Import Required Libraries

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Image Augmentation

```
IMG_SHAPE = 128
IMG_FOLDER = "./flowers/"
BATCH_SIZE = 64

datagen = 
tf.keras.preprocessing.image.ImageDataGenerator(rescale=1/255.0, 
shear_range=0.2, zoom_range=0.2, 

rotation_range=45, horizontal_flip=True, vertical_flip=True, 

validation_split=0.2)

train = datagen.flow_from_directory(IMG_FOLDER, 
target_size=(IMG_SHAPE,IMG_SHAPE), color_mode='rgb',
                                    class_mode='categorical', 
keep_aspect_ratio=True, batch_size=BATCH_SIZE,
                                    shuffle=True, subset='training')
test = datagen.flow_from_directory(IMG_FOLDER, 
target_size=(IMG_SHAPE,IMG_SHAPE), color_mode='rgb',
                                class_mode='categorical', 
keep_aspect_ratio=True, batch_size=BATCH_SIZE,
                                shuffle=False, subset='validation')

Found 3457 images belonging to 5 classes.
Found 860 images belonging to 5 classes.
```

Create the Model

```
model = tf.keras.models.Sequential()
```

Add Layers to the Model

```
model.add(tf.keras.layers.Input((IMG_SHAPE,IMG_SHAPE,3)))
model.add(tf.keras.layers.Conv2D(16, 3, activation='relu'))
model.add(tf.keras.layers.Conv2D(16, 3, padding='same', 
activation='relu'))
model.add(tf.keras.layers.MaxPool2D(2))
model.add(tf.keras.layers.Conv2D(32, 3, activation='relu'))
model.add(tf.keras.layers.Conv2D(32, 3, padding='same', 
activation='relu'))
model.add(tf.keras.layers.MaxPool2D(2))
model.add(tf.keras.layers.Conv2D(64, 3, activation='relu'))
model.add(tf.keras.layers.Conv2D(64, 3, padding='same',
activation='relu'))
model.add(tf.keras.layers.MaxPool2D(2))
model.add(tf.keras.layers.Conv2D(128, 3, activation='relu'))
model.add(tf.keras.layers.Conv2D(128, 3, padding='same',
activation='relu'))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(128, activation='relu'))
model.add(tf.keras.layers.Dense(64, activation='relu'))
model.add(tf.keras.layers.Dense(5, activation='softmax'))

model.summary()
```

```
Model: "sequential"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 16)      448

 conv2d_1 (Conv2D)           (None, 126, 126, 16)      2320

 max_pooling2d (MaxPooling2D  (None, 63, 63, 16)       0
 )

 conv2d_2 (Conv2D)           (None, 61, 61, 32)        4640

 conv2d_3 (Conv2D)           (None, 61, 61, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 32)       0
 2D)

 conv2d_4 (Conv2D)           (None, 28, 28, 64)        18496

 conv2d_5 (Conv2D)           (None, 28, 28, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 conv2d_6 (Conv2D)           (None, 12, 12, 128)       73856

 conv2d_7 (Conv2D)           (None, 12, 12, 128)       147584

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 128)               2359424

 dense_1 (Dense)             (None, 64)                8256
 dense_2 (Dense)             (None, 5)                 325

=================================================================
Total params: 2,661,525
Trainable params: 2,661,525
Non-trainable params: 0
_________________________________________________________________
```

Compile the Model

```
model.compile(loss='categorical_crossentropy',
optimizer=tf.keras.optimizers.Adam(1e-4), metrics=['accuracy'])
```

