Build CNN Model for Classi cation Of Flowers

Download the dataset here.

```
# Unzip data
!unzip '/content/Flowers-Dataset.zip'
```

```
Archive:  /content/Flowers-Dataset.zip
replace flowers/daisy/100080576_f52e8ee070_n.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ena
```

1. Image Augmentation

```
#import lib. from tensorflow.keras.preprocessing.image import

ImageDataGenerator
```

```
#augmentation on flowers

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


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


xrose = rose_datagen.flow_from_directory('/content/flowers',
target_size=(64,64),
class_mode='categorical',
batch_size=100)
```

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

```
xtulip = tulip_datagen.flow_from_directory('/content/flowers',
target_size=(64,64),
class_mode='categorical',
batch_size=100)
```

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

2. Create Model

```
#import lib.

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

3. Add Layers (Convolution,MaxPooling,Flatten,Dense(Hidden Layers),Output)

```
# Add a layers

model = Sequential() # Initializing sequential model
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3))) # convolution l
model.add(MaxPooling2D(pool_size=(2, 2))) # Max pooling layer model.add(Flatten()) #
Flatten layer model.add(Dense(300,activation='relu')) # Hidden layer 1
model.add(Dense(150,activation='relu')) # Hidden layer 2
model.add(Dense(5,activation='softmax')) # Output layer
```

4. Compile The Model

```
# Compiling the model

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

5. Fit The Model

```
model.fit_generator(xrose,
steps_per_epoch=len(xrose),
epochs=10,
validation_data=xtulip,
validation_steps=len(xtulip))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: UserWarning: `Model.
  """
Epoch 1/10
44/44 [==============================] - 45s 1s/step - loss: 1.7388 - accuracy: 0.35
Epoch 2/10
44/44 [==============================] - 44s 1s/step - loss: 1.1245 - accuracy: 0.54
Epoch 3/10
44/44 [==============================] - 45s 1s/step - loss: 1.0298 - accuracy: 0.60
Epoch 4/10
```

```
44/44 [==============================] - 44s 1s/step - loss: 0.9845 - accuracy: 0.61
Epoch 5/10
44/44 [==============================] - 44s 1s/step - loss: 0.8992 - accuracy: 0.64
Epoch 6/10
44/44 [==============================] - 44s 1s/step - loss: 0.8754 - accuracy: 0.66
Epoch 7/10
44/44 [==============================] - 44s 1s/step - loss: 0.8217 - accuracy: 0.69
Epoch 8/10
44/44 [==============================] - 44s 1s/step - loss: 0.7950 - accuracy: 0.69
Epoch 9/10
44/44 [==============================] - 44s 999ms/step - loss: 0.7403 - accuracy: 0
Epoch 10/10
44/44 [==============================] - 44s 999ms/step - loss: 0.7182 - accuracy: 0
<keras.callbacks.History at 0x7f8cfcc75450>
```

6. Save The Model

```
model.save('rose.h5')
```

7. Test The Model

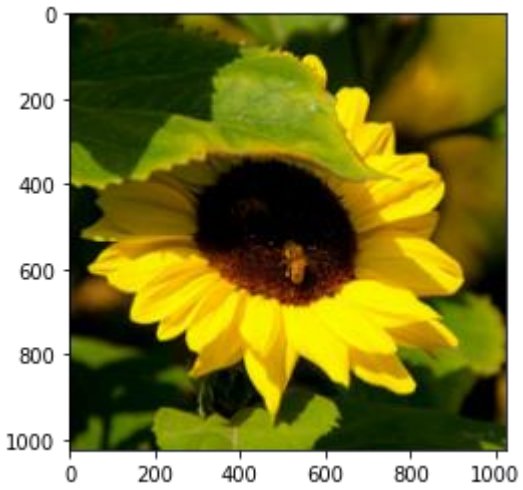
```
from tensorflow.keras.preprocessing import image
 import numpy as np import matplotlib.pyplot as
plt


#testing 1 img =
image.load_img('/content/flowers/sunflower/12471443383_b71e7a7480_m.jpg',target_size x =
image.img_to_array(img) # Converting image into array x = np.expand_dims(x,axis=0) #
expanding Dimensions
pred = np.argmax(model.predict(x)) # Predicting the higher probablity index op
= ['daisy','dandelion','rose','sunflower','tulip'] # Creating list op[pred] #
List indexing with output
```

```
1/1 [==============================] - 0s 101ms/step 'rose'
```

```
img = image.load_img('/content/flowers/sunflower/12471443383_b71e7a7480_m.jpg',target_size
plt.imshow(img)
```

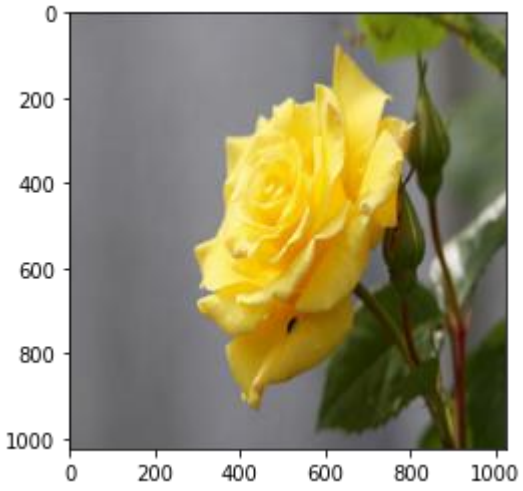
```
<matplotlib.image.AxesImage at 0x7f8cf73ae990>
```

```
img =
```




```
image.load_img('/content/flowers/rose/14145188939_b4de638bd3_n.jpg',target_size=(102
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7f8cf90b8bd0>
```



28s completed at 12.45 PM