=True
)

test_data = image_dataset_from_directory(
    test_dir,
    image_size=img_size,
    batch_size=batch_size
)

# Save class names
class_names = train_data.class_names
print("Class names:", class_names)
```

```
# Prefetch for speed
train_data = train_data.prefetch(buffer_size=tf.data.AUTOTUNE)
test_data = test_data.prefetch(buffer_size=tf.data.AUTOTUNE)
```

```
I0000 00:00:1758103912.926072      36 gpu_device.cc:2022] Created device /job:localh
ost/replica:0/task:0/device:GPU:0 with 13942 MB memory:  -> device: 0, name: Tesla T
4, pci bus id: 0000:00:04.0, compute capability: 7.5
I0000 00:00:1758103912.926760      36 gpu_device.cc:2022] Created device /job:localh
ost/replica:0/task:0/device:GPU:1 with 13942 MB memory:  -> device: 1, name: Tesla T
4, pci bus id: 0000:00:05.0, compute capability: 7.5
Found 15064 files belonging to 10 classes.
Found 3771 files belonging to 10 classes.
Class names: ['Tomato___Bacterial_spot', 'Tomato___Early_blight', 'Tomato___Late_bli
ght', 'Tomato___Leaf_Mold', 'Tomato___Septoria_leaf_spot', 'Tomato___Spider_mites Tw
o-spotted_spider_mite', 'Tomato___Target_Spot', 'Tomato___Tomato_Yellow_Leaf_Curl_Vi
rus', 'Tomato___Tomato_mosaic_virus', 'Tomato___healthy']
```

In [3]:

```
# Load MobileNetV2 with pretrained weights
base_model = MobileNetV2(weights="imagenet", include_top=False, input_shape=(224, 2

# Freeze most layers, keep last 30 trainable for fine-tuning
base_model.trainable = True
for layer in base_model.layers[:-30]:
    layer.trainable = False

# Build model
inputs = keras.Input(shape=(224, 224, 3))
x = data_augmentation(inputs)
x = keras.applications.mobilenet_v2.preprocess_input(x)
x = base_model(x, training=False)
x = layers.GlobalAveragePooling2D()(x)
x = layers.Dropout(0.3)(x)
outputs = layers.Dense(len(class_names), activation="softmax")(x)

model = keras.Model(inputs, outputs)

# Compile with low LR for fine-tuning
model.compile(
    optimizer=keras.optimizers.Adam(learning_rate=1e-5),
    loss="sparse_categorical_crossentropy",
    metrics=["accuracy"]
)

model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/m
obilenet_v2/mobilenet_v2_weights_tf_dim_ordering_tf_kernels_1.0_224_no_top.h5
9406464/9406464 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

Model: "functional_1"

```
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ input_layer_1 (InputLayer)      │ (None, 224, 224, 3)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ sequential (Sequential)         │ (None, 224, 224, 3)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ true_divide (TrueDivide)        │ (None, 224, 224, 3)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ subtract (Subtract)             │ (None, 224, 224, 3)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ mobilenetv2_1.00_224            │ (None, 7, 7, 1280)     │     2,257,984 │
│ (Functional)                    │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ global_average_pooling2d        │ (None, 1280)           │             0 │
│ (GlobalAveragePooling2D)        │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dropout (Dropout)               │ (None, 1280)           │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense (Dense)                   │ (None, 10)             │        12,810 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 2,270,794 (8.66 MB)
 Trainable params: 1,539,210 (5.87 MB)
 Non-trainable params: 731,584 (2.79 MB)
```