Fit the Model

```
hist = model.fit(train, epochs=25)
```

```
Epoch 1/25
55/55 [==============================] - 13s 167ms/step - loss: 1.5647
- accuracy: 0.2583
Epoch 2/25
55/55 [==============================] - 9s 169ms/step - loss: 1.3117
- accuracy: 0.4220
Epoch 3/25
55/55 [==============================] - 9s 168ms/step - loss: 1.2742
- accuracy: 0.4492
Epoch 4/25
55/55 [==============================] - 9s 167ms/step - loss: 1.2004
- accuracy: 0.4782
Epoch 5/25
55/55 [==============================] - 9s 168ms/step - loss: 1.1626
- accuracy: 0.4973
Epoch 6/25
55/55 [==============================] - 9s 168ms/step - loss: 1.1296
- accuracy: 0.5242
Epoch 7/25
55/55 [==============================] - 9s 167ms/step - loss: 1.1192
- accuracy: 0.5236
Epoch 8/25
55/55 [==============================] - 9s 167ms/step - loss: 1.1265
- accuracy: 0.5308
Epoch 9/25
55/55 [==============================] - 9s 168ms/step - loss: 1.0652
- accuracy: 0.5652
Epoch 10/25
55/55 [==============================] - 9s 168ms/step - loss: 1.0664
- accuracy: 0.5652
Epoch 11/25
55/55 [==============================] - 9s 172ms/step - loss: 1.0477
```

```
- accuracy: 0.5710
Epoch 12/25
55/55 [==============================] - 9s 168ms/step - loss: 1.0329
- accuracy: 0.5736
Epoch 13/25
55/55 [==============================] - 9s 169ms/step - loss: 0.9977
- accuracy: 0.5930
Epoch 14/25
55/55 [==============================] - 9s 167ms/step - loss: 0.9795
- accuracy: 0.6075
Epoch 15/25
55/55 [==============================] - 9s 169ms/step - loss: 0.9663
- accuracy: 0.6138
Epoch 16/25
55/55 [==============================] - 9s 167ms/step - loss: 1.0038
- accuracy: 0.5962
Epoch 17/25
55/55 [==============================] - 9s 168ms/step - loss: 0.9437
- accuracy: 0.6193
Epoch 18/25
55/55 [==============================] - 9s 169ms/step - loss: 0.9444
- accuracy: 0.6266
Epoch 19/25
55/55 [==============================] - 9s 169ms/step - loss: 1.0283
- accuracy: 0.5863
Epoch 20/25
55/55 [==============================] - 9s 171ms/step - loss: 0.9238
- accuracy: 0.6231
Epoch 21/25
55/55 [==============================] - 9s 170ms/step - loss: 0.9117
- accuracy: 0.6358
Epoch 22/25
55/55 [==============================] - 9s 168ms/step - loss: 0.8958
- accuracy: 0.6451
Epoch 23/25
55/55 [==============================] - 9s 170ms/step - loss: 0.8946
- accuracy: 0.6404
Epoch 24/25
55/55 [==============================] - 9s 168ms/step - loss: 0.9289
- accuracy: 0.6242
Epoch 25/25
55/55 [==============================] - 9s 168ms/step - loss: 0.9256
- accuracy: 0.6407
```

Save the Model

```
model.save("flowers.h5")
```

Test the Model

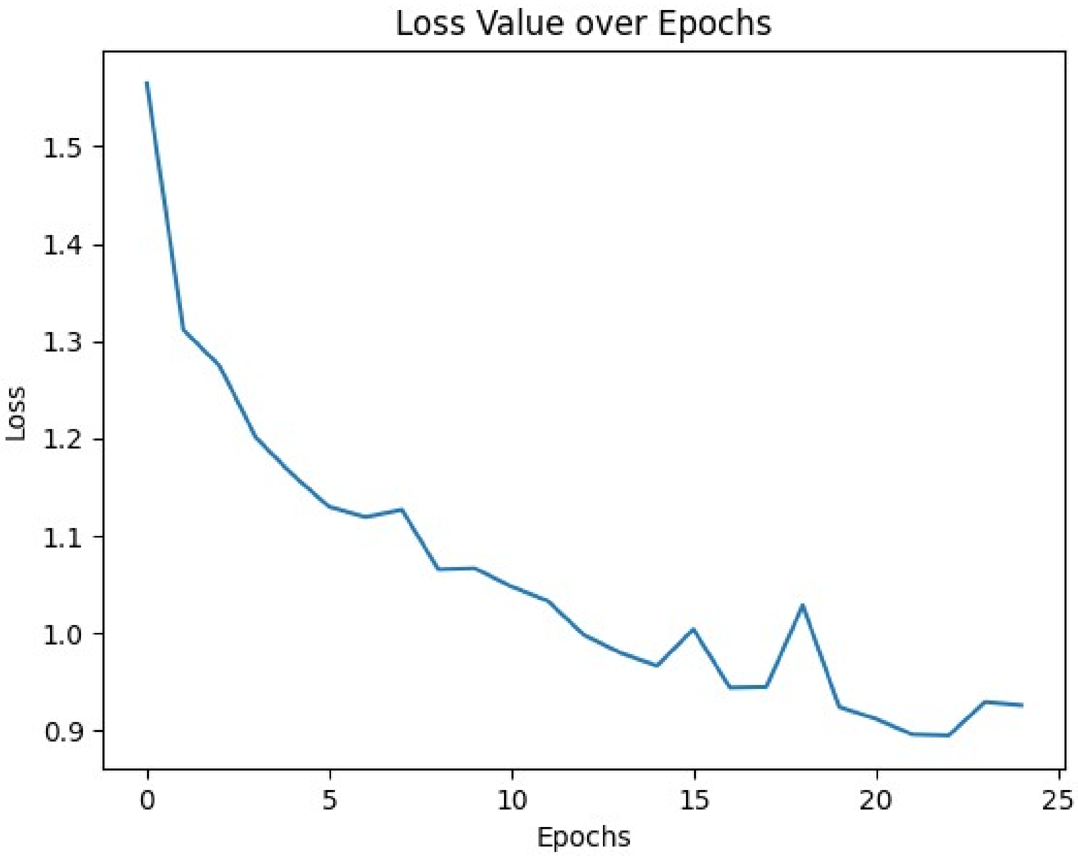
```
loss, acc = model.evaluate(test)
```

```
14/14 [==============================] - 2s 156ms/step - loss: 0.9511
- accuracy: 0.6198
```

