Image Augmentation

```
import tensorflow
```

```
import tensorflow.keras.preprocessing
```

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

```
train_datagen = ImageDataGenerator(rescale=1./255,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
```

```
test_datagen = ImageDataGenerator(rescale=1./255)
```

Load Data

```
xtrain =
train_datagen.flow_from_directory('/content//drive//MyDrive//Flowers-
Dataset//flowers',target_size=(64,64),
                                          class_mode='categorical',
                                          batch_size=100)
```

```
Found 4327 images belonging to 5 classes.
```

```
x_test=test_datagen.flow_from_directory('/content//drive//MyDrive//
Flowers-Dataset//flowers',target_size=(64,64),

class_mode="categorical",batch_size=24)
```

```
Found 4327 images belonging to 5 classes.
```

```
xtrain.class_indices
```

```
{'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4}
```

```
x_test.class_indices
```

```
{'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4}
```

Create Model

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

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

```
model=Sequential()
```

Add Layers

a)Convolution Layer

```
model.add(Convolution2D(32,
(3,3),kernel_initializer="random_uniform",activation="relu",strides=(1
,1),input_shape=(64,64,3)))
```

b)MaxPooling Layer

```
model.add(MaxPooling2D(pool_size=(2,2)))
```

c)Flatten

```
model.add(Flatten())
```

d)Dense(Hidden layer)

```
model.add(Dense(300,activation="relu"))

model.add(Dense(300,activation="relu"))
```

e)Output layer

```
model.add(Dense(5,activation="softmax"))
```

Compile the model

```
model.compile(loss="categorical_crossentropy",metrics=['accuracy'],opt
imizer='adam')
```

Fit the model

```
model.fit(xtrain,epochs=5,steps_per_epoch=len(xtrain),validation_data=
x_test,validation_steps=len(x_test))
```

```
Epoch 1/5
44/44 [==============================] - 675s 15s/step - loss: 1.3175
- accuracy: 0.4292 - val_loss: 1.1927 - val_accuracy: 0.5131
Epoch 2/5
44/44 [==============================] - 60s 1s/step - loss: 1.0988 -
accuracy: 0.5558 - val_loss: 1.1278 - val_accuracy: 0.5621
Epoch 3/5
44/44 [==============================] - 58s 1s/step - loss: 1.0229 -
accuracy: 0.5986 - val_loss: 0.9473 - val_accuracy: 0.6298
Epoch 4/5
44/44 [==============================] - 58s 1s/step - loss: 0.9315 -
accuracy: 0.6499 - val_loss: 0.9684 - val_accuracy: 0.6205
Epoch 5/5
44/44 [==============================] - 60s 1s/step - loss: 0.8719 -
accuracy: 0.6721 - val_loss: 0.8900 - val_accuracy: 0.6644
```

```
<keras.callbacks.History at 0x7febf620a550>
```

Save the model

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

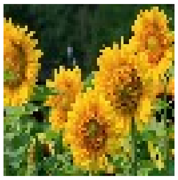
Test the model

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

```
model=load_model("Flowers.h5")
```

```
img=image.load_img('/content/drive/MyDrive/Flowers-Dataset/flowers/
sunflower/1008566138_6927679c8a.jpg',target_size=(64,64))
```

```
img
```



```
x=image.img_to_array(img)
```

```
x=np.expand_dims(x,axis=0)
```

```
pred=model.predict(x)
```

```
1/1 [==============================] - 0s 28ms/step
```

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

```
index[np.argmax(pred)]
```

```
{"type":"string"}
```