# Build CNN model for classification of flowers

## 1. Download the Dataset [dataset]

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

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

```
!unzip "/content/drive/MyDrive/Colab Notebooks/Flowers-Dataset.zip"
```

```
inflating: flowers/daisy/10466290366_cc72e33532.jpg
inflating: flowers/daisy/10466558316_a7198b87e2.jpg
inflating: flowers/daisy/10555749515_13a12a026e.jpg
inflating: flowers/daisy/10555815624_dc211569b0.jpg
inflating: flowers/daisy/10555826524_423eb8bf71_n.jpg
inflating: flowers/daisy/10559679065_50d2b16f6d.jpg
inflating: flowers/daisy/105806915_a9c13e2106_n.jpg
inflating: flowers/daisy/10712722853_5632165b04.jpg
inflating: flowers/daisy/107592979_aaa9cdfe78_m.jpg
inflating: flowers/daisy/10770585085_4742b9dac3_n.jpg
inflating: flowers/daisy/10841136265_af473efc60.jpg
inflating: flowers/daisy/10993710036_2033222c91.jpg
inflating: flowers/daisy/10993818044_4c19b86c82.jpg
inflating: flowers/daisy/10994032453_ac7f8d9e2e.jpg
inflating: flowers/daisy/11023214096_b5b39fab08.jpg
inflating: flowers/daisy/11023272144_fce94401f2_m.jpg
inflating: flowers/daisy/11023277956_8980d53169_m.jpg
inflating: flowers/daisy/11124324295_503f3a0804.jpg
inflating: flowers/daisy/1140299375_3aa7024466.jpg
inflating: flowers/daisy/11439894966_dca877f0cd.jpg
inflating: flowers/daisy/1150395827_6f94a5c6e4_n.jpg
inflating: flowers/daisy/11642632_1e7627a2cc.jpg
inflating: flowers/daisy/11834945233_a53b7a92ac_m.jpg
inflating: flowers/daisy/11870378973_2ec1919f12.jpg
inflating: flowers/daisy/11891885265_ccefec7284_n.jpg
inflating: flowers/daisy/12193032636_b50ae7db35_n.jpg
inflating: flowers/daisy/12348343085_d4c396e5b5_m.jpg

inflating: flowers/daisy/12585131704_0f64b17059_m.jpg
inflating: flowers/daisy/12601254324_3cb62c254a_m.jpg
inflating: flowers/daisy/1265350143_6e2b276ec9.jpg
inflating: flowers/daisy/12701063955_4840594ea6_n.jpg
inflating: flowers/daisy/1285423653_18926dc2c8_n.jpg
inflating: flowers/daisy/1286274236_1d7ac84efb_n.jpg
inflating: flowers/daisy/12891819633_e4c82b51e8.jpg
inflating: flowers/daisy/1299501272_59d9da5510_n.jpg
inflating: flowers/daisy/1306119996_ab8ae14d72_n.jpg
inflating: flowers/daisy/1314069875_da8dc023c6_m.jpg
inflating: flowers/daisy/1342002397_9503c97b49.jpg
inflating: flowers/daisy/134409839_71069a95d1_m.jpg
inflating: flowers/daisy/1344985627_c3115e2d71_n.jpg
inflating: flowers/daisy/1349195645_2cd9df44d6_n.jpg
```

```
inflating: flowers/daisy/13491959645_2cd9df44d6_n.jpg
inflating: flowers/daisy/1354396826_2868631432_m.jpg
inflating: flowers/daisy/1355787476_32e9f2a30b.jpg
inflating: flowers/daisy/13583238844_573df2de8e_m.jpg
inflating: flowers/daisy/1374193928_a52320eafa.jpg
inflating: flowers/daisy/13826249325_f61cb15f86_n.jpg
inflating: flowers/daisy/13901930939_a7733c03f0_n.jpg
inflating: flowers/daisy/1392131677_116ec04751.jpg
inflating: flowers/daisy/1392946544_115acbb2d9.jpg
inflating: flowers/daisy/13953307149_f8de6a768c_m.jpg
inflating: flowers/daisy/1396526833_fb867165be_n.jpg
inflating: flowers/daisy/13977181862_f8237b6b52.jpg
inflating: flowers/daisy/14021430525_e06baf93a9.jpg
inflating: flowers/daisy/14073784469_ffb12f3387_n.jpg
inflating: flowers/daisy/14087947408_9779257411_n.jpg
inflating: flowers/daisy/14088053307_1a13a0bf91_n.jpg
inflating: flowers/daisy/14114116486_0bb6649bc1_m.jpg
inflating: flowers/daisy/14147016029_8d3cf2414e.jpg
```