In [4]:

```python
callbacks = [
    keras.callbacks.EarlyStopping(patience=5, restore_best_weights=True),
    keras.callbacks.ModelCheckpoint("best_model.keras", save_best_only=True)
]

history = model.fit(
    train_data,
    validation_data=test_data,
    epochs=25,
    callbacks=callbacks
)
```

```
Epoch 1/25
I0000 00:00:1758104011.617025      98 cuda_dnn.cc:529] Loaded cuDNN version 90300
```

```
471/471 ━━━━━━━━━━━━━━━━━━━━ 60s 90ms/step - accuracy: 0.3313 - loss: 2.0005 - val_a
ccuracy: 0.5924 - val_loss: 1.2252
Epoch 2/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 42s 89ms/step - accuracy: 0.7361 - loss: 0.8297 - val_a
ccuracy: 0.7346 - val_loss: 0.7959
Epoch 3/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 86ms/step - accuracy: 0.8175 - loss: 0.5784 - val_a
ccuracy: 0.8255 - val_loss: 0.5532
Epoch 4/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.8528 - loss: 0.4554 - val_a
ccuracy: 0.8706 - val_loss: 0.4144
Epoch 5/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.8784 - loss: 0.3754 - val_a
ccuracy: 0.8921 - val_loss: 0.3441
Epoch 6/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.8930 - loss: 0.3289 - val_a
ccuracy: 0.9075 - val_loss: 0.2924
Epoch 7/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9062 - loss: 0.2873 - val_a
ccuracy: 0.9146 - val_loss: 0.2660
Epoch 8/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 87ms/step - accuracy: 0.9189 - loss: 0.2562 - val_a
ccuracy: 0.9223 - val_loss: 0.2412
Epoch 9/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 87ms/step - accuracy: 0.9240 - loss: 0.2339 - val_a
ccuracy: 0.9247 - val_loss: 0.2340
Epoch 10/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9300 - loss: 0.2146 - val_a
ccuracy: 0.9326 - val_loss: 0.2119
Epoch 11/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9351 - loss: 0.1939 - val_a
ccuracy: 0.9321 - val_loss: 0.2098
Epoch 12/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9395 - loss: 0.1825 - val_a
ccuracy: 0.9374 - val_loss: 0.1950
Epoch 13/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 87ms/step - accuracy: 0.9409 - loss: 0.1727 - val_a
ccuracy: 0.9281 - val_loss: 0.2105
Epoch 14/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 87ms/step - accuracy: 0.9427 - loss: 0.1720 - val_a
ccuracy: 0.9361 - val_loss: 0.1980
Epoch 15/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9440 - loss: 0.1655 - val_a
ccuracy: 0.9433 - val_loss: 0.1833
Epoch 16/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 87ms/step - accuracy: 0.9508 - loss: 0.1477 - val_a
ccuracy: 0.9419 - val_loss: 0.1856
Epoch 17/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 42s 88ms/step - accuracy: 0.9538 - loss: 0.1347 - val_a
ccuracy: 0.9451 - val_loss: 0.1766
Epoch 18/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9547 - loss: 0.1365 - val_a
ccuracy: 0.9422 - val_loss: 0.1727
Epoch 19/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9612 - loss: 0.1141 - val_a
ccuracy: 0.9464 - val_loss: 0.1623
```

```
Epoch 20/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 86ms/step - accuracy: 0.9604 - loss: 0.1166 - val_a
ccuracy: 0.9462 - val_loss: 0.1665
Epoch 21/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 86ms/step - accuracy: 0.9612 - loss: 0.1165 - val_a
ccuracy: 0.9483 - val_loss: 0.1625
Epoch 22/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 86ms/step - accuracy: 0.9665 - loss: 0.1038 - val_a
ccuracy: 0.9462 - val_loss: 0.1646
Epoch 23/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9654 - loss: 0.1105 - val_a
ccuracy: 0.9523 - val_loss: 0.1503
Epoch 24/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 87ms/step - accuracy: 0.9713 - loss: 0.0899 - val_a
ccuracy: 0.9480 - val_loss: 0.1564
Epoch 25/25
471/471 ━━━━━━━━━━━━━━━━━━━━ 41s 88ms/step - accuracy: 0.9697 - loss: 0.0930 - val_a
ccuracy: 0.9520 - val_loss: 0.1462
```

In [5]:

```python
test_loss, test_acc = model.evaluate(test_data)
print("Test Accuracy:", test_acc)
```

```
118/118 ━━━━━━━━━━━━━━━━━━━━ 6s 52ms/step - accuracy: 0.9530 - loss: 0.1421
Test Accuracy: 0.9520021080970764
```

In [7]:

```python
import requests
from PIL import Image

# Example image URL
url = "https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQDAr_Ewp3xi3MFHdiXewcu
response = requests.get(url, stream=True)
img = Image.open(response.raw).convert("RGB")

# Preprocess
img = img.resize((224, 224))
img_array = np.array(img) / 255.0
img_array = np.expand_dims(img_array, axis=0)

# Predict
pred = model.predict(img_array)
print("Prediction:", class_names[np.argmax(pred)])
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step
Prediction: Tomato___Late_blight
```

In [8]:

```python
# Save the whole model (architecture + weights + optimizer state)
model.save("plant_disease_model.h5")
```