```
import numpy as np
import os
import cv2
import shutil
import random as rn
from tqdm import tqdm
import matplotlib.pyplot as plt
import tensorflow as tf

from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
     Mounted at /content/drive
```

```
data_dir ="/content/drive/MyDrive/Flowers-Dataset/flowers"
```

```
print(os.listdir("/content/drive/MyDrive/Flowers-Dataset/flowers"))
```

```
     ['rose', 'dandelion', 'sunflower', 'tulip', 'daisy']
```

```
batch_size = 32
img_height = 180
img_width = 180
num_classes = 5
```

```
train_ds = tf.keras.preprocessing.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="training",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

```
     Found 4317 files belonging to 5 classes.
     Using 3454 files for training.
```

```
val_ds = tf.keras.preprocessing.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="validation",
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

```
     Found 4317 files belonging to 5 classes.
     Using 863 files for validation.
```

```
class_names = train_ds.class_names
class_names
```

```
['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']
```

```
AUTOTUNE = tf.data.AUTOTUNE

train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)
```

```
normalization_layer = layers.experimental.preprocessing.Rescaling(1./255)
```

```
normalized_ds = train_ds.map(lambda x, y: (normalization_layer(x), y))
image_batch, labels_batch = next(iter(normalized_ds))
first_image = image_batch[0]
# Notice the pixels values are now in `[0,1]`.
print(np.min(first_image), np.max(first_image))
```

```
0.0 1.0
```

Image Augmentation

```
data_augmentation = keras.Sequential(
  [
    layers.experimental.preprocessing.RandomFlip("horizontal",
                                                 input_shape=(img_height,
                                                              img_width,
                                                              3)),
    layers.experimental.preprocessing.RandomRotation(0.1),
    layers.experimental.preprocessing.RandomZoom(0.1),
  ]
)
```

Model Creation / Adding Layers

```
model = Sequential([
  data_augmentation,
  layers.experimental.preprocessing.Rescaling(1./255),
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(128, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(256, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Dropout(0.3),
  layers.Flatten(),
  layers.Dense(512, activation='relu'),
  layers.Dense(num_classes)
])
```

Compling the Model

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

Fitting the Model

```
epochs=25
model.fit(
  train_ds,
  validation_data=val_ds,
  epochs=epochs
)
```

```
    Epoch 1/25
    108/108 [==============================] - 104s 848ms/step - loss: 1.3140 - accuracy
    Epoch 2/25
    108/108 [==============================] - 3s 29ms/step - loss: 1.0246 - accuracy: 0
    Epoch 3/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.9208 - accuracy: 0
    Epoch 4/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.8606 - accuracy: 0
    Epoch 5/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.8408 - accuracy: 0
    Epoch 6/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.7981 - accuracy: 0
    Epoch 7/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.7698 - accuracy: 0
    Epoch 8/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.7301 - accuracy: 0
    Epoch 9/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.7165 - accuracy: 0
    Epoch 10/25
    108/108 [==============================] - 3s 31ms/step - loss: 0.6941 - accuracy: 0
    Epoch 11/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.6631 - accuracy: 0
    Epoch 12/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.6291 - accuracy: 0
    Epoch 13/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.6053 - accuracy: 0
    Epoch 14/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.5805 - accuracy: 0
    Epoch 15/25
    108/108 [==============================] - 3s 30ms/step - loss: 0.5771 - accuracy: 0
```

```
Epoch 16/25
108/108 [==============================] - 3s 30ms/step - loss: 0.5336 - accuracy: 0
Epoch 17/25
108/108 [==============================] - 3s 30ms/step - loss: 0.5350 - accuracy: 0
Epoch 18/25
108/108 [==============================] - 3s 30ms/step - loss: 0.5016 - accuracy: 0
Epoch 19/25
108/108 [==============================] - 3s 30ms/step - loss: 0.4718 - accuracy: 0
Epoch 20/25
108/108 [==============================] - 3s 30ms/step - loss: 0.4694 - accuracy: 0
Epoch 21/25
108/108 [==============================] - 3s 29ms/step - loss: 0.4698 - accuracy: 0
Epoch 22/25
108/108 [==============================] - 3s 30ms/step - loss: 0.4473 - accuracy: 0
Epoch 23/25
108/108 [==============================] - 3s 31ms/step - loss: 0.3950 - accuracy: 0
Epoch 24/25
108/108 [==============================] - 3s 30ms/step - loss: 0.4096 - accuracy: 0
Epoch 25/25
108/108 [==============================] - 3s 29ms/step - loss: 0.3779 - accuracy: 0
<keras.callbacks.History at 0x7fcdde88a790>
```

Testing on unseen image Data

```
from matplotlib import image as im
from matplotlib import pyplot
from keras.preprocessing import image
from PIL import Image
data = im.imread('/content/drive/MyDrive/th.jpg')
img=image.load_img('/content/drive/MyDrive/th.jpg', target_size=(180, 180))

test_img=np.expand_dims(img , axis=0)

result = model.predict(test_img)
pred = np.argmax(result)
print(result)
print(pred)
print(class_names)
```

```
[[ 0.36658025 -3.2330253   2.590238   -4.2220488   3.3819256 ]]
4
['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']
```

```
image = tf.keras.preprocessing.image.load_img('/content/drive/MyDrive/th (1).jpg', target_
input_arr = tf.keras.preprocessing.image.img_to_array(image)
input_arr = np.array([input_arr])
result = model.predict(input_arr)
pred = np.argmax(result)
print(class_names)
print(pred)
print(class_names[pred])
```

```
['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']
3
sunflower
```

Saving Model

```
model.save("/content/drive/MyDrive/flower_model.h5")
```

```
from tensorflow.keras.models import load_model

savedModel=load_model("/content/drive/MyDrive/flower_model.h5")
savedModel.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 sequential (Sequential)     (None, 180, 180, 3)       0         
                                                                 
 rescaling_2 (Rescaling)     (None, 180, 180, 3)       0         
                                                                 
 conv2d_5 (Conv2D)           (None, 180, 180, 16)      448       
                                                                 
 max_pooling2d_5 (MaxPooling  (None, 90, 90, 16)       0         
 2D)                                                             
                                                                 
 conv2d_6 (Conv2D)           (None, 90, 90, 32)        4640      
                                                                 
 max_pooling2d_6 (MaxPooling  (None, 45, 45, 32)       0         
 2D)                                                             
                                                                 
 conv2d_7 (Conv2D)           (None, 45, 45, 64)        18496     
                                                                 
 max_pooling2d_7 (MaxPooling  (None, 22, 22, 64)       0         
 2D)                                                             
                                                                 
 conv2d_8 (Conv2D)           (None, 22, 22, 128)       73856     
                                                                 
 max_pooling2d_8 (MaxPooling  (None, 11, 11, 128)      0         
 2D)                                                             
                                                                 
 conv2d_9 (Conv2D)           (None, 11, 11, 256)       295168    
                                                                 
 max_pooling2d_9 (MaxPooling  (None, 5, 5, 256)        0         
 2D)                                                             
                                                                 
 dropout_1 (Dropout)         (None, 5, 5, 256)         0         
                                                                 
 flatten_1 (Flatten)         (None, 6400)              0         
                                                                 
 dense_1 (Dense)             (None, 512)               3277312   
                                                                 
 dense_2 (Dense)             (None, 5)                 2565      
                                                                 
=================================================================
Total params: 3,672,485
Trainable params: 3,672,485
Non-trainable params: 0
```