# 2. Image Augmentation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255, horizontal_flip = True, vertical_flip

x_train = train_datagen.flow_from_directory(r"/content/flowers", target_size = (64,64) , c
```

```
Found 4317 images belonging to 5 classes.
```

# 3. Model Building and also Split dataset into training and testing sets

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense

model = Sequential()
```

# 4. Add the layers (Convolution, MaxPooling, Flatten, Dense-(HiddenLayers), Output)

```
model.add(Convolution2D(32, (3,3), activation = "relu", input_shape = (64,64,3) ))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Flatten())
model.add(Dense(300, activation = "relu"))
model.add(Dense(150, activation = "relu")) #multiple dense layers
model.add(Dense(5, activation = "softmax")) #output layer
```

# 5. Compile The Model

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

```
44
```

```
#model.fit(x_train, epochs = 15, validation_data = x_test, steps_per_epoch = len(x_train),

model.fit(x_train, epochs = 15, steps_per_epoch = len(x_train))
```

```
Epoch 1/15
44/44 [==============================] - 22s 284ms/step - loss: 1.4755 - accuracy: 0
Epoch 2/15
44/44 [==============================] - 13s 290ms/step - loss: 1.1200 - accuracy: 0
Epoch 3/15
44/44 [==============================] - 13s 288ms/step - loss: 1.0583 - accuracy: 0
Epoch 4/15
44/44 [==============================] - 13s 290ms/step - loss: 0.9895 - accuracy: 0
Epoch 5/15
44/44 [==============================] - 13s 285ms/step - loss: 0.9492 - accuracy: 0
Epoch 6/15
44/44 [==============================] - 13s 286ms/step - loss: 0.9329 - accuracy: 0
Epoch 7/15
44/44 [==============================] - 13s 289ms/step - loss: 0.8723 - accuracy: 0
Epoch 8/15
44/44 [==============================] - 13s 289ms/step - loss: 0.8797 - accuracy: 0
Epoch 9/15
44/44 [==============================] - 13s 288ms/step - loss: 0.8268 - accuracy: 0
Epoch 10/15
44/44 [==============================] - 13s 284ms/step - loss: 0.8126 - accuracy: 0
Epoch 11/15
44/44 [==============================] - 13s 290ms/step - loss: 0.7847 - accuracy: 0
Epoch 12/15
44/44 [==============================] - 13s 289ms/step - loss: 0.7878 - accuracy: 0
Epoch 13/15
44/44 [==============================] - 13s 292ms/step - loss: 0.7425 - accuracy: 0
Epoch 14/15
44/44 [==============================] - 13s 296ms/step - loss: 0.7069 - accuracy: 0
Epoch 15/15
44/44 [==============================] - 13s 288ms/step - loss: 0.7175 - accuracy: 0
<keras.callbacks.History at 0x7f80748fae90>
```

# 6. Fit The Model

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
```

```
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 flatten (Flatten)           (None, 30752)             0

 dense (Dense)               (None, 300)               9225900

 dense_1 (Dense)             (None, 150)               45150

 dense_2 (Dense)             (None, 5)                 755

=================================================================
Total params: 9,272,701
Trainable params: 9,272,701
Non-trainable params: 0
_________________________________________________________________
```

# 7. Save The Model

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

# 8. Test The Model

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np

model = load_model("/content/flowers.h5")
img = image.load_img("/content/drive/MyDrive/Colab Notebooks/download.jpeg", target_size =
x = image.img_to_array(img)
x = np.expand_dims(x,axis = 0)
pred = model.predict(x)

labels = ['daisy','dandelion','roses','sunflowers','tulips']
print("Input Image is\n")
img
```

```
Input Image is
```

```
print("Classification of Flower is:",labels[np.argmax(pred)])
```

```
Classification of Flower is: sunflowers
```

Colab paid products - Cancel contracts here

✓ 0s completed at 6:08 PM