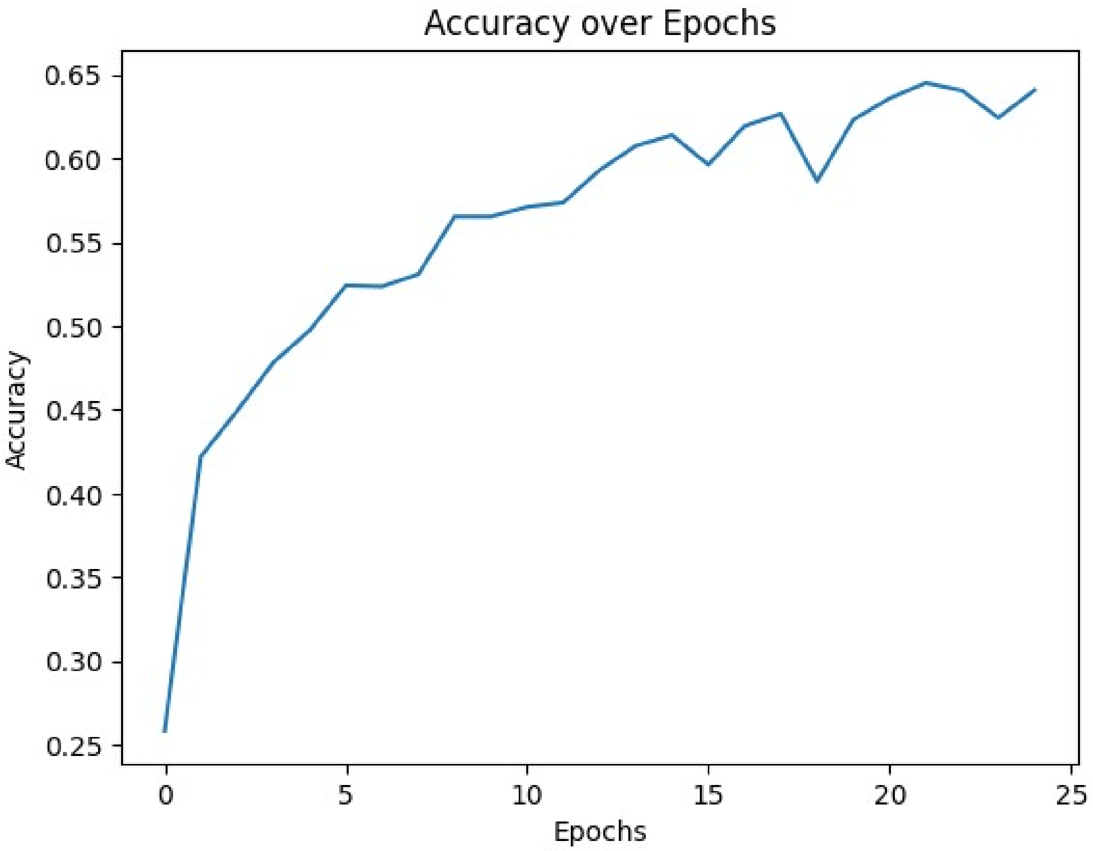
```
print(f"Loss Value for Test Data : {loss:0.2f}")
print(f"Accuracy for Test Data : {acc:0.2f}")
```

```
Loss Value for Test Data : 0.95
Accuracy for Test Data : 0.62
```

```
plt.plot(hist.history['loss'])
plt.title("Loss Value over Epochs")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.show()
```



```
plt.plot(hist.history['accuracy'])
plt.title("Accuracy over Epochs")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.show()
```


Accuracy over Epochs
Accuracy
0.25
0.30
0.35
0.40
0.45
0.50
0.55
0.60
0.65
0
5
10
15
20
25
Epochs